ALIBABA CLOUD

阿里云

负载均衡

SDK参考

文档版本：20200828

阿里云

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
危险	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。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会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，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	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。	单击**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。	在**结果确认**页面，单击**确定**。
`Courier字体`	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	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。	`switch {active|stand}`

目录

1.SDK下载 ---- 05
2.Python SDK示例 ---- 06
2.1. 安装阿里云Python SDK ---- 06
2.2. 准备工作 ---- 08
2.3. 功能示例 ---- 08
2.3.1. 创建和删除SLB实例 ---- 08
2.3.2. 创建TCP监听 ---- 16
2.3.3. 更新HTTPS证书 ---- 24
2.3.4. 查看监控 ---- 34
2.3.5. 克隆实例 ---- 44

1.SDK下载

负载均衡SLB支持Java、Python、Go和PHP开发。

下表列举了各语言SDK的下载地址和开发指南，更多SDK的信息，请参见阿里云开放平台。

Alibaba Cloud SDK	负载均衡SDK	说明文档
Alibaba Cloud SDK for Java	Alibaba Cloud SLB SDK for Java	快速开始
Alibaba Cloud SDK for Python	Alibaba Cloud SLB SDK for Python	快速开始
Alibaba Cloud SDK for Go	Alibaba Cloud SLB SDK for Go	快速开始
Alibaba Cloud SDK for PHP	Alibaba Cloud SLB SDK for PHP	快速开始

2.Python SDK示例

2.1. 安装阿里云Python SDK

本文介绍如何在本地搭建运行负载均衡Python SDK示例的Python开发环境。阿里云支持Python 2.7。要运行负载均衡的Python SDK示例，您需要安装阿里云Python SDK的核心库和SLB Python SDK。Python SDK支持Windows、Linux和Mac操作系统，所有负载均衡Python SDK示例均在Windows系统下运行。

操作步骤

1. 安装Python。
   i. 打开Python下载地址。
   ii. 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为需要安装的版本号。
   iii. 下载后，双击下载包，进入Python安装向导，使用默认设置即可。
   iv. 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，目录之间以分号（;）分隔。

   ```
   C:\Python27;C:\Python27\Scripts
   ```

   v. 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. 安装阿里云SDK核心库。执行如下命令，安装阿里云SDK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SLB SDK。

   ```
   pip install aliyun-python-sdk-slb
   ```

4. 需要结合阿里云云监控服务，查看负载均衡的云监控数据，如连接数、数据包数和流量等，执行如下命令，安装云监控SDK。

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

5. （可选）如果需要使用CloudShell运行SDK文档，执行如下命令，使用--user安装SDK并运行*setup.py*文件。

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

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

2.2. 准备工作

在运行SLB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. 因为使用Python SDK创建SLB实例会产生实例费，所以确保您的阿里云账户金额不少于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。如果尚未获取AK，参见查询获取获取AccessKey。
3. 下载阿里云负载均衡Python SDK场景示例的SLB 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
   ```

2.3. 功能示例

2.3.1. 创建和删除SLB实例

本文介绍如何使用Python SDK创建一个SLB实例，创建成功后删除该实例。

前提条件

在开始运行示例脚本前，您需要确保已完成以下准备工作：

1. 因为使用Python SDK创建SLB实例会产生实例费，所以确保您的阿里云账户金额不少于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。如果尚未获取AK，参见查询获取获取AccessKey。
3. 下载阿里云负载均衡Python SDK场景示例的SLB 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
   ```

背景信息

在华北3（张家口）地域创建一个名为SLB1的SLB实例，主可用区为cn-zhangjiakou-a，备可用区为cn-zhangjiakou-b，实例计费类型为按量计费，规格为slb.s1.small，监听的带宽峰值为1Mbps，其他参数使用默认值。实例创建成功后，删除该实例。

操作步骤

1. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\examples\slb*文件夹。
2. 使用编辑器打开*slb_quick_start.py*文件，您需要配置用户鉴权参数ACS_CLIENT，其他参数根据实际情况配置后，保存退出。

> 说明 因为用户鉴权是每个SDK必须执行的操作，所以可以在常量文件中配置AcsClient参数的值，在场景代码中调用常量文件。

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

# 调用AcsClient参数进行身份验证
from aliyunsdkcore.client import AcsClient
# 使用阿里云官方sdk的异常处理模块
from aliyunsdkcore.acs_exception.exceptions import ServerException, ClientException
# 调用上传证书的API
from aliyunsdkslb.request.v20140515 import UploadServerCertificateRequest
# 调用创建SLB实例API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerRequest
# 调用添加默认服务器组API
from aliyunsdkslb.request.v20140515 import AddBackendServersRequest
# 调用创建HTTPS监听的API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerHTTPSListenerRequest
# 调用修改HTTPS监听的API
from aliyunsdkslb.request.v20140515 import SetLoadBalancerHTTPSListenerAttributeRequest
# 调用删除SLB实例API
from aliyunsdkslb.request.v20140515 import DeleteLoadBalancerRequest
##异常处理逻辑
from sdk_lib.exception import ExceptionHandler
# 命令行导入日志
from sdk_lib.common_util import CommonUtil

# 用户身份鉴权参数配置
ACS_CLIENT = AcsClient(
'your-access-key-id', # your-access-key-id
'your-access-key-secret', # your-access-key-secret
'cn-zhangjiakou', # your-region-id
)
# 失败重试次数
TRY_TIME = 3
'''
创建slb实例->上传服务器证书->创建https监听->上传新的服务器证书
->循环修改https监听配置
'''
```

```
def create_load_balancer(params):
'''
create_load_balancer：创建slb实例
官网API参考链接：help.aliyun.com/document_detail/27577.html
'''
try:
# 设置创建SLB实例的调用参数
request = CreateLoadBalancerRequest.CreateLoadBalancerRequest()
request.set_MasterZoneId(params["master_zone_id"])
request.set_SlaveZoneId(params["slave_zone_id"])
request.set_LoadBalancerName(params["load_balancer_name"])
request.set_PayType(params["pay_balancer_type"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)


def add_backend_servers(params):
'''
add_backend_servers：添加后端服务器

'''
counter = 0
while counter < TRY_TIME:
try:
# 设置添加默认服务器组的调用参数
request = AddBackendServersRequest.AddBackendServersRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
# 要添加的后端服务器列表
request.set_BackendServers(params["backend_servers"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
```

```
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
if ExceptionHandler.server_exception(e):
return e
except ClientException as e:
if ExceptionHandler.client_exception(e):
return e
finally:
counter += 1


def upload_server_certificate(params):
'''
upload_server_certificate：上传服务器证书

'''
counter = 0
while counter < TRY_TIME:
try:
request = UploadServerCertificateRequest.UploadServerCertificateRequest()
# 要上传的公钥证书
request.set_ServerCertificate(params["server_certificate"])
# 需要上传的私钥
request.set_PrivateKey(params["private_key"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)
finally:
counter += 1


def create_https_listener(params):
'''
```

```
create_https_listener：创建HTTPS监听

'''
counter = 0
while counter < TRY_TIME:
try:
request = CreateLoadBalancerHTTPSListenerRequest.CreateLoadBalancerHTTPSListenerRequest(
)
# 负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
# 监听的带宽峰值
request.set_Bandwidth(params["bandwidth"])
# 负载均衡实例前端使用的端口
request.set_ListenerPort(params["listener_port"])
# 是否开启健康检查
request.set_HealthCheck(params["health_check"])
# 是否开启会话保持
request.set_StickySession(params["sticky_session"])
# 负载均衡实例后端使用的端口
request.set_BackendServerPort(params["backend_server_port"])
# 服务器证书的ID
request.set_ServerCertificateId(params["server_certificate_id"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)
finally:
counter += 1


def set_https_listener_attribute(params):
'''
set_https_listener_attribute：修改HTTPS监听的配置

'''
```

```
counter = 0
while counter < TRY_TIME:
try:
request = SetLoadBalancerHTTPSListenerAttributeRequest.SetLoadBalancerHTTPSListenerAttrib
uteRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
# 监听的带宽峰值
request.set_Bandwidth(params["bandwidth"])
# 负载均衡实例前端使用的端口
request.set_ListenerPort(params["listener_port"])
# 是否开启健康检查
request.set_HealthCheck(params["health_check"])
# 是否开启会话保持
request.set_StickySession(params["sticky_session"])
# 服务器证书的ID
request.set_ServerCertificateId(params["server_certificate_id"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)
finally:
counter += 1


def delete_load_balancer(load_balancer_id):
'''
delete_load_balancer：删除slb实例

'''
try:
# 设置删除SLB实例的调用参数
request = DeleteLoadBalancerRequest.DeleteLoadBalancerRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(load_balancer_id)
# 发送调用请求并接收返回数据
```

```
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)


def main():
params = {}
# 设置创建SLB实例的参数
# 设置新建SLB实例的主可用区为cn-zhangjiakou-a
params["master_zone_id"] = "cn-zhangjiakou-a"
# 设置新建SLB实例的备可用区为cn-zhangjiakou-b
params["slave_zone_id"] = "cn-zhangjiakou-b"
# 设置新建SLB实例的名称为SLB1
params["load_balancer_name"] = "SLB1"
# 设置新建SLB实例的计费类型为按量计费
params["pay_balancer_type"] = "PayOnDemand"

# 设置添加到默认服务器组的ECS的实例ID和权重
params["backend_servers"] = [{"ServerId": "i-8vbe8yixxxxxxxxxxxxxxw", "Weight": "100"},
{"ServerId": "i-8vbe8yxxxxxxxxxxxxxj9v", "Weight": "100"}]

# 设置上传服务器证书的参数

# 设置创建HTTPS监听的参数
# 关闭健康检查
params["health_check"] = "off"
# 设置监听的带宽峰值
params["bandwidth"] = 6
# 负载均衡实例前端使用的端口
params["listener_port"] = 80
# 设置负载均衡实例后端使用的端口
params["backend_server_port"] = 443
# 关闭会话保持
params["sticky_session"] = "off"

# 创建slb实例
```

```
# 创建slb实例
# 获取create_load_balancer函数返回值，load_balancer_json为结果的json串
load_balancer_json = create_load_balancer(params)
# 打印 load_balancer_json结果，其中"create_load_balancer"是对json串起的名字
CommonUtil.log("create_load_balancer", load_balancer_json)

# 上传服务器证书
# 要上传的公钥证书
params["server_certificate"] = "-----BEGIN CERTIFICATE-----xxxxxxx-----END CERTIFICATE-----"
params["private_key"] = "-----BEGIN RSA PRIVATE KEY-----xxxxxxxxxxx-----END RSA PRIVATE KEY-----"
result_json = upload_server_certificate(params)
CommonUtil.log("upload_server_certificate", result_json)
params["server_certificate_id"] = result_json["ServerCertificateId"]

# slb实例id
params["load_balancer_id"] = load_balancer_json["LoadBalancerId"]

# 创建https监听
result_json = create_https_listener(params)
CommonUtil.log("create_https_listener", result_json)

# 上传新的服务器证书
# 要上传的新公钥证书
params["server_certificate"] = "-----BEGIN CERTIFICATE-----xxxxxxx-----END CERTIFICATE-----"
params["private_key"] = "-----BEGIN RSA PRIVATE KEY-----xxxxxxxxxxx-----END RSA PRIVATE KEY-----"
result_json = upload_server_certificate(params)
CommonUtil.log("upload_server_certificate", result_json)
params["server_certificate_id"] = result_json["ServerCertificateId"]

# 修改https监听配置
result_json = set_https_listener_attribute(params)
CommonUtil.log("set_https_listener_attribute", result_json)

# 删除slb实例
# 删除返回的LoadBalancerId对应的SLB实例
load_balancer_json = delete_load_balancer(load_balancer_json["LoadBalancerId"])
# 打印 load_balancer_json结果
CommonUtil.log("delete_load_balancer", load_balancer_json)
```

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

3. 进入*slb_quick_start.py*所在的目录，执行如下命令，运行创建和删除SLB实例示例。

```
python slb_quick_start.py
```

系统显示类似如下：

```
----------create_load_balancer----------
{
"VpcId": "",
"AddressIPVersion": "ipv4",
"LoadBalancerName": "SLB1",
"ResourceGroupId": "rg-acfmxxxxxxxxy",
"VSwitchId": "",
"RequestId": "070A9361-EA2B-4A4E-91E8-F70C1E47866A",
"Address": "172.16.XX.XX",
"NetworkType": "classic",
"LoadBalancerId": "lb-8vbninnfxxxxxxxxxxxxx"
}

----------delete_load_balancer----------
{
"RequestId": "4825E5BB-5131-4A1E-AF67-0EF3E8E5B323"
}
```

2.3.2. 创建TCP监听

本文介绍使用Python SDK给SLB实例创建TCP监听，创建完成后删除SLB实例。

前提条件

在华北3张家口地域已经有两台状态为运行中的ECS实例。

在开始运行示例脚本前，您还需确保已完成以下准备工作：

1. 因为使用Python SDK创建SLB实例会产生实例费，所以确保您的阿里云账户金额不少于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。如果尚未获取AK，参见查询获取获取AccessKey。
3. 下载阿里云负载均衡Python SDK场景示例的SLB 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
```

背景信息

本示例将使用以下配置：

- 在华北3张家口地域创建一个名为SLB1的SLB实例，主可用区为cn-zhangjiakou-a，备可用区为cn-zhangjiakou-b，实例计费类型为按量计费，规格为slb.s1.small，其他配置使用系统默认值。
- 将张家口地域的两台ECS实例添加到默认服务器组，用来分发流量，ECS实例的后端服务权重都为100。
- 创建TCP监听，SLB实例前端使用的端口为80，后端服务器开放用来接收请求的端口为80，健康检查协议为TCP，不限制监听带宽峰值，其他使用默认值。
- TCP监听创建完成后，从默认服务器组中移除ECS实例，并删除新建的SLB实例。

操作步骤

1. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\examples\slb*文件夹。
2. 使用编辑器打开*slb_create_tcp_listener.py*文件，您需要配置用户鉴权参数ACS_CLIENT，其他参数根据实际情况配置后，保存退出。

   > 说明 因为用户鉴权是每个SDK必须执行的操作，所以可以在常量文件中配置AcsClient参数的值，在场景代码中调用常量文件。

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


# 调用AcsClient参数进行身份验证
from aliyunsdkcore.client import AcsClient
# 使用阿里云官方sdk的异常处理模块
from aliyunsdkcore.acs_exception.exceptions import ServerException, ClientException
# 调用创建SLB实例API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerRequest
# 调用添加默认服务器组API
from aliyunsdkslb.request.v20140515 import AddBackendServersRequest
```

```
from aliyunsdkslb.request.v20140515 import AddBackendServersRequest
# 调用添加TCP监听API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerTCPListenerRequest
# 调用从默认服务器组中移除ECS实例API
from aliyunsdkslb.request.v20140515 import RemoveBackendServersRequest
# 调用删除SLB实例API
from aliyunsdkslb.request.v20140515 import DeleteLoadBalancerRequest
# 命令行打印日志
from sdk_lib.common_util import CommonUtil
# 异常处理逻辑
from sdk_lib.exception import ExceptionHandler

'''
创建slb实例->添加后端服务器->创建tcp监听->删除后端服务器->删除slb实例
'''
# client参数配置
ACS_CLIENT = AcsClient(
'your-access-key-id', # your-access-key-id
'your-access-key-secret', # your-access-key-secret
'cn-zhangjiakou', # your-region-id
)

TRY_TIME = 3


def create_load_balancer(params):
'''
create_load_balancer：创建slb实例
官网API参考链接：https://help.aliyun.com/document_detail/27577.html
'''
try:
# 设置创建SLB实例的调用参数
request = CreateLoadBalancerRequest.CreateLoadBalancerRequest()
request.set_MasterZoneId(params["master_zone_id"])
request.set_SlaveZoneId(params["slave_zone_id"])
request.set_LoadBalancerName(params["load_balancer_name"])
request.set_PayType(params["pay_balancer_type"])
request.set_LoadBalancerSpec(params["load_balancer_spec"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
```

```
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)


def add_backend_servers(params):
'''
add_backend_servers：添加后端服务器
官网API参考链接: https://help.aliyun.com/document_detail/27632.html
'''
counter = 0
while counter < TRY_TIME:
try:
# 设置添加默认服务器组的调用参数
request = AddBackendServersRequest.AddBackendServersRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
# 要添加的后端服务器列表
request.set_BackendServers(params["backend_servers"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
if ExceptionHandler.server_exception(e):
return e
except ClientException as e:
if ExceptionHandler.client_exception(e):
return e
finally:
counter += 1


def create_tcp_listener(params):
'''
create_tcp_listener：创建tcp监听
官网API参考链接: https://help.aliyun.com/document_detail/27594.html
```

```
'''
counter = 0
while counter < TRY_TIME:
try:
# 设置创建TCP监听的调用参数
request = CreateLoadBalancerTCPListenerRequest.CreateLoadBalancerTCPListenerRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
# 负载均衡实例前端使用的端口
request.set_ListenerPort(params["listener_port"])
# 负载均衡实例后端使用的端口
request.set_BackendServerPort(params["backend_server_port"])
# 监听的健康检查协议
request.set_HealthCheckType(params["listener_health_check"])
# 设置监听的带宽峰值
request.set_Bandwidth(params["listener_bandwidth"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
if ExceptionHandler.server_exception(e):
return e
except ClientException as e:
if ExceptionHandler.client_exception(e):
return e
finally:
counter += 1


def remove_backend_servers(params):
'''
add_backend_servers：删除后端服务器
官网API参考链接: https://help.aliyun.com/document_detail/27633.html
'''
counter = 0
while counter < TRY_TIME:
try:
# 设置删除后端服务器的调用参数
request = RemoveBackendServersRequest.RemoveBackendServersRequest()
```

```
request = RemoveBackendServersRequest.RemoveBackendServersRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
# 要删除的后端服务器列表
request.set_BackendServers(params["backend_servers"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
if ExceptionHandler.server_exception(e):
return e
except ClientException as e:
if ExceptionHandler.client_exception(e):
return e
finally:
counter += 1


def delete_load_balancer(load_balancer_id):
'''
delete_load_balancer：删除slb实例
官网API参考链接：https://help.aliyun.com/document_detail/27579.html
'''
try:
# 设置删除SLB实例的调用参数
request = DeleteLoadBalancerRequest.DeleteLoadBalancerRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(load_balancer_id)
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)
```

```
def main():
params = {}

# 设置创建SLB实例的参数
# 设置新建SLB实例的主可用区为cn-zhangjiakou-a
params["master_zone_id"] = "cn-zhangjiakou-a"
# 设置新建SLB实例的主可用区为cn-zhangjiakou-b
params["slave_zone_id"] = "cn-zhangjiakou-b"
# 设置新建SLB实例的名称为SLB1
params["load_balancer_name"] = "SLB1"
# 设置新建SLB实例的计费类型为按量计费
params["pay_balancer_type"] = "PayOnDemand"
# 设置新建SLB实例的规格为slb.s1.small
params["load_balancer_spec"] = "slb.s1.small"
# 设置添加到默认服务器组的ECS的实例ID和权重
params["backend_servers"] = [{"ServerId": "i-8vbe8yi8krxxxxxxxxxxw", "Weight": "100"},
{"ServerId": "i-8vbe8yi8kccxxxxxxxxxv", "Weight": "100"}]

# 设置添加TCP监听的参数
# 前端使用的端口为80
params["listener_port"] = 80
# 后端服务器开放用来接收请求的端口为80
params["backend_server_port"] = 80
# 健康检查协议为TCP
params["listener_health_check"] = "tcp"
# TCP监听的带宽峰值为-1，即不限制带宽峰值
params["listener_bandwidth"] = -1

# 创建slb实例
# 获取create_load_balancer函数返回值，load_balancer_json为结果的json串
load_balancer_json = create_load_balancer(params)
# 打印 load_balancer_json结果，其中"create_load_balancer"是对json串起的名字
CommonUtil.log("create_load_balancer", load_balancer_json)

# slb实例id
params["load_balancer_id"] = load_balancer_json["LoadBalancerId"]

# 添加后端服务器
load_balancer_json = add_backend_servers(params)
CommonUtil.log("add_backend_servers", load_balancer_json)
```

```
# 创建tcp监听
load_balancer_json = create_tcp_listener(params)
CommonUtil.log("create_tcp_listener", load_balancer_json)

# 删除后端服务器
load_balancer_json = remove_backend_servers(params)
CommonUtil.log("remove_backend_servers", load_balancer_json)

# 删除slb实例
# 删除返回的LoadBalancerId对应的SLB实例
load_balancer_json = delete_load_balancer(load_balancer_json["LoadBalancerId"])
# 打印 load_balancer_json结果
CommonUtil.log("delete_load_balancer", load_balancer_json)



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

3. 进入*slb_create_tcp_listener.py*所在的目录，执行如下命令，运行创建TCP监听示例。

```
python slb_create_tcp_listener.py
```

系统显示类似如下：

```
---------------------------create_load_balancer---------------------------
{
"VpcId": "",
"AddressIPVersion": "ipv4",
"LoadBalancerName": "SLB1",
"ResourceGroupId": "rg-acfmxazb4pxxxxx",
"VSwitchId": "",
"RequestId": "8E72DFDD-E510-4D65-9063-F47B62DE9887",
"Address": "39.98.xx.xx",
"NetworkType": "classic",
"LoadBalancerId": "lb-8vbjd6pzphvjoq29xxxxx"
}

---------------------------add_backend_servers---------------------------
{
"RequestId": "F6BF0E0B-5837-48A3-9CEC-2EE2CED5293A",
"BackendServers": {
"BackendServer": [
```

```
"BackendServer": [
{
"ServerId": "i-8vbe8yi8krqri2axxxxx",
"Type": "ecs",
"Weight": 100
},
{
"ServerId": "i-8vbe8yi8krqri2xxxxxx",
"Type": "ecs",
"Weight": 100
}
]
},
"LoadBalancerId": "lb-8vbjd6pzphvjoq29xxxxx"
}

----------create_tcp_listener----------
{
"RequestId": "6FF090BD-DF9F-49C6-84A6-30F2D9F88489"
}

----------remove_backend_servers----------
{
"RequestId": "F7D62969-D7F0-4022-95FA-1710DA448B6A",
"BackendServers": {
"BackendServer": []
},
"LoadBalancerId": "lb-8vbjd6pzphvjoq29xxxxx"
}

----------delete_load_balancer----------
{
"RequestId": "1A8CB48F-8242-49A9-9958-E762B1E9BA4D"
}
```


2.3.3. 更新HTTPS证书

介绍如何使用Python SDK给一个SLB实例创建一个HTTPS监听，并更新该实例下HTTPS监听使用的服务器证书。

前提条件

在华北3张家口地域已经有两台状态为**运行中**的ECS实例。

在开始运行示例脚本前，您还需确保完成以下准备工作：

1. 因为使用Python SDK创建SLB实例会产生实例费，所以确保您的阿里云账户金额不少于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。如果尚未获取AK，参见查询获取获取AccessKey。
3. 下载阿里云负载均衡Python SDK场景示例的SLB 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
   ```

背景信息

本次示例分为以下几步，为SLB实例添加HTTPS监听，更新HTTPS监听的证书。

1. 在华北3张家口地域创建一个名为SLB1的SLB实例，主可用区为cn-zhangjiakou-a，备可用区为cn-zhangjiakou-b，实例计费类型为按量计费，规格为slb.s1.small，其他配置使用系统默认值。
2. 将张家口地域的两台ECS实例添加到默认服务器组，用来分发流量，ECS实例的后端服务权重都为100。
3. 上传服务器证书。
4. 创建HTTPS监听，SLB实例前端使用的端口为80，后端服务器开放用来接收请求的端口为443，关闭健康检查和会话保持，监听带宽峰值为6Mbps，其他使用默认值。
5. 上传新的服务器证书。
6. 更新新建HTTPS监听的服务器证书。
7. 删除新建的SLB实例。

操作步骤

1. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\examples\slb*文件夹。
2. 使用编辑器打开*upload_server_certificate.py*文件，您需要配置用户鉴权参数ACS_CLIENT，其他参数根据实际情况配置后，保存退出。

   > **说明** 因为用户鉴权是每个SDK必须执行的操作，所以可以在常量文件中配置AcsClient参数的值，在场景代码中调用常量文件。

   ```
   #encoding=utf-8
   """
   创建slb实例->上传服务器证书->创建https监听->上传新的服务器证书
   ->循环修改https监听配置
   """
   import sys
   import json
   import uuid
   ```

```
#调用AcsClient参数进行身份验证
from aliyunsdkcore.client import AcsClient
#使用阿里云官方sdk的异常处理模块
from aliyunsdkcore.acs_exception.exceptions import ServerException, ClientException
#调用上传证书的API
from aliyunsdkslb.request.v20140515 import UploadServerCertificateRequest
#调用创建SLB实例API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerRequest
#调用添加默认服务器组API
from aliyunsdkslb.request.v20140515 import AddBackendServersRequest
#调用创建HTTPS监听的API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerHTTPSListenerRequest
#调用修改HTTPS监听的API
from aliyunsdkslb.request.v20140515 import SetLoadBalancerHTTPSListenerAttributeRequest
#调用删除SLB实例API
from aliyunsdkslb.request.v20140515 import DeleteLoadBalancerRequest

#用户身份鉴权参数配置
ACS_CLIENT = AcsClient(
'your-access-key-id', #your-access-key-id
'your-access-key-secret', #your-access-key-secret
'cn-zhangjiakou', #your-region-id
)
#失败重试次数
TRY_TIME = 3

def log(action, response_json):
print("----------"+action+"----------")
print(json.dumps(response_json, indent=2)+'\n')

def create_load_balancer(params):
'''
create_load_balancer：创建slb实例

'''
try:
#设置创建SLB实例的调用参数
request = CreateLoadBalancerRequest.CreateLoadBalancerRequest()
request.set_MasterZoneId(params["master_zone_id"])
request.set_SlaveZoneId(params["slave_zone_id"])
```

```
request.set_LoadBalancerName(params["load_balancer_name"])
request.set_PayType(params["pay_balancer_type"])
request.set_ClientToken(str(uuid.uuid1()))
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
return e
except ClientException as e:
print(e)

def add_backend_servers(params):
'''
add_backend_servers：添加后端服务器

'''
counter = 0
while counter < TRY_TIME:
try:
#设置添加默认服务器组的调用参数
request = AddBackendServersRequest.AddBackendServersRequest()
#负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
#要添加的后端服务器列表
request.set_BackendServers(params["backend_servers"])
request.set_ClientToken(str(uuid.uuid1()))
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
return e
except ClientException as e:
return e
finally:
counter += 1
```

```
def upload_server_certificate(params):
'''
upload_server_certificate：上传服务器证书

'''
counter = 0
while counter < TRY_TIME:
try:
request = UploadServerCertificateRequest.UploadServerCertificateRequest()
#要上传的公钥证书
request.set_ServerCertificate(params["server_certificate"])
#需要上传的私钥
request.set_PrivateKey(params["private_key"])
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
return e
except ClientException as e:
return e
finally:
counter += 1

def create_https_listener(params):
'''
create_https_listener：创建HTTPS监听

'''
counter = 0
while counter < TRY_TIME:
try:
request = CreateLoadBalancerHTTPSListenerRequest.CreateLoadBalancerHTTPSListenerRequest(
)
#负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
#监听的带宽峰值
request.set_Bandwidth(params["bandwidth"])
#负载均衡实例前端使用的端口
request.set_ListenerPort(params["listener_port"])
```

```
#是否开启健康检查
request.set_HealthCheck(params["health_check"])
#是否开启会话保持
request.set_StickySession(params["sticky_session"])
#负载均衡实例后端使用的端口
request.set_BackendServerPort(params["backend_server_port"])
#服务器证书的ID
request.set_ServerCertificateId(params["server_certificate_id"])
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
return e
except ClientException as e:
return e
finally:
counter += 1

def set_https_listener_attribute(params):
'''
set_https_listener_attribute：修改HTTPS监听的配置

'''
counter = 0
while counter < TRY_TIME:
try:
request = SetLoadBalancerHTTPSListenerAttributeRequest.SetLoadBalancerHTTPSListenerAttrib
uteRequest()
#负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
#监听的带宽峰值
request.set_Bandwidth(params["bandwidth"])
#负载均衡实例前端使用的端口
request.set_ListenerPort(params["listener_port"])
#是否开启健康检查
request.set_HealthCheck(params["health_check"])
#是否开启会话保持
request.set_StickySession(params["sticky_session"])
#服务器证书的ID
```

```
request.set_ServerCertificateId(params["server_certificate_id"])
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
return e
except ClientException as e:
return e
finally:
counter += 1

def delete_load_balancer(load_balancer_id):
'''
delete_load_balancer：删除slb实例

'''
try:
#设置删除SLB实例的调用参数
request = DeleteLoadBalancerRequest.DeleteLoadBalancerRequest()
#负载均衡实例的ID
request.set_LoadBalancerId(load_balancer_id)
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
return e
except ClientException as e:
return e

def main():
params = {}
#设置创建SLB实例的参数
#设置新建SLB实例的主可用区为cn-zhangjiakou-a
params["master_zone_id"] = "cn-zhangjiakou-a"
#设置新建SLB实例的主可用区为cn-zhangjiakou-b
params["slave_zone_id"] = "cn-zhangjiakou-b"
```

```
#设置新建SLB实例的名称为SLB1
params["load_balancer_name"] = "SLB1"
#设置新建SLB实例的计费类型为按量计费
params["pay_balancer_type"] = "PayOnDemand"

#设置添加到默认服务器组的ECS的实例ID和权重
params["backend_servers"] = [{"ServerId":"i-8vbe8yixxxxxxxxxxxxxxw","Weight":"100"},{"ServerId":"
i-8vbe8yxxxxxxxxxxxxxxj9v","Weight":"100"}]

#设置上传服务器证书的参数

#设置创建HTTPS监听的参数
#关闭健康检查
params["health_check"] = "off"
#设置监听的带宽峰值
params["bandwidth"] = 6
#负载均衡实例前端使用的端口
params["listener_port"] = 80
#设置负载均衡实例后端使用的端口
params["backend_server_port"] = 443
#关闭会话保持
params["sticky_session"] = "off"

#创建slb实例
#获取create_load_balancer函数返回值，load_balancer_json为结果的json串
load_balancer_json = create_load_balancer(params)
#打印 load_balancer_json结果，其中"create_load_balancer"是对json串起的名字
log("create_load_balancer", load_balancer_json)

#上传服务器证书
#要上传的公钥证书
params["server_certificate"] = "-----BEGIN CERTIFICATE-----xxxxxxx-----END CERTIFICATE-----"
params["private_key"] = "-----BEGIN RSA PRIVATE KEY-----xxxxxxxxxxx-----END RSA PRIVATE KEY---
-"
result_json = upload_server_certificate(params)
log("upload_server_certificate", result_json)
params["server_certificate_id"] = result_json["ServerCertificateId"]

#slb实例id
params["load_balancer_id"] = load_balancer_json["LoadBalancerId"]
```

```
#创建https监听
result_json = create_https_listener(params)
log("create_https_listener", result_json)

#上传新的服务器证书
#要上传的新公钥证书
params["server_certificate"] = "-----BEGIN CERTIFICATE-----xxxxxxx-----END CERTIFICATE-----"
params["private_key"] = "-----BEGIN RSA PRIVATE KEY-----xxxxxxxxxxx-----END RSA PRIVATE KEY---
--"
result_json = upload_server_certificate(params)
log("upload_server_certificate", result_json)
#
params["server_certificate_id"] = result_json["ServerCertificateId"]

#修改https监听配置
result_json = set_https_listener_attribute(params)
log("set_https_listener_attribute", result_json)

#删除slb实例
#删除返回的LoadBalancerId对应的SLB实例
load_balancer_json = delete_load_balancer(load_balancer_json["LoadBalancerId"])
#打印 load_balancer_json结果
log("delete_load_balancer", load_balancer_json)

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

3. 进入*upload_server_certificate.py*所在的目录，执行如下命令，运行更新HTTPS证书示例。

```
python upload_server_certificate.py
```

系统显示类似如下：

```
----------------------------create_load_balancer----------------------------
{
"VpcId": "",
"AddressIPVersion": "ipv4",
"LoadBalancerName": "SLB1",
"ResourceGroupId": "rg-axxxxxxxxxxxy",
"VSwitchId": "",
"RequestId": "1DEF72EE-CA5C-449D-9E97-2CE460DE7F7A",
"Address": "39.98.97.8",
```

"NetworkType": "classic",
"LoadBalancerId": "lb-8vxxxxxxxxxxxf"
}

----------upload_server_certificate----------
{
"ExpireTimeStamp": 1732169065000,
"RegionIdAlias": "cn-zhangjiakou",
"AliCloudCertificateName": "",
"RegionId": "cn-zhangjiakou",
"CommonName": "test",
"ServerCertificateName": "test",
"ExpireTime": "2024-11-21T06:04:25Z",
"ResourceGroupId": "rg-axxxxxxxxxxxyy",
"RequestId": "D6FB8B7C-B0EA-43E7-AD8F-E0201A9E1A13",
"Fingerprint": "cd:90:1b:7b:49:4d:1d:90:f6:01:de:9a:81:7d:31:a7:38:1d:84:8d",

"AliCloudCertificateId": "",
"IsAliCloudCertificate": 0,
"ServerCertificateId": "12xxxxxxxxxxxx3_16xxxxxxxxxx6_-1xxxxxxxx13_-1xxxxxx72"
}

----------create_https_listener----------
{
"RequestId": "FC70EDC8-06EC-4E1C-97BE-D81571DA23B4"
}

----------upload_server_certificate----------
{
"ExpireTimeStamp": 1567857600000,
"RegionIdAlias": "cn-zhangjiakou",
"AliCloudCertificateName": "",
"RegionId": "cn-zhangjiakou",
"CommonName": "doc.aliyun-slb.top",
"ServerCertificateName": "doc.aliyun-slb.top",
"ExpireTime": "2019-09-07T12:00:00Z",
"ResourceGroupId": "rg-acxxxxxxxxxxxxy",
"RequestId": "913E991F-BCC9-4A3D-9802-19084E2CE271",
"Fingerprint": "8c:47:1b:a7:8f:02:27:3a:35:7c:e3:47:4a:5f:55:02:ed:e3:57:1e",

```
"SubjectAlternativeNames": {
"SubjectAlternativeName": [
"doc.aliyun-slb.top"
]
},
"AliCloudCertificateId": "",
"IsAliCloudCertificate": 0,
"ServerCertificateId": "12xxxxxxxxxxxx3_16xxxxxxx716_4xxxxxxx4_-14xxxxxxxxxx9"
}

----------set_https_listener_attribute----------
--
{
"RequestId": "14275D2B-20F2-4D96-9FC8-7BB2B110155C"
}

----------delete_load_balancer----------
{
"RequestId": "2A1DF87C-842A-4C1E-AA61-23EB365EB86F"
}
```

2.3.4. 查看监控

介绍如何使用Python SDK查看SLB实例的监控参数并配置告警。

前提条件

在华北3张家口地域已经有两台状态为**运行中**的ECS实例。

在开始运行示例脚本前，您还需确保已完成以下准备工作：

1. 因为使用Python SDK创建SLB实例会产生实例费，所以确保您的阿里云账户金额不少于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。如果尚未获取AK，参见查询获取获取AccessKey。
3. 下载阿里云负载均衡Python SDK场景示例的SLB 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
   ```

背景信息

本示例将采用以下配置：

- 在华北3张家口地域新建一个名为SLB1的按量付费实例，主可用区为cn-zhangjiakou-a，备可用区为cn-zhangjiakou-b。
- 为新建SLB实例创建一个TCP监听，SLB实例前端使用的端口为80，后端服务器开放用来接收请求的端口为80，健康检查协议为TCP，将张家口地域的两台ECS实例添加到默认服务器组，用来分发流量，ECS实例的后端服务权重都为100。
- 查询新建SLB实例QPS使用率。
- 创建告警，当新建SLB实例QPS使用率平均值大于等于35时，发起告警。
- 删除新建实例。

操作步骤

1. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\examples\slb*文件夹。
2. 使用编辑器打开*slb_monitor.py*文件，您需要配置用户鉴权参数ACS_CLIENT，其他参数根据实际情况配置后，保存退出。

> **说明** 因为用户鉴权是每个SDK必须执行的操作，所以可以在常量文件中配置AcsClient参数的值，在场景代码中调用常量文件。

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


# 调用AcsClient参数进行身份验证
from aliyunsdkcore.client import AcsClient
# 使用阿里云官方sdk的异常处理模块
from aliyunsdkcore.acs_exception.exceptions import ServerException, ClientException
# 异常处理逻辑
from sdk_lib.exception import ExceptionHandler
# 调用创建SLB实例API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerRequest
# 调用添加默认服务器组API
from aliyunsdkslb.request.v20140515 import AddBackendServersRequest
# 调用添加TCP监听API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerTCPListenerRequest
# 调用云监控的接口查看SLB的最新监控数据
from aliyunsdkcms.request.v20190101 import DescribeMetricLastRequest
# 调用云监控的接口为SLB实例创建报警规则
from aliyunsdkcms.request.v20190101 import PutResourceMetricRuleRequest
# 调用删除SLB实例API
```

```
from aliyunsdkslb.request.v20140515 import DeleteLoadBalancerRequest
# 命令行导入日志
from sdk_lib.common_util import CommonUtil

# 用户身份验证参数配置
ACS_CLIENT = AcsClient(
'your-access-key-id', # your-access-key-id
'your-access-key-secret', # your-access-key-secret
'cn-zhangjiakou', # your-region-id
)
# 失败重试次数
TRY_TIME = 3

'''
创建slb实例->创建tcp监听->创建后端服务器->
查询slb实例端口当前并发连接数->创建告警规则
'''



def create_load_balancer(params):
'''
create_load_balancer：创建slb实例
官网API参考链接：https://help.aliyun.com/document_detail/27577.html
'''
try:
# 设置创建SLB实例的调用参数
request = CreateLoadBalancerRequest.CreateLoadBalancerRequest()
request.set_MasterZoneId(params["master_zone_id"])
request.set_SlaveZoneId(params["slave_zone_id"])
request.set_LoadBalancerName(params["load_balancer_name"])
request.set_PayType(params["pay_balancer_type"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)
```

```
def add_backend_servers(params):
'''
add_backend_servers：添加后端服务器
官网API参考链接: https://help.aliyun.com/document_detail/27632.html
'''
counter = 0
while counter < TRY_TIME:
try:
# 设置添加默认服务器组的调用参数
request = AddBackendServersRequest.AddBackendServersRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
# 要添加的后端服务器列表
request.set_BackendServers(params["backend_servers"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
if ExceptionHandler.server_exception(e):
return e
except ClientException as e:
if ExceptionHandler.client_exception(e):
return e
finally:
counter += 1


def create_tcp_listener(params):
'''
create_tcp_listener：创建tcp监听
官网API参考链接: https://help.aliyun.com/document_detail/27594.html
'''
counter = 0
while counter < TRY_TIME:
try:
# 设置创建TCP监听的调用参数
request = CreateLoadBalancerTCPListenerRequest.CreateLoadBalancerTCPListenerRequest()
# 负载均衡实例的ID
```

```
# 负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
# 负载均衡实例前端使用的端口
request.set_ListenerPort(params["listener_port"])
# 负载均衡实例后端使用的端口
request.set_BackendServerPort(params["backend_server_port"])
# 监听的健康检查协议
request.set_HealthCheckType(params["listener_health_check"])
# 设置监听的带宽峰值
request.set_Bandwidth(params["listener_bandwidth"])
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
if ExceptionHandler.server_exception(e):
return e
except ClientException as e:
if ExceptionHandler.client_exception(e):
return e
finally:
counter += 1


def describe_metric_last(params):
'''
describe_metric_last：查询指定监控对象的最新监控数据
官网API参考链接: https://help.aliyun.com/document_detail/51939.html
'''
counter = 0
while counter < TRY_TIME:
try:
request = DescribeMetricLastRequest.DescribeMetricLastRequest()
# 名字空间，表明监控数据所属产品
request.set_Namespace(params["project"])
# 监控项名称
request.set_MetricName(params["metric_name"])
# 过滤项
request.set_Dimensions(params['dimensions'])
response = ACS_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)
finally:
counter += 1


def put_resource_metric_rule(params):
'''
put_resource_metric_rule：创建报警规则，可以为某一个实例创建报警规则，也可以为多个实例同时创建报警规则
官网API参考链接: https://help.aliyun.com/document_detail/114934.html
'''
counter = 0
while counter < TRY_TIME:
try:
request = PutResourceMetricRuleRequest.PutResourceMetricRuleRequest()
# 报警规则ID，调用者统一生成，保证唯一性，已经存在的ID则修改，不存在则创建
request.set_RuleId(uuid.UUID)
# 产品名称，参考各产品对应的project
request.set_Namespace(params["name_space"])
# 报警联系人组，多个联系组之间用英文逗号分隔
request.set_ContactGroups(params["ContactGroups"])
# 报警规则名称
request.set_RuleName(params["name"])
# 相应产品对应的监控项名称，参考各产品metric定义
request.set_MetricName(params["metric_name"])
# 报警规则对应实例列表
request.set_Resources(params["dimensions"])
# Critical级别报警统计方法
request.set_EscalationsCriticalStatistics(params["statistics"])
# Critical级别阈值比较符
request.set_EscalationsCriticalComparisonOperator(params["comparison_operator"])
# Critical级别报警阈值
request.set_EscalationsCriticalThreshold(params["threshold"])
# Critical级别报警重试次数
request.set_EscalationsCriticalTimes(3)
response = ACS_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)
finally:
counter += 1


def delete_load_balancer(load_balancer_id):
'''
delete_load_balancer：删除slb实例
官网API参考链接：https://help.aliyun.com/document_detail/27579.html
'''
try:
# 设置删除SLB实例的调用参数
request = DeleteLoadBalancerRequest.DeleteLoadBalancerRequest()
# 负载均衡实例的ID
request.set_LoadBalancerId(load_balancer_id)
# 发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
# 返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)


def main():
params = {}
# 设置创建SLB实例的参数
# 设置新建SLB实例的主可用区为cn-zhangjiakou-a
params["master_zone_id"] = "cn-zhangjiakou-a"
# 设置新建SLB实例的主可用区为cn-zhangjiakou-b
params["slave_zone_id"] = "cn-zhangjiakou-b"
# 设置新建SLB实例的名称为SLB1
params["load_balancer_name"] = "SLB1"
```

```
# 设置新建SLB实例的计费类型为按量计费
params["pay_balancer_type"] = "PayOnDemand"

# 设置添加到默认服务器组的ECS的实例ID和权重
params["backend_servers"] = [{"ServerId": "i-8vbfus6jnwjp1c2*****", "Weight": "100"},
{"ServerId": "i-8vbfus6jnwjp1c2*****", "Weight": "100"}]

# 设置添加TCP监听的参数
# 前端使用的端口为80
params["listener_port"] = 80
# 后端服务器开放用来接收请求的端口为80
params["backend_server_port"] = 80
# 健康检查协议为TCP
params["listener_health_check"] = "tcp"
# TCP监听的带宽峰值为-1，即不限制带宽峰值
params["listener_bandwidth"] = -1

# 设置查询监控项的参数
# 设置监控数据所属产品
params["project"] = "acs_slb_dashboard"
# 设置监控项名称
params["traffic_tx_new"] = "InstanceQpsUtilization"

# 设置创建告警的参数
# 设置告警规则所属产品
params["name_space"] = "acs_slb_dashboard"
# 设置告警规则名称
params["name"] = "slb_alarm"
# 设置告警中SLB对应的监控项名称
params["metric_name"] = "InstanceQpsUtilization"
# 设置统计方法
params["statistics"] = "Average"
# 设置报警比较符,大于等于
params["comparison_operator"] = "GreaterThanOrEqualToThreshold"
# 设置报警阈值
params["threshold"] = 35

# 创建slb实例
# 获取create_load_balancer函数返回值，load_balancer_json为结果的json串
load_balancer_json = create_load_balancer(params)
# 打印 load_balancer_json结果，其中"create_load_balancer"是对json串起的名字
```

```
CommonUtil.log("create_load_balancer", load_balancer_json)

# slb实例id
params["load_balancer_id"] = load_balancer_json["LoadBalancerId"]

# 创建tcp监听
result_json = create_tcp_listener(params)
CommonUtil.log("create_tcp_listener", result_json)

# 创建后端服务器
result_json = add_backend_servers(params)
CommonUtil.log("add_backend_servers", result_json)

# 查询slb实例QPS使用率
params["dimensions"] = '[{"instanceId":"' + load_balancer_json["LoadBalancerId"] + '"}]'
result_json = describe_metric_last(params)
CommonUtil.log("describe_metric_last", result_json)

# 创建告警规则
params["metric_name"] = "InstanceQpsUtilization"
# 报警联系人组
params["ContactGroups"] = "test_contact_groups"
result_json = put_resource_metric_rule(params)
CommonUtil.log("put_resource_metric_rule", result_json)

# 删除slb实例
# 删除返回的LoadBalancerId对应的SLB实例
load_balancer_json = delete_load_balancer(load_balancer_json["LoadBalancerId"])
# 打印 load_balancer_json结果
CommonUtil.log("delete_load_balancer", load_balancer_json)


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

3. 进入*slb_monitor.py*所在的目录，执行如下命令，运行查看监控示例。

```
python slb_monitor.py
```

系统显示类似如下：

```
----------create load balancer----------
```

```
{
"VpcId": "",
"AddressIPVersion": "ipv4",
"LoadBalancerName": "SLB1",
"ResourceGroupId": "rg-axxxxxxxxxx",
"VSwitchId": "",
"RequestId": "661DDF49-FD2B-46C1-8E38-FDE5E62C8448",
"Address": "39.xx.xx.xx4",
"NetworkType": "classic",
"LoadBalancerId": "lb-8vbtxxxxxxxxxxx1hn"
}

----------create_tcp_listener----------
{
"RequestId": "7ED453FF-7DF6-4F4F-81A0-670EED9EB997"
}

----------add_backend_servers----------
{
"RequestId": "4B730606-BD44-4A6B-BE9E-118E9606B064",
"BackendServers": {
"BackendServer": [
{
"ServerId": "i-8xxxxxxxxxxf4z",
"Type": "ecs",
"Weight": 100
},
{
"ServerId": "i-8xxxxxxxxxxxxx",
"Type": "ecs",
"Weight": 100
}
]
},
"LoadBalancerId": "lb-8xxxxxxxxxxxxxx"
}

----------describe_metric_last----------
{
"Code": "200",
```

```
"Period": "60",
"RequestId": "15FD2461-FD94-451F-ABE3-D281DB453161",
"Datapoints": "[]"
}

----------------------------put_resource_metric_rule----------------------------
{
"Code": "200",
"Data": "53547DF14168E368073BE8E34E1522DE25854224",
"RequestId": "8652E53F-74AD-413B-B2D9-103BFE557FAC",
"Success": true
}

----------------------------delete_load_balancer----------------------------
{
"RequestId": "D6CCD259-41C2-4D5C-A644-19A0E96F1D4C"
}
```

2.3.5. 克隆实例

介绍如何使用Python SDK克隆SLB实例。

前提条件

在华北3张家口地域已经有两台状态为**运行中**的ECS实例。

在开始运行示例脚本前，您还需确保已完成以下准备工作：

1. 因为使用Python SDK创建SLB实例会产生实例费，所以确保您的阿里云账户金额不少于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。如果尚未获取AK，参见查询获取获取AccessKey。
3. 下载阿里云负载均衡Python SDK场景示例的SLB 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
   ```

背景信息

本次示例分为以下几步，克隆SLB实例：

1. 在华北3张家口地域创建一个名为SLB1的SLB实例，主可用区为cn-zhangjiakou-a，备可用区为cn-zhangjiakou-b，实例计费类型为按量计费，规格为slb.s1.small，其他配置使用系统默认值。
2. 将张家口地域的两台ECS实例添加到默认服务器组，用来分发流量，ECS实例的后端服务权重都为100。
3. 创建TCP监听，SLB实例前端使用的端口为80，后端服务器开放用来接收请求的端口为80，健康检查协议为TCP，不限制监听带宽峰值，其他使用默认值。
4. 查询实例SLB1的信息，作为克隆实例的入参。
5. 克隆新的SLB实例。
6. 删除SLB1实例。
7. 查询克隆实例的信息。
8. 删除克隆实例。

操作步骤

1. 在下载的SDK目录中，打开*$aliyun-openapi-python-sdk-examples\sdk_examples\examples\slb*文件夹。
2. 使用编辑器打开*configration_clone.py*文件，您需要配置用户鉴权参数ACS_CLIENT，其他参数根据实际情况配置后，保存退出。

> **说明** 因为用户鉴权是每个SDK必须执行的操作，所以可以在常量文件中配置AcsClient参数的值，在场景代码中调用常量文件。

```
#encoding=utf-8
"""
创建slb实例->添加tcp监听->查询slb实例->slb实例克隆->删除slb实例
"""
import sys
import json
import uuid


#调用AcsClient参数进行身份验证
from aliyunsdkcore.client import AcsClient
#使用阿里云官方sdk的异常处理模块
from aliyunsdkcore.acs_exception.exceptions import ServerException, ClientException
#命令行打印日志
from sdk_lib.common_util import CommonUtil
#异常处理逻辑
from sdk_lib.exception import ExceptionHandler
#调用创建SLB实例API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerRequest
#调用添加默认服务器组API
```

```
from aliyunsdkslb.request.v20140515 import AddBackendServersRequest
#调用添加TCP监听API
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerTCPListenerRequest
#调用查询SLB实例信息API
from aliyunsdkslb.request.v20140515 import DescribeLoadBalancerAttributeRequest
#调用创建SLB实例API克隆实例
from aliyunsdkslb.request.v20140515 import CreateLoadBalancerRequest
#调用删除SLB实例API
from aliyunsdkslb.request.v20140515 import DeleteLoadBalancerRequest

#用户身份验证参数配置
ACS_CLIENT = AcsClient(
'your-access-key-id', #your-access-key-id
'your-access-key-secret', #your-access-key-secret
'cn-zhangjiakou', #your-region-id
)
#失败重试次数
TRY_TIME = 3

def create_load_balancer(params):
'''
create_load_balancer：创建slb实例

'''
try:
#设置创建SLB实例的调用参数
request = CreateLoadBalancerRequest.CreateLoadBalancerRequest()
request.set_MasterZoneId(params["master_zone_id"])
request.set_SlaveZoneId(params["slave_zone_id"])
request.set_LoadBalancerName(params["load_balancer_name"])
request.set_PayType(params["pay_balancer_type"])
request.set_LoadBalancerSpec(params["load_balancer_spec"])
request.set_ClientToken(str(uuid.uuid1()))
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
```

```
ExceptionHandler.client_exception(e)

def add_backend_servers(params):
'''
add_backend_servers：添加后端服务器

'''
counter = 0
while counter < TRY_TIME:
try:
#设置添加默认服务器组的调用参数
request = AddBackendServersRequest.AddBackendServersRequest()
#负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
#要添加的后端服务器列表
request.set_BackendServers(params["backend_servers"])
request.set_ClientToken(str(uuid.uuid1()))
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
if ExceptionHandler.server_exception(e):
return e
except ClientException as e:
if ExceptionHandler.client_exception(e):
return e
finally:
counter += 1

def create_tcp_listener(params):
'''
create_tcp_listener：创建tcp监听

'''
counter = 0
while counter < TRY_TIME:
try:
#设置创建TCP监听的调用参数
request = CreateLoadBalancerTCPListenerRequest.CreateLoadBalancerTCPListenerRequest()
```

```
request = CreateLoadBalancerTCPListenerRequest.CreateLoadBalancerTCPListenerRequest()
#负载均衡实例的ID
request.set_LoadBalancerId(params["load_balancer_id"])
#负载均衡实例前端使用的端口
request.set_ListenerPort(params["listener_port"])
#负载均衡实例后端使用的端口
request.set_BackendServerPort(params["backend_server_port"])
#监听的健康检查协议
request.set_HealthCheckType(params["listener_health_check"])
#设置监听的带宽峰值
request.set_Bandwidth(params["listener_bandwidth"])
request.set_ClientToken(str(uuid.uuid1()))
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
if ExceptionHandler.server_exception(e):
return e
except ClientException as e:
if ExceptionHandler.client_exception(e):
return e
finally:
counter += 1

#查询slb实例
def describe_load_balancer_attribute(params):
'''
describe_load_balancer_attribute：查询指定负载均衡实例的详细信息

'''
counter = 0
while counter < TRY_TIME:
try:
request = DescribeLoadBalancerAttributeRequest.DescribeLoadBalancerAttributeRequest()
#负载均衡实例的id
request.set_LoadBalancerId(params["load_balancer_id"])
response = ACS_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)
finally:
counter += 1

#克隆slb实例
def create_load_balancer_clone(params):
'''
create_load_balancer：创建slb实例

'''
counter = 0
while counter < TRY_TIME:
try:
request = CreateLoadBalancerRequest.CreateLoadBalancerRequest()
#主可用区
request.set_MasterZoneId(params["master_zone_id"])
#备可用区
request.set_SlaveZoneId(params["slave_zone_id"])
#付费类型
request.set_PayType(params["pay_balancer_type"])
#资源组id
request.set_ResourceGroupId(params["resource_group_id"])
#负载均衡实例的ip版本
request.set_AddressIPVersion(params["address_ip_version"])
#负载均衡实例的网络类型
request.set_AddressType(params["address_type"])
#实例名称
request.set_LoadBalancerName(params["load_balancer_name"])
request.set_ClientToken(str(uuid.uuid1()))
response = ACS_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)
finally:
counter += 1
```

```
def delete_load_balancer(load_balancer_id):
'''
delete_load_balancer：删除slb实例

'''
try:
#设置删除SLB实例的调用参数
request = DeleteLoadBalancerRequest.DeleteLoadBalancerRequest()
#负载均衡实例的ID
request.set_LoadBalancerId(load_balancer_id)
#发送调用请求并接收返回数据
response = ACS_CLIENT.do_action_with_exception(request)
response_json = json.loads(response)
#返回结果
return response_json
except ServerException as e:
ExceptionHandler.server_exception(e)
except ClientException as e:
ExceptionHandler.client_exception(e)

def main():
params = {}
#设置创建SLB实例的参数
#设置新建SLB实例的主可用区为cn-zhangjiakou-a
params["master_zone_id"] = "cn-zhangjiakou-a"
#设置新建SLB实例的主可用区为cn-zhangjiakou-b
params["slave_zone_id"] = "cn-zhangjiakou-b"
#设置新建SLB实例的名称为SLB1
params["load_balancer_name"] = "SLB1"
#设置新建SLB实例的计费类型为按量计费
params["pay_balancer_type"] = "PayOnDemand"
#设置新建SLB实例的规格为slb.s1.small
params["load_balancer_spec"] = "slb.s1.small"

#设置添加到默认服务器组的ECS的实例ID和权重
params["backend_servers"] = [{"ServerId":"i-8vxxxxx","Weight":"100"},{"ServerId":"i-8vbe8xxxxxxxv
","Weight":"100"}]

#设置添加TCP监听的参数
#前端使用的端口为80
params["listener_port"] = 80
```

```
params["listener_port"] = 80
#后端服务器开放用来接收请求的端口为80
params["backend_server_port"] = 80
#健康检查协议为TCP
params["listener_health_check"] = "tcp"
#TCP监听的带宽峰值为-1，即不限制带宽峰值
params["listener_bandwidth"] = -1

#创建slb实例
#获取create_load_balancer函数返回值，load_balancer_json为结果的json串
result_json = create_load_balancer(params)
#打印 load_balancer_json结果，其中"create_load_balancer"是对json串起的名字
CommonUtil.log("create_load_balancer", result_json)

#读取新建SLB实例的ID和名称
load_balancer_id = result_json["LoadBalancerId"]
params["load_balancer_id"] = load_balancer_id
params["LoadBalancerName"] = result_json["LoadBalancerName"]

#创建tcp监听
result_json = create_tcp_listener(params)
CommonUtil.log("create_tcp_listener", result_json)

#查询slb实例
result_json = describe_load_balancer_attribute(params)
CommonUtil.log("describe_load_balancer_attribute", result_json)
#需要显示的查询新建SLB实例信息
params["master_zone_id"] = result_json["MasterZoneId"]
params["slave_zone_id"] = result_json["SlaveZoneId"]
params["pay_balancer_type"] = result_json["PayType"]
params["resource_group_id"] = result_json["ResourceGroupId"]
params["address_ip_version"] = result_json["AddressIPVersion"]
params["address_type"] = result_json["AddressType"]
params["load_balancer_name"] = result_json["LoadBalancerName"]

#根据SLB1的配置，克隆一个新的实例
result_json = create_load_balancer_clone(params)
CommonUtil.log("create_load_balancer_clone", result_json)

#读取克隆实例ID
clone_load_balancer_id = result_json["LoadBalancerId"]
```

```
#删除第一个实例
#删除返回的LoadBalancerId对应的SLB实例
result_json = delete_load_balancer(load_balancer_id)
#打印 load_balancer_json结果
CommonUtil.log("delete_load_balancer", result_json)

#查询克隆出来的实例信息
params["load_balancer_id"] = clone_load_balancer_id
result_json = describe_load_balancer_attribute(params)
CommonUtil.log("describe_load_balancer_attribute", result_json)


#删除克隆出来的实例
#删除返回的LoadBalancerId对应的SLB实例
result_json = delete_load_balancer(clone_load_balancer_id)
#打印 load_balancer_json结果
CommonUtil.log("delete_load_balancer", result_json)

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

3. 进入*configration_clone.py*所在的目录，执行如下命令，运行克隆实例示例。

```
python configration_clone.py
```

系统显示类似如下：

```
----------------------------create_load_balancer----------------------------
{
"VpcId": "",
"AddressIPVersion": "ipv4",
"LoadBalancerName": "SLB1",
"ResourceGroupId": "rg-acfxxxxxxxxxxy",
"VSwitchId": "",
"RequestId": "544656CD-4DC9-47CB-B4DA-9371C0098F37",
"Address": "39.xx.xx.xx",
"NetworkType": "classic",
"LoadBalancerId": "lb-8vbermxxxxxxxxx90l"
}

----------------------------create_tcp_listener----------------------------
```

```
{
"RequestId": "4FCAFA1B-A6A8-4AA7-AB26-788C4F1506DA"
}

----------describe_load_balancer_attribute----------
------
{
"LoadBalancerStatus": "inactive",
"HasReservedInfo": "false",
"InternetChargeType": "paybytraffic",
"EndTime": "2999-09-08T16:00:00Z",
"VpcId": "",
"RegionIdAlias": "cn-zhangjiakou",
"RegionId": "cn-zhangjiakou",
"ListenerPortsAndProtocal": {
"ListenerPortAndProtocal": [
{
"ListenerPort": 80,
"ListenerProtocal": "tcp"
}
]
},
"ResourceGroupId": "rg-acfxxxxxxxaiy",
"CreateTimeStamp": 1551254179000,
"VSwitchId": "",
"Address": "39.98.97.43",
"AddressIPVersion": "ipv4",
"LoadBalancerSpec": "slb.s1.small",
"EndTimeStamp": 32493801600000,
"ListenerPortsAndProtocol": {
"ListenerPortAndProtocol": [
{
"ListenerProtocol": "tcp",
"ListenerPort": 80
}
]
},
"LoadBalancerId": "lb-8vxxxxxxxxxx90l",
"AddressType": "internet",
"RequestId": "9D696BEF-18C2-4EFD-92F4-434580964A51",
"BackendServers": {
```

```
"BackendServer": []
},
"MasterZoneId": "cn-zhangjiakou-a",
"PayType": "PayOnDemand",
"SlaveZoneId": "cn-zhangjiakou-b",
"Bandwidth": 5120,
"LoadBalancerName": "SLB1",
"NetworkType": "classic",
"CreateTime": "2019-02-27T07:56:19Z",
"ListenerPorts": {
"ListenerPort": [
80
]
}
}

----------create_load_balancer_clone----------

{
"VpcId": "",
"AddressIPVersion": "ipv4",
"LoadBalancerName": "SLB1",
"ResourceGroupId": "rg-acxxxxxxxxxxiy",
"VSwitchId": "",
"RequestId": "0E051B90-1C32-4270-A653-F7FBB03E3E5C",
"Address": "xx.xx.xx.xx",
"NetworkType": "classic",
"LoadBalancerId": "lb-8xxxxxxxxxxxxxx"
}

----------delete_load_balancer----------
{
"RequestId": "14703145-BC19-4032-A5F5-6DD49AED6885"
}

----------describe_load_balancer_attribute----------
------
{
"LoadBalancerStatus": "active",
"HasReservedInfo": "false",
```

```
"InternetChargeType": "paybytraffic",
"EndTime": "2999-09-08T16:00:00Z",
"VpcId": "",
"RegionIdAlias": "cn-zhangjiakou",
"RegionId": "cn-zhangjiakou",
"ListenerPortsAndProtocal": {
"ListenerPortAndProtocal": []
},
"ResourceGroupId": "rg-acfxxxxxxxxxiy",
"CreateTimeStamp": 1551254182000,
"VSwitchId": "",
"Address": "39.98.97.36",
"AddressIPVersion": "ipv4",
"MasterZoneId": "cn-zhangjiakou-a",
"EndTimeStamp": 32493801600000,
"ListenerPortsAndProtocol": {
"ListenerPortAndProtocol": []
},
"LoadBalancerId": "lb-8vxxxxxxxxxxx",
"AddressType": "internet",
"RequestId": "6345BAD4-B7EB-4280-9C7B-BB1324A77E32",
"BackendServers": {
"BackendServer": []
},
"PayType": "PayOnDemand",
"SlaveZoneId": "cn-zhangjiakou-b",
"Bandwidth": 5120,
"LoadBalancerName": "SLB1",
"NetworkType": "classic",
"CreateTime": "2019-02-27T07:56:22Z",
"ListenerPorts": {
"ListenerPort": []
}
}

----------delete_load_balancer----------
{
"RequestId": "DC7C11D0-E21E-49AE-9695-F0171960D0F1"
}
```