在STACKIT云上安装和配置Catalyst 8000V

目录

简介

先决条件

- 要求
- 使用的组件

背景信息

经过验证的软件版本和VM类型

- 软件版本
- STACKIT VM风格
- 访客报告的虚拟机监控程序详细信息

配置

- 1.安装堆叠式CLI
- 2.设置环境
- 3.准备部署输入
- 4.上传C8000V映像
- 5.创建引导卷
- 6.创建网络和安全性
- 7.创建有序的NIC
- 8.准备第0天配置
- 9.创建C8000V实例
- 10.验证部署
- 11.删除部署

Appendix

- 有用的STACK命令
- 有用的C8000V命令

简介

本文档介绍在属于Schwarz Digits的STACKIT云上安装和启用Cisco Catalyst 8000V所需的步骤。

先决条件

要求

Cisco 建议您了解以下主题：

- 现有的STACKIT项目和经过身份验证的操作员，有权管理映像、卷、网络接口和服务器。
- 具有兼容Bash的外壳的Linux或macOS工作站。本指南中的命令示例适用于Linux和macOS。
- 批准的网络前缀、安全控制、管理访问以及遵循STACKIT设计的可选公有IP。
- 所选思科IOS® XE版本的授权C8000V QCOW2映像。
- 满足所选版本和功能集发布的C8000V CPU和内存要求的STACKIT计算机类型。
- 适合所选C8000V映像的引导卷大小和固件设置。
- 已安装QEMU，包括用于检查源映像的qemu-img实用程序。
- stackit、jq和base64在命令环境中可用。
- 为自治、控制器或SD-Routing模式准备的0天引导程序。

引导程序可以包含密码、证书或设备身份信息。将其存储在受保护的位置。Base64编码不加密文件。

使用的组件

本文档中的信息基于以下软件版本：

- Cisco IOS XE版本：17.18.5
- Cisco SD-WAN控制器版本：20.18.5，用于控制器（思科SD-WAN）模式

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，请确保您了解所有命令的潜在影响。

背景信息

Cisco Catalyst 8000V是运行Cisco IOS XE的虚拟路由器。在STACKIT上，C8000V映像作为自定义映像上传，并用于创建虚拟服务器的引导卷。

在创建服务器期间，STACKIT会将提供的用户数据放在配置驱动器上。C8000V在初始引导期间读取引导程序。bootstrap内容取决于预期操作模式：

- 自主模式使用iosxe_config.txt。
- 控制器模式使用由Cisco SD-WAN Manager生成的ciscosdwan_cloud_init.cfg。
- SD路由模式使用由Cisco SD-WAN Manager为SD路由生成的ciscosdwan_cloud_init.cfg。

所有三种模式使用相同的映像、卷、接口和服务器创建顺序。引导程序内容和部署后验证因模式而

异。

经过验证的软件版本和VM类型

本文档中的部署工作流程已通过这些软件版本和STACKIT VM风格进行验证。该测试基线记录实验环境；它不会取代当前的Cisco Catalyst 8000V版本说明、平台要求或特定于功能的规模确定指南。

软件版本

- Cisco IOS XE版本：17.18.5
- Cisco SD-WAN控制器版本：20.18.5，用于控制器（思科SD-WAN）模式

STACKIT VM风格

验证中包括以下STACKIT VM类型：

实例大小	vCPUs	内存(GiB)
c3i.4	4	8
c3i.8	8	16
c3i.16	16	32

选择满足预期C8000V部署的CPU、内存、吞吐量和功能要求的风格。

访客报告的虚拟机监控程序详细信息

经过验证的C8000V访客报告了STACKIT虚拟化环境：

<#root>

```
Router#
```

show platform software system hypervisor

Hypervisor:

KVM

Manufacturer:

STACKIT Cloud

Product Name:

OpenStack Nova

Serial Number: <INSTANCE_UUID>
UUID: <INSTANCE_UUID>
Image Variant: None

Router#

配置

1.安装堆叠式CLI

本指南使用Homebrew安装STACKIT CLI。还提供其他软件包管理器和安装方法。有关详细信息，请参阅《STACKIT CLI安装指南》。

```
brew tap stackitcloud/tap
brew install --cask stackit
```

2.设置环境

在运行部署命令之前配置STACKIT CLI环境。

2.1登录和身份验证

登录到您的STACKIT帐户。

```
stackit auth login
```

在您的浏览器中完成身份验证。身份验证成功后，CLI会确认您已登录。

2.2设置项目

为其余命令设置默认项目。

```
stackit config set --project-id xxxxxxxx-yyyy-zzzz-aaaa-bbbbbbbbbbbb
```

如果您不知道项目ID，请列出可用项目。

```
stackit project list
```

3.准备部署输入

收集部署所需的值。本示例创建了两个网络和两个NIC:第一个用于管理或传输，第二个用于服务或数据流量。

```
export PROJECT_ID="<PROJECT_ID>"
export REGION="<REGION>"
export AVAILABILITY_ZONE="<AVAILABILITY_ZONE>"
export MACHINE_TYPE="<MACHINE_TYPE>"

export SERVER_NAME="<C8000V_NAME>"
export IMAGE_NAME="<C8000V_IMAGE_NAME>"
export IMAGE_FILE="/path/to/<C8000V_IMAGE>.qcow2"
export IMAGE_MIN_DISK_SIZE="<MINIMUM_DISK_GB>"
export BOOT_VOLUME_SIZE="<BOOT_VOLUME_GB>"
export IMAGE_UEFI="<true_or_false>"

export MGMT_NETWORK_NAME="${SERVER_NAME}-mgmt"
export DATA_NETWORK_NAME="${SERVER_NAME}-data"
export MGMT_IPV4_PREFIX="<MANAGEMENT_IPV4_PREFIX>"
export DATA_IPV4_PREFIX="<DATA_IPV4_PREFIX>"
export TRUSTED_SOURCE_CIDR="<TRUSTED_SOURCE_IP_OR_NETWORK>/<PREFIX_LENGTH>"
```

使用已批准网络设计中的非重叠前缀。使用CIDR表示法指定TRUSTED_SOURCE_CIDR，包括前

缀长度。使用/32允许一个IPv4地址。列出的管理规则示例仅允许来自TRUSTED_SOURCE_CIDR.F的SSH。

4.上传C8000V映像

4.1检查源映像

验证该文件是否为目标版本的授权映像。在部署记录中记录文件名和校验和。

```
qemu-img info "$IMAGE_FILE"

if command -v shasum >/dev/null 2>&1; then
  shasum -a 256 "$IMAGE_FILE"
else
  sha256sum "$IMAGE_FILE"
fi
```

4.2创建STACKIT自定义映像

将EFI映像的IMAGE_UEFI设置为true，或将BIOS映像设置为false。选择为下载的C8000V映像指定的设置。根据选定的映像和当前产品文档设置最小磁盘大小。

```
IMAGE_RESPONSE="$(
  stackit image create \
    --project-id "$PROJECT_ID" \
    --region "$REGION" \
    --name "$IMAGE_NAME" \
    --disk-format qcow2 \
    --uefi="$IMAGE_UEFI" \
    --min-disk-size "$IMAGE_MIN_DISK_SIZE" \
    --local-file-path "$IMAGE_FILE" \
    --output-format json
)"

export IMAGE_ID="$(printf '%s' "$IMAGE_RESPONSE" | jq -r '.id')"
printf 'IMAGE_ID=%s\n' "$IMAGE_ID"
```

4.3轮询，直到映像可用

```
stackit image describe "$IMAGE_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION" \
```

```
  --output-format json |
  jq '{id, name, status, diskFormat}'
```

映像状态可用时继续。

5.创建引导卷

从自定义映像创建新启动卷。卷大小必须满足所选映像的磁盘要求。

```
BOOT_VOLUME_RESPONSE="$(
  stackit volume create \
    --project-id "$PROJECT_ID" \
    --region "$REGION" \
    --availability-zone "$AVAILABILITY_ZONE" \
    --name "${SERVER_NAME}-boot" \
    --source-id "$IMAGE_ID" \
    --source-type image \
    --size "$BOOT_VOLUME_SIZE" \
    --output-format json
)"

export BOOT_VOLUME_ID="$(printf '%s' "$BOOT_VOLUME_RESPONSE" | jq -r '.id')"
printf 'BOOT_VOLUME_ID=%s\n' "$BOOT_VOLUME_ID"
```

轮询卷直到其状态为可用。

```
stackit volume describe "$BOOT_VOLUME_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION" \
  --output-format json |
  jq '{id, name, status, size}'
```

6.创建网络和安全性

当现有网络和安全组已满足批准的设计时，请使用它们。否则，请如图所示创建资源。

6.1创建或确定网络

在创建新网络之前列出现有网络。

```
stackit network list \
  --project-id "$PROJECT_ID" \
  --region "$REGION"
```

如果不存在合适的网络，请创建管理或传输网络以及服务或数据网络。

```
MGMT_NETWORK_RESPONSE="$(
  stackit network create \
    --project-id "$PROJECT_ID" \
    --region "$REGION" \
    --name "$MGMT_NETWORK_NAME" \
    --ipv4-prefix "$MGMT_IPV4_PREFIX" \
    --output-format json
)"

DATA_NETWORK_RESPONSE="$(
  stackit network create \
    --project-id "$PROJECT_ID" \
    --region "$REGION" \
    --name "$DATA_NETWORK_NAME" \
    --ipv4-prefix "$DATA_IPV4_PREFIX" \
    --output-format json
)"

export MGMT_NETWORK_ID="$(printf '%s' "$MGMT_NETWORK_RESPONSE" | jq -r '.id')"
export DATA_NETWORK_ID="$(printf '%s' "$DATA_NETWORK_RESPONSE" | jq -r '.id')"
printf 'MGMT_NETWORK_ID=%s\nDATA_NETWORK_ID=%s\n' "$MGMT_NETWORK_ID" "$DATA_NETWORK_ID"
```

如果获批网络已存在，请将MGMT_NETWORK_ID和DATA_NETWORK_ID设置为它们的ID。

6.2创建管理安全组

为管理NIC创建有状态安全组。

```
SECURITY_GROUP_RESPONSE="$(
  stackit security-group create \
    --project-id "$PROJECT_ID" \
    --region "$REGION" \
    --name "${SERVER_NAME}-mgmt" \
    --description "Management access for ${SERVER_NAME}" \
    --stateful \
    --output-format json
)"

export MGMT_SECURITY_GROUP_ID="$(printf '%s' "$SECURITY_GROUP_RESPONSE" | jq -r '.id')"
printf 'MGMT_SECURITY_GROUP_ID=%s\n' "$MGMT_SECURITY_GROUP_ID"
```

允许来自已批准的源CIDR的SSH。

```
stackit security-group rule create \
  --project-id "$PROJECT_ID" \
  --region "$REGION" \
  --security-group-id "$MGMT_SECURITY_GROUP_ID" \
  --direction ingress \
  --ether-type IPv4 \
  --protocol-name tcp \
  --port-range-min 22 \
  --port-range-max 22 \
  --ip-range "$TRUSTED_SOURCE_CIDR" \
  --description "SSH from approved source"
```

仅在部署需要时，才添加HTTPS、NETCONF、ICMP或思科SD-WAN控制和数据平面规则。将每个入口规则限制在最窄的实际源上。

7.创建有序的NIC

在创建服务器之前，先创建NIC。将管理安全组连接到NIC 1，然后按照预期的C8000V接口顺序提交生成的NIC ID。

7.1创建网卡1和网卡2

```
MGMT_NIC_RESPONSE="$(
  stackit network-interface create \
    --project-id "$PROJECT_ID" \
    --region "$REGION" \
    --network-id "$MGMT_NETWORK_ID" \
    --name "${SERVER_NAME}-mgmt" \
    --nic-security \
    --security-groups "$MGMT_SECURITY_GROUP_ID" \
    --output-format json
)"

DATA_NIC_RESPONSE="$(
  stackit network-interface create \
    --project-id "$PROJECT_ID" \
    --region "$REGION" \
    --network-id "$DATA_NETWORK_ID" \
    --name "${SERVER_NAME}-data" \
    --nic-security \
    --output-format json
)"

export MGMT_NIC_ID="$(printf '%s' "$MGMT_NIC_RESPONSE" | jq -r '.id')"
export DATA_NIC_ID="$(printf '%s' "$DATA_NIC_RESPONSE" | jq -r '.id')"
printf 'MGMT_NIC_ID=%s\nDATA_NIC_ID=%s\n' "$MGMT_NIC_ID" "$DATA_NIC_ID"
```

server-create示例首先为GigabitEthernet1提交MGMT_NIC_ID，其次为GigabitEthernet2提交DATA_NIC_ID。其他NIC ID按照提交的顺序映射到后续IOS XE接口。在服务器启动后验证来自IOS XE的结果映射。记录每个NIC的NIC ID、网络ID、MAC地址、分配的IP地址和预期的IOS XE接口。

7.2关联可选公有IP

公共IP与特定NIC关联。如果需要外部管理，请将其与NIC 1关联。

当通过专用路径提供管理访问时，省略此步骤。

```
PUBLIC_IP_RESPONSE="$(
  stackit public-ip create \
    --project-id "$PROJECT_ID" \
    --region "$REGION" \
    --associated-resource-id "$MGMT_NIC_ID" \
    --output-format json
)"

export PUBLIC_IP_ID="$(printf '%s' "$PUBLIC_IP_RESPONSE" | jq -r '.id')"
printf '%s\n' "$PUBLIC_IP_RESPONSE" | jq '{id, ip, associatedResourceId}'
```

8.准备第0天配置

8.1选择Bootstrap文件

对于自主模式，使用每行一个IOS XE配置命令创建iosxe_config.txt。

对于控制器模式，在Cisco SD-WAN Manager中生成Controller-mode Cloud-Init bootstrap，并保留文件名ciscosdwan_cloud_init.cfg。

对于SD-Routing模式，在Cisco SD-WAN Manager中生成SD-Routing-mode Cloud-Init bootstrap，并保留文件名ciscosdwan_cloud_init.cfg。

请勿手动编辑Cisco SD-WAN Manager生成的bootstrap。生成的文件包含设备身份和入职信息，必须作为带有凭证的项目加以保护。

8.2自主模式示例

```
hostname <ROUTER_HOSTNAME>
!
ip domain name <DOMAIN_NAME>
!
username <ADMIN_USER> privilege 15 secret <ADMIN_SECRET>
enable secret <ENABLE_SECRET>
!
interface GigabitEthernet1
 description Management
 ip address dhcp
 no shutdown
!
interface GigabitEthernet2
 description Service
 no ip address
 no shutdown
!
crypto key generate rsa modulus 2048
ip ssh version 2
!
line vty 0 4
 login local
 transport input ssh
!
end
```

设置所选引导程序的路径并限制对文件的访问。

```
export BOOTSTRAP_FILE="/path/to/<BOOTSTRAP_FILE>"
chmod 600 "$BOOTSTRAP_FILE"
```

8.3对引导程序编码

```
export USER_DATA_B64="$(base64 < "$BOOTSTRAP_FILE" | tr -d '\r\n')"
```

9.创建C8000V实例

9.1生成服务器创建请求

特定于C8000V的设置是configDrive:true，是userData中的base64编码引导程序、基于映像的引导卷以及有序的NIC ID。

```
jq -n \
```

```
  --arg name "$SERVER_NAME" \
  --arg machineType "$MACHINE_TYPE" \
  --arg availabilityZone "$AVAILABILITY_ZONE" \
  --arg bootVolumeId "$BOOT_VOLUME_ID" \
  --arg mgmtNicId "$MGMT_NIC_ID" \
  --arg dataNicId "$DATA_NIC_ID" \
  --arg userData "$USER_DATA_B64" \
  '{
    name: $name,
    machineType: $machineType,
    availabilityZone: $availabilityZone,
    configDrive: true,
    userData: $userData,
    bootVolume: {
      source: {
        type: "volume",
        id: $bootVolumeId
      }
    },
    networking: {
      nicIds: [$mgmtNicId, $dataNicId]
    },
    labels: {
      product: "cisco-c8000v"
    }
  }' > c8000v-server.json
```

查看请求，但不显示引导程序：

```
jq 'del(.userData)' c8000v-server.json
```

9.2创建服务器

使用针对目标环境记录的当前STACKIT IaaS v2终端。根据需要替换STACKIT API URL。

```
SERVER_RESPONSE="$(
  stackit curl \
    -X POST \
    "https://<STACKIT API URL>/v2/projects/${PROJECT_ID}/regions/${REGION}/servers" \
    -H "Content-Type: application/json" \
    --data @c8000v-server.json \
    --fail
)"

export SERVER_ID="$(printf '%s' "$SERVER_RESPONSE" | jq -r '.id')"
printf '%s\n' "$SERVER_RESPONSE" | jq '{id, name, status, configDrive}'
```

响应示例：

```
{
  "id": "<SERVER_ID>",
  "name": "<C8000V_NAME>",
  "status": "CREATING",
  "configDrive": true
}
```

轮询直到服务器达到预期的运行状态。

```
stackit server describe "$SERVER_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION" \
  --output-format json |
  jq '{id, name, status, powerStatus}'
```

10.验证部署

10.1检验STACKIT资源

确认：

- 服务器正在目标项目、区域和可用区域中运行。
- 已连接预期的启动卷。
- 已连接预期的网卡。
- 可选的公有IP与预定网卡关联。
- 应用的访问控制与已批准的设计相匹配。

10.2检验Cisco IOS XE

通过批准的管理路径进行连接，并运行与所选版本和模式相应的检查。

```
show version
show platform software vnic-if interface-mapping
show ip interface brief
show ip route
```

确认目标模式处于活动状态，已应用第0天配置，并且IOS XE接口与提交的NIC顺序匹配。

控制器模式的其他命令：

```
show sdwan control connections
show sdwan control local-properties
show sdwan system status
```

用于SD-Routing模式的其他命令：

```
show sd-routing control connections summary
show sd-routing control local-properties summary
show sd-routing system status
```

验证后，轮换临时调配凭据并删除本地请求文件和编码后的变量。

```
rm -f c8000v-server.json
unset USER_DATA_B64
```

11.删除部署

此部分中的cleanup命令永久删除资源。确认不再需要部署，并且每个资源ID仅属于此C8000V部署。请勿删除现有或共享网络、安全组、映像或其他资源。

清除示例会保留交互式确认提示。在批准操作之前，查看每个STACKIT CLI确认提示中显示的资源。

对于控制器或SD-Routing模式，请在删除STACKIT资源之前完成所有所需的控制器端停用。

11.1准备删除

检查记录的资源ID，然后继续。

```
printf 'SERVER_ID=%s\n' "$SERVER_ID"
printf 'PUBLIC_IP_ID=%s\n' "${PUBLIC_IP_ID:-not-created}"
printf 'MGMT_NIC_ID=%s\nDATA_NIC_ID=%s\n' "$MGMT_NIC_ID" "$DATA_NIC_ID"
printf 'BOOT_VOLUME_ID=%s\n' "$BOOT_VOLUME_ID"
printf 'MGMT_SECURITY_GROUP_ID=%s\n' "$MGMT_SECURITY_GROUP_ID"
```

```
printf 'MGMT_NETWORK_ID=%s\nDATA_NETWORK_ID=%s\n' "$MGMT_NETWORK_ID" "$DATA_NETWORK_ID"
printf 'IMAGE_ID=%s\n' "$IMAGE_ID"
```

如果部署具有公共IP，请在删除服务器之前将其分离。

```
if [ -n "${PUBLIC_IP_ID:-}" ]; then
  stackit server public-ip detach "$PUBLIC_IP_ID" \
    --server-id "$SERVER_ID" \
    --project-id "$PROJECT_ID" \
    --region "$REGION"
fi
```

11.2删除部署资源

首先删除服务器。在服务器列表中不再出现服务器后继续。

```
stackit server delete "$SERVER_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION"

stackit server list \
  --project-id "$PROJECT_ID" \
  --region "$REGION"
```

删除可选的公共IP、两个有序的NIC和引导卷。

```
if [ -n "${PUBLIC_IP_ID:-}" ]; then
  stackit public-ip delete "$PUBLIC_IP_ID" \
    --project-id "$PROJECT_ID" \
    --region "$REGION"
fi

stackit network-interface delete "$MGMT_NIC_ID" \
  --network-id "$MGMT_NETWORK_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION"

stackit network-interface delete "$DATA_NIC_ID" \
  --network-id "$DATA_NETWORK_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION"

stackit volume delete "$BOOT_VOLUME_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION"
```

如果仅为此部署创建了管理安全组和网络，请在它们的从属资源消失后将其删除。

```
stackit security-group delete "$MGMT_SECURITY_GROUP_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION"

stackit network delete "$MGMT_NETWORK_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION"

stackit network delete "$DATA_NETWORK_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION"
```

当自定义映像可用于其他C8000V部署时，请保留该映像。否则，请在删除所有从属启动卷后将其删除。

```
stackit image delete "$IMAGE_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION"
```

11.3检验清除

列出其余资源并确认删除的ID不再存在。故意保留的共享资源仍然可以显示。

```
stackit server list --project-id "$PROJECT_ID" --region "$REGION"
stackit public-ip list --project-id "$PROJECT_ID" --region "$REGION"
stackit volume list --project-id "$PROJECT_ID" --region "$REGION"
stackit security-group list --project-id "$PROJECT_ID" --region "$REGION"
stackit network list --project-id "$PROJECT_ID" --region "$REGION"
stackit image list --project-id "$PROJECT_ID" --region "$REGION"
```

确认在部署后保留的所有本地引导程序对象都已根据组织的凭证保留策略进行处理。

Appendix

有用的STACK命令

```
# List servers
```

```
stackit server list --project-id "$PROJECT_ID" --region "$REGION"

# List volumes
stackit volume list --project-id "$PROJECT_ID" --region "$REGION"

# List custom images
stackit image list --project-id "$PROJECT_ID" --region "$REGION"

# List network interfaces
stackit network-interface list --project-id "$PROJECT_ID" --region "$REGION"
```

有用的C8000V命令

```
show version
show platform software vnic-if interface-mapping
show ip interface brief
show ip route
show logging
```

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）。