Openstack CVIM에서 SRIOV 네트워크로 DNS VNF 구축 - Prime Network Registrar(DNS)의 컨피그레이션 예

목차

소개

이 문서에서는 SR-IOV 및 Active-Backup 결합을 사용하여 OpenStack CVIM(Cisco Virtualized Infrastructure Manager)에서 CPNR을 단계적으로 구축하는 방법을 설명합니다.

사전 요구 사항

요구 사항

다음 주제에 대한 지식을 보유하고 있으면 유용합니다.

- OpenStack 및 SR-IOV(Single Root Input/Output Virtualization) 개념에 익숙함
- Cisco VIM(Virtual Interface Manager) 및 Cisco Elastic Services Controller와 Linux 명령 및 네트워킹에 대한 실무 지식

사용되는 구성 요소

이 문서는 특정 소프트웨어 및 하드웨어 버전으로 한정되지 않습니다.

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다. 이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다. 현재 네트워크가 작동 중인 경우 모든 명령의 잠재적인 영향을 미리 숙지하시기 바랍니다

배경 정보

현재 네트워킹 환경에서 VNF(Virtual Network Function)는 민첩하고 확장 가능하며 효율적인 네트워크 서비스를 활성화하는 데 중요한 역할을 합니다. 고성능 네트워크 연결이 필요한 VNF의 경우 SR-IOV가 일반적으로 사용되는 기술입니다. SR-IOV를 통해 VNF는 하이퍼바이저 가상 스위치를 우회하고 물리적 NIC(Network Interface Controller) 리소스에 직접 액세스할 수 있으므로 레이턴시를 줄이고 처리량을 높일 수 있습니다.

설정

구축을 계속하기 전에 이러한 전제 조건을 충족하는지 확인하십시오.

1. 하드웨어 요구 사항

- SR-IOV 지원 NIC:
  - BIOS/UEFI(Unified Extensible Firmware Interface)에서 SR-IOV가 활성화된 SR-IOV 지원 물리적 NIC 2개 이상
  - 예: sriov0이 NUMA(Non-Uniform Memory Access) 노드 0에 매핑되고 sriov1이 NUMA 노드 1에 매핑됩니다.

- NUMA 인식 호스트:
  - 컴퓨팅 노드는 NUMA 아키텍처를 지원해야 합니다.
  - NUMA 지원은 호스트 BIOS/UEFI에서 활성화해야 합니다.

2. Intel NIC 카드 식별

Intel XL710 및 E810CQDA2 NIC 카드는 일반적으로 고성능 SR-IOV 네트워킹에 사용됩니다. 호스트에서 NIC 카드 모델을 확인하려면 다음 단계를 참조하십시오.

1단계. lspci 명령 사용

네트워크 컨트롤러와 관련된 PCI(Peripheral Component Interconnect) 디바이스를 나열하려면 이 명령을 실행합니다.

```
lspci | grep -i ethernet
```

출력 예:

```
81:00.0 Ethernet controller: Intel Corporation Ethernet Controller XL710 for 40GbE QSFP+ (rev 02)
82:00.0 Ethernet controller: Intel Corporation Ethernet Controller E810-C for QSFP (rev 03)
```

2단계. XL710 확인

NIC가 Intel XL710인 경우, 출력에서 이더넷 컨트롤러 XL710을 볼 수 있습니다.

3단계. E810CQDA2 확인

NIC가 Intel E810CQDA2인 경우 이더넷 컨트롤러 E810-Cin이 출력에 표시됩니다.

4단계. 드라이버 지원 확인

사용 중인 NIC 드라이버를 확인하려면 다음을 실행합니다.

```
ethtool -i
```

XL710의 출력 예:

```
driver: i40e
version: 2.13.10
```

E810CQDA2의 출력 예:

```
driver: ice
version: 1.7.12
```

드라이버 버전이 OpenStack 및 Linux 배포판의 호환성 매트릭스와 일치하는지 확인합니다.

3. BIOS/UEFI 컨피그레이션

- SR-IOV 활성화:

서버 BIOS/UEFI에서 SR-IOV가 활성화되어 있는지 확인합니다.

- VT-d(Directed I/O)/AMD-Vi를 위한 가상화 기술 활성화:

Intel VT-d 또는 AMD-Vi는 PCI 패스스루 및 SR-IOV 기능을 활성화해야 합니다.

4. OpenStack 설정

- 코어 OpenStack 서비스:

Nova, Neutron, Glance, Keystone 등의 OpenStack 서비스가 설치 및 구성되어 있는지 확인합니다.

- 중성자 구성:

Neutron은 오케스트레이션/관리 네트워크용 OVS(Openvswitch)와 애플리케이션/서비스 네트워크용 SR-IOV를 모두 지원해야 합니다.

- SR-IOV 구성:

컴퓨팅 노드는 NIC에서 생성된 VF(Virtual Function)와 함께 SR-IOV를 지원하도록 구성해야 합니다
.

5. Cisco CPNR(Prime Network Registrar) VNF 이미지

- VNF 이미지 호환성:

CPNR VNF 이미지는 SR-IOV 인터페이스를 지원하고 필요한 드라이버를 포함해야 합니다.

- Upload to Glance(한눈에 업로드):

CPNR VNF 이미지를 OpenStack Glance에서 사용할 수 있는지 확인합니다.

6. 관리 액세스

- OpenStack CLI:

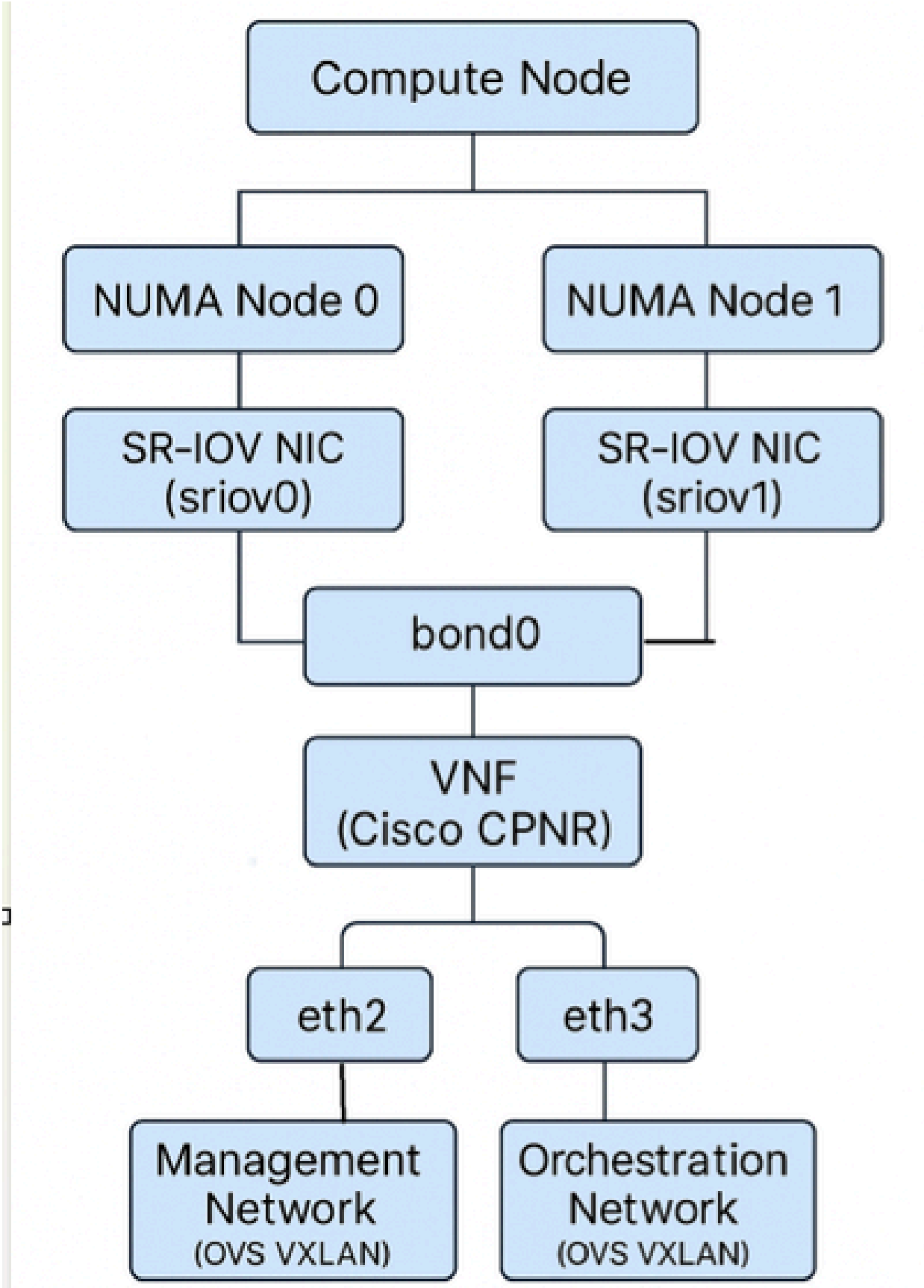
OpenStack CLI에 액세스하여 네트워크 생성, 기능, VNF 시작을 보장합니다.

- 루트 또는 관리자 권한:

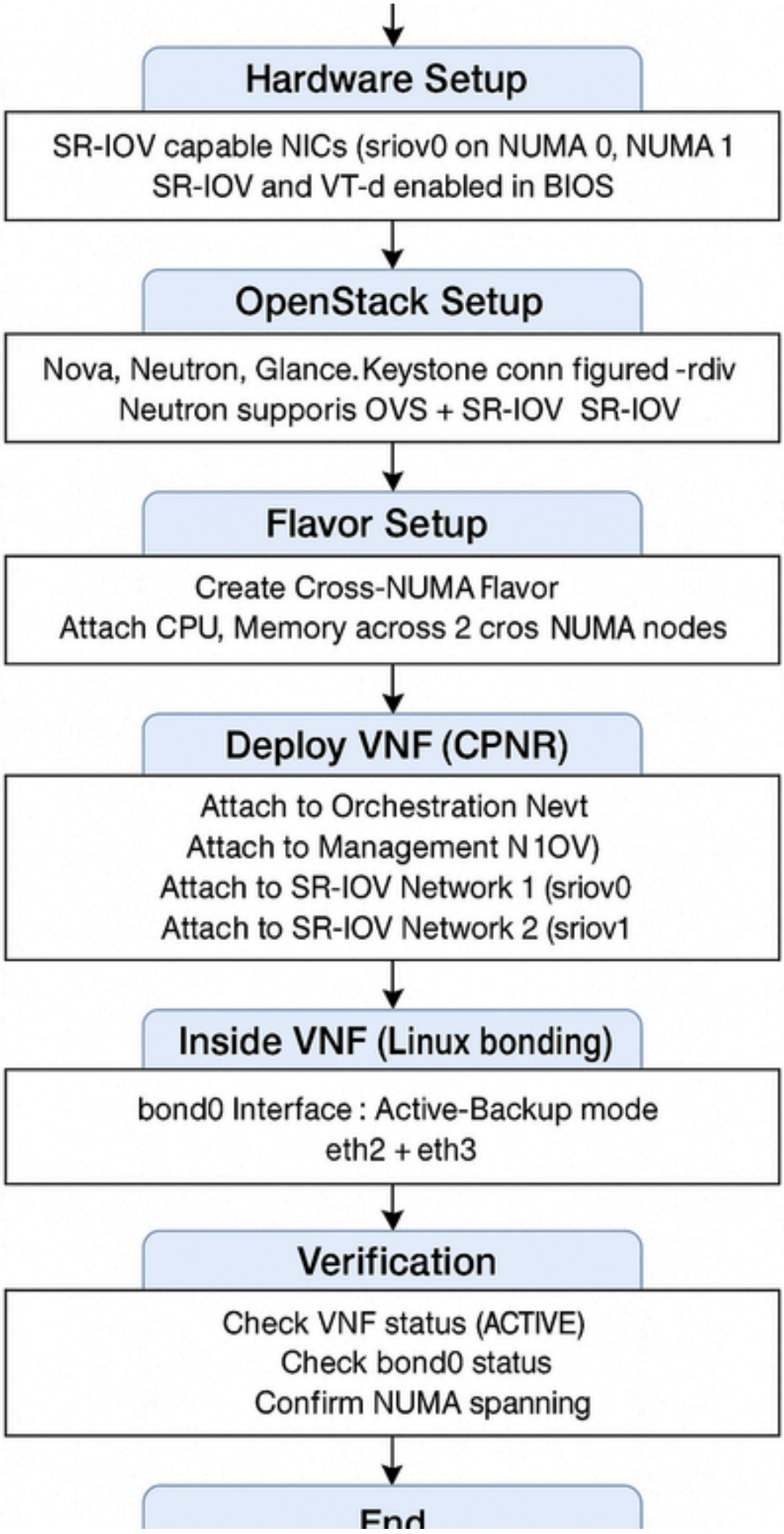
Linux 호스트 및 VNF 내에서 네트워킹을 구성하기 위한 루트 또는 관리 액세스

아키텍처 개요

VNF 네트워크 인터페이스 연결 다이어그램



순서도


Hardware Setup
SR-IOV capable NICs (sriov0 on NUMA 0, NUMA 1
SR-IOV and VT-d enabled in BIOS
OpenStack Setup
Nova, Neutron, Glance.Keystone conn figured -rdiv
Neutron supporis OVS + SR-IOV SR-IOV
Flavor Setup
Create Cross-NUMA Flavor
Attach CPU, Memory across 2 cros NUMA nodes
Deploy VNF (CPNR)
Attach to Orchestration Nevt
Attach to Management N1OV)
Attach to SR-IOV Network 1 (sriov0
Attach to SR-IOV Network 2 (sriov1
Inside VNF (Linux bonding)
bond0 Interface : Active-Backup mode
eth2 + eth3
Verification
Check VNF status (ACTIVE)
Check bond0 status
Confirm NUMA spanning
End

test-tenant

true

false

sriov-vm-deployment

sriov-vm-1-group

vim1

default

sriov-image

custom-flavor

300

30

REBOOT_ONLY

0

mgmt-net

192.168.10.101

1

direct

sriov-net-1

0

sriov-subnet-1

10.10.10.10

2

direct

sriov-net-2

0

sriov-subnet-2

10.10.20.10

1

1

false

```
--user-data



file://tmp/init/sriov-vm-1.cfg
```

핵심 사항

- <interface>의 type>direct</type>는 해당 NIC에 대한 SR-IOV(PCI passthrough)를 활성화합니다.
- 각 SR-IOV 인터페이스에는 자체 네트워크 및 서브넷이 있습니다.
- 필요에 따라 <addresses>에서 IPv4/IPv6를 연결할 수 있습니다.

Cisco ESC XML을 전달할 샘플 Day0 파일:

```
Content-Type: multipart/mixed; boundary="==========267839505026098033O=="
MIME-Version: 1.0`
```

```
--===============2678395050260980330==
MIME-Version: 1.0
Content-Type: text/cloud-boothook; charset="us-ascii"

#cloud-boothook
#!/bin/bash
if [ ! -f /etc/cloud/cloud.cfg.orig ]; then
cp /etc/cloud/cloud.cfg /etc/cloud/cloud.cfg.orig
cp /etc/cloud/cloud.cfg.norootpasswd /etc/cloud/cloud.cfg
fi

--===============2678395050260980330==
MIME-Version: 1.0
Content-Type: text/cloud-config; charset="us-ascii"

#cloud-config
hostname: cpnr
ssh_authorized_keys:
 - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQC7pf8gvOWH/Zv8iAlTv6LWEiPGA3B6t96G6LwTHF6iXOqQxyIUkg8IkqZ6wNwx
 - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQDAmkQGCZUrYqkZOC0J9t7mF9La9zYOqfzzFkk1wWtPga+aANOaFgjqbjj+VlBd
runcmd:
 - /usr/sbin/useradd FMLVL1 -d /home/FMLVL1 -s /bin/bash -g users; (/bin/echo changeme; /bin/echo chang
 - nmcli con add type ethernet con-name eth0 ifname eth0 ip4 10.xx.xx.xx/24
 - nmcli con add type ethernet con-name eth1 ifname eth1 ip4 172.xx.xx.xx/23
 - nmcli connection add type bond con-name bond0 ifname bond0 bond.options "mode=active-backup,miimon=1
 - nmcli connection add type ethernet ifname ens6 master bond0
 - nmcli connection add type ethernet ifname ens7 master bond0
 - nmcli con up eth0
 - nmcli con up eth1
 - nmcli con up bond-slave-ens6
 - nmcli con up bond-slave-ens7
 - nmcli con down bond0
 - nmcli con up bond0
 - nmcli connection reload
 - hostnamectl set-hostname CPNRDNS01CO

--===============2678395050260980330==
```

OpenStack에서 SR-IOV 포트 및 액티브-백업 본드 인터페이스를 사용하여 CPNR VNF 구축

CPNR은 기업 및 서비스 공급자 네트워크를 위해 IPAM(IP 주소 관리), DHCP 및 DNS(Domain Name Server) 서비스를 제공하는 중요한 VNF(Virtual Network Function)입니다. OpenStack에서 CTNR을 VNF로 구축하려면 특히 이중화와 성능을 위해 SR-IOV 포트, 교차 NUMA 컨피그레이션 및 Active-Backup 본드 인터페이스를 활용할 때 신중한 계획이 필요합니다.

이 문서에서는 OpenStack에 CPNR VNF를 구축하기 위한 단계별 프로세스에 대해 설명합니다. 여기에는 다음이 포함됩니다.

- 여러 NUMA 노드에서 SR-IOV NIC에 액세스하는 데 중요한 교차 NUMA 모드 구성
- Active-Backup 연결 설정을 통해 스위치측 구성 없이도 고가용성을 보장합니다.
- OpenStack 네트워크 구성, 기능 및 요약
- IP 주소 계획, ifcfg 파일을 사용한 Linux 네트워킹 구성, Cisco ESC를 사용한 VNF 구축.

구축의 주요 기능

1. NUMA 간 인식:

    - CPNR VNF는 NUMA 노드를 확장하여 SR-IOV NIC(NUMA 0의 sriov0 및 NUMA 1의 sriov1)에 액세스합니다.
    - 단일 NUMA 모드에서는 OpenStack이 VNF가 실행되는 동일한 NUMA 노드에 물리적으로 위치한 NIC에만 VNF를 연결할 수 있도록 허용하므로 Cross-NUMA 모드가 필요합니다. VNF는 교차 NUMA 모드를 활성화함으로써 두 NUMA 노드의 NIC 및 리소스를 활용할 수 있습니다.

2. 액티브-백업 연결:

    - bond0 인터페이스는 SR-IOV NIC를 사용하여 생성됩니다(sriov0의 eth2 및 sriov1의 eth31S의 eth3).
    - Active-Backup 모드는 스위치측 컨피그레이션 없이도 이중화 및 내결함성을 보장합니다.

3. OpenStack 네트워킹:

    - 오케스트레이션 및 관리 네트워크: 제어 및 관리 트래픽을 위한 Openvswitch 기반
    - 애플리케이션/서비스 네트워크: 고성능 트래픽을 위한 SR-IOV 기반.

Cross-NUMA 모드가 필요한 이유

1. OpenStack의 NUMA 인식 네트워킹

NUMA는 각 CPU(및 로컬 메모리 및 디바이스)가 NUMA 노드로 그룹화되는 메모리 아키텍처입니다. OpenStack에서 NUMA 인식 배치는 VNF가 동일한 NUMA 노드의 리소스에 최적으로 할당되도록 하여 레이턴시를 최소화하고 성능을 극대화합니다.

- SR-IOV NIC는 NUMA-Local입니다.
    - 각 물리적 NIC는 특정 NUMA 노드에 연결됩니다. 예를 들면 다음과 같습니다.
        - sriov0은 NUMA 노드 0에 연결됩니다.
        - sriov1은 NUMA 노드 1에 연결됩니다.
- 단일 NUMA 모드 제한:
    - VNF가 단일 NUMA 모드에서 실행되는 경우 OpenStack에서는 VNF가 실행되는 NUMA 노드에 로컬인 NIC에만 연결할 수 있습니다. 이는 다음을 의미합니다.
        - VNF가 NUMA 0에서 실행되는 경우 sriov0의 NIC에만 연결할 수 있습니다.
        - VNF가 NUMA 1에서 실행되는 경우 sriov1의 NIC에만 연결할 수 있습니다.

2. Cross-NUMA 모드가 필요한 이유

CPNR VNF에서는 다음 항목에 액세스해야 합니다.

- 오케스트레이션 네트워크(Openvswitch, NUMA에 구애받지 않음)
- 관리 네트워크(Openvswitch, NUMA에 구애받지 않음)
- SR-IOV 네트워크 1: Connected tosriov0(NUMA 노드 0)
- SR-IOV 네트워크 2: tosriov1(NUMA 노드 1)에 연결되었습니다.

이 구축에서 CPNR VNF는 이중화 및 고가용성을 제공하기 위해 NUMA 0(sriov0) 및 NUMA 1(sriov1) 둘 다에서 SR-IOV NIC에 액세스해야 합니다. 이를 위해 다음을 수행합니다.

- VNF는 OpenStack이 여러 NUMA 노드에서 CPU, 메모리 및 NIC를 할당할 수 있도록 하는 교차 NUMA 모드로 시작해야 합니다.
- 이렇게 하면 VNF가 sriov0 및 sriov1의 NIC에 연결할 수 있으므로 Active-Backup 본드 컨피그레이션에서 SR-IOV 포트를 모두 사용할 수 있습니다.

OVS 포트에 대한 Conntrack 크기 제한

Contrack이란?

Conntrack은 네트워크 연결, 특히 NAT(Network Address Translation) 및 방화벽 규칙을 추적하는 데 사용되는 Linux 커널 기능입니다. OpenStack의 OVS 기반 포트의 경우 conntrack을 사용하여 연결 상태를 관리하고 보안 그룹 규칙을 적용합니다.

Conntrack이 OVS 포트에 미치는 영향

1. Conntrack 테이블:

    - 각 활성 연결은 contrack 테이블의 엔트리를 사용합니다.
    - contrack 테이블의 크기는 f_conntrack_maxparameter에 의해 제한됩니다.

2. 기본 제한:

    - 기본적으로 contrack 테이블 크기는 65536 항목입니다. 높은 연결 속도를 가진 워크로드(예: 동시 흐름이 많은 VNF)의 경우 이 제한을 빠르게 소진하여 패킷이 삭제될 수 있습니다.

3. OVS 포트에 미치는 영향:

    - contrack 테이블이 가득 차면 새 연결이 삭제되어 VNF 성능에 심각한 영향을 미칠 수 있습니다.
    - 이는 특히 OVS 포트를 사용하는 오케스트레이션 및 관리 네트워크와 관련이 있습니다.

Conntrack 제한을 완화하는 방법

1. Conntrack 테이블 크기 증가:

- 현재 제한 보기:

```
sysctl net.netfilter.nf_conntrack_max
```

- 한도 증가:

```
sysctl -w net.netfilter.nf_conntrack_max=262144
```

- 변경 사항을 영구 적용하십시오.

```
echo "net.netfilter.nf_conntrack_max=262144" >> /etc/sysctl.conf
```

2. Conntrack 사용 모니터링:

   contrack 통계를 확인합니다.

```
cat /proc/sys/net/netfilter/nf_conntrack_count
```

3. 보안 그룹 규칙 최적화:

   OVS 포트에 적용되는 규칙의 수를 줄여 contrack 오버헤드를 최소화합니다.

SR-IOV가 Conntrack 문제를 해결하는 방법

1. Conntrack 종속성 제거

SR-IOV 포트는 contrack과 같은 OVS 데이터 경로 및 Linux 커널 기능을 우회합니다. 그러면 연결 추적 오버헤드가 완전히 제거됩니다.

2. 확장성 향상

conntrack 테이블 크기(nf_conntrack_max)에 의해 제한되는 OVS 포트와 달리 SR-IOV 포트는 사실상 무제한의 연결을 처리할 수 있습니다.

3. 지연 시간 감소

SR-IOV 포트는 패킷 처리를 NIC 하드웨어로 오프로드함으로써 소프트웨어 기반 Conntrack 처리로

인한 지연 시간을 제거합니다.

CPNR VM의 SR-IOV 포트에 대해 액티브-백업 모드를 선택하는 이유

Active-Backup 연결 모드는 단순성, 내결함성, SR-IOV 인터페이스와의 호환성 때문에 이 구축에 특히 적합합니다. 그 이유는 다음과 같습니다.

1. 복잡성이 없는 이중화

- 활성-백업 모드: 단 하나의 인터페이스(activeinterface)가 지정된 시간에 트래픽을 전송하고 수신합니다. 다른 인터페이스는 스탠바이 모드로 유지됩니다.
- 액티브 인터페이스에 오류가 발생하는 경우(예: 링크 오류 또는 하드웨어 문제) 본드는 자동으로 스탠바이 인터페이스로 전환됩니다. 이를 통해 수동 작업 없이 지속적인 네트워크 연결이 보장됩니다.

2. LAG(Link Aggregation Group) 필요 없음

- 다른 연결 모드(예: 802.3adorbalance-alb)와 달리 활성 백업 모드에는 LACP(Link Aggregation Control Protocol) 또는 스위치측 컨피그레이션이 필요하지 않습니다.
- SR-IOV VF는 일반적으로 LACP 또는 LAG 컨피그레이션을 지원하지 않으므로 이는 SR-IOV 포트에 특히 중요합니다.

3. 원활한 장애 조치

- 장애 조치는 거의 즉시 실행되며 트래픽에 대한 중단을 최소화합니다.
- 액티브 인터페이스에 오류가 발생하면 본드는 즉시 스탠바이 인터페이스를 액티브 상태로 올립니다.

4. 하드웨어 독립성

액티브-백업 모드는 기본 물리적 스위치 또는 하드웨어와 독립적으로 작동합니다. 장애 조치 로직은 전적으로 Linux 커널에 상주하므로 휴대성과 활용성이 우수합니다.

5. SR-IOV에 최적화

SR-IOV VF는 특정 물리적 NIC 및 NUMA 노드에 연결됩니다. Active-Backup 모드를 사용하면 서로 다른 NUMA 노드의 VF를 단일 논리 본드 인터페이스(bond0)로 결합할 수 있습니다. 이를 통해 고가용성을 보장하는 동시에 NUMA 리소스를 효율적으로 사용할 수 있습니다.

Active-Backup 모드는 Linux 연결에서 가장 간단하고 널리 사용되는 모드 중 하나입니다. 연결된 인터페이스 중 하나에 장애가 발생하더라도 트래픽이 계속 원활하게 흐르도록 하여 높은 가용성을 제공하도록 설계되었습니다. 이는 액티브-백업 모드가 작동하는 방식, 주요 특성 및 이점에 대한 심층적인 설명입니다.

Linux Bond Interface란 무엇입니까?

Linux의 본드 인터페이스는 둘 이상의 네트워크 인터페이스를 하나의 논리적 인터페이스로 결합합니다. 이 논리적 인터페이스는 bond(예: bond0)라고 하며 다음을 제공합니다.

- 이중화: 네트워크 연결의 고가용성 보장
- 성능 향상: 다른 모드(예: balance-rror 802.3ad)에서는 대역폭을 집계할 수도 있습니다.

액티브-백업 모드는 어떻게 작동합니까?

Active-Backup 모드에서는 트래픽을 전송 및 수신하기 위해 지정된 시간에 하나의 인터페이스(액티브 인터페이스라고 함)만 사용됩니다. 다른 인터페이스는 대기 모드로 유지됩니다. 액티브 인터페이스에 오류가 발생하면 스탠바이 인터페이스 중 하나가 액티브 상태로 올려지고, 트래픽이 자동으로 새 액티브 인터페이스로 다시 라우팅됩니다.

액티브-백업 모드의 주요 기능

1. 단일 활성 인터페이스:

    - 임의의 시점에서, 바인딩의 물리적 인터페이스 하나만 트래픽 전송 및 수신에 대해 활성화됩니다.
    - 스탠바이 인터페이스는 장애 조치가 발생하지 않는 한 완전히 패시브 방식입니다.

2. 자동 장애 조치:

    - 하드웨어 문제, 케이블 연결 끊기 또는 링크 장애 등으로 인해 액티브 인터페이스에 장애가 발생하면 본드가 자동으로 스탠바이 인터페이스로 전환됩니다.
    - 페일오버는 원활하며 수동 개입이 필요하지 않습니다.

3. 페일백 지원:

    장애가 발생한 인터페이스가 복원되면 연결 컨피그레이션에 따라 자동으로 다시 액티브 상태가 되거나(활성화되도록 구성된 경우) 스탠바이 모드가 유지됩니다.

4. 스위치측 요구 사항 없음:

    - 다른 본딩 모드(예: 802.3adorbalance-rr)와 달리 활성 백업 모드는 물리적 스위치에 특별한 컨피그레이션(예: LAG 또는 LACP)이 필요하지 않습니다.
    - 따라서 스위치측 컨피그레이션이 가능하지 않거나 LAG를 지원하지 않는 SR-IOV 가상 함수를 결합할 때 이상적입니다.

5. 모니터링:

    - 이 연결은 timeion(Media Independent Interface Monitor) 매개변수를 사용하여 모든 멤버 인터페이스의 상태를 지속적으로 모니터링합니다.

- 링크 장애가 감지되면 본드는 즉시 정상 대기 인터페이스로 전환됩니다.

액티브-백업 모드에서 트래픽 흐름 방식

정상 가동

1. 액티브 인터페이스:

    - 트래픽은 액티브 인터페이스를 통해서만 흐릅니다(예: eth2와 eth3 결합의 eth2).
    - 스탠바이 인터페이스(eth3)는 유휴 상태로 유지되며 트래픽을 전송하거나 수신하지 않습니다.

2. 모니터링:

    - 본드는 모든 멤버 인터페이스의 상태를 주기적으로 모니터링합니다. 이 작업은 다음을 사용하여 수행됩니다.
        - miimon: 구성 가능한 간격(예: 100ms마다)으로 각 인터페이스의 링크 상태를 확인합니다.
        - ARP(Address Resolution Protocol) 모니터링(옵션): 활성 인터페이스에 연결할 수 있도록 ARP 요청을 보냅니다.

장애 조치 시나리오

1. 활성 인터페이스의 링크 실패:

    액티브 인터페이스(eth2)에 장애가 발생하면(예: 케이블 언플러그, NIC 하드웨어 장애 또는 링크 다운) 본드는 ARP 모니터링을 사용하여 장애를 즉시 감지합니다.

2. 자동 장애 조치:

    - 이 연결은 스탠바이 인터페이스(eth3)로 전환되며, 이는 새로운 액티브 인터페이스가 됩니다.
    - 트래픽은 수동 개입 없이 새로운 액티브 인터페이스를 통해 다시 라우팅됩니다.

3. 페일오버 적시성:

    장애 조치 프로세스는 거의 즉각적입니다(일반적으로 몇 밀리초 이내, 시간 간격에 따라).

장애 복구 시나리오

1. 실패한 인터페이스 복원:

    - 이전에 장애가 발생한 인터페이스(eth2)가 복원되면 다음을 수행할 수 있습니다.
        - 활성 역할을 자동으로 재확보합니다(그렇게 구성된 경우).
        - 스탠바이 모드(기본 동작)를 유지합니다.

2. 트래픽 연속성:

   페일백이 원활하여 지속적인 트래픽 플로우에 지장을 주지 않습니다.

사용 사례: SR-IOV 포트와의 액티브-백업 연결

Active-Backup 모드는 다음과 같은 이유로 SR-IOV 인터페이스에 특히 적합합니다.

- SR-IOV VF는 일반적으로 LACP와 같은 링크 집계 프로토콜을 지원하지 않습니다.
- Active-Backup 모드의 결합은 스위치측 컨피그레이션 없이 이중화를 제공할 수 있습니다.

예를 들면 다음과 같습니다.

- eth2는 SR-IOV VF onsriov0(NUMA 노드 0)에 매핑됩니다.
- eth3는 SR-IOV VF onsriov1(NUMA 노드 1)에 매핑됩니다.
- 본드(bond0)는 이러한 인터페이스를 결합하여 SR-IOV VF 간에 원활한 장애 조치를 제공합니다.

1단계. OpenStack 네트워킹

CPNR VNF에는 다음과 같은 4개의 네트워크가 필요합니다.

1. 오케스트레이션 네트워크: 제어 및 오케스트레이션 트래픽(Openvswitch 기반).
2. 관리 네트워크: 관리 액세스(Openvswitch 기반).
3. SR-IOV 네트워크 1: sriov0의 애플리케이션/서비스 트래픽.
4. SR-IOV 네트워크 2: sriov1의 애플리케이션/서비스 트래픽.

단계별 구축:

1.1단계. Openvswitch 네트워크 생성

- 오케스트레이션 네트워크:

```
openstack network create --provider-network-type vxlan orchestration-network
```

- 관리 네트워크:

```
openstack network create --provider-network-type vxlan management-network
```

1.2단계. Openvswitch 네트워크용 서브넷 생성

- 오케스트레이션 서브넷:

```
openstack subnet create --network orchestration-network \
--subnet-range 192.168.100.0/24 orchestration-subnet
```

- 관리 서브넷:

```
openstack subnet create --network management-network \
--subnet-range 10.10.10.0/24 management-subnet
```

1.3단계. SR-IOV 네트워크 생성

- SR-IOV 네트워크 1:

```
openstack network create --provider-network-type vlan \
--provider-physical-network sriov0 --provider-segment 101 sriov-network-1
```

- SR-IOV 네트워크 2:

```
openstack network create --provider-network-type vlan \
--provider-physical-network sriov1 --provider-segment 102 sriov-network-2
```

2단계. OpenStack의 특징

2.1단계. Cross-NUMA Flavor 만들기

VNF가 두 NUMA 노드 모두에서 SR-IOV NIC에 액세스할 수 있도록 하려면 교차 NUMA 지원을 통해 기능을 생성합니다.

```
openstack flavor create --ram 8192 --vcpus 4 --disk 40 cross-numa-flavor
```

2.2단계. NUMA 속성 구성

NUMA 관련 속성을 설정합니다.

```
openstack flavor set cross-numa-flavor \
--property hw:numa_nodes=2 \
--property hw:cpu_policy=dedicated \
--property hw:mem_page_size=large
```

3단계. Active-Backup 모드에서 연결 구성

VNF를 시작한 후 VNF의 SR-IOV 포트(eth2 및 eth3)에 대한 본드 인터페이스를 구성합니다.

3.1단계. 본드 인터페이스 컨피그레이션

Active-Backup 모드에서 본드 인터페이스(bond0)를 생성합니다.

```
vi /etc/sysconfig/network-scripts/ifcfg-bond0
```

```
DEVICE=bond0
BOOTPROTO=static
ONBOOT=yes
BONDING_OPTS="mode=active-backup miimon=100"
IPADDR=172.16.1.10
NETMASK=255.255.255.0
GATEWAY=172.16.1.1
```

3.2단계. 슬레이브 인터페이스 구성

- eth2:

  ```
  vi /etc/sysconfig/network-scripts/ifcfg-eth2
  ```

  ```
  DEVICE=eth2
  ```

```
ONBOOT=yes
MASTER=bond0
SLAVE=yes
```

- eth3:

```
vi /etc/sysconfig/network-scripts/ifcfg-eth3
```

```
DEVICE=eth3
ONBOOT=yes
MASTER=bond0
SLAVE=yes
```

3.3단계. 컨피그레이션 적용

컨피그레이션을 적용하려면 네트워크 서비스를 다시 시작하십시오.

```
systemctl restart network
```

다음을 확인합니다.

VNF를 구축한 후 다음 단계를 사용하여 기능을 확인합니다.

1. VNF 상태 확인

VNF 인스턴스가 활성 상태인지 확인합니다.

```
openstack server show cpnr-instance
```

상태가 ACTIVE인지 확인합니다.

2. 네트워크 연결 확인

- Ping 테스트: VNF가 모든 네트워크를 통해 통신할 수 있는지 확인합니다.

```
ping
```

```
ping
```

- 결합 인터페이스:
  - bond0이 활성 상태인지 확인합니다.

    ```
    cat /proc/net/bonding/bond0
    ```

    검색 대상:

    - 현재 활성 슬레이브: 활성 인터페이스를 나타냅니다.
    - 슬레이브 인터페이스: 2와 3이 모두 본드의 일부임을 확인합니다.

3. NUMA 배치 확인

VNF가 두 NUMA 노드의 리소스를 모두 사용하고 있는지 확인합니다.

```
nova show
```

```
    --human | grep numa
```

모범 사례

- 모니터링 및 문제 해결: SR-IOV 인터페이스를 모니터링하려면 다음과 같은 도구를 사용합니

다.

- Security: 물리적 네트워크에 대한 액세스를 신중하게 관리하고 테넌트 간에 엄격한 격리를 적용합니다.
- 확장: 사용 가능한 VF의 수가 NIC 하드웨어에 의해 제한되므로 SR-IOV 구축을 확장할 때 물리적 NIC 용량을 계획합니다.

문제 해결

구축이 예상대로 작동하지 않을 경우 다음 트러블슈팅 단계를 참조하십시오.

1. SR-IOV 컨피그레이션 확인

- BIOS에서 SR-IOV가 활성화되었는지 확인합니다.

```
dmesg | grep -i "SR-IOV"
```

- VF가 NIC에 생성되었는지 확인합니다.

```
lspci | grep Ethernet
```

2. NUMA 배치 확인

VNF에서 두 NIC에 모두 액세스할 수 없는 경우 cross-NUMA 모드가 활성화되었는지 확인합니다.

- 향의 NUMA 속성을 확인합니다.

```
openstack flavor show cross-numa-flavor
```

3. 채권 인터페이스 문제

- 채권 상태를 확인합니다.

```
cat /proc/net/bonding/bond0
```

- 채권이 작동하지 않는 경우

- 슬레이브 인터페이스(eth2 및 eth3)가 본딩의 일부로 올바르게 구성되었는지 확인합니다.
- 네트워크 서비스를 다시 시작합니다.

```
systemctl restart network
```

4. 네트워크 연결 문제

- OpenStack 포트 바인딩을 확인합니다.

```
openstack port list --server cpnr-instance
```

- VNF 내에서 올바른 IP 컨피그레이션을 확인합니다.

```
ip addr show
```

결론

SR-IOV 포트를 사용하여 OpenStack에 CPNR VNF를 구축하려면 VNF가 두 NUMA 노드의 NIC에 연결할 수 있도록 교차 NUMA 모드가 필요합니다. 이는 OpenStack이 단일 NUMA 모드의 VNF를 VNF가 실행되는 NUMA 노드 내의 리소스(NIC, CPU, 메모리)에만 액세스하도록 제한하기 때문에 필수적입니다. Cross-NUMA 모드와 Active-Backup 결합을 결합하면 고가용성, 내결함성 및 효율적인 리소스 활용이 보장되므로 복원력과 성능이 뛰어납니다.

이 번역에 관하여

Cisco는 전 세계 사용자에게 다양한 언어로 지원 콘텐츠를 제공하기 위해 기계 번역 기술과 수작업 번역을 병행하여 이 문서를 번역했습니다. 아무리 품질이 높은 기계 번역이라도 전문 번역가의 번역 결과물만큼 정확하지는 않습니다. Cisco Systems, Inc.는 이 같은 번역에 대해 어떠한 책임도 지지 않으며 항상 원본 영문 문서(링크 제공됨)를 참조할 것을 권장합니다.