Red Hat Enterprise Linux 8

InfiniBand 및 RDMA 네트워크 구성

Red Hat Enterprise Linux 8에서 InfiniBand 및 RDMA 네트워크 구성 가이드

Last Updated: 2022-01-24
Red Hat Enterprise Linux 8 InfiniBand 및 RDMA 네트워크 구성

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.
Copyright © 2022 | You need to change the HOLDER entity in the en-US/Configuring_InfiniBand_and_RDMA_networks.ent file |

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at http://creativecommons.org/licenses/by-sa/3.0/ . In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

초록

이 문서에서는 InfiniBand 및 RDMA(원격 직접 메모리 액세스)의 개념과 InfiniBand 하드웨어를 구성하는 방법을 설명합니다. 또한 이 문서에서는 InfiniBand 관련 서비스를 구성하는 방법을 설명합니다.
차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체 .................................................. 3
RED HAT 문서에 관한 피드백 제공 ................................................................. 4
1장. INFINIBAND 및 RDMA 이해 ................................................................. 5
2장. ROCE 구성 .......................................................... 6
   2.1. ROCE 프로토콜 버전 개요 .................................................. 6
   2.2. 기본 ROCE 버전 일시적으로 변경 .................................. 6
   2.3. 소프트 로빈 구성 .................................................. 7
3장. 코어 RDMA 하위 시스템 구성 .................................................. 9
   3.1. IPOIB 장치 이름 변경 .................................................. 9
   3.2. 사용자가 시스템에서 고정할 수 있는 메모리 양 증가 ................. 9
   3.3. RDMA 서비스 구성 .................................................. 10
4장. INFINIBAND 서브넷 관리자 구성 ............................................. 12
   4.1. OPNSM 서브넷 관리자 설치 ........................................ 12
   4.2. 간단한 방법을 사용하여 OPNSM 구성 .................................. 12
   4.3. OPNSM.CONF 파일을 편집하여 OPNSM 구성 ......................... 13
   4.4. 여러 OPNSM 인스턴스 구성 ........................................ 14
   4.5. 파티션 설정 생성 .................................................. 15
5장. IPOIB 구성 .......................................................... 17
   5.1. IPOIB 통신 모드 .................................................. 17
   5.2. IPOIB 하드웨어 주소 이해 ........................................ 17
   5.3. NMCLI 명령을 사용하여 IPOIB 연결 구성 ........................... 18
   5.4. NM-CONNECTION-EDITOR를 사용하여 IPOIB 연결 구성 ............ 18
6장. INFINIBAND 네트워크 테스트 ............................................. 21
   6.1. 초기 INFINIBAND RDMA 작업 테스트 .................................. 21
   6.2. PING 유필리티를 사용하여 IPOIB 테스트 ........................... 23
   6.3. IPOIB 구성 후 QPERF를 사용하여 RDMA 네트워크 테스트 ......... 23
보다 포괄적 수용을 위한 오픈 소스 용어 교체

Red Hat은 코드, 문서, 웹 속성에서 문제가 있는 용어를 교체하기 위해 최선을 다하고 있습니다. 먼저 마스터(master), 슬레이브(slave), 블랙리스트(blacklist), 화이트리스트(whitelist) 등 네 가지 용어를 교체하고 있습니다. 이러한 변경 작업은 작업 범위가 크므로 향후 여러 릴리스에 걸쳐 점차 구현할 예정입니다. 자세한 내용은 CTO Chris Wright의 메시지를 참조하십시오.
RED HAT 문서에 관한 피드백 제공

문서 개선을 위한 의견을 보내 주십시오. Red Hat이 어떻게 이를 개선할 수 있는지 알려 주십시오. 이렇게 하려면 다음을 수행합니다.

- 특정 문구에 대한 간단한 주석은 다음과 같습니다.
  1. 문서가 Multi-page HTML 형식으로 표시되는지 확인합니다. 또한 문서 오른쪽 상단에 Feedback (피드백) 버튼이 있는지 확인합니다.
  2. 마우스 커서를 사용하여 주석 처리하려는 텍스트 부분을 강조 표시합니다.
  3. 강조 표시된 텍스트 아래에 표시되는 피드백 추가 팝업을 클릭합니다.
  4. 표시된 지침을 따릅니다.

- 보다 상세하게 피드백을 제출하려면 다음과 같이 Bugzilla 티켓을 생성하십시오.
  1. Bugzilla 웹 사이트로 이동하십시오.
  2. Component로 Documentation을 선택하십시오.
  3. Description 필드에 문서 개선을 위한 제안 사항을 기입하십시오. 관련된 문서의 해당 부분 링크를 알려주십시오.
  4. Submit Bug를 클릭하십시오.
1장. INFINIBAND 및 RDMA 이해

InfiniBand는 다음 두 가지를 참조합니다.

- InfiniBand 네트워크의 물리적 링크 계층 프로토콜
- RDMA(Remote Direct Memory Access) 기술을 구현한 InfiniBand Verbs API

RDMA는 컴퓨터 운영 체계 없이 한 컴퓨터에서 다른 컴퓨터의 메모리에 액세스할 수 있도록 합니다. 이 기술은 낮은 CPU 사용률로 높은 처리량과 대기 시간이 짧은 네트워킹을 활성화합니다.

일반적인 IP 데이터 전송에서는 한 시스템의 애플리케이션이 다른 시스템의 애플리케이션에 데이터를 전송할 때 수신 측에서 다음이 수행됩니다.

1. 커널이 데이터를 수신해야 합니다.
2. 커널은 데이터가 애플리케이션에 속하는지 결정해야 합니다.
3. 커널이 애플리케이션을 활성화합니다.
4. 커널은 애플리케이션 간의 시스템 호출을 수행할 때까지 기다립니다.
5. 애플리케이션은 커널의 내부 메모리 공간에서 애플리케이션에서 제공하는 버퍼에 데이터를 복사합니다.

이 프로세스는 호스트 어댑터가 직접 메모리 액세스(DMA)를 사용하거나 두 번 이상 사용하는 경우 시스템의 주 메모리에 대부분의 네트워크 트래픽이 복사됩니다. 또한 커널은 커널과 애플리케이션 컨텍스트를 전환하기 위해 여러 컨텍스트 스위치를 실행합니다. 두 컨텍스트 스위치 모두 트래픽 속도가 높은 CPU 부하를 발생시키고 다른 작업을 느리게 할 수 있습니다.

RDMA 통신은 일반적인 IP 통신과 달리 동신 프로세스의 커널 개입을 무시합니다. 이렇게 하면 CPU 오버헤드가 줄어듭니다. RDMA 프로토콜을 사용하면 호스트 어댑터가 네트워크에서 패킷이 들어 오는 시기를 알 수 있으며, 애플리케이션은 수신해야 하며 애플리케이션의 메모리 공간에서 패킷을 저장해야 합니다. RDMA를 단순화하려면 다음 사항을 고려해야 합니다. RDMA의 콘텐츠는 애플리케이션 버퍼에 직접 배치합니다. 이 프로세스에는 별도의 API, InfiniBand Verbs API가 필요하며 RDMA를 사용하려면 이 API를 지원해야 합니다.

Red Hat Enterprise Linux 8은 InfiniBand 하드웨어와 InfiniBand Verbs API를 모두 지원합니다. 또한 Red Hat Enterprise Linux는 비InfiniBand 하드웨어에서 InfiniBand Verbs API를 사용할 수 있는 다음과 같은 기술을 지원합니다.

- iWARP(Internet Wide Area RDMA Protocol): IP 네트워크를 통해 RDMA를 구현하는 네트워크 프로토콜입니다.
- RoCE(RDMA over Converged Ethernet)를 IBoE(InfiniBand over Ethernet)라고도 합니다. 이더넷 네트워크를 통해 RDMA를 구현하는 네트워크 프로토콜입니다.

추가 리소스

- RoCE 구성
2장. ROCE 구성

이 섹션에서는 RoCE(RDMA over Converged Ethernet)에 대한 배경 정보와 기본 RoCE 버전을 변경하는 방법과 소프트웨어로 RoCE 어댑터를 구성하는 방법에 대해 설명합니다.

RoCE 하드웨어를 제공하는 Mellanox, Broadcom 및 QLogic과 같은 다양한 벤더가 있습니다.

2.1. ROCE 프로토콜 버전 개요

RoCE는 이더넷을 통해 원격 직접 메모리 액세스(RDMA)를 활성화하는 네트워크 프로토콜입니다.

다음은 다른 RoCE 버전입니다.

RoCE v1

RoCE 버전 1 프로토콜은 동일한 이더넷 브로드캐스트 도메인에서 두 호스트 간의 통신을 활성화하는 이더넷 링크 계층 프로토콜 0x8915 입니다. 기본적으로 Mellanox ConnectX-3 네트워크 어댑터를 사용하는 경우 Red Hat Enterprise Linux는 RDMA_CM(RDMA Connection Manager)에 RoCE v1을 사용합니다.

RoCE v2

RoCE 버전 2 프로토콜은 UDP over IPv4 또는 IPv6 프로토콜의 UDP 위에 존재합니다. UDP 대상 포트 번호 4791은 RoCE v2용으로 예약되어 있습니다. 기본적으로 Mellanox ConnectX-3 Pro, ConnectX-4 Lx 또는 ConnectX-5 네트워크 어댑터를 사용하는 경우 Red Hat Enterprise Linux는 RDMA_CM에 RoCE v2을 사용하지만 하드웨어는 RoCE v1 및 RoCE v2를 모두 지원합니다.

RDMA_CM은 데이터를 전송하는 클라이언트와 서버 간에 안정적인 연결을 설정합니다. RDMA_CM은 연결 설정을 위한 RDMA 전송 중립 인터페이스를 제공합니다. 통신은 특정 RDMA 장치를 사용하며 테이터 전송은 메시지 기반입니다.

중요

서버에서 RoCE v2를 클라이언트 및 RoCE v1에서 사용하는 것은 지원되지 않습니다. 이 경우 RoCE v1에서 통신하도록 서버와 클라이언트 모두를 구성합니다.

추가 리소스

- 기본 RoCE 버전 임시 변경

2.2. 기본 ROCE 버전 일시적으로 변경

서버에서 클라이언트 및 RoCE v1에서 RoCE v2 프로토콜을 사용하는 것은 지원되지 않습니다. 서버의 하드웨어가 RoCE v1만 지원하면 RoCE v1을 사용하여 서버와 통신하도록 클라이언트를 구성합니다. 이 섹션에서는 Mellanox ConnectX-5 Infiniband 장치에 mlx5_0 드라이버를 사용하는 클라이언트에 RoCE v1을 적용하는 방법에 대해 설명합니다. 이 섹션에 설명된 변경 사항은 호스트를 재부팅할 때까지 일시적인 뿐입니다.

사전 요구 사항

- 클라이언트는 기본적으로 RoCE v2 프로토콜을 사용하는 InfiniBand 장치를 사용합니다.
- 서버의 InfiniBand 장치는 RoCE v1만 지원합니다.
## 2장. ROCE 구성

### 절차

1. `/sys/kernel/config/rdma_cm/mlx5_0` 디렉토리를 만듭니다.
   
   ```
   # mkdir /sys/kernel/config/rdma_cm/mlx5_0/
   ```

2. 기본 RoCE 모드를 표시합니다. 예를 들어 포트1에 대한 모드를 표시하려면 다음을 수행합니다.
   
   ```
   # cat /sys/kernel/config/rdma_cm/mlx5_0/ports/1/default_roce_mode
   RoCE v2
   ```

3. 기본 RoCE 모드를 버전 1로 변경합니다.
   
   ```
   # echo "IB/RoCE v1" > /sys/kernel/config/rdma_cm/mlx5_0/ports/1/default_roce_mode
   ```

### 2.3. 소프트 로빈 구성

소프트 로빈은 이더넷을 통한 RDMA(Remote Direct Memory Access) 소프트웨어 구현으로 RXE라고도 합니다. RoCE HCA(호스트 채널 어댑터)가 있는 호스트에서 소프트 로빈을 사용합니다.

#### 중요

field-RoCE 기능은 기술 프리뷰로만 제공됩니다. 기술 프리뷰 기능은 Red Hat 프로덕션 서비스 수준 계약(SLA)에서 지원되지 않을 수 있으며, 기능적으로 완전하지 않을 수 있으며 Red Hat은 프로덕션에 이러한 기능을 사용하지 않는 것이 좋습니다. 이러한 프리뷰를 통해 향후 제품 기능에 모기 앱세스할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 Red Hat 고객 포털의 기술 프리뷰 기능 지원 범위를 참조하십시오.

이 섹션에서는 소프트 로빈을 구성하는 방법에 대해 설명합니다.

### 사전 요구 사항

- 이더넷 어댑터가 시스템에 설치되어 있습니다.

### 절차

1. `iproute libibverbs,libibverbs-utils` 및 `infiniband-diags` 패키지를 설치합니다.
   
   ```
   # yum install iproute libibverbs libibverbs-utils infiniband-diags
   ```

2. 시스템의 the `rdma` 링크를 표시합니다.
   
   ```
   # rdma link show
   ```

3. `rdma_rxe` 컨트롤 모듈을 로드하고 새 RXE 장치를 추가합니다.
   
   ```
   # rdma link add rxe0 type rxe netdev eth0
   ```
다음과 같습니다.

- **rxe0** 는 추가할 rdma 링크의 이름을 지정합니다.
- **RXE**는 사용할 the rdma 유형을 지정합니다.
- **netdev** 는 링크가 바인딩되는 네트워크 장치를 지정합니다.

4. 시스템에서 **all rdma** 링크의 상태를 확인합니다.

    ```
    # rdma link show
    link rxe0/1 state ACTIVE physical_state LINK_UP netdev eth0
    ```

5. 선택 사항: 시스템에서 사용 가능한 RDMA 장치를 나열합니다.

    ```
    # ibv_devices
    device node GUID
    ------ ----------------
    rxe0 505400fffed5e0fb
    ```

또는 the ibstat 용리리를 사용하여 자세한 상태를 표시합니다.

    ```
    # ibstat rxe0
    CA ‘rxe0’
    CA type: 
    Number of ports: 1
    Firmware version: 
    Hardware version: 
    Node GUID: 0x505400fffed5e0fb
    System image GUID: 0x0000000000000000
    Port 1:
    State: Active
    Physical state: LinkUp
    Rate: 100
    Base lid: 0
    LMC: 0
    SM lid: 0
    Capability mask: 0x00890000
    Port GUID: 0x505400fffed5e0fb
    Link layer: Ethernet
    ```
3장. 코어 RDMA 하위 시스템 구성

이 섹션에서는 rdma 서비스를 구성하고 사용자가 시스템에서 고정할 수 있는 메모리 양을 늘리는 방법을 설명합니다.

3.1. IPOIB 장치 이름 변경
기본적으로 커널은 IP를 IP over InfiniBand(IPoIB) 장치(예: ib0, ib1 등)로 지정합니다. 충돌을 방지하기 위해 Red Hat은 udev 장치 관리자에 규칙을 생성하여 mlx4_ib0과 같은 지속적이고 의미 있는 이름을 만들 것을 권장합니다.

사전 요구 사항
- InfiniBand 장치가 호스트에 설치되어 있습니다.

절차
1. 장치의 하드웨어 주소를 표시합니다. 예를 들어 이름이 ib0 인 장치의 주소를 표시하려면 다음을 입력합니다.

```bash
# ip link show ib0
8: ib0: >BROADCAST,MULTICAST,UP,LOWER_UP< mtu 65520 qdisc pfifo_fast state UP
mode DEFAULT qdisc 256
link/infiniband 80:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00 00:02:c9:03:00:31:78:f2
brd 00:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff:ff
```

예제에서 공백 표시된 주소의 마지막 8바이트는 다음 단계에서 udev 규칙을 만드는 데 필요합니다.

2. `/etc/udev/rules.d/70-persistent-ipoib.rules` 파일을 편집하고 ACTION 규칙을 추가합니다. 예를 들어 00:02:c9:03:00:31:78:f2 하드웨어 주소의 장치 이름은 mlx4_ib0 에 추가하는 규칙을 구성하려면 다음 행을 추가합니다.

```bash
ACTION=="add", SUBSYSTEM=="net", DRIVERS=="?*", ATTR{type}=="32",
ATTR{address}=="?*:00:02:c9:03:00:31:78:f2", NAME="mlx4_ib0"
```

3. 호스트를 재부팅합니다.

```
# reboot
```

추가 리소스
- udev(7) 도움말 페이지
- IPoIB 하드웨어 주소 이해

3.2. 사용자가 시스템에서 고정할 수 있는 메모리 양 증가
RDMA 작업을 수행하려면 물리적 메모리를 고정해야 합니다. 즉, 커널이 메모리를 스왑 공간에 쓸 수 없습니 다. 사용자가 너무 많은 메모리를 고정하면 시스템이 메모리를 부족할 수 있으며 커널은 더 많은 메모리를 확보하기 위해 프로세스를 종료합니다. 이러한 이유로 메모리 고정은 권한 있는 작업입니다.
루트가 아닌 사용자가 대규모 RDMA 애플리케이션을 실행하는 경우 이러한 사용자가 시스템에 고정할 수 있는 메모리 양을 늘려야 할 수 있습니다. 이 섹션에서는 rdma 그룹에 대해 무제한 메모리를 구성하는 방법에 대해 설명합니다.

절차

- root 사용자로 다음 콘텐츠를 사용하여 `/etc/security/limits.d/rdma.conf` 파일을 만듭니다.

  ```bash
  @rdma soft memlock unlimited @rdma hard memlock unlimited
  ```

결중 단계

1. `/etc/security/limits.d/ rdma.conf` 파일을 편집한 후 rdma 그룹의 멤버로 로그인합니다. 사용자가 로그인할 때 Red Hat Enterprise Linux는 업데이트된 `ulimit` 설정을 적용합니다.

2. 제한을 표시하려면 `ulimit -l` 명령을 사용합니다.

   ```bash
   $ ulimit -l
   unlimited
   ```

가능한 리소스

- `limits.conf(5)` 메뉴얼 페이지

3.3. RDMA 서비스 구성

The `rdma` 서비스는 커널의 RDMA 스택을 관리합니다. Red Hat Enterprise Linux가 InfiniBand, iWARP 또는 RoCE 장치를 감지하는 경우 `udev` 장치 관리자는 `systemd` 에 rdma 서비스를 시작하도록 지시합니다.

절차

1. `/etc/rdma/rdma.conf` 파일을 편집하고 활성화하려는 모듈의 변수를 `yes`로 설정합니다. 다음은 Red Hat Enterprise Linux 8의 기본 `/etc/rdma/rdma.conf` 입니다.

   ```plaintext
   # Load IPoIB
   IPOIB_LOAD=yes
   # Load SRP (SCSI Remote Protocol initiator support) module
   SRP_LOAD=yes
   # Load SRPT (SCSI Remote Protocol target support) module
   SRPT_LOAD=yes
   # Load iSER (iSCSI over RDMA initiator support) module
   ISER_LOAD=yes
   # Load iSERT (iSCSI over RDMA target support) module
   ISERT_LOAD=yes
   # Load RDS (Reliable Datagram Service) network protocol
   RDS_LOAD=no
   # Load NFSoRDMA client transport module
   XPRTRDMA_LOAD=yes
   # Load NFSoRDMA server transport module
   SVCRDMA_LOAD=no
   # Load Tech Preview device driver modules
   TECH_PREVIEW_LOAD=no
   ```
2. **rdma** 서비스를 다시 시작하십시오.

```bash
# systemctl restart rdma
```
4장. INFINIBAND 서브넷 관리자 구성

모든 InfiniBand 네트워크에는 네트워크가 작동하려면 서브넷 관리자가 실행되어야 합니다. 두 시스템 이 연결된 스위치 없이 직접 연결된 경우에도 마찬가지입니다.

서브넷 관리자가 두 개 이상 있을 수 있습니다. 이 경우 마스터 역할을 하고 다른 서브넷 관리자가 마스터 서브넷 관리자가 실패하는 경우 인수하는 슬레이브 역할을 합니다.

대부분의 InfiniBand 스위치에는 서브넷 관리자가 포함되어 있습니다. 그러나 보다 최신 서브넷 관리자가 필요하거나 더 많은 제어가 필요한 경우 Red Hat Enterprise Linux에서 제공하는 OpenSM 서브넷 관리자를 사용하십시오.

4.1. OPENSM 서브넷 관리자 설치

이 섹션에서는 OpenSM 서브넷 관리자를 설치하는 방법에 대해 설명합니다.

절차

1. opensm 패키지를 설치합니다.

```
# yum install opensm
```

2. 기본 설치가 완경과 일치하지 않는 경우 OpenSM을 구성합니다.
   하나의 InfiniBand 포트만 설치되어 있으며 호스트는 마스터 서브넷 관리자 역할을 하며 사용자 지정 변경이 필요하지 않습니다. 기본 구성은 수정 없이 작동합니다.

3. opensm 서비스를 활성화하고 시작합니다.

```
# systemctl enable --now opensm
```

추가 리소스

- opensm(8) 매뉴얼 페이지

4.2. 간단한 방법을 사용하여 OPENSM 구성

이 섹션에서는 사용자 지정 설정이 필요하지 않은 경우 OpenSM을 구성하는 방법을 설명합니다.

사전 요구 사항

- 하나 이상의 InfiniBand 포트가 서버에 설치되어 있습니다.

절차

1. the ibstat 유ти리티를 사용하여 포트의 GUID를 가져옵니다.

```
# ibstat -d device_name
```

CA 'mlx4_0'
CA type: MT4099
Number of ports: 2
Firmware version: 2.42.5000
Hardware version: 1
Node GUID: 0xf4521403007be130
System image GUID: 0xf4521403007be133
Port 1:
  State: Active
  Physical state: LinkUp
  Rate: 56
  Base lid: 3
  LMC: 0
  SM lid: 1
  Capability mask: 0x02594868
  Port GUID: 0xf4521403007be131
  Link layer: InfiniBand
Port 2:
  State: Down
  Physical state: Disabled
  Rate: 10
  Base lid: 0
  LMC: 0
  SM lid: 0
  Capability mask: 0x04010000
  Port GUID: 0xf65214fffe7be132
  Link layer: Ethernet

참고

일부 InfiniBand 어댑터는 노드, 시스템 및 포트에 동일한 GUID를 사용합니다.

2. `/etc/sysconfig/opensm` 파일을 편집하고 GUID S 매개변수에서 GUID를 설정합니다.

```
GUIDS="GUID_1 GUID_2"
```

3. 필요한 경우 서브넷에서 여러 서브넷 관리자를 사용할 수 있는 경우 PRIORITY 매개 변수를 설정합니다. 예를 들어 다음과 같습니다.

```
PRIORITY=15
```

추가 리소스

- `/etc/sysconfig/opensm`

4.3. OPENS.M.CONF 파일을 편집하여 OPENS M 구성

이 섹션에서는 `/etc/rdma/opensm.conf` 파일을 편집하여 OpenSM을 구성하는 방법을 설명합니다. InfiniBand 포트를 하나만 사용할 수 있는 경우 이 방법을 사용하여 OpenSM 구성을 사용자 지정합니다.

사전 요구 사항

- 하나의 InfiniBand 포트만 서버에 설치됩니다.

절차

1. `/etc/rdma/opensm.conf` 파일을 편집하고 환경에 맞게 설정을 사용자 지정합니다.
**opensm** 패키지를 업데이트한 후, **yum** 유튜리티는 `/etc/rdma/opensm.conf` 를 재정의하고 새 OpenSM 구성 파일 `/etc/rdma/opensm.conf.rpmnew` 를 만듭니다. 따라서 이전 및 새 파일을 비교하여 변경 사항을 식별하고 `opensm.conf` 파일에 수동으로 통합할 수 있습니 다.

2. `opensm` 서비스를 다시 시작하십시오.

```
# systemctl restart opensm
```

4.4. 여러 OPENSM 인스턴스 구성

이 섹션에서는 OpenSM의 여러 인스턴스를 설정하는 방법에 대해 설명합니다.

사전 요구 사항

- 하나 이상의 InfiniBand 포트가 서버에 설치되어 있습니다.

절차

1. 선택적으로 `/etc/rdma/opensm.conf` 파일을 `/etc/rdma/opensm.conf.orig` 파일에 복사합니다.

```
# cp /etc/rdma/opensm.conf /etc/rdma/opensm.conf.orig
```

업데이트된 `opensm` 패키지를 설치할 때, **yum** 유튜리티는 `/etc/rdma/opensm.conf` 를 재정의합니다. 이 단계에서 복사본을 만든 상태에서 이전 및 새 파일을 비교하여 변경 사항을 식별하고 인스턴스별 `opensm.conf` 파일에 수동으로 통합할 수 있습니다.

2. `/etc/rdma/opensm.conf` 파일의 사본을 생성합니다.

```
# cp /etc/rdma/opensm.conf /etc/rdma/opensm.conf.1
```

생성한 각 인스턴스에 대해 고유한 연속 숫자를 구성 파일의 복사본에 추가합니다.

`opensm` 패키지를 업데이트한 후, **yum** 유튜리티는 새 OpenSM 구성 파일을 `/etc/rdma/opensm.conf.rpmnew`로 저장합니다. 이 파일을 사용자 지정된 `/etc/rdma/opensm.conf.*` 파일과 비교하고 변경 사항을 수동으로 통합할 수 있습니다.

3. 이전 단계에서 만든 복사본을 편집하고 환경에 맞게 인스턴스의 설정을 사용자 지정합니다. 예를 들어 `guid`, `subnet_prefix` 및 `logdir` 매개 변수를 설정합니다.

4. 선택적으로 이 서브넷의 고유한 이름을 사용하여 `partition.conf` 파일을 생성하고, `opensm.conf` 파일의 해당 복사본에 있는 `partition_config_file` 매개 변수에서 해당 파일을 참조합니다.

5. 생성한 각 인스턴스에 대해 이전 단계를 반복합니다.

6. `opensm` 서비스를 시작합니다.

```
# systemctl start opensm
```

`opensm` 서비스는 `/etc/rdma/` 디렉터리에 있는 각 `opensm.conf.*` 파일에 대한 고유한 인스턴스를 자동으로 시작합니다. 여러 `opensm.conf.*` 파일이 있는 경우 서비스는 `/etc/sysconfig/opensm` 파일의 설정을 물론 기본 `/etc/rdma/opensm.conf` 파일에서 설정을 무시합니다.

추가 리소스
업데이트된 opensm 패키지를 설치할 때 yum 유틸리티는 새 OpenSM 구성 파일을 /etc/rdma/opensm.conf.rpmnew로 저장합니다. 이 파일을 사용자 지정 /etc/rdma/opensm.conf로 파일과 비교하고 변경 사항을 수동으로 통합합니다.

4.5. 파티션 설정 생성

이 섹션에서는 OpenSM에 대한 InfiniBand 파티션 구성을 생성하는 방법에 대해 설명합니다. 파티션을 사용하면 관리자가 이더넷 VLAN과 유사한 InfiniBand에서 서브넷을 만들 수 있습니다.

주요

40Gbps와 같은 특정 속도로 파티션을 정의한 경우 이 파티션 내의 모든 호스트에서 최소한 이 속도를 지원해야 합니다. 호스트가 속도 요구 사항을 충족하지 않으면 파티션이 참여할 수 없습니다. 따라서 파티션의 속도를 파티션에 가입할 수 있는 권한이 있는 모든 호스트에서 지원하는 최저 속도로 설정합니다.

사전 요구 사항

- 하나 이상의 InfiniBand 포트가 서버에 설치되어 있습니다.

절차

1. /etc/rdma/partitions.conf 파일을 편집하고 파티션을 구성합니다.

참고

모든 패브릭에는 0x7fff 기본 파티션이 포함되어야 하며 모든 스위치와 모든 호스트는 해당 패브릭에 속해야 합니다.

예를 들어 다음 콘텐츠를 파일에 추가하여 0x7fff 기본 파티션을 10Gbps의 속도 및 40Gbps의 속도로 0x7ff 기본 파티션을 만듭니다.

```bash
# For reference:
# IPv4 IANA reserved multicast addresses:
# http://www.iana.org/assignments/multicast-addresses/multicast-addresses.txt
# IPv6 IANA reserved multicast addresses:
# http://www.iana.org/assignments/ipv6-multicast-addresses/ipv6-multicast-addresses.xml
#
# mtu =
#   1 = 256
#   2 = 512
#   3 = 1024
#   4 = 2048
#   5 = 4096
#
# rate =
#   2  =   2.5 GBit/s
#   3  =  10   GBit/s
#   4  =  30   GBit/s
#   5  =   5   GBit/s
#   6  =  20   GBit/s
#   7  =  40   GBit/s
#   8  =  60   GBit/s
#   9  =  80   GBit/s
```
Default=0x7fff, rate=3, mtu=4, scope=2, defmember=full:
  ALL, ALL_SWITCHES=full;
Default=0x7fff, ipoib, rate=3, mtu=4, scope=2:
  mgid=ff12:401b::ffff:ffff  # IPv4 Broadcast address
  mgid=ff12:401b::1          # IPv4 All Hosts group
  mgid=ff12:401b::2          # IPv4 All Routers group
  mgid=ff12:401b::16         # IPv4 IGMP group
  mgid=ff12:401b::fb         # IPv4 mDNS group
  mgid=ff12:401b::fc         # IPv4 Multicast Link Local Name Resolution group
  mgid=ff12:401b::101        # IPv4 NTP group
  mgid=ff12:401b::202        # IPv4 Sun RPC
  mgid=ff12:601b::1          # IPv6 All Hosts group
  mgid=ff12:601b::2          # IPv6 All Routers group
  mgid=ff12:601b::16         # IPv6 MLDv2-capable Routers group
  mgid=ff12:601b::fb         # IPv6 mDNS group
  mgid=ff12:601b::101        # IPv6 NTP group
  mgid=ff12:601b::202        # IPv6 Sun RPC group
  mgid=ff12:601b::1:3        # IPv6 Multicast Link Local Name Resolution group
ALL=full, ALL_SWITCHES=full;

ib0_2=0x0002, rate=7, mtu=4, scope=2, defmember=full:
  ALL, ALL_SWITCHES=full;
ib0_2=0x0002, ipoib, rate=7, mtu=4, scope=2:
  mgid=ff12:401b::ffff:ffff  # IPv4 Broadcast address
  mgid=ff12:401b::1          # IPv4 All Hosts group
  mgid=ff12:401b::2          # IPv4 All Routers group
  mgid=ff12:401b::16         # IPv4 IGMP group
  mgid=ff12:401b::fb         # IPv4 mDNS group
  mgid=ff12:401b::fc         # IPv4 Multicast Link Local Name Resolution group
  mgid=ff12:401b::101        # IPv4 NTP group
  mgid=ff12:401b::202        # IPv4 Sun RPC
  mgid=ff12:601b::1          # IPv6 All Hosts group
  mgid=ff12:601b::2          # IPv6 All Routers group
  mgid=ff12:601b::16         # IPv6 MLDv2-capable Routers group
  mgid=ff12:601b::fb         # IPv6 mDNS group
  mgid=ff12:601b::101        # IPv6 NTP group
  mgid=ff12:601b::202        # IPv6 Sun RPC group
  mgid=ff12:601b::1:3        # IPv6 Multicast Link Local Name Resolution group
ALL=full, ALL_SWITCHES=full;
5장. IPOIB 구성

기본적으로 InfiniBand는 통신에 인터넷 프로토콜(IP)을 사용하지 않습니다. 그러나 IP over InfiniBand(IpIB)는 InfiniBand RDMA(Remote Direct Memory Access) 네트워크 위에 IP 네트워크 애플레이션 계층을 제공합니다. 이렇게 하면 수정되지 않은 기존 애플리케이션에서 InfiniBand 네트워크를 통해 데이터를 전송할 수 있지만 성능은 애플리케이션이 기본적으로 RDMA를 사용하는 것보다 낮습니다.

참고
IWARP(Internet Wide Area RDMA Protocol) 및 RoCE 네트워크는 이미 IP 기반입니다. 따라서 IWARP 또는 RoCE 장치 상단에 IPoIB 장치를 생성할 수 없습니다.

RHEL 8에서 ConnectX-4 이상에서 시작되는 Mellanox 장치는 기본적으로 Enhanced IPoIB 모드를 사용합니다(데이터그램만 해당). 연결 모드는 이러한 장치에서 지원되지 않습니다.

RHEL 8에서 ConnectX-4 이상에서 시작되는 Mellanox 장치는 기본적으로 Enhanced IPoIB 모드를 사용합니다(데이터그램만 해당). 연결 모드는 이러한 장치에서 지원되지 않습니다.

5.1. IPOIB 통신 모드

`Datagram` 또는 `Connected` 모드에서 IPoIB 장치를 구성할 수 있습니다. 차이점은 IPoIB 계층이 통신의 다른 끝에서 시스템을 사용하여 열려고 하는 큐 쌍 유형입니다.

- `Datagram` 모드에서 시스템은 불안정하고 연결이 끊긴 대기열 쌍을 얻습니다. 이 모드는 InfiniBand 링크 계층의 MTU(최대 전송 단위)보다 큰 페킷을 지원하지 않습니다. IPoIB 계층은 전송 중인 IP 페킷 상단에 4바이트 IPoIB 레이어를 추가합니다. 결과적으로 IPoIB MTU는 InfiniBand 링크 계층 MTU보다 4바이트 작아야 합니다. 2048은 일반적인 InfiniBand 링크 계층 MTU이므로 데이터그램 모드의 일반적인 IPoIB 장치 MTU는 2044입니다.

- 연결 모드에서 시스템은 신뢰할 수 있고 연결된 대기열 쌍을 얻습니다. 이 모드에서는 InfiniBand 링크 계층 MTU보다 큰 메시지를 허용하고 호스트 어댑터는 페킷 분할을 처리하고 다시 집계합니다. 결과적으로 연결된 모드에서 InfiniBand 어댑터가 보낼 수 있는 IPoIB 메시지 크기에 제한된 크기를 제한합니다. 그러나 크기 필드와 TCP/IP 헤더로 인해 IP 페킷이 제한됩니다. 이러한 이유로 연결 모드의 IPoIB MTU는 최대 65520 바이트입니다.

연결 모드는 성능이 향상되지만 커널 메모리를 더 많이 사용합니다.

시스템이 연결 모드를 사용하도록 구성된 경우 InfiniBand 스위치와 Fabric은 연결된 모드에서 멀티캐스트 트래픽을 전달할 수 없기 때문에 여전히 멀티캐스트 트래픽을 `Datagram` 모드로 보냅니다. 또한 연결 모드에서 구성되지 않은 호스트와 통신할 때 시스템은 `Datagram` 모드로 대체됩니다.

인터넷에서 최대 MTU까지 멀티캐스트 데이터를 전송하는 애플리케이션을 실행하는 동안 `Datagram` 모드에서 인터페이스를 구성하거나 페킷 전송 크기를 데이터그램 크기만큼 제한하도록 애플리케이션을 구성해야 합니다.

5.2. IPOIB 하드웨어 주소 이해

IPoIB 장치에는 다음 부분으로 구성된 20 바이트 하드웨어 주소가 있습니다.

- 처음 4바이트는 플래그와 대기열 쌍 번호입니다.
- 다음 8바이트는 서브넷 접두사입니다. 기본 서브넷 접두사는 `0x:80:00:00:00:00:00:00` 입니다. 장치가 서브넷 관리자에 연결되면 장치는 서브넷 관리자에 구성된 항목과 일치하도록 이 접두사를 변경합니다.
- 마지막 8바이트는 IPoIB 장치가 연결된 InfiniBand 포트의 GUID(Globally Unique Identifier)입니다.
참고
처음 12바이트는 변경될 수 있으므로 udev 장치 관리자 규칙에 사용하지 마십시오.

추가 리소스
- IPoIB 장치

5.3. NMCLI 명령을 사용하여 IPOIB 연결 구성
다음 절차에서는 nmcli 명령을 사용하여 IPoIB 연결을 구성하는 방법을 설명합니다.

사전 요구 사항
- InfiniBand 장치가 서버에 설치되고 해당 커널 모듈이 로드됩니다.

절차
1. InfiniBand 연결을 만듭니다. 예를 들어 연결 전송 모드에서 mlx4_ib0 인터페이스와 65520 바이트의 최대 MTU를 사용하는 연결을 생성하려면 다음을 입력합니다.

```
# nmcli connection add type infiniband con-name mlx4_ib0 ifname mlx4_ib0 transport-mode Connected mtu 65520
```

2. 선택사항: P_Key 인터페이스를 설정합니다. 예를 들어 0x8002를 mlx4_ib0 연결의 P_Key 인터페이스로 설정하려면 다음을 입력합니다.

```
# nmcli connection modify mlx4_ib0 infiniband.p-key 0x8002
```

3. IPv4 설정을 구성합니다. 예를 들어 mlx4_ib0 연결의 정적 IPv4 주소, 네트워크 마스크, 기본 게이트웨이 및 DNS 서버를 설정하려면 다음을 입력합니다.

```
# nmcli connection modify mlx4_ib0 ipv4.addresses '192.0.2.1/24'
# nmcli connection modify mlx4_ib0 ipv4.gateway '192.0.2.254'
# nmcli connection modify mlx4_ib0 ipv4.dns '192.0.2.253'
# nmcli connection modify mlx4_ib0 ipv4.method manual
```

4. IPv6 설정을 구성합니다. 예를 들어 mlx4_ib0 연결의 정적 IPv6 주소, 네트워크 마스크, 기본 게이트웨이 및 DNS 서버를 설정하려면 다음을 입력합니다.

```
# nmcli connection modify mlx4_ib0 ipv6.addresses '2001:db8:1::1/32'
# nmcli connection modify mlx4_ib0 ipv6.gateway '2001:db8:1::ffe'
# nmcli connection modify mlx4_ib0 ipv6.dns '2001:db8:1::fff'f'
# nmcli connection modify mlx4_ib0 ipv6.method manual
```

5. 연결을 활성화합니다. 예를 들어 mlx4_ib0 연결을 활성화하려면 다음을 수행합니다.

```
# nmcli connection up mlx4_ib0
```

5.4. NM-CONNECTION-EDITOR를 사용하여 IPOIB 연결 구성
다음 절차에서는 nm-connection-editor 애플리케이션을 사용하여 IPoIB 연결을 구성하는 방법을 설명합니다.

사전 요구 사항

- InfiniBand 장치가 서버에 설치되고 해당 커널 모듈이 로드됩니다.
- nm-connection-editor 패키지가 설치되어 있습니다.

절차

1. 터미널을 열고 다음을 입력합니다.

```
$ nm-connection-editor
```

2. 버튼을 클릭하여 새 연결을 추가합니다.

3. InfiniBand 연결 유형을 선택하고 Create(만들기)를 클릭합니다.

4. InfiniBand 탭에서 다음을 수행합니다.
   a. 선택적으로 연결 이름을 변경합니다.
   b. 전송 모드를 선택합니다.
   c. 장치를 선택합니다.
   d. 선택 사항: MTU를 설정합니다.

5. IPv4 Settings(IPv4 설정) 탭에서 IPv4 설정을 구성합니다. 예를 들어 정적 IPv4 주소, 네트워크 마스크, 기본 게이트웨이 및 DNS 서버를 설정합니다.

<table>
<thead>
<tr>
<th>Connection name:</th>
<th>mlx4_ib0</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>InfiniBand</td>
</tr>
<tr>
<td>Method:</td>
<td>Manual</td>
</tr>
<tr>
<td>Addresses</td>
<td></td>
</tr>
<tr>
<td>Address</td>
<td>Prefix</td>
</tr>
<tr>
<td>2001:db8::1</td>
<td>32</td>
</tr>
</tbody>
</table>

DNS servers: 2001:db8::fffd

7. **Save**(저장)을 클릭하여 팀 연결을 저장합니다.

8. **nm-connection-editor** 닫기.

9. 선택사항: **P_Key** 인터페이스를 설정합니다. **nm-connection-editor**에서 설정을 사용할 수 없으므로 명령줄에서 이 매개변수를 설정해야 합니다.
   예를 들어 0x8002를 mlx4_ib0 연결의 **P_Key** 인터페이스로 설정하려면 다음을 입력합니다.

```
# nmcli connection modify mlx4_ib0 infiniband.p-key 0x8002
```
6장. INFINIBAND 네트워크 테스트

이 섹션에서는 InfiniBand 네트워크를 테스트하는 방법을 설명합니다.

6.1. 초기 INFINIBAND RDMA 작업 테스트

이 섹션에서는 InfiniBand RDMA(원격 직접 메모리 액세스) 작업을 테스트하는 방법을 설명합니다.

참고

이 섹션은 InfiniBand 장치에만 적용됩니다. IP 기반 iWARP 또는 RoCE/IBoE 장치를 사용하는 경우 다음을 참조하십시오.

- ping 유틸리티를 사용하여 IPoIB 테스트
- IPoIB를 구성한 후 qperf를 사용하여 RDMA 네트워크 테스트

사전 요구 사항

- RDMA가 구성되었습니 다.
- libibverbs-utils 및 infiniband-diags 패키지가 설치됩니다.

절차

1. 사용 가능한 InfiniBand 장치를 나열합니다.

```
# ibv_devices

device       node GUID
------       ---------------
mlx4_0       0002c903003178f0
mlx4_1       f4521403007bcba0
```

2. 특정 InfiniBand 장치의 정보를 표시합니다. 예를 들어 mlx4_1 장치의 정보를 표시하려면 다음을 입력합니다.

```
# ibv_devinfo -d mlx4_1

hca_id: mlx4_1
transport: InfiniBand (0)
fw_ver: 2.30.8000
node_guid: f452:1403:007b:cba0
sys_image_guid: f452:1403:007b:cba3
vendor_id: 0x02c9
vendor_part_id: 4099
hw_ver: 0x0
board_id: MT_1090120019
phys_port_cnt: 2
port: 1
state: PORT_ACTIVE (4)
max_mtu: 4096 (5)
active_mtu: 2048 (4)
sm_lid: 2
```
3. InfiniBand 장치의 기본 상태를 표시합니다. 예를 들어 mlx4_1 장치의 상태를 표시하려면 다음을 입력합니다.

```
# ibstat mlx4_1
```

CA 'mlx4_1'
CA type: MT4099
Number of ports: 2
Firmware version: 2.30.8000
Hardware version: 0
Node GUID: 0xf4521403007bcba0
System image GUID: 0xf4521403007bcba3
Port 1:
  State: Active
  Physical state: LinkUp
  Rate: 56
  Base lid: 2
  LMC: 1
  SM lid: 2
  Capability mask: 0x0251486a
  Port GUID: 0xf4521403007bcba1
  Link layer: InfiniBand
Port 2:
  State: Active
  Physical state: LinkUp
  Rate: 40
  Base lid: 0
  LMC: 0
  SM lid: 0
  Capability mask: 0x04010000
  Port GUID: 0xf65214fffe7bcba2
  Link layer: Ethernet

4. InfiniBand를 사용하여 클라이언트에서 서버로 ping하는 데 the ibping 유틸리티를 사용합니다.

a. 서버 역할을 하는 호스트에서 서버 모드에서 시작합니다.

```
# ibping -S -C mlx4_1 -P 1
```

이 명령은 다음 매개변수를 사용합니다.

- **-S**: 서버 모드를 활성화합니다.
6장. INFINIBAND 네트워크 테스트

- **-C InfiniBand_CA_name**: 사용할 CA 이름을 설정합니다.
- **-p port_number**: InfiniBand에서 여러 포트를 제공하는 경우 사용할 포트 번호를 설정합니다.

b. 클라이언트 역할을 하는 호스트에서 다음과 같이 ibping을 사용합니다.

```
# ibping -c 50 -C mlx4_0 -P 1 -L 2
```

- **-c 번호**: 이러한 패킷 수를 서버로 전송합니다.
- **-C InfiniBand_CA_name**: 사용할 CA 이름을 설정합니다.
- **-p port_number**: InfiniBand에서 여러 포트를 제공하는 경우 사용할 포트 번호를 설정합니다.
- **-L port_LID**: 사용할 로컬 식별자(LID)를 설정합니다.

추가 리소스

- ibping(8) 매뉴얼 페이지

**6.2. PING 유틸리티를 사용하여 IPOIB 테스트**

IPOIB를 구성한 후 ping 유틸리티를 사용하여 ICMP 패킷을 보내 IPOIB 연결을 테스트합니다.

사전 요구 사항

- 두 개의 RDMA 호스트는 RDMA 포트와 동일한 InfiniBand 패브릭에 연결됩니다.
- 두 호스트의 IPOIB 인터페이스는 동일한 서브넷 내의 IP 주소로 구성됩니다.

절차

- ping 유틸리티를 사용하여 ICMP 패킷을 원격 호스트의 InfiniBand 어댑터로 전송합니다.

```
# ping -c5 192.0.2.1
```

이 명령은 IP 주소 192.0.2.1 로 5개의 ICMP 패킷을 전송합니다.

**6.3. IPOIB 구성 후 QPERF를 사용하여 RDMA 네트워크 테스트**

이 절차에서는 InfiniBand 어댑터 구성의 표시하고 qperf 유틸리티를 사용하여 두 호스트 간의 대역폭과 대기 시간을 측정하는 방법을 설명합니다.

사전 요구 사항

- qperf 패키지는 두 호스트 모두에 설치됩니다.
- IPOIB는 두 호스트 모두에 구성됩니다.

절차

1. 서비스 역할을 할 옵션 없이 호스트 중 하나에서 qperf를 시작합니다.
2. 클라이언트에서 다음 명령을 사용합니다. 명령은 클라이언트의 mlx4_0 호스트 채널 어댑터의 포트 1을 사용하여 서버의 InfiniBand 어댑터에 할당된 IP 주소 192.0.2.1에 연결합니다.

a. 구성표시하려면 다음을 입력합니다.

```
# qperf -v -i mlx4_0:1 192.0.2.1 conf
```

```
conf:
loc_node   = rdma-dev-01.lab.bos.redhat.com
loc_cpu    = 12 Cores: Mixed CPUs
loc_os     = Linux 4.18.0-187.el8.x86_64
loc_qperf  = 0.4.11
rem_node   = rdma-dev-00.lab.bos.redhat.com
rem_cpu    = 12 Cores: Mixed CPUs
rem_os     = Linux 4.18.0-187.el8.x86_64
rem_qperf  = 0.4.11
```

b. 양방향 대역폭을 스트리밍하는 RC(Reliable Connection)를 표시하려면 다음을 입력합니다.

```
# qperf -v -i mlx4_0:1 192.0.2.1 rc_bi_bw
```

```
rc_bi_bw:
bw        = 10.7 GB/sec
msg_rate  = 163 K/sec
loc_id    = mlx4_0
rem_id    = mlx4_0:1
loc_cpus_used = 65 % cpus
rem_cpus_used = 62 % cpus
```

c. RC 스트리밍을 일방향 대역폭으로 표시하려면 다음을 입력합니다.

```
# qperf -v -i mlx4_0:1 192.0.2.1 rc_bw
```

```
rc_bw:
bw        = 6.19 GB/sec
msg_rate  = 94.4 K/sec
loc_id    = mlx4_0
rem_id    = mlx4_0:1
send_cost = 63.5 ms/GB
recv_cost = 63 ms/GB
send_cpus_used = 39.5 % cpus
recv_cpus_used = 39 % cpus
```

추가 리소스

- qperf(1) 메뉴얼 페이지