STACKIT 클라우드에 Cisco Catalyst 8000V 설치 및 구성

목차

소개

이 문서에서는 Schwarz Digits에 속하는 STACKIT 클라우드에 Cisco Catalyst 8000V를 설치하고 가동하는 데 필요한 단계를 설명합니다.

배경

Cisco Catalyst 8000V는 Cisco IOS XE를 실행하는 가상 라우터입니다. STACKIT에서는 C8000V 이미지가 사용자 지정 이미지로 업로드되고 가상 서버의 부팅 볼륨을 만드는 데 사용됩니다.

서버를 생성하는 동안 STACKIT는 제공된 사용자 데이터를 구성 드라이브에 저장합니다. C8000V는 초기 부팅 중에 부트스트랩을 읽습니다. 부트스트랩 콘텐츠는 의도된 운영 모드에 따라 달라집니다.

- 자동 모드에서는 iosxe_config.txt를 사용합니다.
- 컨트롤러 모드에서는 Cisco SD-WAN Manager에서 생성한 ciscosdwan_cloud_init.cfg를 사용합니다.
- SD 라우팅 모드에서는 SD 라우팅을 위해 Cisco SD-WAN Manager에서 생성한 ciscosdwan_cloud_init.cfg를 사용합니다.

세 가지 모드 모두 동일한 이미지, 볼륨, 인터페이스 및 서버 생성 시퀀스가 사용됩니다. 부트스트랩 콘텐츠 및 구축 후 확인은 모드마다 다릅니다.

사전 요구 사항

요구 사항

- 이미지, 볼륨, 네트워크 인터페이스 및 서버를 관리할 권한이 있는 기존 STACKIT 프로젝트 및 인증된 운영자
- Bash 호환 셸이 있는 Linux 또는 macOS 워크스테이션. 이 설명서의 명령 예는 Linux 및 macOS를 위한 것입니다.
- 승인된 네트워크 접두사, 보안 제어, 관리 액세스 및 STACKIT 설계를 따르는 선택적 공용 IP
- 선택한 Cisco IOS XE 릴리스의 공인 C8000V QCOW2 이미지.
- 선택한 릴리스 및 기능 집합에 대해 게시된 C8000V CPU 및 메모리 요구 사항을 충족하는 STACKIT 시스템 유형.
- 선택한 C8000V 이미지에 적합한 부팅 볼륨 크기 및 펌웨어 설정.
- 소스 이미지를 검사하는 데 사용된 qemu-img 유틸리티를 포함하여 QEMU가 설치되었습니다 .
- stackit, jq 및 base64는 명령 환경에서 사용할 수 있습니다.
- 자동, 컨트롤러 또는 SD 라우팅 모드를 위해 준비된 Day-0 부트스트랩.

부트스트랩에는 비밀번호, 인증서 또는 디바이스 ID 정보가 포함될 수 있습니다. 보호된 위치에 저장하십시오. Base64 인코딩은 파일을 암호화하지 않습니다.

검증된 소프트웨어 릴리스 및 VM 유형

이 문서의 구축 워크플로는 다음 소프트웨어 릴리스 및 STACKIT VM 버전으로 검증되었습니다. 이 테스트된 베이스라인은 랩 환경에 대한 문서입니다. 현재의 Cisco Catalyst 8000V 릴리스 노트, 플랫폼 요구 사항 또는 기능별 크기 조정 지침을 대체하지는 않습니다.

소프트웨어 릴리스

- Cisco IOS XE 릴리스: 17.18.5
- Cisco SD-WAN 컨트롤러 릴리스: 20.18.5, 컨트롤러(Cisco SD-WAN) 모드에 사용

스택형 VM 유형

다음 STACKIT VM 유형이 검증에 포함되었습니다.

인스턴스 크기	vCPU	메모리(GiB)
c3i.4	4	8
c3i.8	8	16
c3i.16	16	32

의도한 C8000V 구축의 CPU, 메모리, 처리량 및 기능 요구 사항을 충족하는 버전을 선택합니다.

게스트가 보고한 하이퍼바이저 세부사항

검증된 C8000V 게스트가 다음과 같이 STACKIT 가상화 환경을 보고했습니다.

<#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#
```

구성

1. STACKIT CLI 설치

이 설명서에서는 Homebrew를 사용하여 STACKIT CLI를 설치합니다. 다른 패키지 관리자와 설치 방법을 사용할 수 있습니다. 자세한 내용은 STACKIT CLI 설치 가이드를 참조하십시오.

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

2. 환경 설정

구축 명령을 실행하기 전에 STACKIT CLI 환경을 구성합니다.

2.1 로그인 및 인증

STACKIT 계정에 로그인합니다.

```
stackit auth login
```

브라우저에서 인증을 완료합니다. 인증이 성공하면 CLI에서 로그인되었음을 확인합니다.

2.2 프로젝트 설정

나머지 명령에 대한 기본 프로젝트를 설정합니다.

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

프로젝트 ID를 모르는 경우 사용 가능한 프로젝트를 나열합니다.

```
stackit project list
```

3. 구축 입력 준비

구축에 필요한 값을 수집합니다. 이 예에서는 2개의 네트워크 및 2개의 NIC를 생성합니다. 첫 번째는 관리 또는 전송, 두 번째는 서비스 또는 데이터 트래픽입니다.

```
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>"
```

승인된 네트워크 설계에서 겹치지 않는 접두사를 사용합니다. 접두사 길이를 포함하여 CIDR 표기법으로 TRUSTED_SOURCE_CIDR을 지정합니다. 1개의 IPv4 주소를 허용하려면 /32를 사용합니다. 아래의 관리 규칙 예에서는 TRUSTED_SOURCE_CIDR의 SSH만 허용합니다.

4. C8000V 이미지 업로드

4.1 소스 이미지 검사

해당 파일이 의도한 릴리스에 대해 승인된 이미지인지 확인합니다. 구축 레코드에 파일 이름과 체크섬을 기록합니다.

```
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 STACKIT 사용자 지정 이미지 생성

EFI 이미지의 경우 IMAGE_UEFI를 true로 설정하고 BIOS 이미지의 경우 false로 설정합니다. 다운로드한 C8000V 이미지에 대해 지정된 설정을 선택합니다. 선택한 이미지와 현재 제품 설명서에 따라 최소 디스크 크기를 설정합니다.

```
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 이미지를 사용할 수 있을 때까지 폴링

```
stackit image describe "$IMAGE_ID" \
  --project-id "$PROJECT_ID" \
  --region "$REGION" \
  --output-format json |
  jq '{id, name, status, diskFormat}'
```

이미지 상태가 AVAILABLE(사용 가능)일 때 계속합니다.

5. 부팅 볼륨을 만듭니다.

사용자 지정 이미지에서 새 부팅 볼륨을 만듭니다. 볼륨 크기는 선택한 이미지의 디스크 요구 사항

을 충족해야 합니다.

```
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"
```

상태가 사용 가능이 될 때까지 볼륨을 폴링합니다.

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

6. 네트워크 및 보안 생성

기존 네트워크 및 보안 그룹이 승인된 설계를 이미 충족한 경우 이를 사용합니다. 그렇지 않으면 아래 표시된 대로 리소스를 생성합니다.

6.1 네트워크 생성 또는 식별

새 네트워크를 생성하기 전에 기존 네트워크를 나열합니다.

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

적합한 네트워크가 존재하지 않는 경우 관리 또는 전송 네트워크와 서비스 또는 데이터 네트워크를 생성합니다.

```
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"
```

승인된 네트워크가 이미 있는 경우 MGMT_NETWORK_ID 및 DATA_NETWORK_ID를 대신 ID로 설정합니다.

6.2 관리 보안 그룹 생성

관리 NIC에 대한 스테이트풀 보안 그룹을 생성합니다.

```
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"
```

승인된 소스 CIDR에서 SSH를 허용합니다.

```
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"
```

구축에 필요한 경우에만 HTTPS, NETCONF, ICMP 또는 Cisco SD-WAN 제어 및 데이터 플레인 규칙을 추가합니다. 각 인그레스 규칙을 가장 좁은 실제 소스로 제한합니다.

7. 순서가 지정된 NIC 생성

서버를 생성하기 전에 NIC를 생성합니다. 관리 보안 그룹을 NIC 1에 연결하고 원하는 C8000V 인터페이스 순서로 결과 NIC ID를 제출합니다.

7.1 NIC 1 및 NIC 2 생성

```
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 예에서는 GigabitEthernet1에 대해 먼저 MGMT_NIC_ID를 제출하고 GigabitEthernet2에 대해 DATA_NIC_ID를 두 번째 제출합니다. 추가 NIC ID는 제출된 순서대로 후속 IOS XE 인터페이스에 매핑됩니다. 서버가 부팅된 후 IOS XE에서 결과 매핑을 확인합니다.

각 NIC에 대해 NIC ID, 네트워크 ID, MAC 주소, 할당된 IP 주소, 원하는 IOS XE 인터페이스를 기록합니다.

7.2 선택적 공용 IP 연결

공용 IP는 특정 NIC와 연결됩니다. 외부 관리가 필요한 경우 NIC 1과 연결합니다.

프라이빗 경로를 통해 관리 액세스가 제공되는 경우 이 단계를 생략합니다.

```
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. Day-0 컨피그레이션 준비

8.1 부트스트랩 파일을 선택합니다

Autonomous(자동) 모드의 경우 한 행당 IOS XE 컨피그레이션 명령을 하나씩 사용하여 iosxe_config.txt를 생성합니다.

컨트롤러 모드의 경우 Cisco SD-WAN Manager에서 Controller-mode Cloud-Init 부트스트랩을 생성하고 ciscosdwan_cloud_init.cfg 파일 이름을 유지합니다.

SD-Routing 모드의 경우 Cisco SD-WAN Manager에서 SD-Routing-mode Cloud-Init 부트스트랩을 생성하고 ciscosdwan_cloud_init.cfg라는 파일 이름을 유지합니다.

Cisco SD-WAN Manager에서 생성된 부트스트랩을 수동으로 편집하지 마십시오. 생성된 파일에는 디바이스 ID 및 온보딩 정보가 포함되어 있으며, 자격 증명이 포함된 아티팩트로 보호해야 합니다.

8.2 자동 모드 예

```
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
```

선택한 부트스트랩에 대한 경로를 설정하고 파일에 대한 액세스를 제한합니다.

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

8.3 부트스트랩 인코딩

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

9. C8000V 인스턴스 생성

9.1 서버 만들기 요청 작성

C8000V 관련 설정은 configDrive입니다. true이면 userData의 base64 인코딩 부트스트랩, 이미지 기반 부트 볼륨 및 순서가 지정된 NIC ID입니다.

```
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
```

부트스트랩을 표시하지 않고 요청을 검토합니다.

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

9.2 서버 만들기

대상 환경에 대해 문서화된 현재 STACKIT IaaS v2 엔드포인트를 사용합니다. 필요에 따라 STACKIT API URL을 교체합니다.

```
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}'
```

응답 예:

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

서버가 예상 실행 상태에 도달할 때까지 폴링합니다.

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

10. 배포 확인

10.1 STACKIT 리소스 확인

다음을 확인합니다.

- 서버가 대상 프로젝트, 지역 및 가용 영역에서 실행되고 있습니다.
- 의도한 부팅 볼륨이 연결되어 있습니다.
- 예상 NIC가 연결됩니다.
- 선택 사항인 공용 IP는 해당 NIC와 연결됩니다. 및
- 적용된 액세스 제어는 승인된 설계와 일치합니다.

10.2 Cisco IOS XE 확인

승인된 관리 경로를 통해 연결하고 선택한 릴리스 및 모드에 적합한 검사를 실행합니다.

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

의도한 모드가 활성 상태이고 Day-0 컨피그레이션이 적용되었으며 IOS XE 인터페이스가 제출된 NIC 순서와 일치하는지 확인합니다.

컨트롤러 모드의 추가 명령:

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

SD 라우팅 모드의 추가 명령:

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

확인 후 임시 프로비저닝 자격 증명을 회전하고 로컬 요청 파일과 인코딩된 변수를 제거합니다.

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

11. 배포를 제거합니다.

이 섹션의 cleanup 명령은 리소스를 영구적으로 삭제합니다. 더 이상 구축이 필요하지 않으며 각 리소스 ID가 이 C8000V 구축에만 속하는지 확인합니다. 기존 또는 공유 네트워크, 보안 그룹, 이미지 또는 기타 리소스를 삭제하지 마십시오.

정리 예에서는 대화형 확인 프롬프트가 유지됩니다. 작업을 승인하기 전에 각 STACKIT CLI 확인 프롬프트에 표시된 리소스를 검토합니다.

컨트롤러 또는 SD 라우팅 모드의 경우 STACKIT 리소스를 삭제하기 전에 필요한 모든 컨트롤러측 서비스 해제를 완료합니다.

11.1 제거 준비

계속하기 전에 기록된 리소스 ID를 검토합니다.

```
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"
```

배포에 공용 IP가 있는 경우 서버를 삭제하기 전에 이를 분리하십시오.

```
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 배포 리소스 삭제

먼저 서버를 삭제합니다. 서버가 더 이상 서버 목록에 나타나지 않으면 계속합니다.

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

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

선택적 공용 IP, 순서가 지정된 2개의 NIC 및 부팅 볼륨을 삭제합니다.

```
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"
```

이 구축에 대해서만 관리 보안 그룹 및 네트워크를 생성한 경우 종속 리소스가 사라진 후 삭제합니다.

```
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"
```

다른 C8000V 구축에 사용할 때 사용자 지정 이미지를 유지합니다. 그렇지 않으면 모든 종속 부팅 볼륨을 제거한 후 삭제합니다.

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

11.3 정리 확인

나머지 리소스를 나열하고 삭제된 ID가 더 이상 존재하지 않는지 확인합니다. 의도적으로 보존된 공유 리소스는 계속 표시됩니다.

```
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"
```

구축 후 보존된 로컬 부트스트랩 객체가 조직의 자격 증명 보존 정책에 따라 처리되었는지 확인합니다.

부록

유용한 STACKIT 명령

```
# 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"
```

유용한 C8000V 명령

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

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.