阿里云
专有网络VPC

SDK 参考

文档版本：20190418

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的”现状“、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含”阿里云”、Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **禁止:** 重置操作将丢失用户配置数据。
	该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告:** 重启操作将导致业务中断，恢复业务所需时间约10分钟。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明:** 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 确定。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
*## *	表示参数、变量。	`bae log list --instanceid` *`Instance_ID`*
[]或者[a|b]	表示可选项，至多选择一个。	`ipconfig` *`[-all|-t]`*
{}或者{a|b}	表示必选项，至多选择一个。	`swich` *`{stand | slave}`*

目录

1 SDK开发指南

专有网络VPC支持Java、Python、Go、.NET和PHP SDK开发。

下表列举了各语言SDK的GitHub下载地址和开发指南，更多SDK的信息，请访问阿里云SDK平台。

SDK	GitHub下载地址	说明文档
Java SDK	VPC Java SDK	快速开始
Python SDK	VPC Python SDK	快速开始
Go SDK	VPC Go SDK	快速开始
.NET SDK	VPC .NET SDK	快速开始
PHP SDK	VPC PHP SDK	快速开始

2 Python SDK 示例

2.1 安装Alibaba Cloud SDK for Python

本文为您介绍如何在本地搭建可以运行专有网络Python SDK示例的Python开发环境，Alibaba Cloud SDK for Python支持Python 2.7，要运行专有网络的Python SDK示例，您需要安装Alibaba Cloud SDK for Python的核心库和VPC Python SDK。

操作步骤

1. 安装Python。
   a) 登录Python下载地址。
   b) 下载2.7版本的Python。

      在下载列表中选择平台安装包，包格式为：*python-XYZ.msi* 文件， XYZ 为您要安装的版本号。
   c) 下载后，双击下载包，进入Python安装向导，使用默认设置即可。
   d) 设置环境变量。

      在Path行添加python安装路径和pip命令运行程序所在的目录，目录之间以;分隔。

      ```
      C:\Python27;C:\Python27\Scripts
      ```
   e) 在cmd命令行，执行python命令，显示类似如下，进入Python交互式环境，表示Python安装成功。

      ```
      Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:30:26) [MSC v.
      1500 64 bit (AM
      D64)] on win32
      Type "help", "copyright", "credits" or "license" for more
      information.
      >>>
      ```
2. 执行如下命令，安装Alibaba Cloud SDK for Python核心库。

   ```
   pip install aliyun-python-sdk-core
   ```

   关于Python及PIP的使用说明，请参见Python文档。

3. 执行如下命令，安装VPC SDK。

```
pip install aliyun-python-sdk-vpc
```

4. 执行如下命令，使用--user安装SDK。

```
pip install --user aliyun-python-sdk-cms
```

2.2 准备工作

在运行专有网络场景功能的SDK文件前，您需要完成公共配置。

在开始运行示例脚本前，确保您已完成以下准备工作：

1. 确保您的账户金额不少于100元。
2. 获取AccessKey。

   确保您已经有了用于身份验证的AccessKey ID和AccessKey Secret。详细说明，请参见创建*AccessKey*。
3. 下载阿里云专有网络python SDK场景示例的*VPC Python Example*库。
4. 进入*setup.py*所在的目录，执行如下命令，完成环境初始化配置。

```
python setup.py install --user
```

2.3 专有网络（VPC）

2.3.1 创建和删除VPC/VSwitch

本文介绍如何使用Python SDK创建和删除VPC/VSwitch。

背景信息

在华北3张家口地域创建一个VPC，并在该VPC下创建一个VSwitch。VPC和VSwitch创建成功后，删除VPC和VSwitch。

操作步骤

1. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\sdk_lib*文件夹。
2. 使用编辑器打开*consts.py*文件，配置用户鉴权参数ACS_CLIENT。
3. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\examples\vpc*文件夹。

4. 使用编辑器打开*vpc_quick_start.py*文件，根据实际情况配置相关参数，保存退出。

```
#encoding=utf-8
import sys
import json

from aliyunsdkcore.acs_exception.exceptions import ServerException,
ClientException
from aliyunsdkvpc.request.v20160428 import CreateVpcRequest
from aliyunsdkvpc.request.v20160428 import CreateVSwitchRequest
from aliyunsdkvpc.request.v20160428 import DeleteVSwitchRequest
from aliyunsdkvpc.request.v20160428 import DeleteVpcRequest
from aliyunsdkvpc.request.v20160428 import DescribeVSwitchAttri
butesRequest
from aliyunsdkvpc.request.v20160428 import DescribeVpcAttribute
Request
from sdk_lib.exception import ExceptionHandler
from sdk_lib.check_status import CheckStatus
from sdk_lib.consts import *
from sdk_lib.common_util import CommonUtil

class VpcQuickStart(object):
    def __init__(self, client):
        self.client = client

    def create_vpc(self):
        """
        create_vpc: 创建VPC
        官网API参考链接: https://help.aliyun.com/document_detail/35737
.html
        """
        try:
            request = CreateVpcRequest.CreateVpcRequest()
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            # 判断VPC状态是否可用
            if CheckStatus.check_status(TIME_DEFAULT_OUT, DEFAULT_TI
ME,
                                        self.describe_vpc_status,
                                        AVAILABLE, response_json['
VpcId']):
                return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def delete_vpc(self, params):
        """
        delete_vpc: 删除VPC
        官网API参考链接: https://help.aliyun.com/document_detail/35738
.html
        """
        try:
            request = DeleteVpcRequest.DeleteVpcRequest()
            # 要删除的VPC的ID
            request.set_VpcId(params['vpc_id'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
```

```
            ExceptionHandler.client_exception(e)

    def describe_vpc_attribute(self, vpc_id):
        """
        describe_vpc_attribute: 查询指定地域已创建的vpc信息
        官网API参考: https://help.aliyun.com/document_detail/94565.
html
        """
        try:
            request = DescribeVpcAttributeRequest.DescribeVp
cAttributeRequest()
            # 要查询的VPC ID
            request.set_VpcId(vpc_id)
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def describe_vpc_status(self, vpc_id):
        """
        describe_vpc_status: 查询指定地域已创建的vpc的状态
        官网API参考: https://help.aliyun.com/document_detail/94565.
html
        """
        response = self.describe_vpc_attribute(vpc_id)
        return response["Status"]

    def create_vswitch(self, params):
        """
        create_vswitch: 创建vswitch实例
        官网API参考: https://help.aliyun.com/document_detail/35745.
html
        """
        try:
            request = CreateVSwitchRequest.CreateVSwitchRequest()
            # 交换机所属区的ID，您可以通过调用DescribeZones接口获取地域ID
            request.set_ZoneId(params['zone_id'])
            # 交换机所属的VPC ID
            request.set_VpcId(params['vpc_id'])
            # 交换机的网段
            request.set_CidrBlock(params['cidr_block'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            # 判断VSwitch状态是否可用
            if CheckStatus.check_status(TIME_DEFAULT_OUT, DEFAULT_TI
ME,
                                        self.describe_vswitch_status
,
                                        AVAILABLE, response_json['
VSwitchId']):
                return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def describe_vswitch_attribute(self, vswitch_id):
        """
        describe_vswitch_attribute: 查询指定地域已创建的vswitch的状态
        官网API参考: https://help.aliyun.com/document_detail/36010.
html
```

```
        """
        try:
            request = DescribeVSwitchAttributesRequest.DescribeVS
witchAttributesRequest()
            request.set_VSwitchId(vswitch_id)
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def describe_vswitch_status(self, vswitch_id):
        """
        describe_vswitch_status: 查询指定地域已创建的vswitch的状态
        官网API参考: https://help.aliyun.com/document_detail/36010.
html
        """
        response = self.describe_vswitch_attribute(vswitch_id)
        return response["Status"]

    def delete_vswitch(self, params):
        """
        delete_vswitch: 删除vswitch实例
        官网API参考: https://help.aliyun.com/document_detail/35746.
html
        """
        try:
            request = DeleteVSwitchRequest.DeleteVSwitchRequest()
            # 要删除的交换机的ID
            request.set_VSwitchId(params['vswitch_id'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            # 判断VSwitch是否被删除成功
            if CheckStatus.check_status(TIME_DEFAULT_OUT, DEFAULT_TI
ME * 5,
                                        self.describe_vswitch_status
,
                                        '', params['vswitch_id']):
                return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)


def main():
    client = ACS_CLIENT

    vpc_quick_start = VpcQuickStart(client)

    params = {}
    params['zone_id'] = "cn-zhangjiakou-b"
    params['cidr_block'] = "172.16.0.0/16"

    # 创建vpc
    vpc_json = vpc_quick_start.create_vpc()
    CommonUtil.log("create_vpc", vpc_json)

    # 创建vswitch
    params['vpc_id'] = vpc_json['VpcId']
    vswitch_json = vpc_quick_start.create_vswitch(params)
    CommonUtil.log("create_vswitch", vswitch_json)
```

```
    # 删除vswitch
    params['vswitch_id'] = vswitch_json['VSwitchId']
    vswitch_json = vpc_quick_start.delete_vswitch(params)
    CommonUtil.log("delete_vswitch", vswitch_json)

    # 删除vpc
    vpc_json = vpc_quick_start.delete_vpc(params)
    CommonUtil.log("delete_vpc", vpc_json)


if __name__ == "__main__":
    sys.exit(main())
```

5. 进入*vpc_quick_start.py*所在的目录，执行如下命令，运行创建和删除VPC/VSwitch示例。

```
python vpc_quick_start.py
```

系统显示类似如下：

```
----------------------------create_vpc---------------------------
{
  "ResourceGroupId": "rg-acfmxazxxxxxxxx",
  "RouteTableId": "vtb-8vbf9ud7xrcn9xxxxxxxx",
  "VRouterId": "vrt-8vb1qjnxcm03nxxxxxxxx",
  "VpcId": "vpc-8vb67v4ozd8wfxxxxxxxx",
  "RequestId": "5052F988-75CC-46AD-A1A6-0E9E445BD0D5"
}

----------------------------create_vswitch---------------------------
{
  "VSwitchId": "vsw-8vbqn2at0kljjxxxxxxxx",
  "RequestId": "0BA1ABF7-21CF-4460-9A86-0BB783886E58"
}

----------------------------delete_vswitch---------------------------
{
  "RequestId": "D691F04B-A6EE-49A7-A434-4A45DD3AA0B8"
}

----------------------------delete_vpc---------------------------
{
  "RequestId": "4570F816-AB8D-45EA-8913-6AE787C1632C"
}
```

2.3.2 创建自定义路由表

本文介绍如何使用Python SDK创建自定义路由表。

背景信息

本次示例分为以下几步，创建自定义路由表：

1. 在华北3张家口地域创建一个VPC。
2. 在新建的VPC下创建一个VSwitch。
3. 创建一个名为sdk_route_table的自定义路由表。

4. 查询新创建的VSwitch。
5. 将新创建的路由表与和同一VPC内的VSwitch进行绑定。
6. 将新创建的路由表与和同一VPC内的VSwitch进行解绑。
7. 删除新创建的自定义路由表。
8. 删除新创建的VSwitch。
9. 删除新创建的VPC。

操作步骤

1. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\sdk_lib*文件夹。
2. 使用编辑器打开*consts.py*文件，配置用户鉴权参数ACS_CLIENT。
3. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\examples\vpc*文件夹。
4. 使用编辑器打开*vpc_route_table.py*文件，根据实际情况配置相关参数，保存退出。

```
#encoding=utf-8
import sys
import json
import time

from aliyunsdkcore.acs_exception.exceptions import ServerException,
ClientException
from aliyunsdkvpc.request.v20160428 import CreateRouteEntryRequest
from aliyunsdkvpc.request.v20160428 import DeleteRouteEntryRequest
from aliyunsdkvpc.request.v20160428 import AssociateEipAddressR
equest
from aliyunsdkvpc.request.v20160428 import UnassociateEipAddres
sRequest
from aliyunsdkvpc.request.v20160428 import DescribeRouteTablesR
equest
from sdk_lib.common_util import CommonUtil
from sdk_lib.sdk_vpc import Vpc
from sdk_lib.sdk_vswitch import VSwitch
from sdk_lib.exception import ExceptionHandler
from sdk_lib.check_status import CheckStatus
from sdk_lib.consts import *

class RouteTable(object):
    def __init__(self, client):
        self.client = client


    def create_route_table(self, params):
        """
        create_route_table: 创建一个自定义路由表
        官网API参考链接: https://help.aliyun.com/document_detail/87586
.html
        """
        try:
            request = CreateRouteEntryRequest.CreateRouteEntryRequ
est()
```

```
            request.set_action_name("CreateRouteTable")
            # 自定义路由表所属的VPC ID
            request.add_query_param("VpcId", params['vpc_id'])
            # 路由表的名称
            request.add_query_param("RouteTableName", params['
route_table_name'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            route_table_id = response_json['RouteTableId']
            # 判断route table状态是否可用
            if CheckStatus.check_status(TIME_DEFAULT_OUT, DEFAULT_TI
ME * 5,
                                        self.describe_route_table
_status,
                                        route_table_id, route_tabl
e_id):
                return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def associate_route_table(self, params):
        """
        associate_route_table: 将创建的自定义路由表和同一VPC内的交换机绑定
        官网API参考链接: https://help.aliyun.com/document_detail/87599
.html
        """
        try:
            request = AssociateEipAddressRequest.AssociateE
ipAddressRequest()
            request.set_action_name("AssociateRouteTable")
            # 路由表ID
            request.add_query_param("RouteTableId", params['
route_table_id'])
            # 要绑定的交换机ID
            request.add_query_param("VSwitchId", params['vswitch_id
'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            time.sleep(DEFAULT_TIME * 5)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def unassociate_route_table(self, params):
        """
        unassociate_route_table: 将路由表和交换机解绑
        官网API参考链接: https://help.aliyun.com/document_detail/87628
.html
        """
        try:
            request = UnassociateEipAddressRequest.Unassociat
eEipAddressRequest()
            request.set_action_name("UnassociateRouteTable")
            # 路由表ID
            request.add_query_param("RouteTableId", params['
route_table_id'])
            # 要解绑的交换机ID
            request.add_query_param("VSwitchId", params['vswitch_id
'])
            response = self.client.do_action_with_exception(request)
```

```
            response_json = json.loads(response)
            time.sleep(DEFAULT_TIME * 5)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def delete_route_table(self, params):
        """
        delete_route_table: 删除自定义路由表
        官网API参考链接: https://help.aliyun.com/document_detail/87601
.html
        """
        try:
            request = DeleteRouteEntryRequest.DeleteRouteEntryRequ
est()
            request.set_action_name("DeleteRouteTable")
            # 路由表ID
            request.add_query_param("RouteTableId", params['
route_table_id'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            # 判断route table是否被删除成功
            if CheckStatus.check_status(TIME_DEFAULT_OUT, DEFAULT_TI
ME * 5,
                                        self.describe_route_table
_status,
                                        '', params['route_table_id
']):
                return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def describe_route_table(self, route_table_id):
        """
        describe_route_table: 查询路由表
        官网API参考链接: https://help.aliyun.com/document_detail/36014
.html
        """
        try:
            request = DescribeRouteTablesRequest.DescribeRo
uteTablesRequest()
            # 路由表的ID
            request.set_RouteTableId(route_table_id)
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def describe_route_table_status(self, route_table_id):
        """
        describe_route_table_status: 查询路由表状态
        """
        response = self.describe_route_table(route_table_id)
        if len(response["RouteTables"]["RouteTable"]) == 0:
            return ''
        return response["RouteTables"]["RouteTable"][0]["RouteTable
Id"]
```

```
def main():
    client = ACS_CLIENT

    vpc = Vpc(client)
    vswitch = VSwitch(client)
    route_table = RouteTable(client)

    params = {}
    params['cidr_block'] = "172.16.0.0/16"
    params['zone_id'] = "cn-zhangjiakou-b"
    params['route_table_name'] = "sdk_route_table"

    #创建vpc
    vpc_json = vpc.create_vpc()
    CommonUtil.log("create_vpc", vpc_json)

    #创建vswitch
    params['vpc_id'] = vpc_json['VpcId']
    vswitch_json = vswitch.create_vswitch(params)
    CommonUtil.log("create_vswitch", vswitch_json)

    #创建route table
    route_table_json = route_table.create_route_table(params)
    CommonUtil.log("create_route_table", route_table_json)

    #查询vswitch
    params['vswitch_id'] = vswitch_json['VSwitchId']
    vswitch_json = vswitch.describe_vswitch_attribute(params['
vswitch_id'])
    CommonUtil.log("describe_vswitch_attribute", vswitch_json)

    #route table绑定vswitch
    params['route_table_id'] = route_table_json['RouteTableId']
    associate_json = route_table.associate_route_table(params)
    CommonUtil.log("associate_route_table", associate_json)

    #route table解绑vswitch
    unassociate_json = route_table.unassociate_route_table(params)
    CommonUtil.log("unassociate_route_table", unassociate_json)

    #删除route table
    delete_route_table_json = route_table.delete_route_table(params)
    CommonUtil.log("delete_route_table", delete_route_table_json)

    #删除vswitch
    delete_vswitch_json = vswitch.delete_vswitch(params)
    CommonUtil.log("delete_vswitch", delete_vswitch_json)

    #删除vpc
    delete_vpc_json = vpc.delete_vpc(params)
    CommonUtil.log("delete_vpc", delete_vpc_json)

if __name__ == "__main__":
```

```
    sys.exit(main())
```

5. **进入*vpc_route_table.py*所在的目录，执行如下命令，运行创建自定义路由条目示例。**

```
python vpc_route_table.py
```

系统显示类似如下：

```
----------------------------create_vpc---------------------------
{
  "ResourceGroupId": "rg-acfmxazxxxxxxxx",
  "RouteTableId": "vtb-8vb65a5hqy8pcxxxxxxxx",
  "VRouterId": "vrt-8vbbbiftzizc3xxxxxxxx",
  "VpcId": "vpc-8vbebihln001gxxxxxxxx",
  "RequestId": "862F279B-4A27-4300-87A1-047FB9961AF2"
}

----------------------------create_vswitch---------------------------
{
  "VSwitchId": "vsw-8vb30klhn2is5xxxxxxxx",
  "RequestId": "1DA17173-CB61-4DCE-9C29-AABFDF3001A6"
}

----------------------------create_route_table
----------------------------
{
  "RouteTableId": "vtb-8vbc4iwpo13apxxxxxxxx",
  "RequestId": "01E66E67-7801-4705-A02A-853BA7EEA89F"
}

----------------------------describe_vswitch_attribute
----------------------------

{
  "Status": "Available",
  "NetworkAclId": "",
  "VpcId": "vpc-8vbebihln001gxxxxxxxx",
  "Description": "",
  "RouteTable": {
    "RouteTableId": "vtb-8vb65a5hqy8pcxxxxxxxx",
    "RouteTableType": "System"
  },
  "CidrBlock": "172.16.0.0/16",
  "CreationTime": "2019-04-12T03:08:43Z",
  "CloudResources": {
    "CloudResourceSetType": []
  },
  "ZoneId": "cn-zhangjiakou-b",
  "ResourceGroupId": "rg-acfmxazbxxxxxxxx",
  "VSwitchId": "vsw-8vb30klhn2is5xxxxxxxx",
  "RequestId": "C5A20BA3-E998-498D-8900-35AE5FDFFB77",
  "Ipv6CidrBlock": "",
  "VSwitchName": "",
  "AvailableIpAddressCount": 252,
  "IsDefault": false
}

----------------------------associate_route_table
----------------------------
{
  "RequestId": "5FC0143B-D34B-47DC-8D49-AFD222EA5876"
}
```

```
----------------------------unassociate_route_table
----------------------------
{
  "RequestId": "F0194718-6E4C-496C-9DA8-1B88DF1D6FAD"
}

---------------------------delete_route_table
---------------------------
{
  "RequestId": "B5C068A6-137C-4337-8E3A-9E30E1726703"
}

---------------------------delete_vswitch---------------------------
{
  "RequestId": "26DEDBF8-2F0D-4A13-8CB3-23A84C947704"
}

----------------------------delete_vpc----------------------------
{
  "RequestId": "E1B2641F-5911-40E4-9F36-CC0B2EDD1747"
}
```

2.3.3 创建自定义路由条目

本文介绍如何使用Python SDK在VPC路由器或边界路由器（VBR）上创建自定义路由条目。

前提条件

在创建自定义路由条目之前，您必须先完成以下操作：

- 在华北3张家口地域创建了VPC和VSwitch。
- 在华北3张家口地域创建了自定义路由的下一跳，下一跳类型可以为ECS实例、高可用虚拟IP、VPN网关、NAT网关、辅助弹性网卡或路由器接口。

背景信息

本次示例分为以下几步，创建自定义路由条目：

1. 在华北3张家口地域创建一个名为sdk_route_table的自定义路由表。
2. 查询专有网络vpc-8vb7ztbjqomi9xxxxxxxx下的VSwitch。
3. 将新创建的自定义路由表与交换机vsw-8vbfqpcijj0d1xxxxxxxx进行绑定。
4. 为新建的自定义路由表创建自定义路由条目，目的网络为168.168.0.0/16，下一跳类型为ECS实例，下一跳ID为i-8vbgsnt7046axxxxxxxx。
5. 删除新创建的路由条目。
6. 将新创建的自定义路由表与VSwitch解绑。
7. 删除新创建的自定义路由表。

操作步骤

1. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\sdk_lib*文件夹。

2. 使用编辑器打开*consts.py*文件，配置用户鉴权参数ACS_CLIENT。
3. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\examples\vpc*文件夹。
4. 使用编辑器打开*vpc_route_entry.py*文件，根据实际情况配置相关参数，保存退出。

```
#encoding=utf-8
import sys
import json
import time

from aliyunsdkcore.acs_exception.exceptions import ServerException,
ClientException
from aliyunsdkvpc.request.v20160428 import CreateRouteEntryRequest
from aliyunsdkvpc.request.v20160428 import DeleteRouteEntryRequest
from aliyunsdkvpc.request.v20160428 import DescribeRouteTablesR
equest
from sdk_lib.exception import ExceptionHandler
from sdk_lib.check_status import CheckStatus
from sdk_lib.common_util import CommonUtil
from sdk_lib.sdk_vswitch import VSwitch
from sdk_lib.sdk_route_table import RouteTable
from sdk_lib.consts import *

class RouteEntry(object):
    def __init__(self, client):
        self.client = client

    def create_route_entry(self, params):
        """
        create_route_entry: 创建route_entry路由条目
        官网API参考链接: https://help.aliyun.com/document_detail/36012
.html
        """
        try:
            request = CreateRouteEntryRequest.CreateRouteEntryRequ
est()
            # 路由表ID
            request.set_RouteTableId(params['route_table_id'])
            # 自定义路由条目的目标网段
            request.set_DestinationCidrBlock(params['destinatio
n_cidr_block'])
            # 下一跳的类型
            request.set_NextHopType(params['nexthop_type'])
            # 下一跳实例的ID
            request.set_NextHopId(params['nexthop_id'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            # 判断router entry状态是否可用
            if CheckStatus.check_status(TIME_DEFAULT_OUT, DEFAULT_TI
ME,
                                        self.describe_route_entry
_status,
                                        AVAILABLE, params['
route_table_id']):
                return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)
```

```
    def delete_route_entry(self, params):
        """
        delete_route_entry: 删除route_entry路由条目
        官网API参考链接: https://help.aliyun.com/document_detail/36013
.html
        """
        try:
            request = DeleteRouteEntryRequest.DeleteRouteEntryRequ
est()
            # 路由条目所在的路由表的ID
            request.set_RouteTableId(params['route_table_id'])
            # 路由条目的目标网段
            request.set_DestinationCidrBlock(params['destinatio
n_cidr_block'])
            # 下一跳实例的ID
            request.set_NextHopId(params['nexthop_id'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            time.sleep(DEFAULT_TIME)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def describe_route_entry_vrouter(self, params):
        """
        describe_route_entry_vrouter: 查询route_entry路由条目
        官网API参考链接: https://help.aliyun.com/document_detail/36014
.html
        """
        try:
            request = DescribeRouteTablesRequest.DescribeRo
uteTablesRequest()
            # 路由表所属的VPC路由器或边界路由器的ID
            request.set_VRouterId(params['vrouter_id'])
            # 路由表的ID
            request.set_RouteTableId(params['route_table_id'])
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)

    def describe_route_entry(self, route_table_id):
        """
        describe_route_entry: 查询route_entry路由条目
        官网API参考链接: https://help.aliyun.com/document_detail/36014
.html
        """
        try:
            request = DescribeRouteTablesRequest.DescribeRo
uteTablesRequest()
            request.set_RouteTableId(route_table_id)
            response = self.client.do_action_with_exception(request)
            response_json = json.loads(response)
            return response_json
        except ServerException as e:
            ExceptionHandler.server_exception(e)
        except ClientException as e:
            ExceptionHandler.client_exception(e)
```

```
    def describe_route_entry_status(self, route_table_id):
        """
        describe_route_entry_status: 查询route_entry路由条目当前状态
        """
        response = self.describe_route_entry(route_table_id)
        return response["RouteTables"]["RouteTable"][0]["RouteEntrys
"]["RouteEntry"][0]["Status"]

def main():
    client = ACS_CLIENT
    vswitch = VSwitch(client)
    route_table = RouteTable(client)
    route_entry = RouteEntry(client)

    params = {}
    params['route_table_name'] = "sdk_route_table"
    params['destination_cidr_block'] = "0.0.0.0/0"
    params['nexthop_id'] = "i-bp1e82xlhob2xxxxxxxx"
    params['nexthop_type'] = "Instance"

    params['vpc_id'] = "vpc-bp15opprpg0rgxxxxxxxx"
    params['vswitch_id'] = "vsw-bp1e67w26n2sjxxxxxxxx"

    #创建route table
    route_table_json = route_table.create_route_table(params)
    CommonUtil.log("create_route_table", route_table_json)

    #查询vswitch
    vswitch_json = vswitch.describe_vswitch_attribute(params)
    CommonUtil.log("describe_vswitch_attribute", vswitch_json)

    #route table绑定vswitch
    params['route_table_id'] = route_table_json['RouteTableId']
    associate_json = route_table.associate_route_table(params)
    CommonUtil.log("associate_route_table", associate_json)

    #创建路由条目
    create_route_entry_json = route_entry.create_route_entry(params)
    CommonUtil.log("create_route_entry", create_route_entry_json)

    #删除路由条目
    delete_route_entry_json = route_entry.delete_route_entry(params)
    CommonUtil.log("delete_route_entry", delete_route_entry_json)

    #route table解绑vswitch
    unassociate_json = route_table.unassociate_route_table(params)
    CommonUtil.log("unassociate_route_table", unassociate_json)

    #删除route table
    delete_route_table_json = route_table.delete_route_table(params)
    CommonUtil.log("delete_route_table", delete_route_table_json)


if __name__ == "__main__":
```

```
    sys.exit(main())
```

5. **进入*vpc_route_entry.py*所在的目录，执行如下命令，运行创建自定义路由条目示例。**

```
python vpc_route_entry.py
```

系统显示类似如下：

```
----------------------------create_route_table
----------------------------
{
  "RouteTableId": "vtb-8vbn7px9zxwr2xxxxxxxx",
  "RequestId": "8B351EE1-614F-44E4-93AF-1CADA4BF02E8"
}

----------------------------describe_vswitch_attribute
----------------------------

{
  "Status": "",
  "NetworkAclId": "",
  "VpcId": "",
  "Description": "",
  "Ipv6CidrBlock": "",
  "CreationTime": "",
  "CloudResources": {
    "CloudResourceSetType": []
  },
  "ZoneId": "",
  "ResourceGroupId": "",
  "VSwitchId": "",
  "RequestId": "5E199415-BBA3-443D-B1EC-06341FE267F4",
  "VSwitchName": "",
  "CidrBlock": ""
}

----------------------------associate_route_table
----------------------------
{
  "RequestId": "5F33E444-5CCD-4677-91AB-3E234A9A64E4"
}

----------------------------create_route_entry
----------------------------
{
  "RequestId": "D6035ECA-DD81-4FAB-B084-55BE60FB18ED"
}

----------------------------delete_route_entry
----------------------------
{
  "RequestId": "54108FD7-8609-4111-919D-B2983466F480"
}

----------------------------unassociate_route_table
----------------------------
{
  "RequestId": "0F36A76A-1E54-41DC-852E-1D970FDE8F3F"
}

----------------------------delete_route_table
----------------------------
```

```
{
  "RequestId": "F3151A59-4F90-4531-AFDC-B7B7CF70A8C1"
}
```