# 在Openstack CVIM上使用SRIOV网络部署DNS VNF - Prime Network Registrar(DNS)配置示例

## 目录


简介

先决条件

- 要求
- 使用的组件

背景信息

配置

- 1.硬件要求
- 2.识别Intel NIC卡
  - 步骤1.使用lspci命令
  - 步骤2.检验XL710
  - 步骤3.检验E810CQDA2
  - 步骤4.确认驱动程序支持
- 3. BIOS/UEFI配置
- 4. OpenStack设置
- 5. Cisco Prime Network Registrar(CPNR)VNF映像
- 6.管理权限

体系结构概述

- VNF网络接口连接图
- 流程图
- 示例配置
- 要点

在OpenStack上使用SR-IOV端口和主备绑定接口部署CPNR VNF

部署的主要功能

为什么需要跨NUMA模式

- 1. OpenStack中的NUMA感知网络
- 2.为什么需要跨网络联盟模式

OVS端口的Contrack大小限制

- 什么是Conntrack?
- Conntrack对OVS端口的影响
- 如何缓解跟踪限制

SR-IOV如何解决连接问题

- 1.消除连接跟踪依赖关系
- 2.更高的可扩展性
- 3.减少延迟

为CPNR VM上的SR-IOV端口选择主用备份模式的原因

- 1.无复杂性的冗余
- 2.不需要链路汇聚组(LAG)

- 3.无缝故障切换
- 4.硬件独立性
- 5.针对SR-IOV进行了优化

什么是Linux Bond接口？

Active-Backup模式如何工作？

- 主用 — 备用模式的主要功能

流量在主用 — 备用模式中的传输方式

- 正常运行
- 故障切换场景
- 故障恢复场景

使用案例:与SR-IOV端口的主用 — 备用绑定

步骤1. OpenStack网络

- 步骤1.1.创建Openvswitch网络
- 步骤1.2.为Openvswitch网络创建子网
- 步骤1.3.创建SR-IOV网络

步骤2. OpenStack风格

- 步骤2.1.创建跨NUMA口味
- 步骤2.2.配置NUMA属性

步骤3.在Active-Backup模式下配置绑定

- 步骤3.1.绑定接口配置
- 步骤3.2.配置从接口
- 步骤3.3.应用配置

验证

- 1.检验VNF状态
- 2.检验网络连接
- 3.检验NUMA位置

最佳实践

故障排除

- 1.检验SR-IOV配置
- 2.检验NUMA位置
- 3.债券接口发行
- 4.网络连接问题

结论


---

# 简介

本文档介绍使用SR-IOV和Active-Backup绑定在OpenStack Cisco Virtualized Infrastructure Manager(CVIM)上逐步部署CPNR。

# 先决条件

### 要求

Cisco 建议您了解以下主题：

- 熟悉OpenStack和单根输入/输出虚拟化(SR-IOV)概念
- 有关Cisco Virtual Interface Manager(VIM)和Cisco Elastic Services Controller和Linux命令和网络的工作知识

### 使用的组件

本文档不限于特定的软件和硬件版本。

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您的网络处于活动状态，请确保您了解任何命令的潜在影响

## 背景信息

在当前网络环境中，虚拟网络功能(VNF)在实现灵活、可扩展且高效的网络服务方面发挥着关键作用。对于需要高性能网络连接的VNF，SR-IOV是一种常用的技术。SR-IOV允许VNF绕过虚拟机监控程序虚拟交换机并直接访问物理网络接口控制器(NIC)资源，从而减少延迟并提高吞吐量。

## 配置

在继续部署之前，请确保满足这些前提条件。

### 1.硬件要求

- 支持SR-IOV的NIC:
  - 在BIOS/统一可扩展固件接口(UEFI)中至少有两个启用了SR-IOV的物理NIC。
  - 示例：sriov0映射到非统一内存访问(NUMA)节点0,sriov1映射到NUMA节点1。

- NUMA感知主机：
  - 计算节点必须支持NUMA架构。
  - 必须在主机BIOS/UEFI中启用NUMA支持。

### 2.识别Intel NIC卡

Intel XL710和E810CQDA2 NIC卡通常用于高性能SR-IOV网络。要验证主机上的NIC卡型号，请参阅以下步骤：

步骤1.使用lspci命令

执行以下命令以列出与网络控制器相关的外围组件互联(PCI)设备：

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

输出示例：

```
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)
```

步骤2.检验XL710

如果网卡是Intel XL710，您可以在输出中看到以太网控制器XL710。

步骤3.检验E810CQDA2

如果网卡是Intel E810CQDA2，您可以在输出中看到以太网控制器E810-C。

步骤4.确认驱动程序支持

要检查正在使用的网卡驱动程序，请运行：

```
ethtool -i
```

XL710的输出示例：

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

E810CQDA2的输出示例：

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

确保驱动程序版本与OpenStack和Linux分发版的兼容性矩阵相匹配。

## 3. BIOS/UEFI配置

- 启用SR-IOV:

确保在服务器BIOS/UEFI中启用SR-IOV。

- 为定向I/O(VT-d)/AMD-Vi启用虚拟化技术：

必须启用Intel VT-d或AMD-Vi才能实现PCI直通和SR-IOV功能。

## 4. OpenStack设置

- 核心OpenStack服务：

确保安装和配置了OpenStack服务，如Nova、Neutron、Glance和Keystone。

- Neutron配置：

Neutron必须同时支持用于协调/管理网络的Openvswitch(OVS)和用于应用/服务网络的SR-IOV。

- SR-IOV配置：

计算节点必须配置为支持SR-IOV，在NIC上创建虚拟功能(VF)。

## 5. Cisco Prime Network Registrar(CPNR)VNF映像

- VNF映像兼容性：

CPNR VNF映像必须支持SR-IOV接口并包含必需的驱动程序。

- 上传至概览：

确保CPNR VNF映像在OpenStack Glance中可用。

### 6.管理权限

- OpenStack CLI:

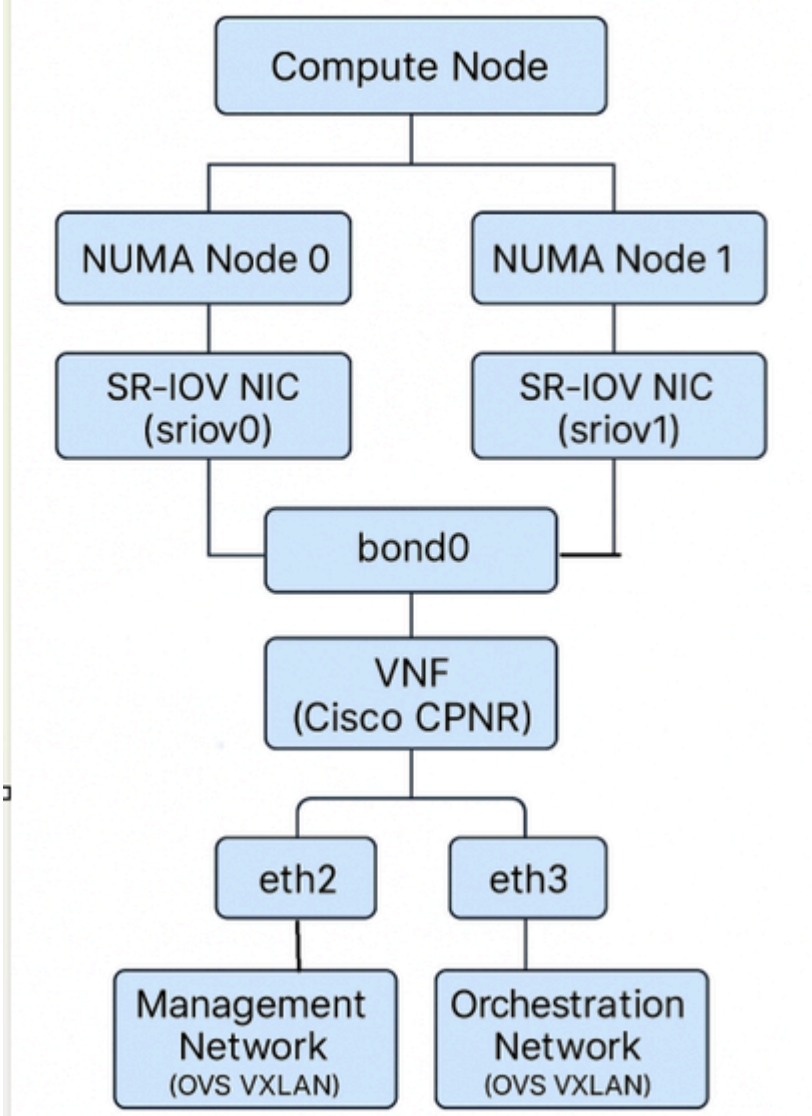
确保访问OpenStack CLI以创建网络、风格和启动VNF。

- 根或管理员权限：

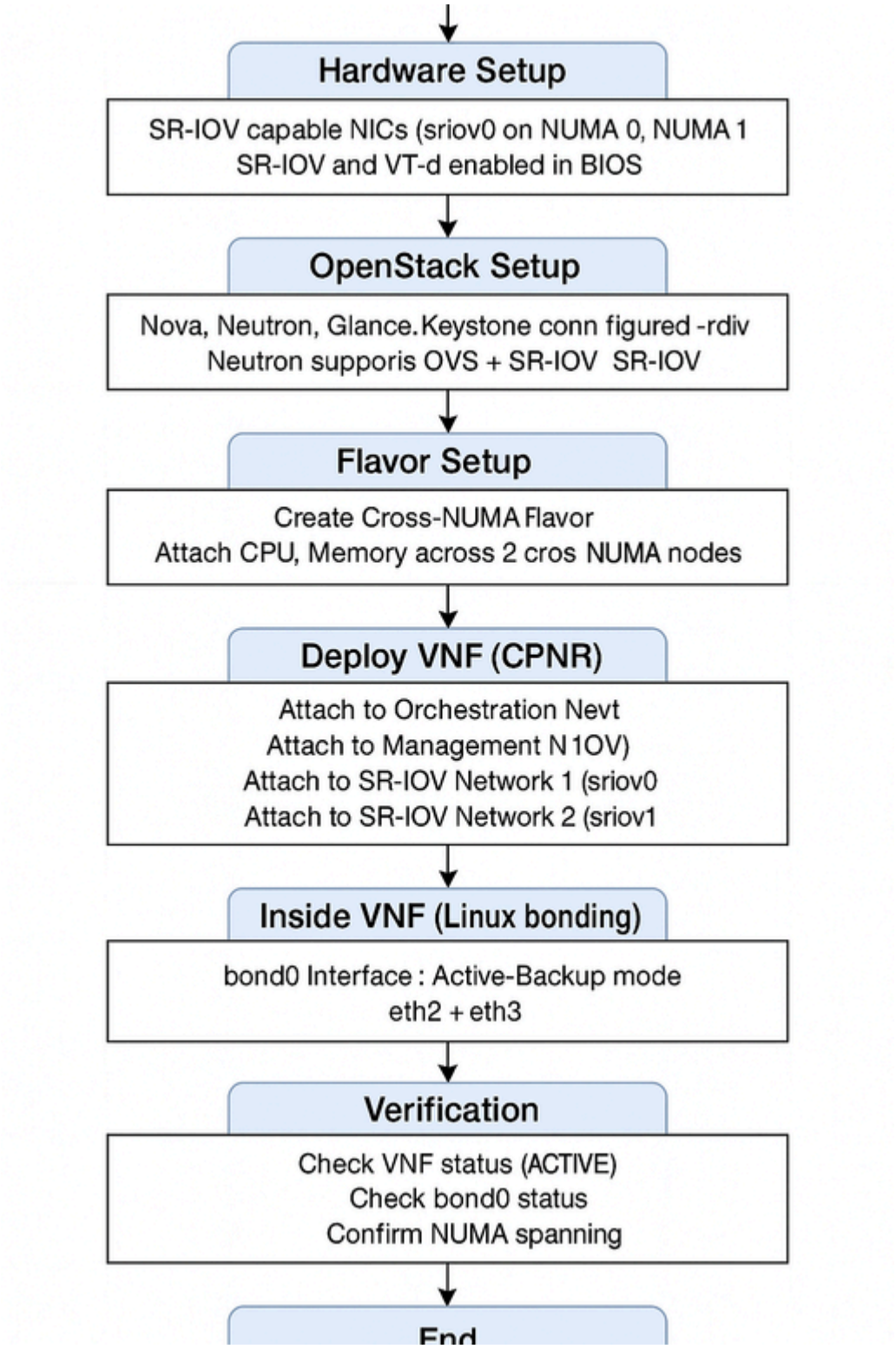
在Linux主机和VNF内配置网络的Root或管理权限。

## 体系结构概述

### VNF网络接口连接图


Compute Node
NUMA Node 0
NUMA Node 1
SR-IOV NIC
(sriov0)
SR-IOV NIC
(sriov1)
bond0
VNF
(Cisco CPNR)
eth2
eth3
Management
Network
(OVS VXLAN)
Orchestration
Network
(OVS VXLAN)

流程图




斯廖夫

## 示例配置

以下是思科弹性服务控制器(ESC)XML示例，演示使用SR-IOV网络（使用type>direct</type>）为名为test-tenant的租户和名为sriov-vm-1的VM部署虚拟机(VM)。这包括多个接口，包括SR-IOV（直接）接口:

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.x.x.x

1

direct

sriov-net-1

*0*

*sriov-subnet-1*

*10.x.x.x*

2

direct

sriov-net-2

*0*

*sriov-subnet-2*

*10.y.y.y*

1

1

false

```
--user-data



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

## 要点

- type>direct</type>在<interface>下为该NIC启用SR-IOV（PCI直通）。
- 每个SR-IOV接口都有自己的网络和子网。
- 您可以根据需要在<addresses>中关联IPv4/IPv6。

在Cisco ESC XML上传递的Day0文件示例：

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

--================267839505026098033O==
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 TEST1KEYAAAAB3NzaC1yc2EAAAADAQABAAACAQC7pf8gvOWH/Zv8iAlTv6LWEiPGA3B6t96G6LwTHF6iXOqQxyIUkg8I
 - ssh-rsa TEST2KEYAAAAB3NzaC1yc2EAAAADAQABAAACAQDAmkQGCZUrYqkZOC0J9t7mF9La9zYOqfzzFkk1wWtPga+aANOaFgjq
runcmd:
 - /usr/sbin/useradd Test1 -d /home/Test1 -s /bin/bash -g users; (/bin/echo changeme; /bin/echo changeme
 - 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 CPNRDNSTEST

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

# 在OpenStack上使用SR-IOV端口和主备绑定接口部署CPNR VNF

CPNR是重要的虚拟网络功能(VNF)，为企业和服务提供商网络提供IP地址管理(IPAM)、DHCP和域名服务器(DNS)服务。在OpenStack中将CPNR部署为VNF需要仔细规划，特别是在利用SR-IOV端口、跨NUMA配置和Active-Backup绑定接口以实现冗余和性能时。

本文解释在OpenStack上部署CPNR VNF的逐步过程。此命令包括：

- 配置跨NUMA模式，这对于从多个NUMA节点访问SR-IOV NIC至关重要。
- 设置Active-Backup绑定，确保高可用性，而无需进行交换机端配置。
- 配置OpenStack网络、风格和概览。
- IP地址规划、使用ifcfg文件配置Linux网络，以及使用Cisco ESC部署VNF。

# 部署的主要功能

1. 跨NUMA认知：

    - CPNR VNF跨越NUMA节点以访问SR-IOV NIC（NUMA 0上的sriov0和NUMA 1上的sriov1）。
    - 需要跨NUMA模式，因为在单NUMA模式下，OpenStack仅允许VNF连接到物理上位于启动VNF的同一NUMA节点上的NIC。通过启用跨NUMA模式，VNF可以利用来自两个NUMA节点的NIC和资源。

2. 主用 — 备用绑定：

    - 使用SR-IOV NIC创建绑定0接口（eth2来自sriov0,eth3来自sriov1）。
    - Active-Backup模式可确保冗余和容错能力，无需进行交换机端配置。

3. OpenStack网络：

    - 协调和管理网络：基于Openvswitch的控制和管理流量。
    - 应用/服务网络：基于SR-IOV的高性能流量。

# 为什么需要跨NUMA模式

## 1. OpenStack中的NUMA感知网络

NUMA是一种内存架构，其中每个CPU（及其本地内存和设备）分组为一个NUMA节点。在OpenStack中，可感知NUMA的位置可确保VNF被优化分配到同一NUMA节点上的资源，以最大程度地减少延迟并最大限度地提高性能。

- SR-IOV NIC为NUMA-Local:
    - 每个物理NIC都绑定到特定的NUMA节点。例如：
        - sriov0连接到NUMA节点0。
        - sriov1连接到NUMA节点1。

- 单NUMA模式限制：
    - 在单NUMA模式下启动VNF时，OpenStack仅允许VNF连接到启动VNF的NUMA节点本地的NIC。这意味着：
        - 如果VNF在NUMA 0上启动，则它只能连接到sriov0上的NIC。
        - 如果VNF在NUMA 1上启动，则它只能连接到sriov1上的NIC。

## 2.为什么需要跨网络联盟模式

CPNR VNF需要访问：

- 协调网络(Openvswitch、NUMA-agnostic)
- 管理网络（Openvswitch、NUMA无关）
- SR-IOV网络1:连接到sriov0（NUMA节点0）
- SR-IOV网络2:已连接到sriov1（NUMA节点1）。

在此部署中，CPNR VNF需要从NUMA 0(sriov0)和NUMA 1(sriov1)访问SR-IOV NIC，以提供冗余和高可用性。为了实现这一目标：

- 必须在cross-NUMA模式下启动VNF，这允许OpenStack从多个NUMA节点分配CPU、内存和NIC。
- 这可以确保VNF可以连接到sriov0和sriov1上的NIC，从而在主用 — 备用绑定配置中使用两个SR-IOV端口。

# OVS端口的Contrack大小限制

## 什么是Conntrack?

Contrack是Linux内核功能，用于跟踪网络连接，尤其是网络地址转换(NAT)和防火墙规则。对于OpenStack中基于OVS的端口，conntrack用于管理连接状态和实施安全组规则。

## Conntrack对OVS端口的影响

1. Conntrack表：
    - 每个活动连接会消耗一个连接表中的条目。
    - contrack表的大小受nf_contrack_max参数限制。
2. 默认限制：
    - 默认情况下，contrack表大小为65536个条目。对于高连接率的工作负载（例如，具有许多并发流的VNF），此限制可以快速耗尽，导致数据包丢失。
3. OVS端口的影响：
    - 如果contrack表已满，则会丢弃新连接，这会严重影响VNF性能。
    - 这尤其与使用OVS端口的协调和管理网络相关。

### 如何缓解跟踪限制

1. 增大跟踪表大小：
    - 查看当前限制：

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

    - 增加限制：

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

    - 使更改持续进行：

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

2. 监控连接使用情况：

   检查conntrack统计信息：

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

3. 优化安全组规则：

   减少应用于OVS端口的规则数量，以最大程度减少控制跟踪开销。

## SR-IOV如何解决连接问题

### 1.消除连接跟踪依赖关系

SR-IOV端口绕过OVS数据路径和Linux内核功能，如contrack。这样可以完全消除连接跟踪开销。

### 2.更高的可扩展性

与OVS端口不同，OVS端口受控制跟踪表大小(nf_conntrack_max)的限制，SR-IOV端口可以处理几乎无限数量的连接。

### 3.减少延迟

通过将数据包处理卸载到NIC硬件，SR-IOV端口消除了基于软件的连接处理带来的延迟。

## 为CPNR VM上的SR-IOV端口选择主用备份模式的原因

Active-Backup绑定模式由于其简单性、容错性以及SR-IOV接口的兼容性，特别适合此部署。原因如下：

### 1.无复杂性的冗余

- Active-Backup模式：在任何给定时间只有一个接口（主用接口）发送和接收流量。其他接口保持备用模式。
- 如果主用接口发生故障（例如，由于链路故障或硬件问题），绑定会自动切换到备用接口。这样无需人工干预即可确保持续的网络连接。

### 2.不需要链路汇聚组(LAG)

- 与其他绑定模式（例如，802.3ad或balance-alb）不同，主用 — 备用模式不需要链路聚合控制协议(LACP)或交换机端配置。
- 这一点对于SR-IOV端口尤其重要，因为SR-IOV VF通常不支持LACP或LAG配置。

### 3.无缝故障切换

- 故障切换几乎是瞬时的，对流量的中断最小。
- 当主用接口发生故障时，绑定会立即将备用接口升级为主用状态。

### 4.硬件独立性

Active-Backup模式独立于底层物理交换机或硬件运行。故障切换逻辑完全驻留在Linux内核中，使其具有高度便携性和通用性。

### 5.针对SR-IOV进行了优化

SR-IOV VF绑定到特定物理NIC和NUMA节点。通过使用主用 — 备用模式，您可以将来自不同NUMA节点的VF合并到单个逻辑绑定接口(bond0)。 这可确保高可用性，同时有效利用NUMA资源。

Active-Backup模式是Linux绑定中最简单、使用最广泛的模式之一。它旨在提供高可用性，从而确保即使绑定接口之一发生故障流量也能继续无缝流。这深入说明了活动备份模式的工作方式、主要特性和优势。

# 什么是Linux Bond接口？

Linux中的bond interface将两个或多个网络接口合并为一个逻辑接口。此逻辑接口称为绑定（例如bond0），用于提供：

- 冗余：确保网络连接的高可用性。
- 性能改进：在其他模式（例如balance-rr或802.3ad）中，它也可以聚合带宽。

# Active-Backup模式如何工作？

在Active-Backup模式下，在任何指定时间都仅使用一个接口（称为active interface）来传输和接收流量。其他接口仍处于standby模式。如果主用接口发生故障，其中一个备用接口将升级为active状态，流量将自动重新路由到新的主用接口。

## 主用 — 备用模式的主要功能

1. 单个活动接口：
    - 在任何指定时间，绑定中只有一个物理接口处于活动状态，用于发送和接收流量。
    - 备用接口完全被动，除非发生故障转移。

2. 自动故障切换:
    - 如果主用接口发生故障（例如，由于硬件问题、电缆断开或链路故障），绑定会自动切换到备用接口。
    - 故障切换是无缝的，无需手动干预。

3. 故障恢复支持：

    故障接口恢复后，根据绑定配置，它可以自动再次变为活动状态（如果已进行配置）或保持备用模式。

4. 无交换机端要求：

    - 与其他绑定模式（例如，802.3ad或balance-rr）不同，Active-Backup模式不需要在物理交换机（例如，LAG或LACP）上进行任何特殊配置。
    - 这使它非常适合于无法进行交换机端配置或绑定SR-IOV虚拟功能（不支持LAG）的场景。

5. 监控：

    - 该绑定使用miimon（媒体独立接口监控器）参数持续监控所有成员接口的运行状况。
    - 如果检测到链路故障，绑定会立即切换到正常备用接口。

# 流量在主用 — 备用模式中的传输方式

## 正常运行

1. 活动接口：

    - 流量专门流经活动接口（例如，eth2和eth3的绑定中的eth2）。
    - 备用接口(eth3)保持空闲，不传输或接收流量。

2. 监控：

    - 绑定定期监视所有成员接口的状态。这通过以下方式完成：
        - 最小值：以可配置的时间间隔（例如，每100毫秒）检查每个接口的链路状态。
        - 地址解析协议(ARP)监控（可选）：发送ARP请求以确保可访问活动接口。

## 故障切换场景

1. 活动接口上的链路故障：

    如果主用接口(eth2)发生故障（例如，电缆未插好、网卡硬件故障或链路断开），绑定会立即使用miimon或ARP监控检测故障。

2. 自动故障切换:

    - 绑定交换到备用接口(eth3)，该接口成为新的活动接口。
    - 流量通过新的活动接口重新路由，无需人工干预。

3. 故障转移及时性：

   故障切换过程几乎是瞬时的（通常在几毫秒内，具体取决于最小间隔）。

### 故障恢复场景

1. 故障接口恢复：

   - 当恢复以前发生故障的接口(eth2)时，它可以：
     - 自动回收活动角色（如果配置为这样做）。
     - 保持备用模式（默认行为）。

2. 流量连续性：

   回切是无缝的，确保不会中断持续流量。

## 使用案例:与SR-IOV端口的主用 — 备用绑定

Active-Backup模式尤其适用于SR-IOV接口，因为：

- SR-IOV VF通常不支持链路聚合协议，例如LACP。
- 主用 — 备用模式中的绑定可以提供冗余，而无需任何交换机端配置。

例如：

- eth2映射到sriov0（NUMA节点0）上的SR-IOV VF。
- eth3映射到sriov1（NUMA节点1）上的SR-IOV VF。
- 绑定(bond0)将这些接口组合在一起，在SR-IOV VF之间提供无缝故障切换。

## 步骤1. OpenStack网络

CPNR VNF需要以下四个网络：

1. 协调网络：用于控制和协调流量（基于Openvswitch）。
2. 管理网络:用于管理访问（基于Openvswitch）。
3. SR-IOV网络1:sriov0上的应用/服务流量。
4. SR-IOV网络2:sriov1上的应用/服务流量。

以下是分步部署：

## 步骤1.1.创建Openvswitch网络

- 协调网络：

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

- 管理网络:

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

## 步骤1.2.为Openvswitch网络创建子网

- 协调子网：

```
openstack subnet create --network orchestration-network \
--subnet-range 192.xx.xx.0/24 orchestration-subnet
```

- 管理子网：

```
openstack subnet create --network management-network \
--subnet-range 10.x.x.0/24 management-subnet
```

## 步骤1.3.创建SR-IOV网络

- SR-IOV网络1:

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

- SR-IOV网络2:

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

# 步骤2. OpenStack风格

## 步骤2.1.创建跨NUMA口味

为了确保VNF可以从两个NUMA节点访问SR-IOV NIC，请创建具有跨NUMA支持的风格：

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

## 步骤2.2.配置NUMA属性

设置NUMA特定属性：

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

# 步骤3.在Active-Backup模式下配置绑定

启动VNF后，为VNF上的SR-IOV端口（eth2和eth3）配置绑定接口。

## 步骤3.1.绑定接口配置

在Active-Backup模式下创建绑定接口(bond0):

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

```
DEVICE=bond0
BOOTPROTO=static
ONBOOT=yes
BONDING_OPTS="mode=active-backup miimon=100"
IPADDR=172.xx.xx.10
NETMASK=255.xx.xx.0
GATEWAY=172.xx.xx.1
```

## 步骤3.2.配置从接口

- 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
```

### 步骤3.3.应用配置

重新启动网络服务以应用配置：

```
systemctl restart network
```

## 验证

部署VNF后，使用以下步骤验证其功能：

### 1.检验VNF状态

检查VNF实例是否处于活动状态：

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

确保状态为ACTIVE。

### 2.检验网络连接

- Ping 测试:验证VNF是否可以通过所有网络通信：

```
ping
```

```
ping
```

- 绑定接口：
    - 确认bond0处于活动状态：

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

查找：

- 当前活动的从设备：指示活动接口。
- 从接口：确认eth2和eth3都是绑定的一部分。

### 3.检验NUMA位置

确保VNF使用来自两个NUMA节点的资源：

```
nova show
      --human | grep numa
```

## 最佳实践

- 监控和故障排除：使用tcpdump和ethtool之类的工具监控SR-IOV接口。
- 安全:小心管理对物理网络的访问，并在租户之间实施严格的隔离。
- 扩展：在扩展SR-IOV部署时规划物理NIC容量，因为可用VF的数量受NIC硬件的限制。

## 故障排除

如果部署未按预期运行，请参阅以下故障排除步骤：

## 1.检验SR-IOV配置

- 检查BIOS中是否启用了SR-IOV:

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

- 确认已在NIC上创建VF:

```
lspci | grep Ethernet
```

## 2.检验NUMA位置

如果VNF无法访问两个网卡，请确保启用跨NUMA模式：

- 检查风味的NUMA属性：

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

## 3.债券接口发行

- 检查绑定状态：

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

- 如果绑定不起作用：
  - 确保从接口（eth2和eth3）已正确配置为绑定的一部分。
  - 重新启动网络服务：

```
systemctl restart network
```

### 4.网络连接问题

- 验证OpenStack端口绑定：

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

- 检查VNF中正确的IP配置：

```
ip addr show
```

## 结论

在具有SR-IOV端口的OpenStack上部署CPNR VNF需要cross-NUMA模式，以使VNF能够从两个NUMA节点连接到NIC。这是至关重要的，因为OpenStack在单NUMA模式下将VNF限制为仅访问启动VNF的NUMA节点内的资源（NIC、CPU、内存）。将跨NUMA模式与Active-Backup绑定相结合，可确保高可用性、容错能力和高效的资源利用率，从而使此部署具有极高的弹性和性能。

关于此翻译
思科采用人工翻译与机器翻译相结合的方式将此文档翻译成不同语言，希望全球的用户都能通过各自的语言得到支持性的内容。

请注意：即使是最好的机器翻译，其准确度也不及专业翻译人员的水平。

Cisco Systems, Inc. 对于翻译的准确性不承担任何责任，并建议您总是参考英文原始文档（已提供链接）。