

BLB 文档



【版权声明】

版权所有©百度在线网络技术（北京）有限公司、北京百度网讯科技有限公司。 未经本公司书面许可，任何单位和个人不得擅自摘抄、复制、传播本文档内容，否则本公司有权依法追究法律责任。

【商标声明】



和其他百度系商标，均为百度在线网络技术（北京）有限公司、北京百度网讯科技有限公司的商标。 本文档涉及的第三方商标，依法由相关权利人所有。未经商标权利人书面许可，不得擅自对其商标进行使用、复制、修改、传播等行为。

【免责声明】

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导。 如您购买本文档介绍的产品、服务，您的权利与义务将依据百度智能云产品服务合同条款予以具体约定。本文档内容不作任何明示或暗示的保证。

目录	
目录	2
功能发布记录	6
2025年	6
2024年	6
2023年	6
产品描述	7
产品介绍	7
基本概念	10
基本架构	11
产品优势	12
产品特性	12
应用场景	15
使用限制	15
性能规格说明	17
产品定价	18
负载均衡实例计费	18
负载均衡专属集群计费	21
快速入门	22
操作指南	28
实名认证	28
普通型BLB实例	29
创建BLB普通型实例	29
管理BLB普通型实例	51
BLB实例带宽变更：	53
应用型BLB实例	54
普通型IPv6 BLB实例	69
扩展域名	78
访问控制	80
访问日志	81
定制配置	83
负载均衡专属集群	85
服务发布点	88
标签管理	90
多用户访问控制	91
BLB监控项说明	95
应用型IPv6实例	103
IP组使用指南	116
负载均衡前后端使用协议对应关系	119
使用VPC流日志记录访问日志	121

Baidu 百度智能云文档	目录
典型实践	125
通过应用型BLB实现多站点托管	125
通过Terraform管理BLB	128
通过Terraform实现BLB关联监听器、后端服务器、安全组等	133
健康检查异常排查	136
场景介绍	136
通过应用型BLB实例获取客户端信息	137
1. 获取源IP	138
配置Proxy Protocol协议获取客户端信息	139
负载均衡安全防护策略	139
BLB选型指南	145
概览	146
产品概述	146
产品类型	146
使用IP组功能在同地域跨VPC挂载后端服务器	147
使用IP组功能在跨地域VPC挂载后端服务器	150
API参考	156
简介	156
API服务域名	156
访问控制	157
接口规范	157
错误码	159
普通型BLB接口	161
普通型BLB实例相关接口	161
UpdateLoadBalancerAcl更新实例ACL功能	171
普通型BLB监听器相关接口	172
普通型BLB后端服务器相关接口	214
普通型BLB安全组相关接口	220
应用型BLB接口	227
应用型BLB实例相关接口	227
应用型BLB监听器相关接口	237
应用型BLB服务器组相关接口	277
应用型BLB安全组相关接口	298
BindEnterpriseSecurityGroups绑定企业安全组	302
普通型IPv6BLB接口	305
普通型IPv6BLB相关接口	305
应用型IPv6BLB接口	306
应用型IPv6BLB实例相关接口	306
应用型IPv6BLB监听器相关接口	316
应用型IPv6BLB服务器组相关接口	353
应用型IPv6BLB安全组相关接口	375

Baidu 百度智能云文档	目录
BindEnterpriseSecurityGroups绑定企业安全组	379
负载均衡专属集群LBDC接口	382
LBDC实例相关接口	382
服务发布点接口	392
服务发布点相关接口	392
BLB订单相关接口	405
BLB退订接口	405
BLB询价	406
BLB计费变更-预付费转后付费	408
BLB-变配	410
BLB计费变更-取消预付费转后付费	411
BLB计费变更-后付费转预付费	412
附录	414
文档更新记录	432
SDK	432
Java-SDK	432
概述	433
安装SDK工具包	433
初始化	434
普通型BLB实例	436
应用型BLB实例	462
IPv6型BLB实例	493
负载均衡专属集群LBDC	495
服务发布点	500
版本更新说明	507
Python-SDK	507
概述	507
安装	507
运行环境	507
初始化	508
确认Endpoint	508
普通型blb实例	509
应用型blb实例	535
IPv6型BLB实例	565
负载均衡专属集群LBDC	567
服务发布点	571
版本更新说明	577
PHP-SDK	577
概述	577
安装	577
初始化	578

Baidu 百度智能云文档	功能发布记录
普通型blb实例	581
GO-SDK	605
概述	605
安装	605
运行环境	605
初始化	606

功能发布记录

2025年

发布时间	功能概述
2025-05	支持超大型4、超大型5新规格
2025-04	监听支持配置连接优雅中断超时时间
2025-04	支持在迁移中心中拉取友商云上负载均衡资源清单
2025-04	支持以VPC内流日志的形式创建访问日志，并使用BLS平台进行分析
2025-03	支持QPS和带宽监控项

2024年

发布时间	功能概述
2024-12	API支持计费方式转换、API支持HTTPS类型监听器添加扩展域名
2024-12	支持按百分比查看监控项
2024-09	实例支持按使用量计费（LCU计费）
2024-05	发布了支持国密加密算法的专属集群
2024-04	IP组，支持IP格式定义的目标组
2024-02	应用型IPv6实例，支持IPv6 to IPv6的访问转发

2023年

发布时间	功能概述
2023-12	普通型BLB、应用型BLB、IPv6型BLB、LBDC、服务发布点 Java-SDK 增加接口示例。
2023-10	负载均衡实例支持 性能规格实例 ，满足您对BLB实例性能可预期的需求。
2022-07	负载均衡实例支持 定制配置 ，允许用户设置BLB 7层监听的配置参数，如 client_header_timeout、client_body_timeout、proxy_buffer_size、proxy_buffers、proxy_set_header等，满足您的定制配置需求。
2021-11	新增 负载均衡专属集群 ，提供性能可控、资源独享、物理资源隔离的专属负载均衡服务，满足超高性能和独占资源需求。
2021-09	负载均衡实例支持 安全访问控制 ，通过配置负载均衡所在子网对应的ACL或者关联安全组，实现负载均衡实例级别的访问控制。
2021-09	普通型 BLB 监听器、应用型服务器组 -- UDP 端口增加 ICMP 健康检查类型。
2021-09	新增 DescribeAppAllListeners 、 DescribeAllListeners 接口，通过此接口可以查询所有监听。
2021-09	新增 LBDC实例相关接口 ，包括创建、更新、升级、查询详情。
2020-03	负载均衡实例支持 扩展域名 ，HTTPS监听支持挂载多个证书，通过配置扩展域名，实现将来自不同访问域名的请求转发至不同的后端服务器或者后端服务器组。
2018-09	负载均衡实例支持 多用户访问控制 。多用户访问控制功能实现了多用户协同开发，项目管理者可以基于应用或者环境为其他开发测试人员开放查看、代码管理及部署等权限。

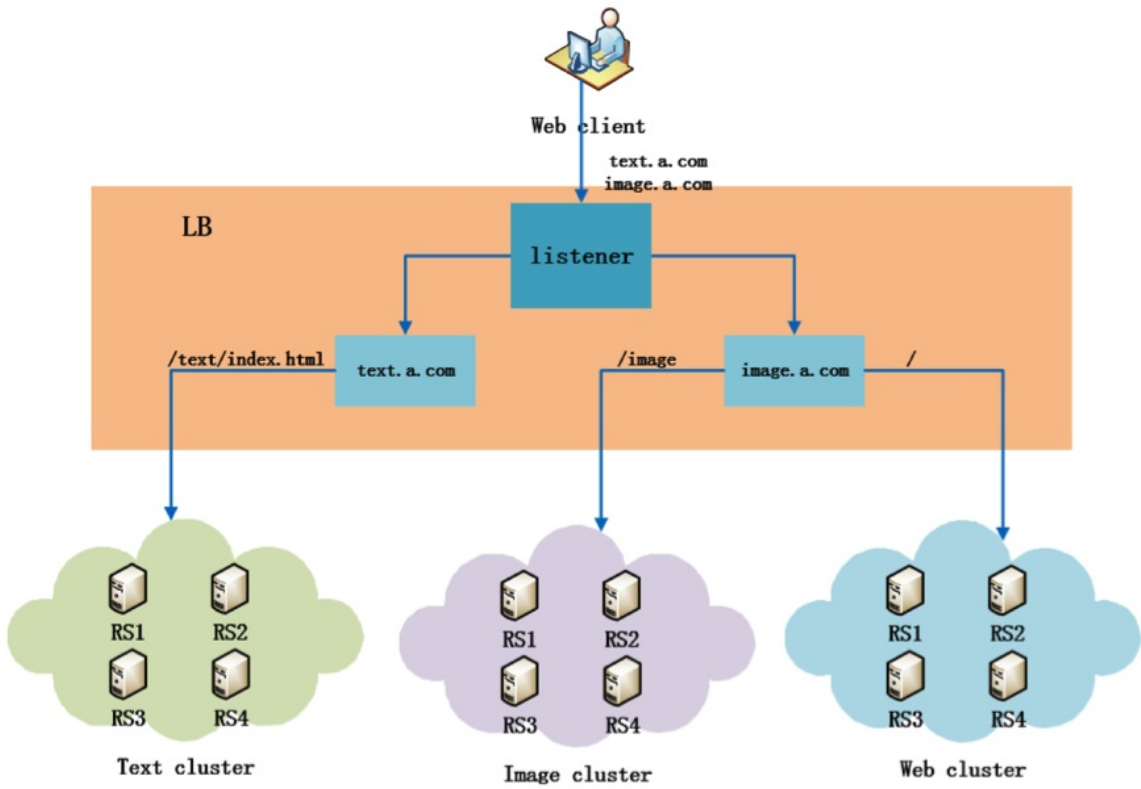
产品描述

产品介绍

负载均衡实例

百度负载均衡BLB（Baidu Load Balance）通过将同一地域的多台百度智能云服务器虚拟成一个组，设置一个内网或外网的服务地址，将前端并发访问转发给后台多台云服务器，实现应用程序的流量均衡，性能上实现业务水平扩展。负载均衡还通过故障自动切换及时地消除服务的单点故障，提升服务的可用性。

现在新推出应用型BLB实例，其在解决用户某一场景时的部署架构如下：



实例类型对比：

实例类型	说明	区别
普通型 BLB	提供了负载均衡常规功能组件，包括监听与后端服务器，能够满足用户对负载均衡的通用需求，可以将来自互联网或内网的IPv4或IPv6流量按一定负载均衡协议分发至多台IPv4后端服务器，具备轻量化、配置简单、价格便宜等特点。	按实例维度组织后端服务器，每个监听器可转发至该实例的所有后端服务器，且对应监听的后端服务器后端端口相同。 流量转发方式：监听器统一调度，支持加权轮询、加权最小连接数、源IP的流量转发。
应用型 BLB	业务场景主要面向复杂应用部署架构，特别是大型网站架构。使用基于策略的流量转发框架构建，可以将来自互联网或内网的IPv4或IPv6流量基于特定策略分发至后端多个IPv4服务器组，实现业务驱动 的流量负载均衡。	支持服务器组，可以按监听器维度组织后端服务器，每个监听器可以通过策略对应不同的后端服务器组，IP组场景支持同一服务器组不同服务器设置不同后端端口。 流量转发方式：按策略转发，支持根据域名/URL/源IP/Cookie等做流量转发, 同时支持策略设置权重。
普通型 IPv6 BLB	提供了负载均衡常规功能组件，包括监听与后端服务器，能够满足用户对负载均衡的通用需求，可以将来自互联网或内网的IPv6流量按一定负载均衡协议分发至多台IPv6后端服务器，具备轻量化、配置简单等特点。	按实例维度组织后端服务器，每个监听器可转发至该实例的所有后端服务器，且对应监听的后端服务器后端端口相同。 流量转发方式：监听器统一调度，支持加权轮询、加权最小连接数、源IP的流量转发。仅支持将前端的IPv6流量分发至后端IPv6服务器。
应用型 IPv6BLB	业务场景主要面向复杂应用部署架构，特别是大型网站架构。使用基于策略的流量转发框架构建，可以将来自互联网或内网的IPv6流量基于特定策略分发至后端多个IPv6服务器组，实现业务驱动 的流量负载均衡。	支持服务器组，可以按监听器维度组织后端服务器，每个监听器可以通过策略对应不同的后端服务器组，IP组场景支持同一服务器组不同服务器设置不同后端端口。 流量转发方式：按策略转发，支持根据域名/URL/源IP/Cookie等做流量转发, 同时支持策略设置权重。仅支持将前端的IPv6流量分发至后端IPv6服务器

实例规格对比：

实例规格	性能上限
标准型1	并发连接数：50,000，每秒新建连接数：5,000，每秒查询数：5,000，带宽上限：1Gbps
标准型2	并发连接数：100,000，每秒新建连接数：10,000，每秒查询数：10,000，带宽上限：2Gbps
增强型1	并发连接数：200,000，每秒新建连接数：20,000，每秒查询数：20,000，带宽上限：4Gbps
增强型2	并发连接数：500,000，每秒新建连接数：50,000，每秒查询数：30,000，带宽上限：6Gbps
超大型1	并发连接数：1,000,000，每秒新建连接数：100,000，每秒查询数：50,000，带宽上限：10Gbps
超大型2	并发连接数：2,000,000，每秒新建连接数：200,000，每秒查询数：100,000，带宽上限：20Gbps
超大型3	并发连接数：4,000,000，每秒新建连接数：400,000，每秒查询数：200,000，带宽上限：40Gbps
超大型4	并发连接数：6,000,000，每秒新建连接数：600,000，每秒查询数：300,000，带宽上限：60Gbps
超大型5	并发连接数：8,000,000，每秒新建连接数：800,000，每秒查询数：800,000，带宽上限：80Gbps

负载均衡专属集群

负载均衡专属集群提供将负载均衡实例部署在用户专属的负载均衡集群的能力，以支持性能可控、资源独享、物理资源隔离的目的，满足用户对负载均衡实例超高性能和独占资源需求。

说明：若未购买负载均衡专属集群，默认创建的BLB实例在共享集群，多租户性能共享，当出现资源争抢时，性能可能会受到限制。若需要超高性能和独占资源，建议您使用负载均衡专属集群。

负载均衡专属集群与负载均衡共享集群的区别：

对比项	说明	专属集群	共享集群
配置独享	独享特殊业务要求配置	支持	不支持
流量隔离	与其他用户的业务流量隔离	支持	不支持
精准导流	将访问的业务流量精确导流至此集群	支持	不支持
业务隔离	与其他用户的业务部署隔离	支持	不支持
集群性能	整体集群的性能上线	支持指定性能规格和一键扩容升级	集群性能存在上限，遇到性能瓶颈多用户相互争抢
集群监控	以集群为粒度监控业务水位	支持	不支持
关联实例	支持查看集群上的全部实例	支持	不支持
价格		按集群收费，预付费形式，其上的BLB实例不再收费	仅收BLB配置费用

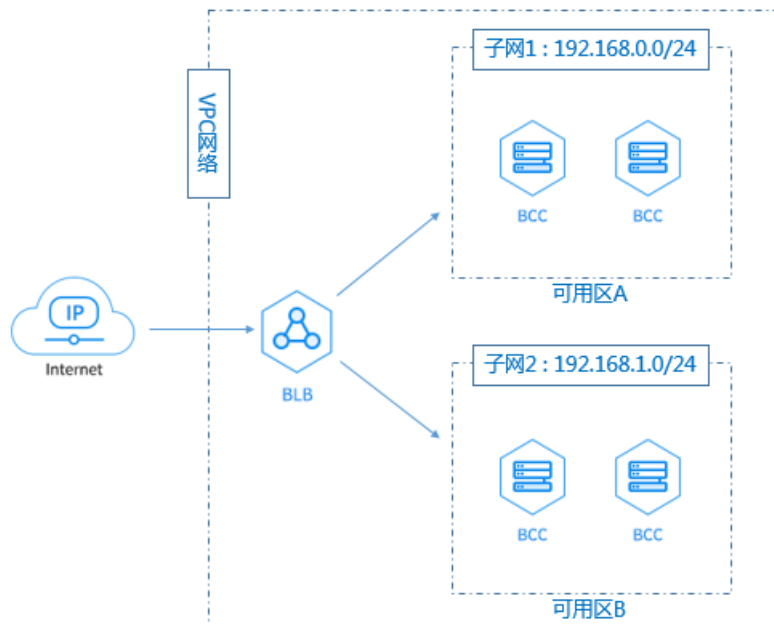
基本概念

🔗 基本概念：

- 负载均衡实例：由百度负载均衡集群提供的虚拟服务。
- 监听器：监听器：监听器用于将用户请求转发至后端服务器，再把后端服务器的响应返回给用户。监听器配置有监听协议和监听端口，当前支持TCP、UDP、HTTP、HTTPS、SSL等多种协议，并按照转发规则将请求转发给特定后端服务器。监听器还提供了基于后端服务器的健康检查端口的健康检查功能。
- 服务地址：负载均衡对外提供服务的访问地址。BLB实例默认提供内网IP作为服务地址，也可以通过开启负载均衡实例的公网访问获得公网IP，对公网流量进行负载均衡。
- 后端服务器：负责接收前端并发访问的一组百度智能云服务器，前端的并发访问会根据BLB实例定义的规则转发到后端服务器进行处理。
- VPC：私有网络VPC(Virtual Private Cloud) 是一个用户能够自定义的虚拟网络。BLB是一个VPC级别的资源，归属于某个VPC。

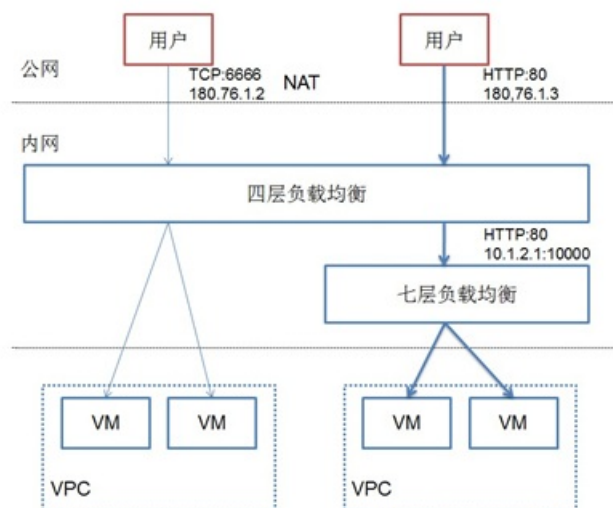
🔗 BLB与VPC的从属关系包括：

- BLB的后端服务器需要是该VPC内的计算资源：云服务器 BCC、专属服务器 DCC、弹性裸金属服务器 BBC、弹性网卡ENI等
- 只有该VPC内的实例才能通过BLB的内网服务地址访问这个BLB实例。如果用户通过专线、VPN、对等连接等将VPC和其他网络打通，则其他网络内的实例也可通过BLB的内网服务地址访问这个BLB实例



基本架构

BLB通过集群的方式提供负载均衡服务，当负载均衡系统集群中的某一台服务器出现故障时，其他服务器将自动接管故障服务器的工作，从而保障服务高可用。BLB基本架构如下：



以TCP监听器为例介绍一个访问请求的处理流程：

1. 用户发送请求即向服务的监听地址和端口发起TCP握手请求；
2. 在四层负载均衡中，查找到用户配置的所有后端云服务器的地址，然后再根据配置的负载均衡策略（加权轮询、加权最少连接数、源IP），选择一个当前可用的云服务器，向其发起TCP握手请求。
3. 连接成功建立后，即完成了一个连接之间的映射变换，从而将前端请求转发到各个云服务器中去。并且在整个BLB工作期间，四层负载均衡设备会不断根据配置对每个云服务器做健康检查，屏蔽不可用的云服务器。

以HTTP监听器为例介绍一个访问请求的处理流程：

1. 用户发送请求即向服务的监听地址和端口发起TCP握手请求；
2. 四层负载均衡会直接将请求转发给七层负载均衡设备，由它向后端云服务器发起TCP握手请求。七层负载均衡在功能上除了完全支持四层之外，还附带获取真实IP、获取监听协议、开启会话保持、启用HTTPS重定向、定制配置等功能。
3. 连接成功建立后，即完成了一个连接之间的映射变换，从而将前端请求转发到各个云服务器中去。并且在整个BLB工作期间，七层负载均衡设备会不断根据配置对每个云服务器做健康检查，屏蔽不可用的云服务器。

产品优势

☞ 高可用

百度负载均衡基于多层次容灾设计，为业务系统提供企业级高可用保障：

- 百度负载均衡BLB（Baidu Load Balance），采用自研4/7层负载均衡器，通过多机房与容灾实例的集群化部署，提供99.95%的服务可用性保证。百度负载均衡专属集群LBDC（Load Balance Dedicated Cluster）提供99.975%的服务可用性保证。
- BGP多线接入，覆盖三大运营商骨干网节点。动态检测线路延迟/丢包率，自动规避网络拥塞节点，为在全国不同运营商不同地域下的快速稳定访问，提供了有力保障。

☞ 高性能

- BLB性能规格实例，单实例（非单集群）最高支持400万并发连接，40万每秒新建连接与40Gbps带宽，轻松应对电商、社区、游戏等业务场景。
- LBDC四层负载均衡集群，单个集群性能容量单位CCU支持8000万并发连接，1200万新建连接与120Gbps集群带宽，可满足超大规模业务负载需求。

☞ 弹性扩展

- BLB无缝扩容，支持与BCC云服务器实例动态绑定，通过水平扩展计算资源实现服务能力的线性增长，业务升级零感知。
- LBDC灵活调整，支持CCU规格扩展升级，以满足超大规模业务增长需求。

☞ 低成本

- 通过OpenAPI与可视化控制台提供标准化的云服务，免除传统负载均衡硬件采购、部署及运维人力成本。
- 提供丰富的实例规格产品线，可根据业务特点按需采购，通过与多种支付方式组合使用，有效优化总拥有成本。

产品特性

启用公网访问

BLB实例支持启用公网访问：

- 内网负载均衡模式：关闭公网访问（默认），通过自动分配的内网IP地址，实现私有网络流量负载。该模式适用于内网各系统间的业务交互，如微服务架构下的内部API调用、数据库集群访问等场景。
- 公网访问模式：开启公网访问，系统自动实现公网IP的购买、绑定及安全防护配置。通过绑定弹性公网IP资源，构建面向互联网应用的流量负载均衡系统。该模式适用于Web门户、移动应用后台、电商平台等需要公网暴露的业务场景。

转发规则

BLB支持加权轮询和加权最小连接数、源IP三种转发规则。

- 加权轮询：基于预设的服务器权重比例进行流量分配，将入站请求按照权重值序列依次分发至后端服务器。该算法适用于服务器硬件配置存在差异的场景，可通过权重设置实现性能最优化的负载分配。
- 加权最小连接数：调度算法结合预设权重与实时连接数状态，自动将新请求导向当前活跃连接数最少的后端服务器。此策略能够动态平衡各节点的会话负载压力，特别适用于处理长连接较多的业务场景。
- 源IP：采用一致性哈希算法对客户端源IP地址进行哈希运算，生成固定映射关系将同一源IP的请求持久化分配到指定后端服务器。该策略有效保证会话连续性，适用于需要保持客户端与服务端会话状态的应用场景。

监听器类型

BLB支持4层（TCP、UDP、SSL协议）和7层（HTTP、HTTPS协议）四种监听器类型。BLB的4/7层负载均衡是在对后台的服务

器进行负载均衡时，依据4层的信息或7层的信息来决定怎样转发流量。

TCP监听器负责监听发送至服务地址和监听端口的TCP连接。监听器根据转发规则挑选一台健康的后端服务器建立TCP连接，并对流量进行NAT处理。在这个过程中会有客户到监听器和监听器到后端服务器的两个TCP连接，后续该TCP连接的所有流量都会转发到同一台后端服务器。

HTTP监听器是在4层协议的基础上增加应用层的特征，以HTTP请求为粒度进行转发。在这个过程中会有多个TCP连接：客户到监听器的一个连接和监听器到每个健康后端服务器的连接。通过这样的方式，HTTP监听器确保每个HTTP请求能够转发至相应的后端服务器。

获取访问者的真实IP

- TCP/UDP监听器类型的后端服务器可以直接获得访问者真实IP，不需要单独设置。
- SSL监听器目前不支持获取真实IP。
- HTTP/HTTPS监听器默认开启“获取真实IP”功能。开启“获取真实IP”功能后，BLB会在每个请求的Header里植入“X-Forwarded-For”字段来标识真实IP地址。

注意：当client端发送过来的http请求，不存在X-Forwarded-For请求头时，BLB侧会添加一行X-Forwarded-For: client_ip的请求头；当client端发送过来的http请求，已有X-Forwarded-For 请求头时，BLB侧会将client_ip追加在X-Forwarded-For请求头的末尾。

会话保持

BLB负载均衡服务提供会话保持机制，确保客户端请求在会话周期内持续转发至同一后端服务器实例。该功能根据网络协议层次差异提供以下实现方案：

- 四层监听器会话保持（TCP、UDP、SSL协议），用户可以通过设置监听器的转发规则为源IP转发规则（来自同一个客户端IP的请求始终转发给同一台后端服务器）来实现会话保持。
- 七层监听器会话保持（HTTP、HTTPS协议），提供两种Cookie管理方案实现会话绑定：
 - 主动植入Cookie（推荐方案）：通过在用户的请求中插入标识后端服务器地址的cookie项实现会话保持。
 - 定制Cookie改写：通过识别并改写指定名称的客户端Cookie来标识后端服务器地址实现会话保持。

健康检查

BLB支持主动检查后端服务器的运行状况，当探测到后端服务器工作异常时，会自动从内部转发列表中摒弃该实例；当异常服务器恢复正常时，会自动将其加入到内部转发列表，继续提供服务。健康检查分别支持TCP、HTTP、UDP、ICMP四种探测协议。

- TCP/SSL监听器支持TCP协议的健康检查。
 - 健康检查协议配置为TCP时，如果后端服务器端口访问正常则视为后端服务器正常，否则即为异常。
- UDP监听器支持UDP、ICMP两种健康检查协议。
 - 健康检查协议配置为UDP时，实例根据配置向后端服务器发送特定字符，如果后端接收请求并返回UDP包则视为后端服务器正常，否则即为异常。
 - 健康检查协议配置为ICMP时，实例根据配置向后端服务器发送 ICMP报文，如果云服务器正常且网络可达则视为正常，否则即为异常。
- HTTP/HTTPS监听器支持HTTP、TCP两种健康检查协议。
 - 健康检查协议配置为HTTP时，实例根据配置向后端服务器端口发起HTTP GET请求，如果返回2xx或3xx则判断 后端服务器运行正常，否则即为异常。

- 健康检查协议配置为TCP时，如果后端服务器端口访问正常则视为后端服务器正常，否则即为异常。

注意：可以设置HTTP健康检查的正常状态码，默认返回2xx和3xx即为正常。

针对健康检查BLB提供了建议参数如下：

- 响应超时时间：3s
- 健康检查间隔：3s
- 不健康阈值：3次
- 健康阈值：3次

数据实时监控

BLB提供给您负载均衡实例和后端服务器实时的一些状态信息，帮助用户了解服务的工作情况，包括：

- 实例监控：
 - 活跃连接数
 - 出入向流量
 - 出入向数据包数量
 - 出入向带宽
 - 新建连接数使用率
 - 并发连接数使用率
 - 出入向流量使用率
- 端口监控
 - 健康检查
 - 网络流量
 - 网络数据包
 - 并发连接数
 - 新建连接数
 - 出入向带宽
 - 流入数据包数
 - 流出数据包数
 - 限速流量
 - 限速数据包
 - 丢弃流量
 - 丢弃数据包
 - 丢弃连接数
 - 请求数（仅7层监听提供）

- 每秒查询数（仅7层监听提供）
- 响应时间（仅7层监听提供）
- 响应状态码（仅7层监听提供）
- 响应状态比例（仅7层监听提供）
- 后端服务器响应状态码（仅7层监听提供）
- 后端服务器响应时间（仅7层监听提供）

此外，BLB提供短信和邮件报警服务，当数据不满足用户设定值时，能够及时对用户进行提醒。

应用场景

网站服务能力水平扩展

随着业务扩展和网站流量的增长，单台Web服务器难以满足高并发访问的需求。通过引入负载均衡服务，可以将请求分配至多台Web服务器，实现服务能力的水平扩展，从而有效应对业务增长。

服务高可用

当服务部署在单一主机上时，网络故障、硬件故障或程序错误等因素可能导致服务中断。为了提高服务的可用性，可以采用负载均衡方案，并在后端配置多个相同的云服务器。这种方式确保即使单个模块发生故障，也不会影响整体服务的正常运行，从而显著提升服务的稳定性和可用性。

公网BLB典型使用场景

您可以通过将弹性IP（EIP）绑定到BLB，实现公网流量的负载均衡。公网BLB的一些典型应用场景包括：

- **社区门户网站**：当单个BCC实例面临访问压力时，可通过添加多个BCC实例并配置负载均衡来分散流量。
- **移动应用开发**：通过部署多个BCC实例和负载均衡，在各实例上安装相同的更新包，以支持应用更新及应对用户增长的需求。
- **电商网站促销**：在进行促销活动期间，电商网站可能会遇到瞬时流量高峰。此时，通过部署多台BCC实例与负载均衡，并设置带宽峰值限制，可以有效缓解服务器压力，确保网站稳定性。

内网BLB典型使用场景

BLB默认提供内网服务地址，适用于多种内部架构场景。其典型应用包括：

- **多层应用架构**：在具有多层结构的应用中，比如前端Web服务器连接到公网，而后端数据库服务器仅允许Web服务器访问的情况下，可以在Web服务器层和数据库服务器层之间部署私网BLB，用于均衡来自Web层到数据库层的访问请求。

使用限制

配额

BLB 支持的产品配额详情如下，如需申请更多配额，请到[配额管理](#)申请。

描述	限额
每个账号可创建的BLB实例数量	60
每个BLB实例可添加的监听数量	50
每个HTTPS监听器可添加的扩展域名数量	50
每个应用型BLB实例可添加的规则数	10
每个应用型BLB实例规则条件数	5
每个应用型BLB实例服务器组数	20
每个应用型BLB实例服务器组内服务器数	50
每个应用型BLB实例IP组数	10
每个应用型BLB实例IP组内成员数	50
每个应用型BLB实例健康检查数	20
每个地域可创建的服务发布点数量	10
每个服务发布点可授权用户的数量	50

🔗 IP组跨VPC场景

注意：同地域跨VPC、跨地域跨VPC可以采用对等连接打通，具体打通方式可以参考 典型实践 ["使用IP组功能在跨地域VPC挂载后端服务器"](#) 和 ["使用IP组功能在同地域跨VPC挂载后端服务器"](#)。注意：7层监听（HTTPS/HTTP/SSL）不支持同地域跨VPC挂载。当前，使用IP组进行不同VPC之间的访问场景支持情况如下：

后端服务IP版本	是否跨地域	负载均衡实例到后端的连接环境	负载均衡监听协议	是否支持
后端为IPv4	同地域	VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
		CSN	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
		对等连接	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
	跨地域	VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
		CSN	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持
		对等连接	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
	IDC	专线	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
		VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
后端为IPv6	同地域	VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
		CSN	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
		对等连接	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
	跨地域	VPN	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持
		CSN	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持
		对等连接	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持
	IDC	专线	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
		VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持

性能规格说明

本篇文档说明负载均衡实例的性能规格限速逻辑。

性能衡量维度说明

负载均衡性能衡量指标有如下几个维度区分：

性能指标	指标说明
并发连接数	单位时间内已建立的总连接数量。
每秒新建连接数	每秒新建连接数定义了新建连接的速率。当新建连接的速率超过规格定义的每秒新建连接数时，新建连接请求将被丢弃。
每秒查询数（QPS）	每秒查询率（QPS，Queries-per-second），即单位时间单位计算资源处理的请求数据量。
带宽上限	带宽是指在单位时间（一般指的是1秒钟）内能传输的数据量，带宽数值越大表示传输能力越强，即在单位时间内传输的数据量越多。

实例说明

性能规格

当前负载均衡有如下规格：

规格	规格指标
标准型1	并发连接数：50,000，每秒新建连接数：5,000，每秒查询数：5,000，带宽上限：1Gbps
标准型2	并发连接数：100,000，每秒新建连接数：10,000，每秒查询数：10,000，带宽上限：2Gbps
增强型1	并发连接数：200,000，每秒新建连接数：20,000，每秒查询数：20,000，带宽上限：4Gbps
增强型2	并发连接数：500,000，每秒新建连接数：50,000，每秒查询数：30,000，带宽上限：6Gbps
超大型1	并发连接数：1,000,000，每秒新建连接数：100,000，每秒查询数：50,000，带宽上限：10Gbps
超大型2	并发连接数：2,000,000，每秒新建连接数：200,000，每秒查询数：100,000，带宽上限：20Gbps
超大型3	并发连接数：4,000,000，每秒新建连接数：400,000，每秒查询数：200,000，带宽上限：40Gbps
超大型4	并发连接数：6,000,000，每秒新建连接数：600,000，每秒查询数：300,000，带宽上限：60Gbps
超大型5	并发连接数：8,000,000，每秒新建连接数：800,000，每秒查询数：400,000，带宽上限：80Gbps

对于所有计费方式的负载均衡，均支持选择如上规格。

不限速规格

当选择的实例为后付费-按使用量计时，支持选择不限速规格。不限速规格的实例不限制使用量，可以无限扩容，但当极端情况下，网络访问量激增时，性能可能会有所损耗。如果有远超过超大型5规格的业务访问量，并希望尽可能保障性能，可提前通过提交工单联系反馈。

产品定价

负载均衡实例计费

计费模式

计费方式	计费维度	计费说明	计费项
预付费	规格	按整月购买实例。	规格费
按使用规格后付费	规格	根据您的使用规格，按使用时间长短实时计费并扣费。	规格费
按使用量后付费	实际发生的连接数、带宽、QPS等使用量	根据您的实际使用量较大的性能维度收取使用量费用	实例费、使用量费（CU费）

预付费说明

🔗 计费规则

- 按购买时长预付费，不同规格的购买价格不同。
- 以月为最小购买单位，购买时长满一年则有包年8.3折优惠。

具体情参见[BLB价格详情](#)，您也可以[在BLB价格计算器](#)中预设BLB配置并查看对应的价格。购买BLB服务时须绑定EIP，有关EIP的产品定价请参考[EIP产品定价](#)。

🔗 按使用规格后付费说明

🔗 计费规则

具体规则如下：

- 按分钟计费，不足1分钟按1分钟计。
- 按小时扣费，即北京时间整点扣费并生成账单。出账单时间是当前计费周期结束后1小时内。例如，10:00-11:00的账单会在12:00之前生成，具体以系统出账时间为准。
- 购买前需保证账户无欠款，且保证账户余额和可用代金券总和大于或等于100元。

🔗 计费公式

费用=使用时长×实例单价×实例个数

🔗 计费价格

不同地域的价格可能存在差异，具体情参见[BLB价格详情](#)，您也可以[在BLB价格计算器](#)中预设BLB配置并查看对应的价格。购买BLB服务时须绑定EIP，有关EIP的产品定价请参考[EIP产品定价](#)。

🔗 余额不足提醒和欠费处理

- 余额不足提醒：
 - 根据您最近3天的账单金额来判断您的账户余额（含可用代金券）是否足够支付未来3天的费用，若不足以支付，系统发送续费提醒。
 - 根据您最近1天的账单金额来判断您的账户余额（含可用代金券）是否足以支付未来1天的费用，若不足以支付，系统发送续费提醒。
- 欠费处理：
 - 北京时间整点检查您的账户余额是否足以支付本次BLB账单的费用（如北京时间11点整检查账户余额是否足以支付10点至11点的账单费用），若不足以支付，即为欠费，欠费时系统会发送欠费通知。
 - 欠费后立即停服，系统会发送欠费停服通知，数据为您保留7天，期间不收取费用，7天内未充值则释放，释放前1天和释放时系统都会发送释放通知。

🔗 按使用量后付费说明

🔗 计费逻辑

按使用量后付费的负载均衡实例会产生如下两个计费项：**实例费** 按使用时长计费：

- 按分钟计费，不足1分钟按1分钟计。
- 按小时扣费，即北京时间整点扣费并生成账单。出账单时间是当前计费周期结束后1小时内。例如，10:00-11:00的账单会在12:00之前生成，具体以系统出账时间为准。

实例费价格：普通型BLB和普通型IPv6BLB价格为0.15元/小时，应用型BLB和应用型IPv6实例为0.2元/小时。

使用量费（CU费） 将实际使用量换算为CU（Capacity Unit），按CU个数计费。

具体CU换算规则如下：

协议	单个 LCU 对应的性能
HTTP/HTTPS 协议	每秒25个新建连接，或 每分钟3000个并发连接，或 每分钟1000个QPS 或 每小时1GB流量
TCP 协议	每秒800个新建连接，或 每分钟10W个并发连接，或 每小时1GB流量
UDP 协议	每秒400个UDP新建流，或 每分钟5W个活跃UDP流，或 每小时1GB流量
TCP SSL 协议	每秒800个新建连接，或 每分钟10W个并发连接，或 每秒50个新建TLS流，或 每分钟3000个活跃TLS流，或 每小时1GB流量

CU费价格：

普通型BLB和普通型IPv6BLB价格为0.04元/CU/小时，应用型BLB和应用型IPv6实例为0.049元/CU/小时。

LCU汇聚逻辑：

对于每实例而言：每个协议的CU个数=各个维度产生的CU数的最大值。

实例产生的CU个数=在该实例上各个协议产生的CU个数之和。

按使用量计费的计费示例

用户购买了1个应用型BLB实例，建立了TCP和HTTP两种监听器，使用了1小时后释放，在1个小时内，用户使用的各个指标如下：

时间	TCP小时内 流量	TCP每分钟并发 连接	TCP每秒新建 连接	HTTP小时内 流量	HTTP每分钟并发 连接	HTTP每秒新建 连接	HTTP每分钟 QPS
第一小时	1G	50000	1600	2G	3000	75	2000
第一小时产生CU 使用量	1CU	0.5CU	2CU	2CU	1CU	3CU	2CU

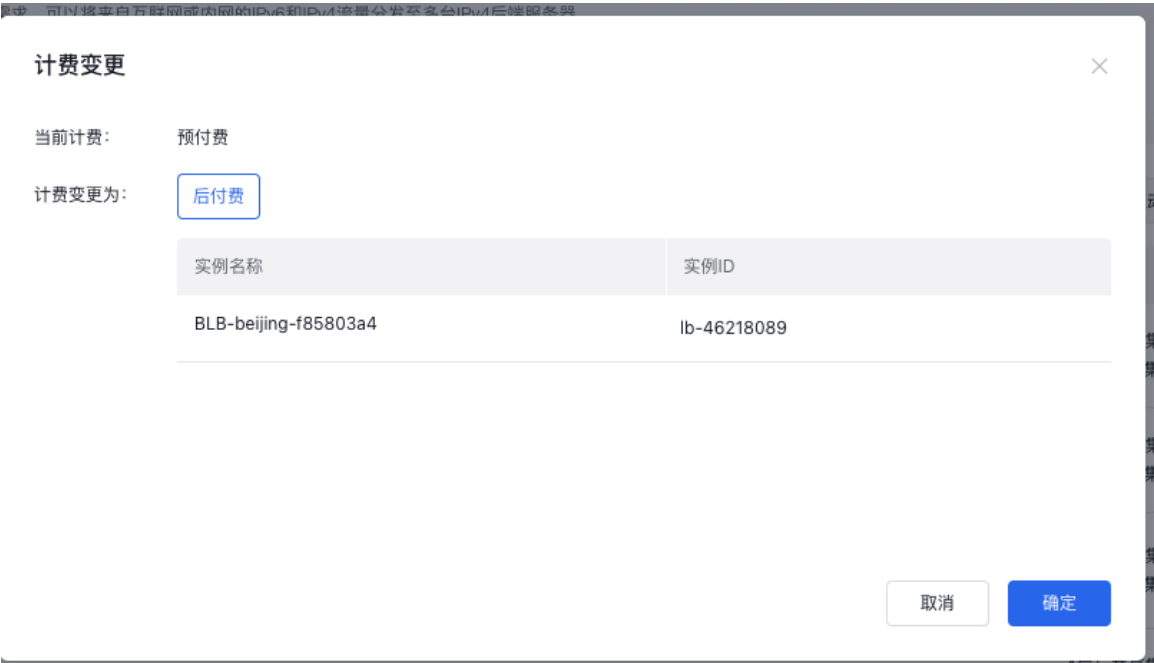
在第一个小时内：

- TCP产生CU消耗为： $\text{Max}(1\text{CU}, 0.5\text{CU}, 2\text{CU})=2\text{CU}$
- HTTP产生CU消耗为： $\text{Max}(2\text{CU}, 1\text{CU}, 3\text{CU}, 2\text{CU})=3\text{CU}$
- 第一小时总计消耗=2CU+3CU=5CU
- 第一小时出账费用为：0.2（实例费）+5*0.049（CU费）=0.445(元)

④ 计费转换

预付费实例和后付费实例，在创建后可以**进行计费转换**。勾选一个希望进行计费转换的实例，在列表上方下拉框中选择计费变





中选择希望的计费方式，点击确定。继续完成订单即可。**预付费转后付费计费规则：**预付费实例转为后付费后，会按照使用情况将剩余未使用的金额退还，详情请参见[计费变更说明文档](#) **转换为LCU计费生效时间** 当计费方式变更为后付费-按使用量计时，需要等待到最近的一个整点时间，账单系统计入后才能生效。

负载均衡专属集群计费

🔗 计费模式

负载均衡专属集群使用集群性能容量单位CCU（Cluster Capacity Unit）来衡量BLB集群处理流量时涉及各个指标。

🔗 计费规则

专属集群收费方式为：包年包月计费，采用预付费方式进行支付。

🔗 计费价格

单个CCU为4台机器，刊例价为：

地域	价格
除武汉外的其他大陆地域	34000元/CCU/月
武汉	44200元/CCU/月
香港	42000元/CCU/月

🔗 性能说明

单个CCU的性能：

- 4层集群：最大并发连接数 8000万、新建连接速度 1200万、入向bps 120Gbps、出向bps 120Gbps。
- 7层集群：最大并发连接数 200万、新建http连接速度 120万、新建https连接速度 22.4万、入向bps 40Gbps、出向bps 40Gbps、http qps 120万、https qps 22.4万。

🔗 下单后搭建时效

- 专属集群下单后到服务可用的搭建时效为3-7天。
- 专属集群下单后，账期开始时间为：服务可用时间。即：服务准备时间不会计入账期

🔗 到期提醒和处理

- 到期提醒：服务到期前7天，系统会给您发送即将到期提醒通知。
- 到期后处理：到期后立即停服，系统会发送欠费停服通知。数据为您保留7天，期间不收取费用，7天内未充值则释放，释放前1天和释放时系统都会发送释放通知。集群到期后，其上的负载均衡实例服务不可用，为避免对业务造成影响，请您关注专属集群的状态。

🔗 专属集群上的实例

- 预付费实例：不允许迁移到专属集群，请转换为后付费后加入专属集群。
- 后付费-按规格计费实例：在专属集群上不另外收取费用。
- 后付费-按使用量计费实例：在专属集群上，会收取实例费，免除使用量费用。

快速入门

如果您需要快速创建一个能公网访问的负载均衡BLB（Baidu Load Balance）实例，并将来自客户端的访问请求转发至后端服务器，请参考本文操作。

🔗 部署思路

1. 创建BCC实例并部署应用服务
2. 创建公网访问的BLB实例
3. 添加后端服务器
4. 添加TCP监听器
5. 公网访问后端服务
6. 查看监控数据

🔗 创建BCC实例并部署应用服务

创建2个包年包月的BCC实例：

1. 登录百度智能云[官网](#)。
2. 登录百度智能云平台：

说明：

- 若没有用户名，请先完成注册，操作请参考[注册](#)
- 若有用户名，登录操作请参考[登录](#)

3. 登录成功后，选择“产品>计算>云服务器 BCC”，点击“立即购买”。
4. 根据页面提示，填写相关配置信息，详情参考[创建BCC实例](#)。

说明：创建BLB实例时可配置公网访问，所以创建BCC实例时不需要选择公网访问。

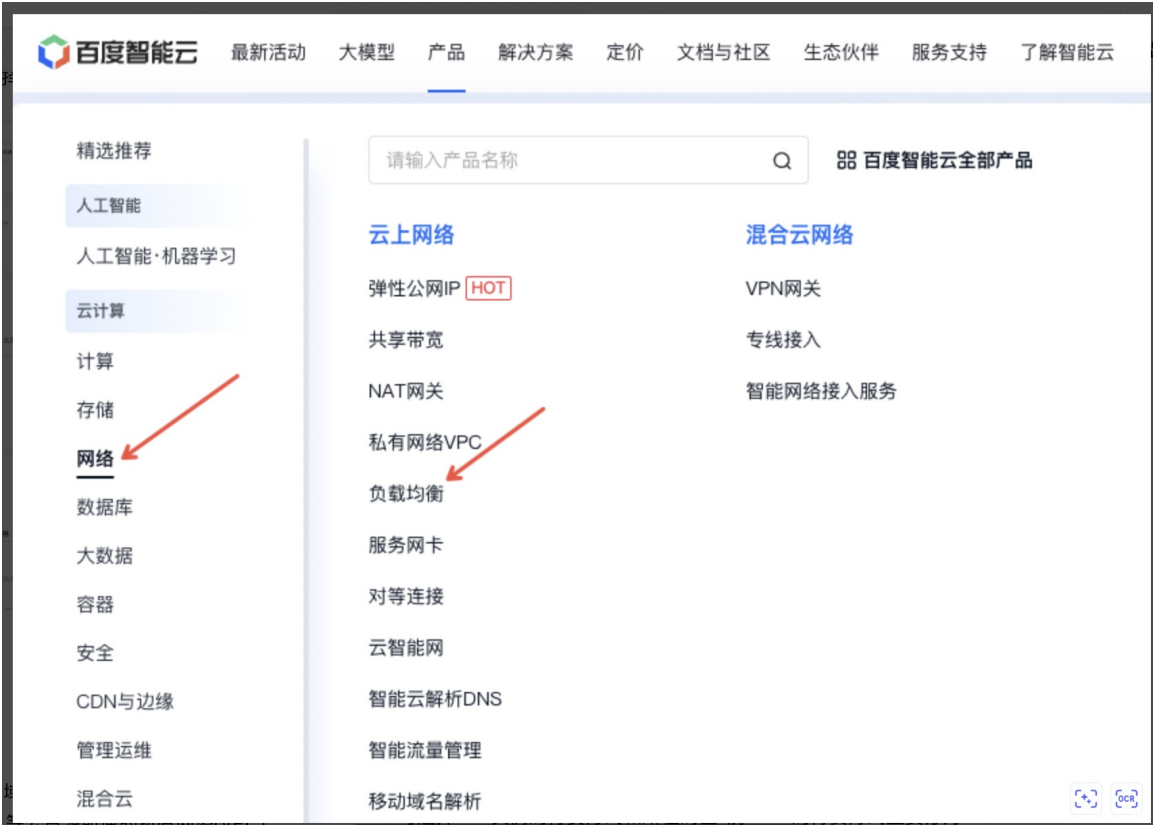
6. 按步骤完成支付后，系统在后台进行云服务器的创建。
7. 进入“实例列表”界面，查看创建实例的状态，创建成功后该实例状态显示为“运行中”。
8. 实例创建成功后，系统将以短信形式提示用户。

9. BCC实例创建成功后，可以使用[SmartTerm](#)远程连接该实例，这里以Centos系统为例，部署nginx应用服务参考命令如下：

```
yum install -y nginx
systemctl start nginx.service
```

🔗 创建公网访问BLB实例

1. 登录控制台，选择“产品>网络>负载均衡”，点击“立即购买”。



2. 选择BLB实例的付费方式及所在地域。



BLB实例的付费方式和所在的区域。BLB的付费方式主要分为预付费和后付费，详见[负载均衡实例计费](#)。

- 说明：
- 通常情况下，推荐BLB实例和后端服务器购买在同一地域，这样配置简单快速且BLB实例与后端服务器的网络时延更低。
 - 如有BLB跨地域挂载后端服务器的需求，则考虑购买应用型BLB实例，具体实践可参考[使用IP组功能在跨地域VPC挂载后端服务器](#)。

- 配置公网访问：可根据实际情况选择“启用”或“不启用”公网访问
 - 如果选择“OFF”，不启用。表示在创建该BLB实例时不绑定公网IP，您可以在购买成功后根据需要绑定公网IP。
 - 如果选择“ON”，启用。表示在创建该BLB实例时购买并绑定公网IP。
- 购买信息：选择购买数量，以及配置是否自动释放。

说明：用户在创建BLB实例时，可直接进行BLB实例[自动释放设置](#)，BLB实例会在设定的时间进行自动释放。

3.选择BLB实例配置信息

基础信息

实例类型：

普通实例

应用实例

普通型IPv6 实例

应用型IPv6 实例

关于各个类型实例的区别详见[说明文档](#)。一旦创建，BLB类型不可更改。[了解详情](#)

名称：

请输入实例名称

0/256

支持大小写字母、数字、中文以及_./特殊字符，必须以字母或者中文开头，长度1~256个字符。

计费方式：

按固定规格计费

实例性能：

性能规格

标准型1

▼

该规格最大并发连接数：50000，每秒新建连接数：5000，每秒查询数：5000，带宽上限：1Gbps

所在网络：

default (192.168.0.0/16)

▼

系统预定义子网 (192.168.0.0/20)

▼

当前可用区A，可用IP共4093个；如需创建子网，您可以到[私有网络-子网](#)去创建

IP地址：

自动分配

指定

安全组：

普通安全组

企业安全组

请选择

▼

描述：

请输入

可填写实例名称，选择BLB实例类型、性能和所在网络等基础信息。

- 实例类型：BLB 各种类型实例的区别详见[BLB选型指南](#)，请注意一旦创建BLB实例类型不可更改。
- 所在网络：指当前创建的 BLB 和相应 BCC 等云资源所属的私有网络(VPC)。

4. 选择其他信息（用户按需可选）

配置公网访问

温馨提示：如不启用公网访问，则默认提供内网IP来进行内网流量的均衡服务；如启用，则为您购买并绑定公网IP来实现对公网访问流量的均衡服务。您也可选择暂不启用，在购买成功后再绑定已有的公网IP。

公网访问：

开启

关闭

购买弹性公网IP

绑定已有弹性公网IP

标签

绑定标签：

请选择或手动输入标签键

请选择或手动输入标签值

+ 添加标签

标签包含键和值两部分，支持按用途、所有者或项目等对资源进行分类，重复的标签会自动去重处理。[帮助文档](#)

资源分组

资源分组： ⓘ

默认分组

▼

如需创建新的资源分组，可前往资源管理 [去创建](#)。更多内容请查看 [帮助文档](#)

配置

公网访问：用户可选，如果未启用公网访问，系统将自动跳过此步骤。另外用户选择开启公网访问后，还可选择购买弹性公网IP或者绑定已有弹性公网IP

说明：

- 选择购买弹性公网IP，创建BLB实例成功后系统会自动创建一个弹性公网IP并绑定到该实例，用户无需额外操作。
- 绑定已有弹性公网IP，适用于用户已有能被绑定的弹性公网EIP场景。

5. 选择购买信息：选择购买的时长、实例数量，以及是否开启自动续费。

购买信息

购买时长：

1个月

2个月

3个月

4个月

5个月

6个月

7个月

8个月

9个月

1年

2年

3年

购买数量：

-

1

+

① 最多可开通60个BLB实例，还可开通60个

自动续费：?

关闭

开启

下一步

配置费用：
¥150.00

已选配置：
查看详情

说明：*用户在创建BLB实例时，可直接进行BLB实例自动释放设置，BLB实例会在设定的时间进行自动释放。

6. 配置公网访问（可选，如果未启用公网访问，系统将自动跳过此步骤） 配置EIP实例信息，有关EIP实例的配置方法，请参看[创建EIP实例](#)。

基本配置

配置公网访问

确认订单

购买须知

备案说明

1. 弹性公网IP（EIP）需绑定在负载均衡（BLB）或预付费云服务器（BCC）上方可进行备案。

2. 若您涉及网站部署并需要使用独立域名，请绑定弹性公网IP（EIP）并进行备案。

付费及地域

付费方式：

包年包月-预付费

先付费后使用，价格更低廉

按量付费-后付费

先使用后付费，按需开通

当前地域：

华北 - 北京

弹性公网IP配置信息

名称：

请输入名称

大小写字母、数字、中文以及-./_特殊字符，必须以字母或者中文开头

线路类型：

标准型BGP

增强型BGP

可购买带宽上限200Mbps，适用于多IP小带宽场景，提供5G DDoS基础原生防护，可以根据业务需要购买流量突发服务包，提升您的防护能力

线路类型选定后不可更改。

地址池：

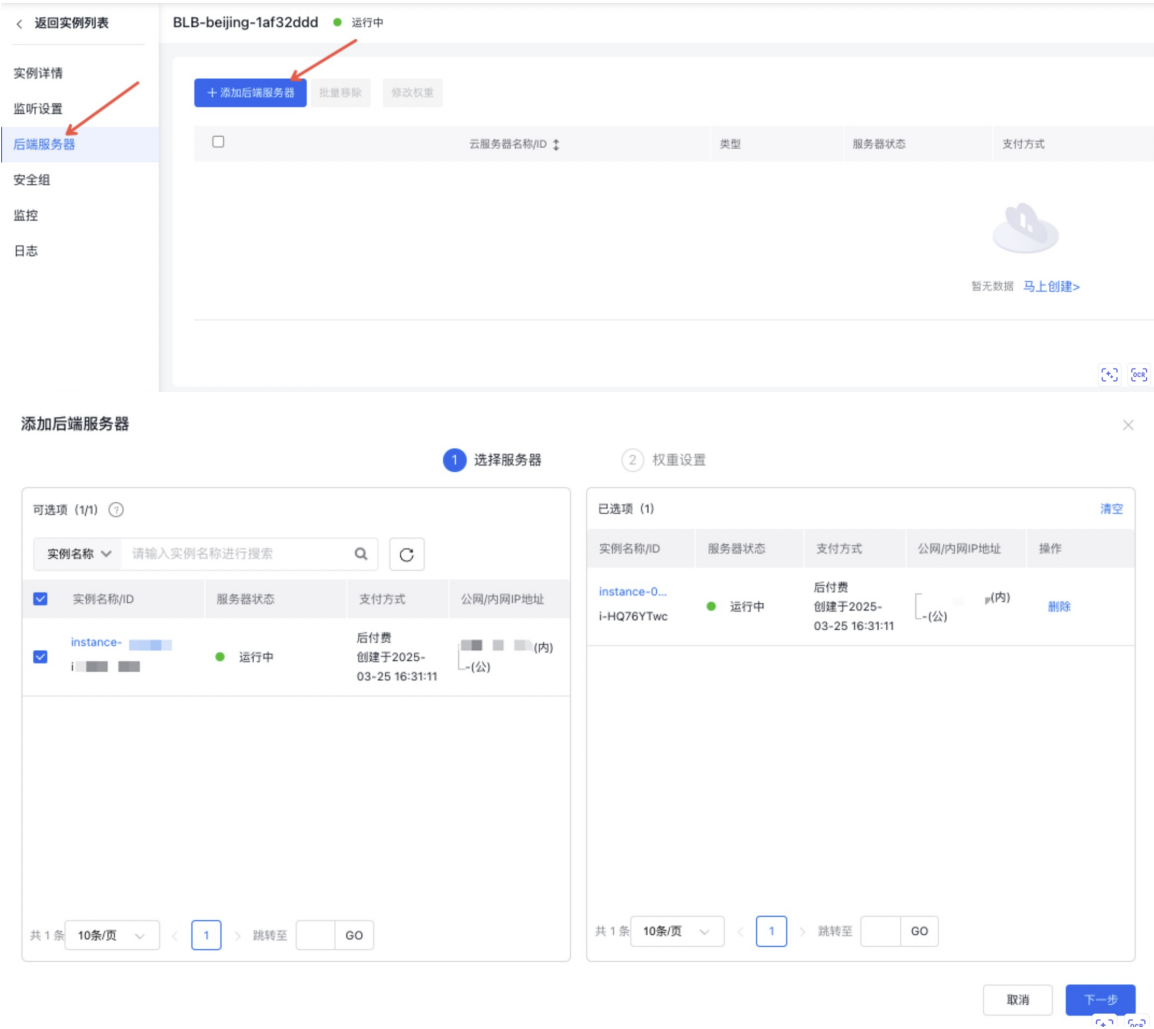
共享地址池

7. 根据系统提示，完成订单确认及支付。创建成功后，用户将收到短信提示。

添加后端服务器

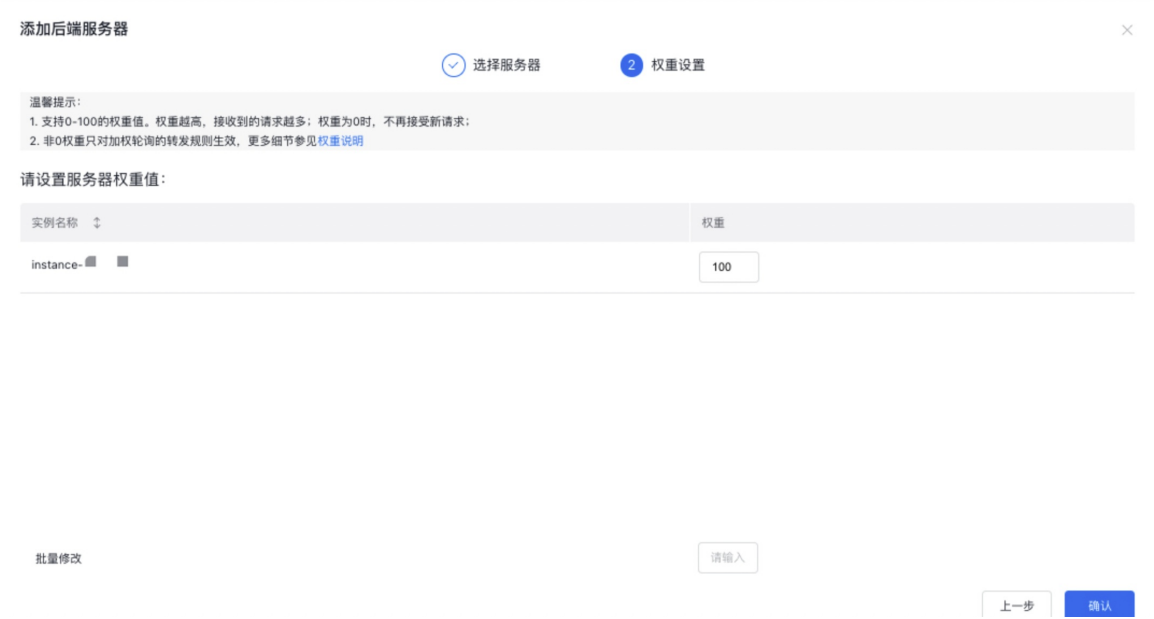
1. 选择“产品服务>负载均衡BLB”，进入负载均衡BLB列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击“后端服务器”页签，添加需要进行负载均衡的BCC服务器。

26



4. 设置BLB实例中BCC服务器的权重。

权重是用来针对不同的后端服务器实例设置不同的请求转发比例。权重值是一个相对值，如果设置成相同数值，其转发比例是一样的。



添加TCP监听器

配置监听包括“配置信息”和“健康检查设置”两部分。

1. 选择“产品>网络>负载均衡”，进入BLB控制台列表，下面以普通型BLB实例添加TCP监听为例进行配置演示。
2. 首先在左边栏选择普通型实例，点击“负载均衡ID/名称”列的ID链接，进入实例详情页。



3. 点击“监听设置”页签，点击“添加监听设置”按钮，下面以配置TCP监听器为例，进行监听器的基本设置和健康检查设置。



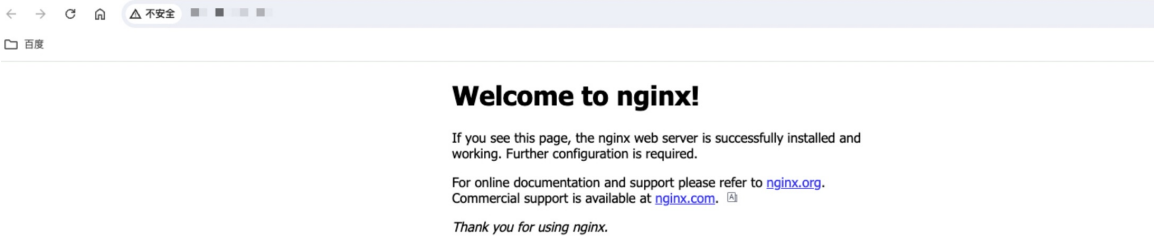
- 配置信息可根据页面提示进行监听器端口的相关设置。
- 健康检查设置可根据页面提示进行设健康检查的相关参数设置。





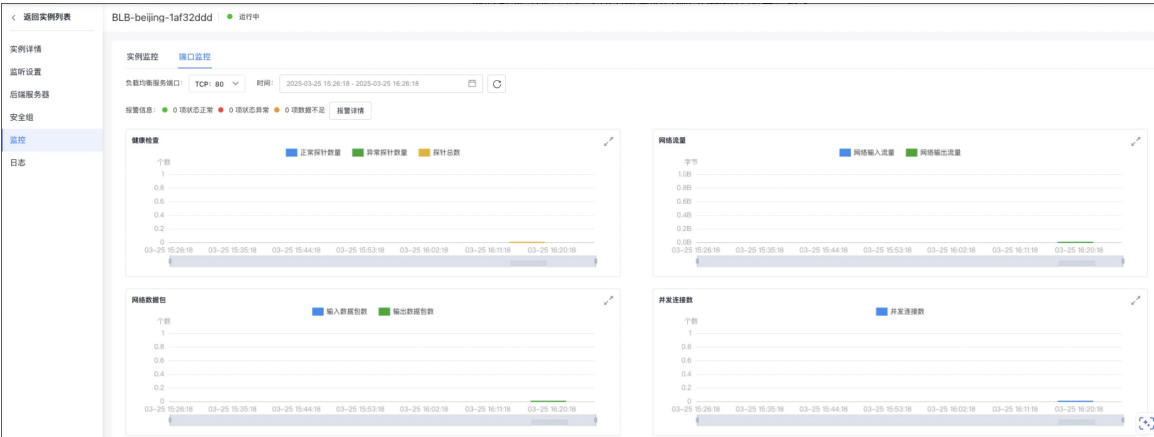
公网访问后端服务

此时可通过浏览器输入BLB绑定的公网EIP地址访问后端服务。



查看监控数据

完成BLB实例、后端服务器及监听器配置后可以查看BLB实例的监控数据。点击BLB的详情页的“实例监控”页签，就可以看到监控的各项图表数据。



操作指南

实名认证

应用场景

所有使用BLB的用户都需要进行实名认证，每个账户只需要认证一次。认证通过以后，才能使用相应的服务。

操作步骤

如果您在创建BLB实例的时候仍未完成实名认证，您可以点击页面上的 认证 提示按钮到实名认证页面完成相关操作。

请根据用户实际情况进入“实名认证”页面进行“企业认证”或者“个人认证”，详细的操作步骤请参见 [个人认证](#)和[企业认证](#)。

普通型BLB实例

创建BLB普通型实例

购买普通型实例

- 1. 登录百度智能云平台，选择“产品导览>负载均衡BLB>普通型实例”，进入“实例列表页面”。
- 2. 点击“创建负载均衡”，进入创建实例界面。

说明：为了方便用户批量对BLB实例进行统一管理如服务地址及后端服务器等，“实例列表”页面支持资源列表下载，下载的资源文件为CSV格式，编码格式为UTF-8，文件命名规范为“产品名地域日期_list.CSV”。所下载的实例与当前实例列表的筛选条件保持一致，内容包含BLB实例的所有字段。



- 3. 选择配置信息。

付费方式：

包年包月-预付费

先付费后使用，价格更低廉

按量付费-后付费

先使用后付费，按需开通

[详见计费说明文档](#)

当前地域：

华北 - 北京

华北 - 保定

西南 - 成都

华南 - 广州

华东 - 苏州

华东 - 上海

华中 - 武汉

中国香港

基础信息

实例类型：

普通实例

应用实例

普通型IPv6实例

应用型IPv6实例

关于各个类型实例的区别详见[说明文档](#)。一旦创建，BLB类型不可更改。[了解详情](#)

名称：

请输入实例名称0/256

支持大小写字母、数字、中文以及_./特殊字符，必须以字母或者中文开头，长度1-256个字符。

计费方式：

按固定规格计费

实例性能：

性能规格

标准型1

该规格最大并发连接数：50000，每秒新建连接数：5000，每秒查询数：5000，带宽上限：1Gbps

所在网络：

默认私有网络（192.168.0.0/16）

系统预定义子网（192.168.0.0/20）

30

配置公网访问

温馨提示：如不启用公网访问，则默认提供内网IP来进行内网流量的均衡服务；如启用，则为您购买并绑定公网IP来实现对公网访问流量的均衡服务。您也可选择暂不启用，在购买成功后再绑定已有的公网IP。

公网访问：

标签

绑定标签：

[+ 添加标签](#)
标签包含键和值两部分，支持按用途、所有者或项目等对资源进行分类。重复的标签会自动去重处理。 [帮助文档](#)

资源分组

资源分组：
如需创建新的资源分组，可前往资源管理 [去创建](#)。更多内容请查看 [帮助文档](#)

购买信息

购买时长：

1年 8.3折 2年 8.3折 3年 8.3折

购买数量：
 ① 最多可开通60个BLB实例。还可开通60个

自动续费：

- 基本信息：可选择区域和设置实例名称。所在网络，指当前创建的 BLB 和相应 BCC 等云资源所属的私有网络(VPC)。

说明：

- 负载均衡 BLB和其绑定的后端服务器 BCC 需要建立在同一个私有网络VPC 下。系统提供默认私有网络，如需自定义 VPC，请参考[创建私有网络VPC](#)。
- 用户在创建BLB实例时需要指定子网。

- 配置公网访问：可根据实际情况选择“启用”或“不启用”公网访问

说明：

- 如果选择“OFF”，不启用。表示在创建该BLB实例时不绑定公网IP，您可以在购买成功后根据需要再绑定公网IP。
- 如果选择“ON”，启用。表示在创建该BLB实例时购买并绑定公网IP。

- 购买信息：预付费时选择购买时长、购买数量，以及配置自动续费。后付费时选择购买数量，以及配置是否自动释放。

说明： 用户在创建BLB实例时，可直接进行BLB实例[自动释放设置](#)，BLB实例会在设定的时间进行自动释放。

(可选，如果未启用公网访问，系统将自动跳过此步骤) 配置EIP实例信息，有关EIP实例的配置方法，请参看[创建EIP实例](#)。

付费及地域

付费方式：

包年包月-预付费

先付费后使用，价格更低廉

按量付费-后付费

先使用后付费，按需开通

当前地域：

华北 - 北京

弹性公网IP配置信息

名称：

请输入名称

大小写字母、数字、中文以及-/_/特殊字符，必须以字母或者中文开头

线路类型：

标准型BGP

增强型BGP

可购买带宽上限200Mbps，适用于多IP小带宽场景，提供5G DDoS基础原生防护，可以根据业务需要购买流量突发服务包，提升您的防护能力

线路类型选定后不可更改。

地址池：

共享地址池

*公网带宽：

按带宽付费

按流量计费

*带宽峰值：

1

Mbps

1Mbps100Mbps200Mbps

单地域所有按使用流量计费EIP的累计购买带宽峰值总和/配额：1Mbps/5000Mbps

按流量计费的EIP带宽峰值不作为业务承诺指标，当出现资源争抢时，可能会受到限制，按固定带宽计费的EIP带宽峰值有保证。

标签

绑定标签：

请选择或手动输入标签键

请选择或手动输入标签值

+ 添加标签

标签包含键和值两部分，支持按用途、所有者或项目等对资源进行分类，重复的标签会自动去重处理。[帮助文档](#)

资源分组

资源分组：?

默认分组

如需创建新的资源分组，可前往资源管理 [去创建](#)，更多内容请查看 [帮助文档](#)

购买信息

购买个数：

1

释放保护：

开启

关闭

上一步

立即购买

IP闲置费用：EIP绑定云资源后不收取闲置费用：

¥0.00333/分钟

(预计¥4.79/天 ¥ 143.85 /月)

流量费用：

¥0.76/GB

已选配置：

[查看详情](#)

4. 确认配置信息并根据系统提示，完成订单确认及支付。创建成功后，用户将收到短信提示。

配置监听器

配置TCP监听器

操作步骤

配置TCP监听包括“配置信息”和“健康检查设置”两部分。
开启健康检查后，BLB将自动屏蔽不健康的后端服务器。

1. 选择“产品导览>负载均衡BLB>普通型实例”，进入普通型实例列表。
2. 点击“实例名称/ID”列的名称链接，进入实例详情页。
3. 点击“监听设置>添加监听设置”，进行监听器的基本设置和健康检查设置。BLB协议[端口]选择“TCP”。
4. 设置配置信息。

配置信息

* BLB协议[端口]:

TCP

80

?

支持获取真实IP

* 后端协议[端口]:

TCP

80

?

转发规则:

加权轮询

?

* 超时时间:

900

秒

?

ProxyProtocol协议:

☐ 开启

☒ 关闭

?

描述:

0/200

项目	描述
BLB协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“80”。
后端协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1~65535间的整数。默认“80”。
转发规则	加权轮询：依据后端服务器的权重，将请求轮流发送给后端服务器，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP：将请求的源IP进行hash运算后派发请求至某匹配的服务器，这可以保证同一个客户端IP的请求始终被派发至某特定的服务器。源IP算法为TCP监听器提供会话保持机制。此算法下，服务器权重不生效。
超时时间	指定TCP连接的超时时间，输入范围为10~4000间的整数，默认900秒。
ProxyProtocol协议	启用ProxyProtocol协议以获取客户端源IP、源Port，访问目的IP、目的Port，访问协议，开启前请务必先配置后端服务器支持ProxyProtocol协议。

5. 设置健康检查。

健康检查设置

开启健康检查：

☒ 开启 ☐ 关闭 ?

健康检查协议：

TCP

检查端口：

?

* 响应超时时间：

秒 ?

* 健康检查间隔：

秒 ?

* 不健康阈值：

2345

?

* 健康阈值：

2345

?

确定

取消

项目	描述
开启健康检查	开启健康检查后将对外端服务器发起健康探测请求，并根据请求结果返回健康检查的状态，关闭健康检查后，健康检查的状态始终显示可用。
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。
健康检查间隔	对后端服务器进行检查的时间间隔。
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。

6. 完成上述配置后，点击“确认”按钮完成TCP监听器添加。

配置UDP监听器

场景简介

UDP流量的常见应用场景有：DNS服务、物联网及一些对数据传输可靠性要求不高的场景。

UDP健康检查介绍

UDP健康检查协议支持ICMP协议和UDP协议两种。UDP协议时，BLB给后端服务器发送UDP包来实现健康检查。如果后端服务器运行正常，则能够接收BLB的健康检查包并给予返回。如果BLB接收到了后端服务器返回的UDP包，就认为其是健康的。用户需指定UDP包所携带的内容，即检查字符串。

例如：用户可以发一个内容是“解析www.abc.com的DNS请求”的UDP包给后端服务器（假设后端服务器是DNS server），这时候后端服务器会返回含有DNS解析结果的UDP包。

检查字符串

检查字符串就是用于健康检查的UDP包的有效载荷内容。由于UDP包的有效载荷是二进制，为了表达方便，可以统一用16进制来表示，如果载荷里有ASCII字符，也可以用ASCII字符来替代相应内容。

配置检查字符串有两种方法：

方法一：如果可以调整后端服务器上的应用程序。可以把检查字符串设置为类似于“hello”的简单字符串，然后调整应用程序来响应携带“hello”的UDP包。响应内容可以任意填写，BLB只要接收到响应包就认为是健康的。

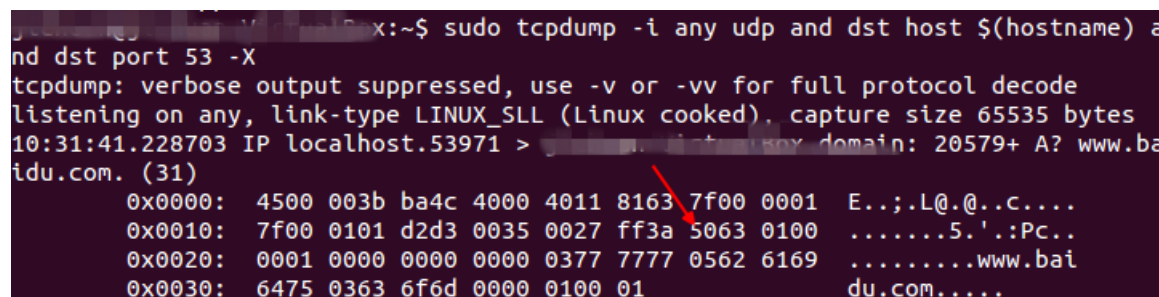
方法二：如果不方便调整后端服务器上的应用程序，例如后端服务器上运行的是通用的DNS解析服务，这时可以配置一个正常服务请求用作健康检查。

获取一个正常请求所对应的有效载荷：可以通过tcpdump命令在后端服务器上抓取本机接收到的UDP包，或者在客户端机器上抓取本机发出的UDP包。

举例：假设后端服务器需要提供DNS解析服务，我们在后端服务器上进行抓包：

```
tcpdump -i any udp and dst host $(hostname) and dst port 53 -X
```

这个命令会抓取所有通过53端口发给本机的UDP包。



请求里前28个字节为IP头+UDP头，需要跳过。从第29个字节（5063）开始，就是UDP包的有效载荷。所以检查字符串可以设置

为：“\50\63\01\00\00\01\00\00\00\00\00\00\03\77\77\77\05\62\61\69\64\75\03\63\6f\6d\00\00\01\00\01”；

为了便于设置检查字符串，在页面上预先填入了对baidu.com的DNS解析请

求“\00\01\01\00\00\01\00\00\00\00\00\00\05baidu\03com\00\00\01\00\01”。

说明：字符串中“\”代表一个16进制的byte字节。例如“\77”代表如下byte：01110111（16bit二进制表示），也可以用所对应的ASCII码“w”来替代。

操作步骤

配置UDP监听包括“配置信息”和“健康检查设置”两部分。

开启健康检查后，BLB将自动屏蔽不健康的后端服务器。

1. 选择“产品导览>负载均衡BLB>普通型实例”，进入普通型实例列表。
2. 点击“实例名称/ID”列的名称链接，进入实例详情页。
3. 点击“监听设置>添加监听设置”，进行监听器的基本设置和健康检查设置。BLB协议[端口]选择“UDP”。
4. 设置配置信息。

配置信息

BLB协议[端口]:

UDP

53

?

后端协议[端口]:

UDP

53

?

转发规则:

加权轮询

?

超时时间:

30

秒

?

描述:

0/200

项目	描述
BLB协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1～65535间的整数。默认“53”。
后端协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1～65535间的整数。默认“53”。
转发规则	加权轮询：依据后端服务器的权重，将请求轮流发送给后端服务器。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP：将请求的源IP进行hash运算后派发请求至某匹配的服务器，这可以保证同一个客户端IP的请求始终被派发至某特定的服务器。源IP算法为UDP监听器提供会话保持机制。此算法下，服务器权重不生效。
超时时间	指定UDP连接的超时时间，输入范围为5～4000间的整数，默认30秒。

5. 设置健康检查。

36

健康检查设置

开启健康检查：

☒ 开启

☐ 关闭

?

健康检查协议：

☒ UDP

☐ ICMP

检查端口：

?

* 检查字符串：

\00\01\01\00\00\01\00\00\00\01

?

* 响应超时时间：

3

秒

?

* 健康检查间隔：

3

秒

?

* 不健康阈值：

2345

3

^

v

?

* 健康阈值：

2345

3

^

v

?

确定

取消

项目	描述
开启健康检查	开启健康检查后将对外端服务器发起健康探测请求，并根据请求结果返回健康检查的状态，关闭健康检查后，健康检查的状态始终显示可用。
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。
检查字符串	BLB采用给后端服务器发送带有用户指定字符串的UDP包实现健康检查。 您需要配置应用程序来完成响应。对于后端服务器是DNS server的情况，您可以配置DNS解析请求作为健康检查包，默认值为对baidu.com的DNS解析请求（00\01\01\00\01\00\00\00\00\00\00\05baidu\03com\00\00\01\00\01）。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。 输入范围为1~60间的整数，建议设置为3秒。
健康检查间隔	对外端服务器进行检查的时间间隔。输入范围为1~10间的整数，建议设置为3秒
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。

6. 完成上述配置后，点击“确认”按钮完成UDP监听器添加。

SSL监听器

场景简介

SSL协议常见应用场景有：需要超高性能和大规模TLS卸载的场景。

操作步骤

开启健康检查后，BLB将自动屏蔽不健康的后端成员。

- 1. 选择“产品导览>负载均衡BLB>普通型实例”，进入应用型实例列表。
- 2. 点击“实例名称/ID”列的名称链接，进入实例详情页。
- 3. 点击“监听设置>添加监听设置”，进行监听器的基本设置。BLB协议[端口]选择“SSL（安全TCP）”。
- 4. 设置配置信息。

配置信息

• BLB协议[端口]:

SSL（安全TCP）

443

?

• 后端协议[端口]:

TCP

80

?

转发规则:

加权轮询

?

• 服务器证书:

-请选择一个证书-

添加证书

管理我的证书

申购SSL证书

双向认证:

☐ 开启

☒ 关闭

加密选项:

☒ 预定义的安全策略

☐ 自定义安全策略

?

tls_cipher_policy_default

加密协议:

☒ TLSv1.0协议

☒ TLSv1.1协议

☒ TLSv1.2协议

加密套件:

全部（19项）

超时时间:

900

秒

?

ProxyProtocol协议:

☐ 开启

☒ 关闭

?

描述:

0/200

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“443”。
后端协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1~65535间的整数。默认“80”。
转发规则	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP：将请求的源IP进行hash运算后派发请求至某匹配的服务器，这可以保证同一个客户端IP的请求始终被派发至某特定的服务器。源IP算法为UDP监听器提供会话保持机制。此算法下，服务器权重不生效。
服务器证书	在配置SSL监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看证书管理。 说明：专属集群的BLB实例支持国际标准、国密（SM2）标准证书；共享集群的BLB实例只支持国际标准证书。
双向认证	选择是否开启双向认证。
加密选项	安全策略包含HTTPS可选的TLS协议版本和配套的加密算法套件。
超时时间	等待后端成员响应的超时时间；如果后端成员需较长时间处理请求，可适当调大该值，默认900秒。
ProxyProtocol协议	启用ProxyProtocol协议以获取客户端源IP、源Port，访问目的IP、目的Port，访问协议，开启前请务必先配置后端服务器支持ProxyProtocol协议。
描述	监听的描述，不查过200字符。

5. 设置健康检查。

健康检查设置

健康检查协议：

TCP

* 响应超时时间：

3

秒

?

* 健康检查间隔：

3

秒

?

* 不健康阈值：

2

3

4

5

3

^

v

?

* 健康阈值：

2

3

4

5

3

^

v

?

确定

取消

项目	描述
健康检查协议	指定检查后端服务器健康状态时使用的协议类型，SSL监听只支持TCP健康检查协议。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。
健康检查间隔	对后端服务器进行检查的时间间隔。
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。

6. 完成上述配置后，点击“确认”按钮完成SSL监听器添加。

配置HTTP监听器 操作步骤

开启健康检查后，BLB将自动屏蔽不健康的后端服务器。

- 1. 选择“产品导览>负载均衡BLB>普通型实例”，进入负载均衡BLB列表。
- 2. 点击“实例名称/ID”列的ID链接，进入实例详情页。
- 3. 点击“监听设置>添加监听设置”，进行监听器的基本设置和健康检查设置。BLB协议[端口]选择“HTTP”。
- 4. 设置配置信息。

配置信息

BLB协议[端口]:

HTTP

▼

80

?

后端协议[端口]:

HTTP

80

?

转发规则:

加权轮询

▼

?

会话保持:

☐ 开启

☒ 关闭

附加HTTP头部:

☒ 开启

☐ 关闭

?

☒ 获取真实IP;X-Forwarded-For方式获取客户端真实IP

☐ 获取监听协议;X-Forwarded-Proto方式获取监听协议

启用HTTPS重定向:

☐ 开启

☒ 关闭

?

定制配置:

☐ 开启

☒ 关闭

?

HTTP请求允许非ASCII字符:

☐ 开启

☒ 关闭

?

超时时间:

30

秒

?

描述:

0/200

项目	描述
BLB协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“80”。
后端协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1~65535间的整数。默认“80”。
转发规则	加权轮询：依据后端服务器的权重，将请求轮流发送给后端服务器，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。
会话保持	TCP监听器可以通过源IP转发规则实现会话保持；7层监听器（HTTP、HTTPS）通过插入/修改cookie实现会话保持。可选择“开启”或“关闭”。
附加HTTP头部	在HTTP请求中附加头字段，在后端服务器处理报文时获取客户端真实IP或监听协议信息。 获取真实IP：开启该功能后，可将用户IP地址记录在X-Forwarded-For头域中； 获取监听协议：开启该功能后，可将监听协议记录在X-Forwarded-Proto头域中。
启用HTTPS重定向	将此监听器的HTTP请求重定向至此BLB的HTTPS监听器。请确保相应端口的HTTPS监听器工作正常。可选择“开启”或“关闭”。默认“关闭”。
定制配置	BLB 7层监听支持定制配置，配置项和说明可参见 配置参数详情
HTTP请求允许非ASCII字符	开启此选项后如果HTTP请求中含有非ASCII字符，BLB会放行并转发给后端服务器处理。开启此选项可能带来安全风险，非特殊使用场景建议保持关闭。默认“关闭”。
超时时间	等待后端服务器响应的超时时间；如果后端服务器需较长时间处理请求，可适当调大该值，默认30秒。

5. 设置健康检查信息。

健康检查设置

开启健康检查：

☒ 开启

☐ 关闭

?

健康检查协议：

☒ HTTP

☐ TCP

Host头域：

?

检查端口：

?

*

 检查路径：

/

?

*

 响应超时时间：

3

秒

?

*

 健康检查间隔：

3

秒

?

*

 不健康阈值：

2345

3

^

v

?

*

 健康阈值：

2345

3

^

v

?

*

 正常状态码：

☒ http_2xx

☒ http_3xx

☐ http_4xx

☐ http_5xx

健康检查正常的http状态码

确定

取消

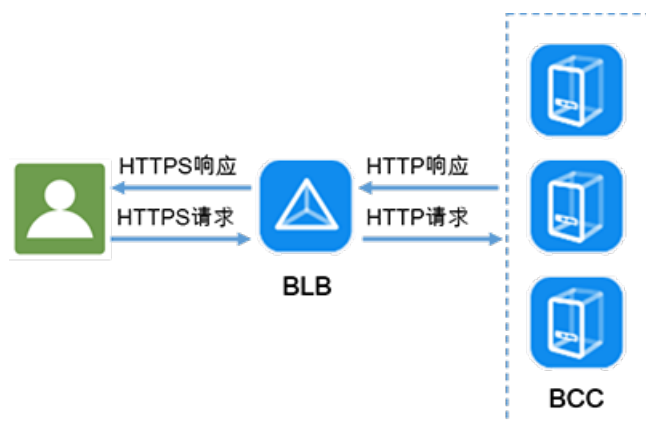
项目	描述
开启健康检查	开启健康检查后将对外端服务器发起健康探测请求，并根据请求结果返回健康检查的状态，关闭健康检查后，健康检查的状态始终显示可用。
健康检查协议	指定检查后端服务器健康状态时使用的协议类型，HTTP监听支持HTTP和TCP两种健康检查协议。
Host头域	发往后端服务器的健康检查请求的Host头域，默认为空。
检查端口	指定后端服务器提供服务的端口。本字段只对HTTP健康检查有效。
检查路径	用于健康检查页面文件的URI，建议对静态页面进行检查。 长度限制为1-80个字符，只能使用字母、数字、‘-’、‘/’、‘.’、‘%’、‘?’、‘#’、‘&’这些字符。本字段只对HTTP健康检查有效。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。
健康检查间隔	对后端服务器进行检查的时间间隔。
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。
正常码状态	健康检查正常的http状态码，可选“http_2xx”、“http_3xx”、“http_4xx”、“http_5xx”，本字段只对HTTP健康检查有效

42

6. 完成上述配置后，点击“确认”按钮完成HTTP监听器添加。

配置HTTPS监听器

为了保证信息安全，用户可以配置BLB监听器的协议为HTTPS。BLB对于用户的HTTPS的请求进行解密，然后发送至后端服务；后端服务处理完请求后的返回包首先发送至BLB，由BLB进行加密后，再传回用户侧。整个过程如下图所示：



前置任务

在配置HTTPS监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看[证书管理](#)。

操作步骤

配置HTTPS监听包括“配置信息”和“健康检查设置”两部分。

开启健康检查后，BLB将自动屏蔽不健康的后端服务器。

1. 选择“产品导航>负载均衡BLB>普通型实例”，进入负载均衡BLB列表。
2. 点击“实例名称/ID”列的ID链接，进入实例详情页。
3. 点击“监听设置>添加监听设置”，进行监听器的基本设置和健康检查设置。BLB协议[端口]选择“HTTPS”。
4. 设置配置信息。

配置信息

BLB协议[端口]:

HTTPS

▼

443

?

后端协议[端口]:

HTTP

▼

80

?

转发规则:

加权轮询

▼

?

服务器证书:

-请选择一个证书-

▼

添加证书

[管理我的证书](#)

[申购SSL证书](#)

扩展域名:

开启

关闭

双向认证:

开启

关闭

加密选项:

预定义的安全策略

自定义安全策略

?

tls_cipher_policy_default

▼

加密协议:

TLsv1.0协议

TLsv1.1协议

TLsv1.2协议

加密套件:

全部 (19项)

▼

会话保持:

开启

关闭

附加HTTP头部:

获取真实IP;X-Forwarded-For方式获取客户端真实IP

获取监听协议;X-Forwarded-Proto方式获取监听协议

定制配置:

开启

关闭

?

HTTP请求允许非ASCII字符:

开启

关闭

?

超时时间:

30

秒

?

描述:

0/200

44

项目	描述
BLB协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“443”。
后端协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1~65535间的整数。默认“80”。
转发规则	加权轮询：依据后端服务器的权重，将请求轮流发送给后端服务器，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。
服务器证书	在配置HTTPS监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看 证书管理 。
扩展域名	选择是否开启扩展域名。
双向认证	选择是否开启双向认证。
加密选项	安全策略包含HTTPS可选的TLS协议版本和配套的加密算法套件。
会话保持	TCP监听器可以通过源IP转发规则实现会话保持；7层监听器（HTTP、HTTPS）通过插入/修改cookie实现会话保持。可选择“开启”或“关闭”。默认“关闭”。
附加HTTP头部	在HTTP请求中附加头字段，在后端服务器处理报文时获取客户端真实IP或监听协议信息。 获取真实IP：开启该功能后，可将用户IP地址记录在X-Forwarded-For头域中； 获取监听协议：开启该功能后，可将监听协议记录在X-Forwarded-Proto头域中。
定制配置	BLB 7层监听支持定制配置，配置项和说明可参见 配置参数详情 。
HTTP请求允许非ASCII字符	开启此选项后如果HTTP请求中含有非ASCII字符，BLB会放行并转发给后端服务器处理。开启此选项可能带来安全风险，非特殊使用场景建议保持关闭。默认“关闭”。
超时时间	等待后端服务器响应的超时时间；如果后端服务器需较长时间处理请求，可适当调大该值，默认30秒。

5. 设置健康检查信息。

健康检查设置

开启健康检查：

☒ 开启

☐ 关闭

?

健康检查协议：

☒ HTTP

☐ TCP

Host头域：

?

检查端口：

?

* 检查路径：

?

* 响应超时时间：

秒

?

* 健康检查间隔：

秒

?

* 不健康阈值：

2345

^v

?

* 健康阈值：

2345

^v

?

* 正常状态码：

☒ http_2xx

☒ http_3xx

☐ http_4xx

☐ http_5xx

健康检查正常的http状态码

确定

取消

项目	描述
开启健康检查	开启健康检查后将对外端服务器发起健康探测请求，并根据请求结果返回健康检查的状态，关闭健康检查后，健康检查的状态始终显示可用。
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。可选择“HTTP”或“TCP”。
Host头域	发往后端服务器的健康检查请求的Host头域，默认为空。
检查端口	指定后端服务器提供服务的端口。本字段只对HTTP健康检查有效。
检查路径	用于健康检查页面文件的URI，建议对静态页面进行检查。 长度限制为1-80个字符，只能使用字母、数字、‘-’、‘/’、‘.’、‘%’、‘?’、‘#’、‘&’这些字符。本字段只对HTTP健康检查有效。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。
健康检查间隔	对外端服务器进行检查的时间间隔。
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。
正常码状态	健康检查正常的http状态码，可选“http_2xx”、“http_3xx”、“http_4xx”、“http_5xx”，本字段只对HTTP健康检查有效

6. 完成上述配置后，点击“确认”按钮完成HTTP监听器添加。

替换证书

对于已经添加HTTPS监听器用户，可以通过以下方式完成证书替换，具体操作如下：

1. 创建新服务器证书，关于证书的创建方法，请参看[证书管理](#)
2. 选择“产品导航>负载均衡BLB>普通型实例”，进入负载均衡BLB列表。
3. 点击“实例名称/ID”列的ID链接，进入实例详情页。
4. 点击“监听设置”，找到已经配置的HTTPS监听器，点击“编辑”进入配置页面。



5. 点击“证书”项下拉菜单，从证书列表中找到步骤1中创建的服务器证书，点击“确定”，完成证书替换。

编辑监听设置

在“监听设置”页签，点击<编辑>，跳入编辑界面，编辑后点击<确定>。



释放监听设置

1. 在“监听设置”页签，勾选要释放的监听设置，点击<释放>按钮。

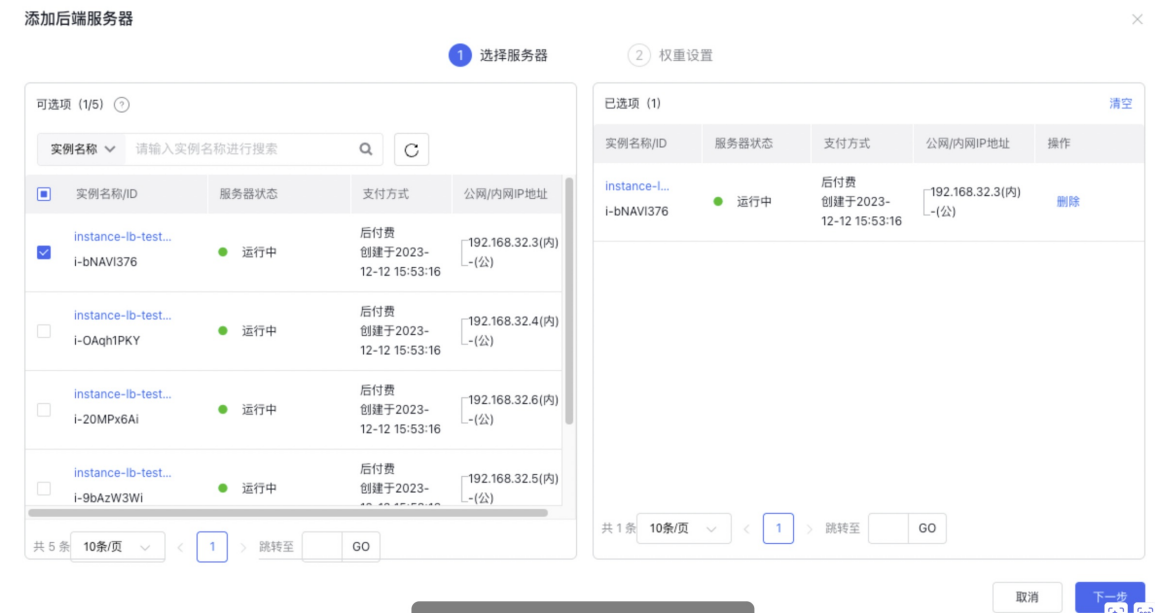


2. 在弹出的界面，点击<确定>，完成释放。

配置后端服务器

添加后端服务器

1. 选择“产品导航>负载均衡BLB>普通型实例”，进入负载均衡BLB列表。
2. 点击“实例名称/ID”列的ID链接，进入实例详情页。
3. 点击“后端服务器>添加后端服务器”，进入后端服务器列表。后端服务器列表中显示了当前BLB所在私有网络VPC下的BCC实例名称、状态等信息。
4. 配置后端服务器权重。权重是用来针对不同的后端服务器实例设置不同的请求转发比例。例如有两台配置不同的后端服务器实例，那么对于高配置的实例可以选择设置更高的权重来接收更多的外部访问请求。
 - 权重值是一个相对值，如果设置成相同数值，其转发比例是一样的。
 - 支持设置0-100的权重值。权重越高，接收的请求越多；如果权重设置为“0”，则不再接收新的请求。
 - 非0权重对加权轮询外的转发规则不生效。



设置后端服务器安全组

目前BCC默认安全组的入站规则是允许所有流量。如果您需要更精细化地对安全组规则进行配置，请参考以下指导。

BLB通过特定的内网网段与BCC交互来完成健康状态检查。所以您在配置BCC安全组时，需要注意为安全组的入站规则放开这些内网IP段。

由于4层监听器（eg.TCP协议）默认开启真实IP功能（数据包的源IP就是client IP），所以用户BCC安全组的入站规则还需要额外放开client IP地址；7层监听器（eg.HTTP协议）的真实IP功能是通过X-Forwarded-For header实现，所以7层监听器的后端BCC安全组无需放开client IP地址。

BCC安全组的入站规则需要对源IP进行如下配置：

监听器类型	公网BLB	内网BLB
4层监听器 (eg.TCP协议)	- 允许用户Client所在的公网IP段 - 允许100.64.0.0/10（健康检查）	- 允许10.0.0.0/8网段（内网BCC数据包的IP地址会被转换为10.0.0.0/8网段的IP地址，所以在BLB的后端服务器看来Client请求是来自10.0.0.0/8网段的） - 允许100.64.0.0/10网段（健康检查）
7层监听器 (eg.HTTP协议)	允许100.79.0.0/16网段	允许100.79.0.0/16网段

安全组开放的端口应至少包含BLB后端服务端口和健康检查端口。

示例

BLB类型为公网BLB，用户Client所在网段为7.0.0.0/8，后端服务端口为80，监听器配置如下图所示：

☐ BLB协议：端口	后端协议：端口	转发规则	获取真实IP	会话保持	健康检查	描述	操作
☐ TCP：80	TCP：80	加权轮询	已开启	-	已开启	- 编辑	编辑

根据上述场景，BCC安全组的配置如下图所示，关于安全组的具体配置，请参看[管理安全组](#)。

入场 出站 允许访问所有端口：☐ + 添加规则 删除							
☐	类型	协议	端口	来源	备注	更新时间	操作
☐	IPv4	全部协议	1-65535	源IP：100.64.0.0/10	健康检查	2025-03-25 14:55:05	编辑
☐	IPv4	全部协议	1-65535	源IP：7.0.0.0/8	Client IP	2025-03-25 14:55:14	编辑

编辑后端服务器

1. 点击“后端服务器”>“添加后端服务器”，进入后端服务器列表。
2. 用户可重新选择后端服务器并配置其权重值，具体配置方法可参看[添加后端服务器](#)。

移除后端服务器

1. 在“后端服务器”页签勾选需要移除的后端服务器实例。
2. 点击<批量移除>，完成移除操作。

查看监控数据

通过控制台查看

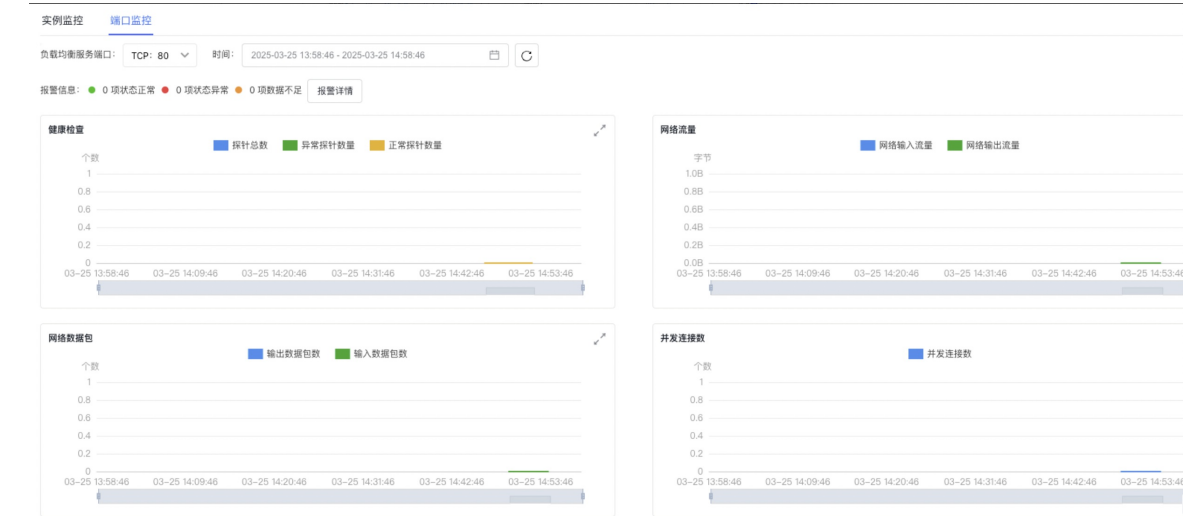
点击BLB的详情页的“监控”页签，就可以看到监控的各项图表数据。

系统默认给出最近1小时内的监控信息，如果需要查看更多内容，可点击下拉菜单，切换时间范围，如下图所示：



实例监控信息代表与BLB实例绑定的EIP实例的监控信息，包括：

- 从外网进入的流量：进站流量，即用户访问站点的流量，单位：字节/分钟。
- 流向外网的流量：出站流量，即站点到用户方向的流量，单位：字节/分钟。
- 出口带宽：EIP实例出方向瞬时带宽，单位：比特/秒。



对于4层监听器，端口监控信息包括：

- 健康检查
 - 探针总数：已添加的后端服务器数量。

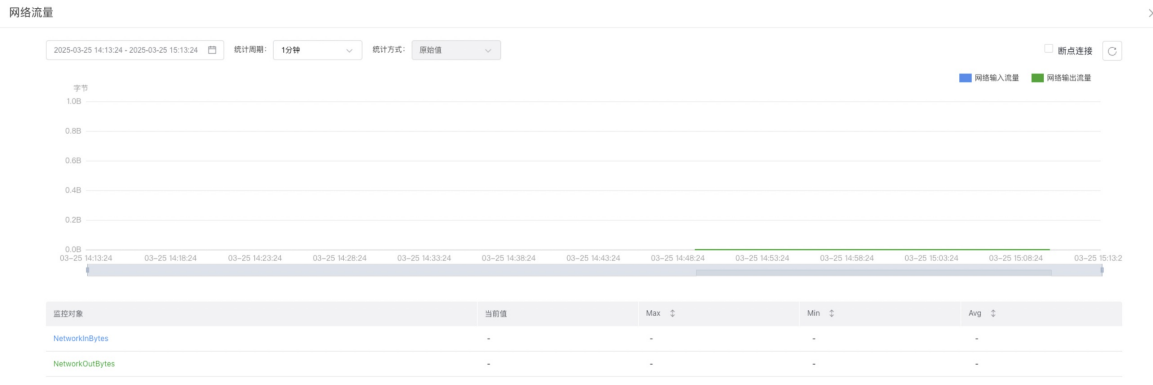
- 正常探针数量：健康检查正常的后端服务器数量。
- 异常探针数量：健康检查异常的后端服务器数量。
- 网络流量
 - 网络输入流量：监听器接收到的用户流量，单位：字节/分钟。
 - 网络输出流量：监听器发往用户的流量，单位：字节/分钟。
- 网络数据包
 - 输入数据包数：监听器接受到的用户数据包（Packet）数量，单位：个/分钟。
 - 输出数据包数：监听器发往用户的数据包（Packet）数量，单位：个/分钟。
- 并发连接数：当前并发的连接数。
- 丢弃流量
 - 丢弃输入流量：监听器丢弃的接收用户的流量，单位：字节/分钟。
 - 丢弃输出流量：监听器丢弃的发往用户的流量，单位：字节/分钟。
- 丢弃数据包
 - 丢弃输入数据包数：监听器丢弃的接收用户的数据包数，单位：个/分钟。
 - 丢弃输出数据包数：监听器丢弃的发往用户的数据包数，单位：个/分钟。
- 丢弃连接数：监听器丢弃的连接数，单位：个/分钟。
- 新建连接数：监听器新建的连接数，单位：个/分钟。
- 网络输入带宽：监听器接收到的用户带宽，单位：bps。
- 网络输出带宽：监听器发往用户的流量，单位：bps。
- 限速流量
 - 限速前输入流量：监听器在限速前接收到的用户流量，单位：字节/分钟。
 - 限速前输出流量：监听器在限速前发往用户的流量，单位：字节/分钟。
- 限速数据包
 - 限速前数据包数：监听器在限速前接受到的用户数据包（Packet）数量，单位：个/分钟。
 - 限速前数据包数：监听器在限速前发往用户的数据包（Packet）数量，单位：个/分钟。

对于7层监听，端口监控信息除了包括以上内容之外，还包括：

- 请求数：监听器接收到的HTTP或HTTPS请求的数量，单位：个/分钟。
- 响应时间：从客户端发送请求到后端服务器并返回响应的时间，单位：ms。
- 响应状态码：
 - 2xx：监听器每秒返回的2xx状态码的数量，单位：个/分钟。
 - 3xx：监听器每秒返回的3xx状态码的数量，单位：个/分钟。
 - 4xx：监听器每秒返回的4xx状态码的数量，单位：个/分钟。
 - 5xx：监听器每秒返回的5xx状态码的数量，单位：个/分钟。

- 其他状态码：监听器每秒返回的除2xx、3xx、4xx、5xx状态码以外的状态码数量，单位：个/分钟。
- 响应状态码比例：
 - 2xx：监听器返回的2xx状态码的比例。
 - 3xx：监听器返回的3xx状态码的比例。
 - 4xx：监听器返回的4xx状态码的比例。
 - 5xx：监听器返回的5xx状态码的比例。
 - 其他状态码：监听器返回的除2xx、3xx、4xx、5xx状态码以外的状态码的比例。
- 后端服务器响应状态码：
 - 4xx：后端服务器返回的4xx状态码的数量，单位：个/分钟。
 - 5xx：后端服务器返回的5xx状态码的数量，单位：个/分钟。
- 每秒查询数：监听器每秒接收到的HTTP或者HTTPS请求数，单位：qps。
- QPS使用率：监听器当前QPS占用配置上限的百分比。

您可以通过点击图表右上方的图标，查看详细监控信息，如下图所示：



系统每分钟产生一个监控数据。用户可以通过调整“统计项”和“采样周期”来获取不同维度的统计信息。例如“采样周期”设为1小时，则每个采样周期内会有60个监控数据；然后如果“统计项”设的是平均值，则每个采样周期内，会为用户展示60个监控数据的均值。

采样周期包括：1分钟、5分钟、20分钟、1小时、6小时、12小时和1天。

统计项包括：

- 平均值：采样周期内所有采样点的平均值。
- 和值：采样周期内所有采样点的取值之和。
- 最大值：采样周期内所有采样点中的最大值。
- 最小值：采样周期内所有采样点中的最小值。
- 样本数：采样周期内的样本数。

通过API查看

详细的接口请参考云监控BCM[查询数据接口](#)

其中，获取BLB监控的请求参数“scope”, “metricName”和“dimensions”，请参照下表：

scope	metricName	dimensions
BCE_BLB	UnhealthyCount 异常实例数量	BlbId + BlbPort 通过"."号拼接起来，例如：BlbId=lb-dd45e123，BlbPort=80 那么instance为：lb-dd45e123.80
	TotalCount 实例总数	
	HealthyCount 正常实例数量	
	RequestCount 请求数	
	NetworkInBytes 网络流入流量	
	NetworkOutBytes 网络流出流量	
	ActiveConnCount 活跃连接数	
	NetworkInPkgCount 输入数据包数	
	NetworkOutPkgCount 输出数据包数	

管理报警

在“实例监控”页签，点击<报警详情>按钮，进入报警管理页面，可以为端口的报警策略进行管理。详细操作步骤请见[BCM管理报警](#)。



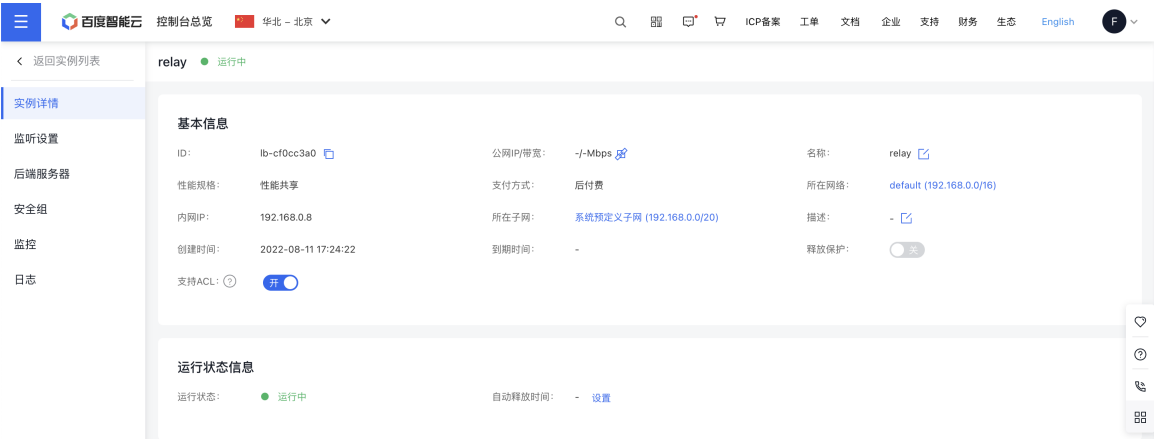
管理BLB普通型实例

查看BLB实例

1. 选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型实例”可查看BLB实例相关信息，具体内容包括：

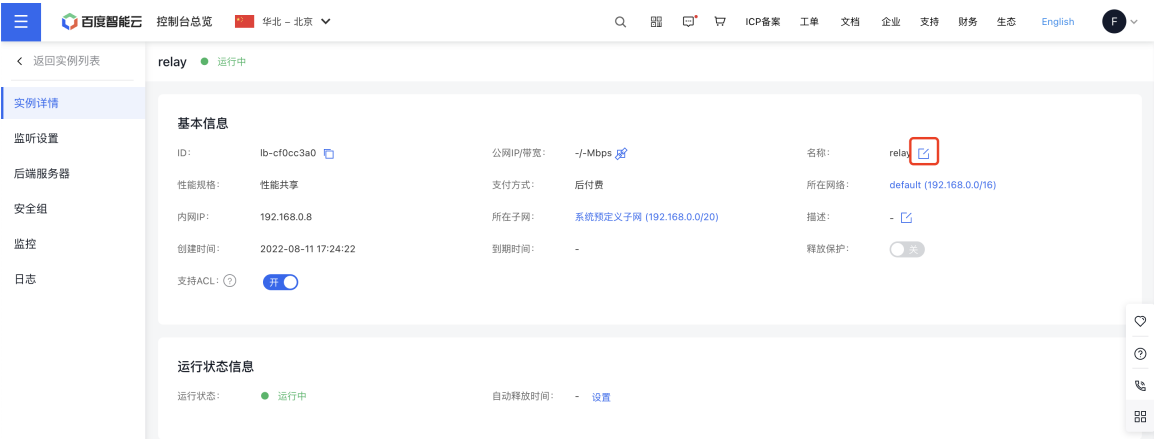
项目	描述
实例名称/ID	BLB实例的名称及ID，通过点击实例名称，可以进入BLB实例详情页面，如步骤2所示。
状态	BLB实例当前的工作状态，若此处显示为“运行中”，代表BLB实例正常工作。
公网IPv4IP/线路类型/带宽	BLB提供公网IP作为访问地址来访问BLB服务。
内网IP	BLB提供私网IP作为访问地址来访问BLB服务。
所在网络	BLB所在的私有网络VPC，如无特殊指定，系统提供默认私有网络。
前端协议/端口	BLB实例所关联的监听器的协议（eg. HTTP, TCP）及端口号。
后端协议/端口	BLB实例所关联的监听器转发至后端BCC的协议及端口号。
健康检查	探测server能提供服务后端BCC是否正常提供服务
后端服务器	BLB实例所关联的BCC。

2. 点击“实例名称/ID”列的名称链接，查看BLB的详细信息



修改实例名称

1. 选择“产品服务>负载均衡BLB”，进入BLB列表。
2. 在实例详情页面中，点击“名称”后面的修改按钮，在弹出的界面修改BLB实例名称。



3. 参照提示的命名规范命名后，点击<确定>，完成修改。

修改BLB实例IP地址

用户在修改IP地址之前需要新建EIP实例并配置EIP实例的计费方式，用新建的EIP实例替换当前正在使用的EIP实例。

注意：

- 用户无法修改已经创建的EIP实例的计费方式。
- 用户在更换EIP实例时，应尽量避免影响当前业务。以网站站点为例，如果网站域名被DNS解析到当前BLB实例的EIP地址，为保证当前业务不受影响，应将DNS解析到其它公网EIP地址，待BLB实例完成更换公网EIP操作后，再将DNS解析改为更换后BLB实例EIP地址。
- 如需更改内网IP地址，可提交工单申请。

1. 创建EIP实例，按照业务需求配置新EIP实例的计费方式。
2. 找到需要更换IP地址的BLB实例，点击如下图所示图标，解绑EIP实例。

☐	负载均衡名称/ID	状态	服务地址	负载均衡服务端	后端服务器端口
☐	BLB-beijing-6...	● 运行中	180.76.170.214 (公网)	HTTP : 80	HTTP : 80
☐	4a4c3641764...				

3. 刷新BLB实例列表，待BLB实例状态为“无公网IP”时，可为其绑定新EIP实例。点击如下图所示图标>“选择新创建的EIP实

例">"确定"，为BLB绑定新EIP实例。

☐	负载均衡名称/ID	状态	服务地址	负载均衡服务端	后端服务器端口
☐	BLB-beijing-6... 4a4c3641764...	● 无公网IP	- 	HTTP : 80	HTTP : 80

变更带宽

网络带宽的变更包含BLB实例绑定的EIP带宽变更和BLB实例带宽变更，用户如果需要变更BLB绑定的EIP带宽，需要通过修改与BLB实例绑定的EIP实例的带宽实现。

修改BLB实例EIP地址

1. 创建EIP实例，按照业务需求配置新EIP实例。
2. 找到需要更换IP地址的BLB实例，点击如下图所示图标，解绑EIP实例。

☐	实例名称/ID	状态	公网IPv4IP/线路类型/带宽	内网IP	内网及公网IPv6IP/线路类型/带宽
☐	wendangtest lb-bf2f1d8a	● 运行中	180.76.104.149 /标准型BGP /1Mbps 	192.168.0.45	- - (EIP) 

3. 点击对应的BLB实例图标，如下图：

☐	实例名称/ID	状态	公网IPv4IP/线路类型/带宽	内网IP	内网及公网IPv6IP/线路类型/带宽
☐	wendangtest lb-bf2f1d8a	● 运行中	- / - / 	192.168.0.45	- - (EIP) 

4. 添加新的EIP：

弹性公网IP绑定

+ 创建公网IP

实例名称 请输入

实例名称	公网IP	线路类型	最大带宽
☒ wendangtest_1	180.76.104.253	标准型BGP	1Mbps
☐ wendangtest	180.76.104.149	标准型BGP	1Mbps
☐ EIP171502	180.76.104.226	标准型BGP	1Mbps
☐ 测试	180.76.104.141	标准型BGP	1Mbps

共 4 条 5条/页 1 跳转至 GO

取消

确定

BLB实例带宽变更：

进入“普通型实例”页面后，选中要变更带宽的BLB实例，在操作这一栏点击“性能调整”，根据业务带宽需要来调整BLB实例性能

普通型实例

云网协同您快速感知和化解云上资源风险，实现最佳实践。 立即免费检查>>

普通型实例——轻量，简单

提供了负载均衡基础功能组件，能够满足用户对负载均衡的通用需求，可以将来自互联网或内网的IPv6和IPv4流量分发至多台IPv4后端服务器

规格

+ 创建负载均衡 释放 编辑标签 设置 批量操作

自动识别 请输入关键字进行搜索 下载选中

实例名称/ID	状态	公网IPv4IP/线路类型/带宽	内网IP	内网及公网IPv6IP/线路类型/带宽	支付方式	所在集群	性能规格	所在网络	所在子网	操作
☒ BLB-beijing-6... lb-yp...	● 运行中	- / - /	192.168.0.45	- (EIP)	后付费-按使用量	4 星：byd 7层：共享集群	不限速	默认私有网络 vpc-xxxw	系统指定	性能调整

🔗 释放BLB实例

手动释放BLB实例

1. 选择“产品 > 网络 > 负载均衡 > 普通型实例”，进入负载均衡BLB管理控制台页面。
2. 勾选要释放的BLB实例，点击“释放”。



3. 在弹出的对话框中，点击“确定”。

自动释放BLB实例

设置自动释放BLB实例后，BLB实例会在设置的时间自动进行释放。

有两种方式自动释放BLB实例：

- 方式一：对于新建BLB实例，可在创建BLB实例的页面进行设置。
1. 点击“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型实例”，然后选择“创建负载均衡”。
 2. 在创建BLB实例的同时设置该BLB实例的自动释放时间，如下图所示。

购买信息

购买数量：

1

^
v

!

最多可开通60个BLB实例，还可开通59个

自动释放：

开

自动释放时间：

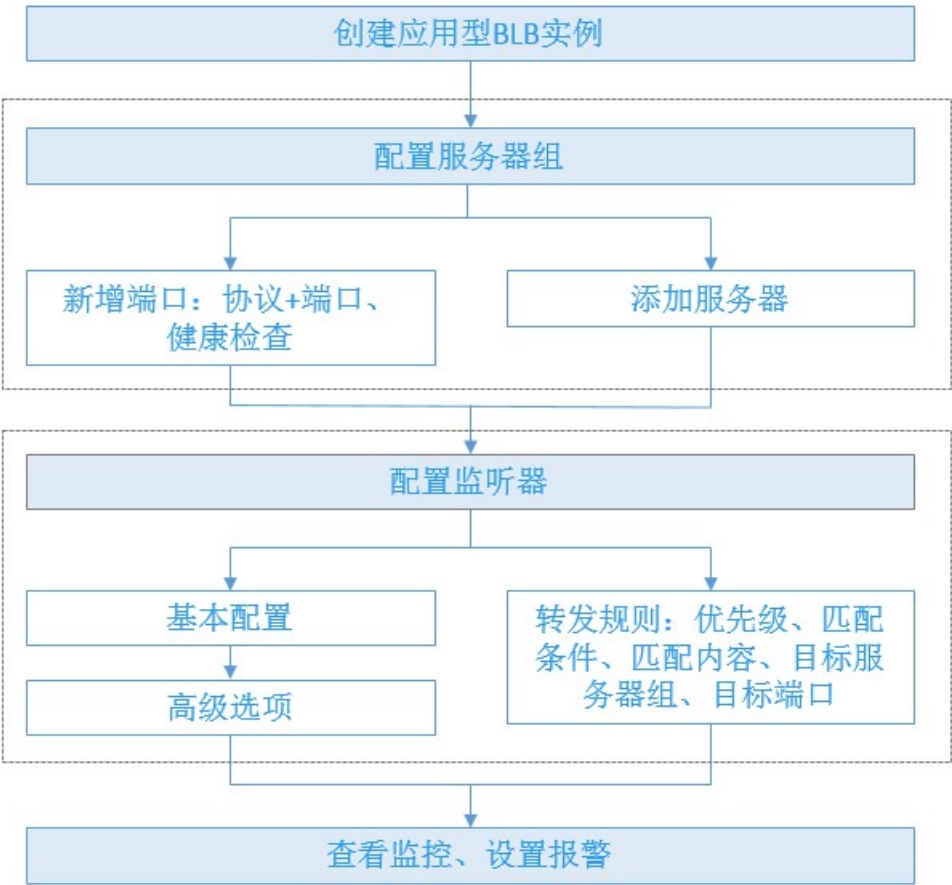
2023-09-11 03:01:59

📅

- 方式二：对于已经创建的BLB实例，可在BLB实例详情页面进行设置。
1. 点击“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型实例”，然后选择“创建负载均衡”。
 2. 点击“实例名称/ID”列的名称链接，进入实例详情页。
 3. 进入“实例详情”页签，点击“自动释放时间-设置”。
 4. 在弹出窗口中选择“设置自动释放”，同时在下方配置释放日期和具体时间，点击“确认”完成设置。

应用型BLB实例

用户使用应用型BLB实例的流程图如下所示：



应用型BLB配置流程图

分为以下四个步骤：

- 第一步：创建应用型BLB实例；
- 第二步：配置服务器组，主要包括端口配置、健康检查和添加服务器；
- 第三步：配置监听器；
- 第四步：查看监控、设置报警。

🔗 创建应用型BLB实例

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB>应用型实例”，进入“实例列表页面”。
2. 点击“创建负载均衡”，进入创建实例界面。

说明：
创建BLB实例之前，应先创建BCC实例，否则无法成功创建BLB实例。

3. 选择配置信息。
 - 基础信息：可选择区域和设置实例名称。所在网络，指当前创建的 BLB 和相应 BCC 等云资源所属的私有网络(VPC)。

说明： 用户在创建BLB实例时需要指定子网。

- 配置公网访问：可根据实际情况选择“启用”或“不启用”公网访问

说明：

- 如果选择“OFF”，不启用。表示在创建该BLB实例时不绑定公网IP，您可以在购买成功后根据需要再绑定公网IP。
- 如果选择“ON”，启用。表示在创建该BLB实例时购买并绑定公网IP。

- 购买信息：选择购买数量，以及配置是否自动释放。

说明： 用户在创建BLB实例时，可直接进行BLB实例自动释放设置，BLB实例会在设定的时间进行自动释放。

(可选，如果未启用公网访问，系统将自动跳过此步骤) 配置EIP实例信息，有关EIP实例的配置方法，请参看[创建EIP实例](#)。

三

百度智能云

控制台总览

华北 - 北京

付费及地域

付费方式：

后付费

当前地域：

华北 - 北京

!

基础信息

名称：

请输入实例名称

实例类型：

普通型实例

应用型实例

IPv6实例

实例性能：

性能共享

创建的BLB实例在共享集群，多租户性能共享，在极端情况下出现资源争抢时，性能可能会受到限制。若需要超高性能和独占资源，建议您使用[负载均衡专属集群](#)。

所在网络：

test-cym (192.168.0.0/16)

test-a (192.168.0.0/24)

IP地址：

☒ 自动分配

☐ 指定

安全组：

普通安全组

企业安全组

请选择

描述：

配置公网访问

温馨提示：如不启用公网访问，则默认提供内网IP来进行内网流量的均衡服务；如启用，则为您购买并绑定公网IP来实现对公网访问流量的均衡服务。您也可以选择不启用，在购买成功后再绑定已有的公网IP。

公网访问：

并

启用

购买弹性公网IP

绑定已有弹性公网IP

标签

绑定标签：

温馨提示：标签支持您按各种标准（如用途、所有者或项目）对资源进行分类；每个标签包含键和值两部分

标签键：

请选择已有或手动输入

值：

请选择已有或手动输入

+ 添加标签

帮助文档

前往标签管理

资源分组

资源分组：

默认分组

①

如需创建新的资源分组，您可前往资源管理 [去创建](#) [帮助文档](#)

购买信息

购买数量：

1

①

① 最多可开通60个BLB实例，还可开通59个

自动释放：

关

下一步

配置费用：

¥0.00036/分钟（预计¥0.51/天 ¥ 15.55 /月）

已选配置：

查看详情

4. 确认配置信息并根据系统提示，完成订单确认及支付。创建成功后，用户将收到短信提示。

配置服务器组

配置服务器组步骤主要包括：

- 1. 服务器组基本信息填写
- 2. 配置开放端口；
- 3. 添加服务器组；

注意：一个服务器组可以对应多个开放端口，例如：A服务器组，可以同时开放 TCP 80和HTTP 80两个端口。

服务器组基本信息填写

操作步骤

- 1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
- 2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
- 3. 选择左侧导航栏中的“目标组“，详情页面选择“服务器组”页签，点击“添加服务器组“，进入添加服务器组配置页面,完成服务器组名称和描述填写。

配置TCP协议开放端口

操作步骤

- 1.进入添加服务器组配置页面,点击"新增端口"进行监听器的基本设置和健康检查设置,开放协议[端口]选择TCP。
- 2.设置配置信息。

58

项目	描述
开放协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1~65535间的整数。默认“80”。

3.设置健康检查。

项目	描述
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。
健康检查间隔	对后端服务器进行检查的时间间隔。
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。

4.完成上述配置后，点击“确认”按钮完成TCP协议开放端口的配置。

🔗 配置UDP协议开放端口

操作步骤

- 1.进入添加服务器组配置页面,点击"新增端口"进行监听器的基本设置和健康检查设置,开放协议[端口]选择UDP。
- 2.设置配置信息。

项目	描述
开放协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1~65535间的整数。默认“80”。

3.设置健康检查。

项目	描述
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。
检查字符串	BLB采用给后端服务器发送带有用户指定字符串的UDP包实现健康检查。 您需要配置应用程序来完成响应。对于后端服务器是DNS server的情况，您可以配置DNS解析请求作为健康检查包，默认值为对baidu.com的DNS解析请求 （00\01\01\00\01\00\00\00\00\00\00\05baidu\03com\00\00\01\00\01）。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。 输入范围为1~60间的整数，建议设置为3秒。
健康检查间隔	对后端服务器进行检查的时间间隔。输入范围为1~10间的整数，建议设置为3秒
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。

4.完成上述配置后，点击“确认”按钮完成UDP协议开放端口的配置。

🔗 配置HTTP协议开放端口

- 1.进入添加服务器组配置页面,点击"新增端口"进行监听器的基本设置和健康检查设置,开放协议[端口]选择HTTP。

2.设置配置信息。

项目	描述
开放协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1～65535间的整数。默认“80”。

3.设置健康检查信息。

项目	描述
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。可选择“HTTP”或“TCP”
Host头域	发往后端服务器的健康检查请求的Host头域，默认为空。
检查端口	指定后端服务器提供服务的端口。本字段只对HTTP健康检查有效。
检查路径	用于健康检查页面文件的URI，建议对静态页面进行检查。 长度限制为1-80个字符，只能使用字母、数字、‘-’、‘/’、‘.’、‘%’、‘?’、‘#’、‘&’这些字符。本字段只对HTTP健康检查有效。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。
健康检查间隔	对后端服务器进行检查的时间间隔。
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。
正常码状态	健康检查正常的http状态码，可选“http_2xx”、“http_3xx”、“http_4xx”、“http_5xx”，本字段只对HTTP健康检查有效

4.完成上述配置后，点击“确认”按钮完成HTTP协议服务器组配置。

添加服务器组

在服务器组详情页面点击“添加服务器”，选择需要添加进入该服务器组的服务器名称，然后选择后点击“下一步”，进行服务器权重设置，最后点击“确认”，完成服务器组的添加步骤。

注意：

- 添加服务器过程中，所有的服务器都需要和应用型BLB实例在同一个VPC下。
- 配置后端服务器权重。权重是用来针对不同的后端服务器实例设置不同的请求转发比例。例如有两台配置不同的后端服务器实例，那么对于高配置的实例可以选择设置更高的权重来接收更多的外部访问请求。
- 权重值是一个相对值，如果设置成相同数值，其转发比例是一样的。
- 支持设置0-100的权重值，权重越高，接收的请求越多。
- 权重对加权轮询外的转发规则不生效，如果权重设置为“0”，则不再接收新的请求。

服务器安全组设置

应用型BLB实例的服务器安全组设置可以参考[普通型BLB实例的安全组设置](#)

编辑端口

在“开放端口”页签，点击<编辑>，跳入编辑界面，编辑后点击<确定>。

🔗 删除端口

1. 在“开放端口”页签，勾选要删除的监听设置，点击<删除>按钮。
2. 在弹出的界面，点击<确定>，完成释放。

完成服务器组基本信息填写、添加服务器和配置开放端口后，点击页面下侧“确认”，完成服务器组的创建步骤。

🔗 配置IP组

配置IP组步骤主要包括：

1. IP组基本信息填写
2. 配置开放协议；
3. 添加IP。

说明：IP组目前仅对白名单用户开放。如需，请提交[工单](#)。

🔗 IP组基本信息填写

操作步骤

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 选择左侧导航栏中的“目标组”，详情页面选择“IP组”页签，点击“添加IP组”，进入添加IP组配置页面,完成IP组名称和描述填写。

🔗 配置TCP开放协议

操作步骤

- 1.进入添加IP组配置页面,点击"新增协议"进行监听器的基本设置和健康检查设置,开放协议选择TCP。
- 2.设置配置信息。

项目	描述
开放协议	指定后端成员提供服务的协议。BLB服务转发请求至后端成员。

- 3.设置健康检查。

项目	描述
健康检查协议	指定检查后端成员健康状态时使用的协议类型。
响应超时时间	如果在指定时间内没有收到后端成员的响应，则认为后端成员响应超时。
健康检查间隔	对后端成员进行检查的时间间隔。
不健康阈值	如果后端成员连续健康检查失败的次数超过阈值，则认为该后端成员不健康。
健康阈值	对于不健康的后端成员，如果连续健康检查成功的次数超过阈值，则认为该后端成员健康。

- 4.完成上述配置后，点击“确认”按钮完成TCP开放协议的配置。

🔗 配置UDP开放协议

操作步骤

- 1.进入添加IP组配置页面,点击"新增协议"进行监听器的基本设置和健康检查设置,开放协议选择UDP。
- 2.设置配置信息。

项目	描述
开放协议	指定后端成员提供服务的协议和端口。BLB服务转发请求至后端成员。

- 3.设置健康检查。

项目	描述
健康检查协议	指定检查后端成员健康状态时使用的协议类型。
检查字符串	BLB采用给后端成员发送带有用户指定字符串的UDP包实现健康检查。 您需要配置应用程序来完成响应。对于后端服务器是DNS server的情况，您可以配置DNS解析请求作为健康检查包，默认值为对baidu.com的DNS解析请求（\00\01\01\00\00\01\00\00\00\00\00\05baidu\03com\00\00\01\00\01）。
响应超时时间	如果在指定时间内没有收到后端成员的响应，则认为后端成员响应超时。 输入范围为1～60间的整数，建议设置为3秒。
健康检查间隔	对后端成员进行检查的时间间隔。输入范围为1～10间的整数，建议设置为3秒
不健康阈值	如果后端成员连续健康检查失败的次数超过阈值，则认为该后端成员不健康。
健康阈值	对于不健康的后端成员，如果连续健康检查成功的次数超过阈值，则认为该后端成员健康。

- 4.完成上述配置后，点击“确认”按钮完成UDP开放协议的配置。

配置HTTP开放协议

- 1.进入添加IP组配置页面,点击"新增协议"进行监听器的基本设置和健康检查设置,开放协议选择HTTP。
- 2.设置配置信息。

项目	描述
开放协议[端口]	指定后端成员提供服务的协议。BLB服务转发请求至后端成员。

- 3.设置健康检查信息。

项目	描述
健康检查协议	指定检查后端成员健康状态时使用的协议类型。可选择“HTTP”。
Host头域	发往后端服务器的健康检查请求的Host头域，默认为空。
检查端口	指定后端成员提供服务的端口。本字段只对HTTP健康检查有效。
检查路径	用于健康检查页面文件的URI，建议对静态页面进行检查。 长度限制为1-80个字符，只能使用字母、数字、‘-’、‘/’、‘.’、‘%’、‘?’、‘#’、‘&’这些字符。本字段只对HTTP健康检查有效。
响应超时时间	如果在指定时间内没有收到后端成员的响应，则认为后端成员响应超时。
健康检查间隔	对后端成员进行检查的时间间隔。
不健康阈值	如果后端成员连续健康检查失败的次数超过阈值，则认为该后端成员不健康。
健康阈值	对于不健康的后端成员，如果连续健康检查成功的次数超过阈值，则认为该后端成员健康。
正常码状态	健康检查正常的http状态码，可选“http_2xx”、“http_3xx”、“http_4xx”、“http_5xx”，本字段只对HTTP健康检查有效

4.完成上述配置后，点击“确认”按钮完成HTTP开发协议的配置。

添加IP组

在服务器组详情页面点击“添加IP”，进入“添加IP”菜单栏，填写需要添加进入该IP组的IP地址+端口，并设置权重，最后点击“确认”，完成IP的添加。

说明：

- 添加IP过程中，IP可以跟应用型BLB实例在同一个VPC下；若内网可达，也可以是跨地域的VPC或者用户IDC内的IP。
- 配置后端成员即IP+端口的权重。权重是用来针对不同的后端成员设置不同的请求转发比例。例如有两个配置不同的后端成员，那么对于高配置的成员可以选择设置更高的权重来接收更多的外部访问请求。
- 权重值是一个相对值，如果设置成相同数值，其转发比例是一样的。
- 支持设置0-100的权重值，权重越高，接收的请求越多。
- 权重对加权轮询外的转发规则不生效，如果权重设置为“0”，则不再接收新的请求。

配置应用型BLB实例监听器

主要包括TCP、UDP、SSL、HTTP和HTTPS五种监听器的配置

TCP监听器

场景简介

TCP协议常见的应用场景有：注重可靠性、对数据准确性要求高和速度可以相对较慢的场景。

操作步骤

- 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
- 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
- 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“TCP”。

4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“80”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP算法为TCP监听器提供会话保持机制。此算法下，服务器权重不生效。

5. 高级选项

项目	描述
超时时间	指定TCP连接的超时时间，输入范围为10~4000间的整数，默认900秒。

6. 设置转发规则。

项目	描述
目标组	把匹配规则的流量转发至指定的服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：TCP监听器只有一条默认转发规则，且该规则不允许删除。例如：将该监听器接收到的流量转发到A服务器组的TCP 80端口。则在目标服务器组中，选择A服务器组，在目标端口中，选择TCP 80。

7. 完成上述配置后，点击“确认”按钮完成TCP监听器添加。

🔗 UDP监听器

场景简介

UDP协议常见的应用场景有：DNS服务、物联网及一些对数据传输可靠性要求不高的场景。

操作步骤

开启健康检查后，BLB将自动屏蔽不健康的后端成员。

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“UDP”。
4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“53”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP：将请求的源IP进行hash运算后派发请求至某匹配的服务器，这可以保证同一个客户端IP的请求始终被派发至某特定的服务器。源IP算法为UDP监听器提供会话保持机制。此算法下，服务器权重不生效。

5. 高级选项

项目	描述
超时时间	指定TCP连接的超时时间，输入范围为10~4000间的整数，默认900秒。

6. 设置转发规则。

项目	描述
目标组	把匹配规则的流量转发至指定服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：UDP监听器只有一条默认转发规则，且该规则不允许删除。例如：将该监听器接收到的流量转发到A服务器组的UDP 53端口。则在目标服务器组中，选择A服务器组，在目标端口中，选择UDP 53。

7. 完成上述配置后，点击“确认”按钮完成UDP监听器添加。

🔗 SSL监听器

场景简介

SSL协议常见应用场景有：需要超高性能和大规模TLS卸载的场景。

操作步骤

开启健康检查后，BLB将自动屏蔽不健康的后端成员。

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“SSL”。
4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“443”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP：将请求的源IP进行hash运算后派发请求至某匹配的服务器，这可以保证同一个客户端IP的请求始终被派发至某特定的服务器。源IP算法为UDP监听器提供会话保持机制。此算法下，服务器权重不生效。
服务器证书	在配置SSL监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看 证书管理 。 说明： 专属集群的BLB实例支持国际标准、国密（SM2）标准证书；共享集群的BLB实例只支持国际标准证书。
双向认证	选择是否开启双向认证。
加密选项	安全策略包含HTTPS可选的TLS协议版本和配套的加密算法套件。

5. 高级选项

项目	描述
超时时间	等待后端成员响应的超时时间；如果后端成员需较长时间处理请求，可适当调大该值，默认30秒。

6. 设置转发规则。

项目	描述
目标组	把匹配规则的流量转发至指定服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：SSL监听器只有一条默认转发规则，且该规则不允许删除。

7. 完成上述配置后，点击“确认”按钮完成SSL监听器添加。

🔗 HTTP监听器

场景简介

HTTP协议常见应用场景有：需要对数据内容进行识别的应用场景。

操作步骤

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“HTTP”。
4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“80”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。
超时时间	等待后端成员响应的超时时间；如果后端成员需较长时间处理请求，可适当调大该值，默认30秒。

5. 高级选项

项目	描述
会话保持	TCP监听器可以通过源IP转发规则实现会话保持；7层监听器（HTTP、HTTPS）通过插入/修改cookie实现会话保持。可选择“开启”或“关闭”。
附加HTTP头部	在HTTP请求中附加头字段，在后端成员处理报文时获取客户端真实IP或监听协议信息。 获取真实IP：开启该功能后，可将用户IP地址记录在X-Forwarded-For头域中； 获取监听协议：开启该功能后，可将监听协议记录在X-Forwarded-Proto头域中。
启用HTTPS重定向	将此监听器的HTTP请求重定向至此BLB的HTTPS监听器。请确保相应端口的HTTPS监听器工作正常。可选择“开启”或“关闭”。默认“关闭”。
HTTP请求允许非ASCII字符	默认关闭。开启此选项后如果HTTP请求中含有非ASCII字符，BLB会放行并转发给后端服务器处理。开启此选项可能带来安全风险，非特殊使用场景建议保持关闭。
数据压缩	开启将对特定文件类型进行压缩；关闭则不会对任何文件类型进行压缩，默认关闭
定制配置	BLB 7层监听支持定制配置，配置项和说明可参见 配置参数详情 。
HTTP2.0	开启后，支持HTTP2.0 版本协议连接请求

6. 设置转发规则。

项目	描述
优先级	优先级数值越小，则优先级越高，优先级不允许重复
匹配条件	可以设置域名、URL和匹配全部的转发规则
目标组	把匹配规则的流量转发至指定服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

- 注意：
- 单条规则同时出现域名和URL条件时，则需要同时满足条件，才算匹配成功
 - 域名转发规则匹配内容限制：
 - 限制长度<100
 - 通配符仅支持*
 - 通配符*可出现0次或1次
 - 通配符*可匹配0-n个字符
 - 仅支持前缀匹配，即 * 只可出现在域名的开头，如： *.baidu.com

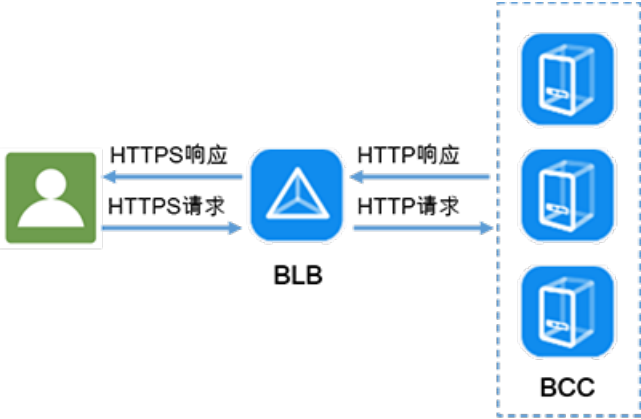
- URL转发规则匹配内容限制：
 - 限制长度<100
 - 通配符仅支持*
 - 通配符*可出现0次或1次
 - 通配符*可匹配0-n个字符
 - 支持前、中、后缀匹配，即*可出现在匹配规则的任意位置

例如：将该监听器接收到的流量转发到A服务器组的HTTP 80端口。则在目标服务器组中，选择A服务器组，在目标端口中，选择HTTP 80。

7. 完成上述配置后，点击“确认”按钮完成HTTP监听器添加。

🔗 HTTPS监听器

为了保证信息安全，用户可以配置BLB监听器的协议为HTTPS。BLB对于用户的HTTPS的请求进行解密，然后发送至后端成员；后端成员处理完请求后的返回包首先发送至BLB，由BLB进行加密后，再传回用户侧。整个过程如下图所示：



前置任务

在配置HTTPS监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看[证书管理](#)。

操作步骤

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“HTTPS”。
4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“443”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设置的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。
服务器证书	在配置HTTPS监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看 证书管理 。 说明： 专属集群的BLB实例支持国际标准、国密（SM2）标准证书；共享集群的BLB实例只支持国际标准证书。
扩展域名	选择是否开启扩展域名。
双向认证	选择是否开启双向认证。
加密选项	安全策略包含HTTPS可选的TLS协议版本和配套的加密算法套件。
超时时间	等待后端成员响应的超时时间；如果后端成员需较长时间处理请求，可适当调大该值，默认30秒。

5. 高级选项

项目	描述
会话保持	TCP监听器可以通过源IP转发规则实现会话保持；7层监听器（HTTP、HTTPS）通过插入/修改cookie实现会话保持。可选择“开启”或“关闭”。默认“关闭”。
附加HTTP头部	在HTTP请求中附加头字段，在后端成员处理报文时获取客户端真实IP或监听协议信息。 获取真实IP：开启该功能后，可将用户IP地址记录在X-Forwarded-For头域中； 获取监听协议：开启该功能后，可将监听协议记录在X-Forwarded-Proto头域中。
HTTP请求允许非ASCII字符	开启此选项后如果HTTP请求中含有非ASCII字符，BLB会放行并转发给后端成员处理。开启此选项可能带来安全风险，非特殊使用场景建议保持关闭。默认“关闭”。
数据压缩	开启将对特定文件类型进行压缩；关闭则不会对任何文件类型进行压缩，默认关闭
定制配置	BLB 7层监听支持定制配置，配置项和说明可参见 配置参数详情 。
HTTP2.0	开启后，支持HTTP2.0 版本协议连接请求

6. 设置转发规则。

项目	描述
优先级	优先级数值越小，则优先级越高，优先级不允许重复
转发规则	可以设置域名、HOST和两者合集的转发规则
目标组	把匹配规则的流量转发至指定服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：

- 单条规则同时出现域名和URL条件时，则需要同时满足条件，才算匹配成功
- 域名转发规则匹配内容限制：
 - 限制长度<100

- 通配符仅支持*
- 通配符*可出现0次或1次
- 通配符*可匹配0-n个字符
- 仅支持前缀匹配，即 * 只可出现在域名的开头，如： *.baidu.com
- URL转发规则匹配内容限制：
 - 限制长度<100
 - 通配符仅支持*
 - 通配符*可出现0次或1次
 - 通配符*可匹配0-n个字符 - 支持前、中、后缀匹配，即*可出现在匹配规则的任意位置

例如：将该监听器接收到的流量转发到A服务器组的HTTP 80端口。则在目标服务器组中，选择A服务器组，在目标端口中，选择HTTP 80。

7. 完成上述配置后，点击“确认”按钮完成HTTPS监听器添加。

🔗 编辑监听

在“监听设置”页签，点击<编辑>，跳入编辑界面，编辑后点击<确定>。

🔗 释放监听

1. 在“监听设置”页签，勾选要释放的监听设置，点击<释放>按钮。
2. 在弹出的界面，点击<确定>，完成释放。

普通型IPv6 BLB实例

IPv6 负载均衡是基于 IPv6 单栈技术实现的负载均衡，和 IPv4 负载均衡协同工作，实现 IPv6/IPv4 双栈通信。IPv6 负载均衡创建后，系统会为实例分配一个IPv6地址，转发来自IPv6客户端的请求。

🔗 创建普通型IPv6实例

1. 登录百度智能云平台，选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面。
2. 点击“创建负载均衡”，进入创建实例界面。



3. 选择配置信息。

配置项	说明
付费及地域	可选择付费方式与当前地域
名称	可自定义IPv6实例名称
实例类型	选择普通型IPv6实例
所在网络	当前创建的 BLB 和相应 BCC 等云资源所属的私有网络(VPC)
分配IPv6	IPv6实例必须分配一个IPv6 IP，可选择自动分配或者指定
安全组	可选择已创建的安全组或默认安全组
标签	按各种标准（如用途、所有者或项目）对资源进行分类；每个标签包含键和值两部分。
资源分组	可选择资源分组
购买信息	可选择购买数量并设置自动释放信息等

4. 确认配置信息并根据系统提示，完成订单确认及支付。创建成功后，用户将收到短信提示。
5. 选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面，查看BLB实例相关信息，具体内容包包括：

项目	描述
实例名称/ID	BLB实例的名称及ID，通过点击实例名称，可以进入BLB实例详情页面，如步骤2所示
状态	BLB实例当前的工作状态，若此处显示为“运行中”，代表BLB实例正常工作
内网IP	创建时分配或制定的IPv6 IP，BLB提供均衡服务的访问地址
所在网络	BLB所在的私有网络VPC，如无特殊指定，系统提供默认私有网络
所在子网	BLB所在的私有网络VPC，如无特殊指定，系统提供默认私有网络
前端协议/端口	BLB实例所关联的监听器的协议（eg. HTTP, TCP）及端口号
后端协议/端口	BLB实例所关联的监听器转发至后端BCC的协议及端口号
后端服务器	BLB实例所关联的BCC
资源分组	用户自定义资源分组

配置监听器

配置TCP监听器

- 配置TCP监听包括“配置信息”和“健康检查设置”两部分。开启健康检查后，BLB将自动屏蔽不健康的后端服务器。
- 1.选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面。
2. 点击负载均衡的名称，进入实例详情页。
3. 点击左侧导航栏的“监听设置”>“添加监听设置”，进行监听器的基本设置和健康检查设置。



健康检查设置

开启健康检查：

☒ 开启 ☐ 关闭 ?

健康检查协议：

TCP

检查端口：

?

* 响应超时时间：

 秒 ?

* 健康检查间隔：

 秒 ?

* 不健康阈值：

2345

3

^

v

?

* 健康阈值：

2345

3

^

v

?

4. 设置配置信息。

配置项	说明
BLB协议端口	选择TCP类型并填写端口
后端协议端口	填写后端服务端口
转发规则	可选择加权轮询、加权最小连接数或源IP
超时时间	指定TCP连接的超时时间，输入范围为10～4000间的整数，默认900秒
ProxyProtocol协议	启用ProxyProtocol协议以获取客户端源IP、源Port，访问目的IP、目的Port，访问协议，开启前请务必先配置后端服务器支持ProxyProtocol协议。

5. 设置健康检查。

配置项	说明
响应超时时间	输入范围为1～60间的整数，建议设置为3秒
健康检查间隔	输入范围为1～10间的整数，建议设置为3秒
不健康阈值	连续健康检查失败次数，超过这个阈值，服务器将被认定为异常，从服务器池中摘除，直到恢复正常
健康阈值	连续健康检查成功次数，超过这个阈值，服务器将被认定为异常排除，重新加入服务器池。

6. 完成上述配置后，点击“确认”按钮完成TCP监听器添加。

配置UDP监听器

场景简介

UDP流量的常见应用场景有：DNS服务、物联网及一些对数据传输可靠性要求不高的场景。

操作步骤

配置UDP监听包括“配置信息”和“健康检查设置”两部分。 开启健康检查后，BLB将自动屏蔽不健康的后端服务器。 1.选择“产品

- > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面。
2. 点击负载均衡的名称，进入实例详情页。
3. 点击“监听设置”>“添加监听设置”，进行监听器的基本设置和健康检查设置。BLB协议选择“UDP”。

返回实例列表

实例详情

监听设置

后端服务器

安全组

监控

配置信息

BLB协议[端口]:

UDP

53

后端协议[端口]:

TCP

转发规则:

UDP

超时时间:

30

秒

健康检查设置

健康检查协议:

UDP

ICMP

检查字符串:

\001010100100100100100101

响应超时时间:

3

秒

健康检查间隔:

3

秒

不健康阈值:

2

3

4

5

3

健康阈值:

2

3

4

5

3

确定

取消

4. 设置配置信息。

配置项	说明
BLB协议端口	选择UDP类型并填写端口
后端协议端口	填写后端服务端口
转发规则	可选择加权轮询、加权最小连接数或源IP
超时时间	指定UDP连接的超时时间，输入范围为5～4000间的整数，默认30秒。
ProxyProtocol协议	可选择是否启用ProxyProtocol协议以获取客户端源IP、源Port，访问目的IP、目的Port，访问协议

5. 设置健康检查。

配置项	说明
响应超时时间	输入范围为1～60间的整数，建议设置为3秒
健康检查间隔	输入范围为1～10间的整数，建议设置为3秒
不健康阈值	连续健康检查失败次数，超过这个阈值，服务器将被认定为异常，从服务器池中摘除，直到恢复正常
健康阈值	连续健康检查成功次数，超过这个阈值，服务器将被认定为异常排除，重新加入服务器池。

6. 完成上述配置后，点击“确认”按钮完成UDP监听器添加。

编辑监听设置

在“监听设置”页签，点击“编辑”，跳入编辑界面，编辑后点击“确定”。

返回实例列表

BLB-shanghai-db0b27a2

运行中

实例详情

+ 添加监听设置

释放

监听设置

BLB协议: 端口

后端协议: 端口

转发规则

获取真实IP

会话保持

健康检查

操作

UDP: 53

UDP: 53

加权轮询

已开启

-

已开启

编辑

后端服务器

安全组

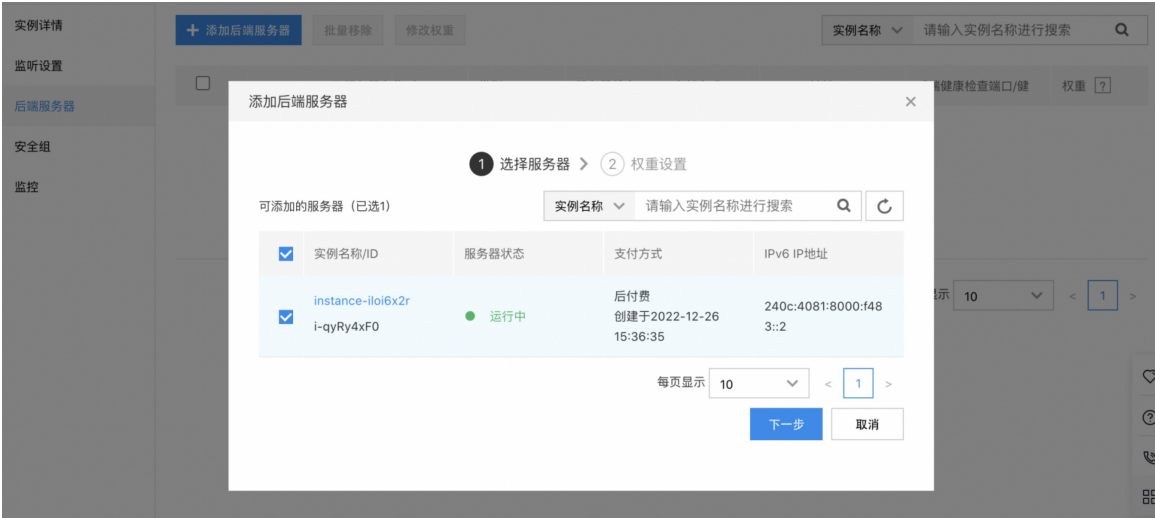
🔗 释放监听设置

1. 在“监听设置”页签，勾选要释放的监听设置，点击“释放”按钮。
2. 在弹出的界面，点击“确定”，完成释放。

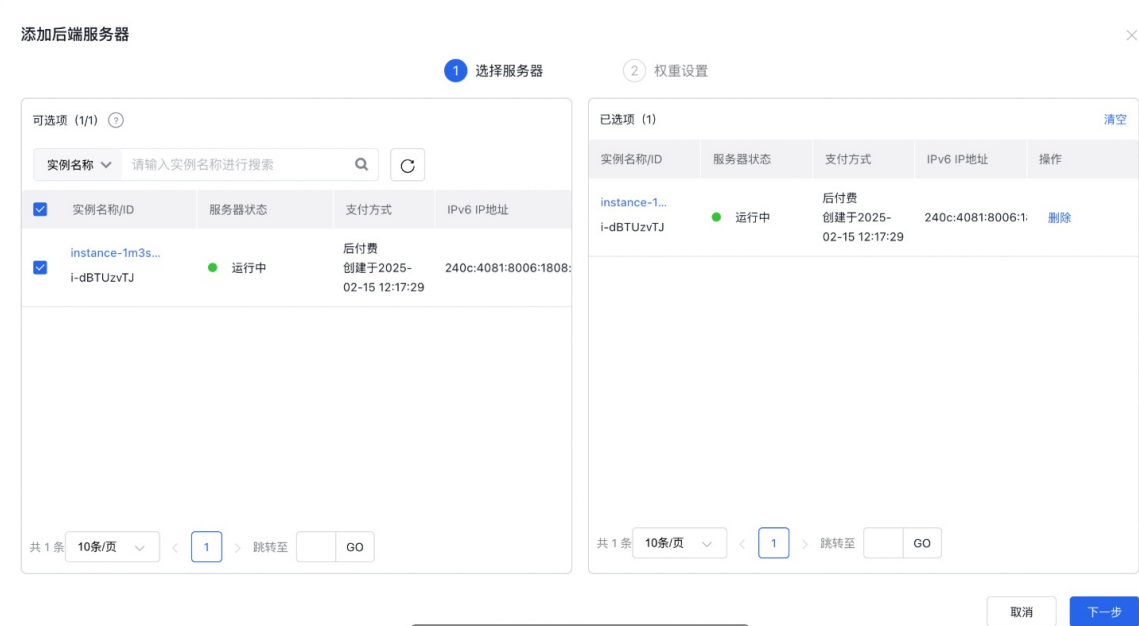
🔗 配置后端服务器

🔗 添加后端服务器

1. 选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面。
2. 点击负载均衡的名称，进入实例详情页。
3. 点击“后端服务器”>“添加后端服务器”，进入后端服务器列表。后端服务器列表中显示了与当前BLB所在私有网络VPC下的IPv6 BCC实例名称、状态等信息。说明：此处后端服务器列表仅展示IPv6 类型实例，并不是全部展示。



4. 配置后端服务器权重。权重是用来针对不同的后端服务器实例设置不同的请求转发比例。例如有两台配置不同的后端服务器实例，那么对于高配置的实例可以选择设置更高的权重来接收更多的外部访问请求。
- 权重值是一个相对值，如果设置成相同数值，其转发比例是一样的。
 - 支持设置0-100的权重值。权重越高，接收的请求越多；如果权重设置为“0”，则不再接收新的请求。
 - 非0权重对加权轮询和加权最小连接数外的转发规则不生效。



5. 勾选要添加的一台或多台后端服务器实例，点击<确定>，将其加入负载均衡服务。

🔗 修改后端服务器权重

1. 在“后端服务器”页签勾选需要修改的后端服务器实例。
2. 点击“修改权重”，可重新选择后端服务器并配置其权重值，具体配置方法可参看添加后端服务器。

🔗 移除后端服务器

1. 在“后端服务器”页签勾选需要移除的后端服务器实例。
2. 点击“批量移除”，完成移除操作。

🔗 设置安全组

目前BCC默认安全组的进站规则是允许所有流量。如果您需要更精细化地对安全组规则进行配置，请参考以下指导。BLB通过特定的内网网段与BCC交互来完成健康状态检查。所以您在配置BCC安全组时，需要注意为安全组的进站规则放开这些内网IP段。由于4层监听器（eg.TCP协议）默认开启真实IP功能（数据包的源IP就是client IP），所以用户BCC安全组的进站规则还需要额外放开client IP地址；7层监听器（eg.HTTP协议）的真实IP功能是通过X-Forwarded-For header实现，所以7层监听器的后端BCC安全组无需放开client IP地址。BCC安全组的进站规则需要对源IP进行如下配置：

监听器类型	公网BLB	内网BLB
4层监听器 (eg.TCP协议)	允许用户Client所在的公网IP段 且 允许100.64.0.0/10（健康检查）	允许10.0.0.0/8网段（内网BCC数据包的IP地址会被转换为10.0.0.0/8网段的IP地址，所以在BLB的后端服务器看来Client请求是来自10.0.0.0/8网段的）且 允许100.64.0.0/10网段（健康检查）
7层监听器 (eg.HTTP协议)	允许100.79.0.0/16网段	允许100.79.0.0/16网段

安全组开放的端口应至少包含BLB后端服务端口和健康检查端口。

🔗 关联安全组

1. 选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击“安全组”，进入安全组页面。当前应用的安全组规则列表中选择出站或者进站规则；关联的安全组规则列表中

返回实例列表

实例详情

监听设置

后端服务器

安全组

监控

BLB-beijing-4cdc24ab运行中

当前应用的安全组规则

进站

出站

协议	端口范围	Source	描述
all	1-65535	源IP: 180	

关联的安全组列表

关联安全组

安全组名称	所在网络	描述	操作
默认安全组	对等连接-北京-跨账号		展开规则 取消关联

4. 点击“关联安全组”或者“立即关联”，进入可关联的安全组列表



5. 勾选负载均衡需要关联的安全组并确定，则该安全组被关联，安全组规则应用于当前实例。

关联安全组

温馨提示：勾选的安全组为负载均衡关联的所有安全组，并非增加关联的安全组，请勾选负载均衡需要关联的安全组。普通安全组和企业安全组不能同时使用，即负载均衡关联的安全组只能是同一种类型。

您选择的需要关联的负载均衡为lb-4cba0ab4，请在下方选择需要关联的安全组

安全组类型：普通安全组 企业安全组

☐	安全组名称	所在网络	安全组描述
☐	KAFKA默认安全组	default	KAFKA默认安全组，禁止任何操作和删除，后果自负
☐	默认安全组	default	default

取消 确定

取消关联安全组

- 1. 在“安全组”页，“关联的安全组列表”操作列签点击取消关联。
- 2. 在确认弹窗中点击<确定>，完成取消关联，当前应用的安全组规则相应移除。



查看监控数据

点击BLB的详情页的“监控”页签，就可以看到监控的各项图表数据：包括端口监控和后端服务器监控信息。

查看端口监控

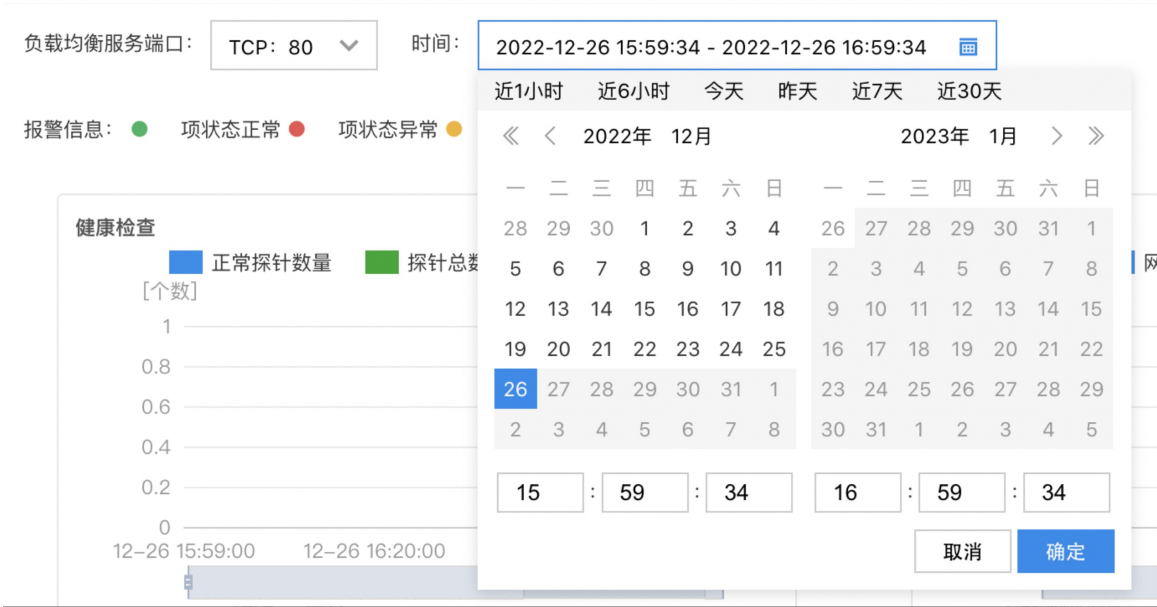
- 1. 选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面。
- 2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
- 3. 点击“监控”，进入监控页面，点击“端口监控”，显示监控情况。



4.对

于4层监听器，端口监控信息包括：

- 健康检查
 - 探针总数：已添加的后端服务器数量。
 - 正常探针数量：健康检查正常的后端服务器数量。
 - 异常探针数量：健康检查异常的后端服务器数量。
 - 网络流量
 - 网络流入流量：监听器接收到的用户流量，单位：字节/分钟。
 - 网络流出流量：监听器发往用户的流量，单位：字节/分钟。
 - 网络数据包
 - 输入数据包数：监听器接受到的用户数据包（Packet）数量，单位：个/分钟。
 - 输出数据包数：监听器发往用户的数据包（Packet）数量，单位：个/分钟。
 - 并发连接数：并发TCP连接数。
 - 丢弃流量：丢弃的输入输出流量。
 - 丢弃数据包：丢弃的输入输出数据包。
 - 丢弃连接数：未正确建立的被丢弃的连接
 - 新建连接数：时间段内新建的连接数
- 5.对于7层监听，端口监控信息除了包括以上内容之外，还包括：
- 请求数：监听器接收到的HTTP请求的数量，单位：个/分钟。
- 6.系统默认给出最近1小时内的监控信息，如果需要查看更多内容，可点击时间，切换时间范围，如下图所示：



通过API查看

详细的接口请参考云监控BCM查询数据接口。其中，获取BLB监控的请求参数“scope”, “metricName”和“dimensions”，请参照下表：

scope	metricName	dimensions
BCE_BLB	UnhealthyCount 异常实例数量	BlbId + BlbPort 通过"."号拼接起来，例如：BlbId=lb-dd45e123，BlbPort=80 那么instance为：lb-dd45e123.80
	TotalCount 实例总数	
	HealthyCount 正常实例数量	
	RequestCount 请求数	
	NetworkInBytes 网络流入流量	
	NetworkOutBytes 网络流出流量	
	ActiveConnCount 活跃连接数	
	NetworkInPkgCount 输入数据包数	
	NetworkOutPkgCount 输出数据包数	

管理报警

在监控页面点击“报警详情”进入报警策略配置页面，可以管理报警策略，详细操作步骤请见[BCM管理报警](#)。



释放IPv6实例

手动释放BLB实例

1. 选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面。
2. 勾选要释放的IPv6实例，点击“释放”。
3. 在弹出的对话框中，点击“确定”。

自动释放BLB实例

设置自动释放IPv6实例后，实例会在设置的时间自动进行释放。有两种方式自动释放IPv6实例：

- 方式一：对于新建IPv6实例，可在创建BLB实例的页面进行设置。 1.点击“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面，选择“创建负载均衡”。 2.在创建BLB实例的同时设置该BLB实例的自动释放时间，如下图所示。

购买信息

购买数量：

-

1

+

最多可开通60个BLB实例，还可开通17个

自动释放：

关闭

开启

自动释放时间：

2025-04-03 19:06:46

📅

- 方式二：对于已经创建的BLB实例，可在BLB实例详情页面进行设置。

- 选择“产品 > 网络 > 负载均衡”，进入负载均衡BLB管理控制台页面，点击“普通型IPv6 实例”进入实例列表页面。
- 点击负载均衡名称，进入实例详情页。
- 点击“自动释放时间-设置”。

wendangtest 运行中

基本信息

ID:

lb-4cba0ab4

性能规格:

标准型1

前端IP:

240c:4081:8006:1842::4

到期时间:

-

IPv6 EIP:

-

所在网络:

default (240c:4081:8006:1800::/56)

支付方式:

后付费-按固定规格

创建时间:

2025-04-02 20:51:32

释放保护:

🔔 关

四层所在集群:

共享集群 迁移集群

名称:

wendangtest

所在子网:

系统预定义子网D (240c:4081:8006:1842::/64)

描述:

-

支持ACL:

🔔 开

七层所在集群:

共享集群 迁移集群

实例性能

并发连接数:

50000

每秒新建连接数:

5000

每秒查询数:

5000

带宽上限:

1Gbps

运行状态信息

运行状态:

运行中

自动释放时间:

- 设置

- 在弹出窗口中选择“设置自动释放”，同时在下方配置释放日期和具体时间，点击“确认”完成设置。

自动释放设置

自动释放设置：

☐ 不设置自动释放

☒ 设置自动释放

2025-04-03

18

时

40

分

取消

确定

扩展域名

负载均衡HTTPS监听支持挂载多个证书，通过配置扩展域名，实现将来自不同访问域名的请求转发至不同的后端服务器或者后端服务器组。

添加扩展域名

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB”，点击BLB名称进入实例详情页面。
2. 点击左侧导航栏“监听设置”，编辑已有HTTPS协议监听或添加监听设置。
3. 在服务器证书设置好的前提下，点击开启“扩展域名”。此时，可以看到扩展域名配置项，域名和证书名称一一对应。

返回监听设置列表

添加监听设置

基本配置

BLB协议[端口]:

HTTPS

443

?

调度方法:

加权轮询

?

大搜泛域名证书:

开启

关闭

服务器证书:

quic.codoge.cc/quic.codoge.cc

添加证书

管理我的证书

申购SSL证书

扩展域名:

开启

关闭

域名	证书名称（证书域名）	操作
请输入扩展域名	-请选择一个证书-	删除
+ 添加扩展名		

双向认证:

开启

关闭

加密选项:

预定义的安全策略

自定义安全策略

?

配置项说明如下，

配置项	说明
域名	填写域名，域名支持精确域名和泛域名
证书名称	选择该域名关联的证书

4. 在应用型BLB实例中，开启扩展域名后，可以根据需要在监听规则中设置匹配条件为域名，这样可以为不同的域名指定不同的后端目标组和目标协议。

+ 添加监听设置

释放

☐	BLB协议: 端口	调度方法	获取真实IP	会话保持	转发规则数量	描述
☐	HTTPS: 443	加权轮询	已开启	已关闭	0	- 编辑

+ 添加规则

优先级	匹配条件	匹配内容	目标组	目标端口	备注

说明：

- 证书中的域名和您添加的证书域名必须是一致的。
- 当前端请求同时匹配多条域名策略时，策略的匹配优先级为最长匹配原则，比如：精确匹配高于小范围通配符匹配，小

80

范围通配符匹配高于大范围通配符匹配。

- 扩展域名需要和转发策略配合使用才能生效，需要配置相同域名转发策略才能生效。

访问控制

创建负载均衡后，您可以通过配置负载均衡所在子网对应的ACL或者关联安全组，实现负载均衡实例级别的访问控制。

说明： Ping负载均衡的内网IP或者绑定的EIP，由负载均衡集群响应，不会转发到后端服务器，不受ACL和安全组的访问控制。

🔗 BLB支持ACL

您能够在负载均衡所在的子网对应的ACL中配置相应的规则，实现负载均衡实例级别的访问控制。ACL使用可参照[ACL操作指南](#)。

🔗 操作步骤

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB”，点击BLB名称进入实例详情页面。
2. 在BLB实例详情页面开启支持ACL开关，即负载均衡所在的子网对应的ACL规则将对该BLB实例生效。
3. 在负载均衡所在的子网对应的ACL中配置相应的规则。



🔗 BLB支持安全组

您可以将该安全组关联绑定到对应的负载均衡实例上，这样该负载均衡实例才能按安全组规则的设定实现网络访问控制功能。安全组的操作详见[安全组文档](#)。

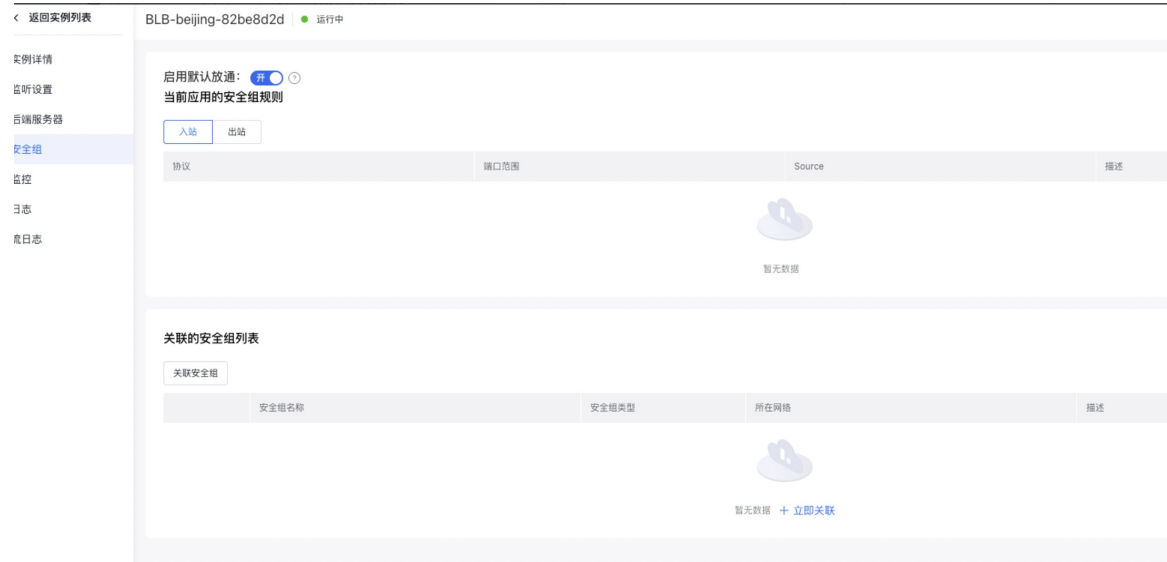
🔗 前提条件

将负载均衡实例关联到安全组之前，请确认以下信息：

- 您必须已经成功创建负载均衡实例。
- 您可以根据业务需求，将负载均衡实例关联一个或多个安全组，每个负载均衡实例最多关联10个安全组。
- 负载均衡实例关联安全组时，目标安全组与负载均衡必须在同一个私有网络。

🔗 操作步骤

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB”，点击BLB名称进入实例详情页面。
2. 点击左侧导航栏“安全组”，选择关联安全组。
3. 关联安全组后，即可查看当前应用的安全组规则。



说明：

- 普通IV及以上机型，后端服务器绑定的安全组不会阻断BLB的流量，BLB和BCC之间默认放行。
- 普通III及以下机型，来自BLB的流量将受到BCC安全组的管控。

访问日志

BLB支持7层（HTTP/HTTPS）访问日志收集。访问日志可以帮助您了解客户端请求，分析用户行为，辅助故障排查等。开启访问日志服务后，我们会将您的原始访问日志推送至您选择的对象存储（BOS）bucket，BOS存储可以保证您的日志永久性存储并方便您进行日志的管理、下载与分析。在配置访问日志前，您需要确保：

- 已经创建了七层负载均衡。
- 已经开通了BOS服务。

费用说明：关于访问日志功能的计费，负载均衡侧暂不收费，BOS侧收费参照[BOS计费规则](#)。

配置访问日志

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB”，点击BLB实例名称进入BLB实例详情页面。
2. 点击左侧导航栏“日志”，进入日志配置页面。



3. 填写下列配置信息：

配置项	说明
存储到	必填，bucket必须存在，如不存在，需要新建；folder可不存在，如不存在，系统会自动创建
日志前缀	非必填，可自定义日志前缀

4. 点击“保存”，日志设置完成。
5. 配置负载均衡访问日志后，您可以在BOS中下载并查看访问日志。

字段	类型	说明
time_local	string	记录访问日志的时间
vip	string	BLB IP
vport	int	BLB监听端口
response_status	int	BLB应答报文的状态码
vdum_ups_id	string	流量访问的服务器组ID
upstream_addr	string	流量访问的后端服务器地址
upstream_status	string	流量访问后端服务器的响应码
remote_addr	string	与BLB建连的远端client IP，如果BGW不开启cip_forward，则为BGW BIP
cip	string	与BLB建连的远端client IP
cport	int	与BLB建连的远端client 端口
server_addr	string	访问的BLB的虚拟服务器IP
server_port	int	访问的BLB的虚拟服务器端口
scheme	string	协议类型（HTTP/HTTPS）
host	string	域名
request	string	请求行信息
request_length	int	请求长度
body_bytes_sent	int	发送的响应的字节数
http_referer	string	BLB收到请求报文中的refer header内容
http_user_agent	string	BLB收到请求报文中的user-agent header内容
http_x_forwarded_for	string	BLB收到请求报文中的x-forward-for内容
upstream_connect_time	float	BLB与后端服务器的建连时间
upstream_response_time	float	BLB从转发请求到后端服务器到收到后端服务器的响应的的时间
request_time	float	BLB从接收到client的请求到返回响应给client的时间
ssl_handshake_time	float	SSL握手时间
ssl_session_reused	string	SSL会话是否复用，复用该字段为r，否则为.
vdum_rs_proto	string	后端服务器协议类型
upstream_local_addr	string	BLB业务节点内部使用ip，用户无需关注
dylimit_id	string	BLB实例id
dylimit_status	string	BLB实例限速状态，0表示不限速，1表示限速
dylimit_raw_qps	long	BLB实例在当前业务节点上的qps，不限速时显示为0
dylimit_token	long	BLB实例在当前业务节点上的qps令牌个数，不限速时显示为0

定制配置

BLB支持定制配置功能，允许用户设置BLB 7层监听的配置参数，如 client_header_timeout、client_body_timeout、proxy_buffer_size、proxy_buffers、proxy_set_header等，满足用户的定制配置需求。

说明：

- 个性化配置在每个地域暂无个数限制。
- 当前一个BLB实例允许绑定多个个性化配置，一类个性化配置可以绑定多个实例。
- 个性化配置仅针对负载均衡（原“应用型负载均衡”）的七层 HTTP/HTTPS 监听器生效。

添加定制配置

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB”，点击BLB名称进入实例详情页面。
2. 点击左侧导航栏“监听设置”，选择HTTP或HTTPS协议配置7层监听。
3. 开启定制配置开关，添加定制配置。

华北 - 北京

返回监听设置列表

添加监听设置

基本配置

BLB协议[端口]:

HTTP

80

调度方法:

加权轮询

超时时间:

30

秒

高级选项

会话保持:

开启

关闭

附加HTTP头部:

开启

关闭

获取真实IPX-Forwarded-For方式获取客户端真实IP

获取监听协议X-Forwarded-Proto方式获取监听协议

启用HTTPS重定向:

开启

关闭

HTTP请求允许非ASCII字符:

开启

关闭

定制配置:

开启

关闭

配置参数说明

当前BLB的定制配置支持如下字段：

配置字段	参数范围	默认值	配置说明
client_header_timeout	30-120	60s	读取客户端请求头的超时时间(s)，如果客户端在这个时间内未发送完整的请求头，BLB会中断请求
client_header_buffer_size	1-16	1k	存放客户端请求头的缓冲区大小(k)
client_body_timeout	30-120	60s	读取客户端请求体的超时时间(s)，该超时时间指BLB连续两次成功读到请求体的间隔时间，而非整个请求体传输时间。如果客户端在这个时间内没有发送任何数据，BLB会中断请求
proxy_buffer_size	1-64	4k	BLB读取来自后端服务器响应头的缓冲区大小 (k)，如果后端服务器响应头超过这个大小，BLB会返回502。使用 proxy_buffer_size 时，必须同时设置 proxy_buffers
proxy_buffers	[3-8] [4-8]	8 4k	BLB读取来自后端服务器响应体的缓冲区数量和每个缓冲区大小(k)
proxy_buffering	on , off	on	- on 表示缓存来自后端服务器的响应：BLB 会缓存响应，全部接收完成后返回给后端服务器 - off 表示不缓存来自后端服务器的响应：BLB 不会缓存响应，此时会导致后端服务器有一定性能压力 也可以通过在响应头里添加X-Accel-Buffering控制是否缓存，X-Accel-Buffering: yes表示缓存，X-Accel-Buffering: no表示不缓存
proxy_set_header	- X-BLB-Cport \$cport - X-BLB-Vip \$vip - X-BLB-Request-Id \$bce_blb_request_id - X-BLB-Client-Cert \$ssl_client_cert - X-Real-IP \$cip	无	向后端传递的请求头 - 客户端端口 - BLB的Vip - 唯一请求ID - HTTPS请求的客户端证书（仅HTTPS监听生效） - 客户端IP
gzip_json	on , off	off	- on 表示在数据压缩中启用gzip数据压缩功能时，gzip压缩支持的类型额外包含 application/json。当client 支持gzip压缩，且响应Content-Type类型在gzip压缩支持类型列表中，BLB侧会使用gzip压缩响应内容。 - off 表示gzip压缩支持的类型不额外包含 application/json 数据类型，具体支持的数据类型列表请参考数据压缩介绍。

说明： 其中，proxy_buffer_size 和 proxy_buffers 配置的值需要满足约束条件： $2 \max (proxy\_buffer\_size, proxy\_buffers.size) \leq (proxy\_buffers.num - 1) \times proxy\_buffers.size$ 。例如，配置 proxy_buffer_size 为 24k，proxy_buffers 为 8 8k，则 $2 \times 24k = 48k$ ， $(8 - 1) \times 8k = 56k$ ，此时 $48k \leq 56k$ ，因此配置不会报错，否则报错。

配置代码示例：

```
client_header_timeout:60
client_body_timeout:60
proxy_buffer_size:4
proxy_buffers:8 4
proxy_set_header:X-Blb-Cport $cport
```

负载均衡专属集群

负载均衡专属集群提供性能可控、资源独享、物理资源隔离的专属负载均衡服务，满足超高性能和独占资源需求。

负载均衡专属集群使用专属物理服务器实现网络资源层面的硬隔离，用户可为不同业务部门申请创建多个独享专属集群，不同业务部门在与之对应的专属集群上产品化创建负载均衡实例。如需使用，可[提交申请](#)。

创建专属集群

- 登录百度智能云平台，选择“产品服务>负载均衡BLB>BLB专属集群”，进入BLB专属集群列表页面。
- 点击“创建专属集群”，进入创建专属集群界面。
- 填写下列配置信息。

配置项	说明
付费方式	仅支持预付费
当前地域	支持北京、保定、广州、苏州、武汉、上海、香港、阳泉、成都、南京，通过左上角区域进行切换
专属集群名称	用户自定义BLB专属集群名称
集群类型	选择集群类型，支持4层或者7层
性能容量	选择CCU的个数，性能容量单位CCU是用来衡量BLB集群处理流量时涉及各个指标，专属集群是互为主备机房容灾架构，购买的CCU是互为主备两个机房性能之和
描述	编辑BLB专属集群相关描述信息
购买时长	选择购买的时长
自动续费	开启后将自动续费

1个CCU的集群性能容量参数：

- 4层集群：最大并发连接数 8000万、新建连接速度 1200万、入向bps 120Gbps、出向bps 120Gbps。
- 7层集群：最大并发连接数 200万、新建http连接速度 120万、新建https连接速度 22.4万、入向bps 40Gbps、出向bps 40Gbps、http qps 120万、https qps 22.4万。

- 确认配置信息并根据系统提示，完成订单确认及支付。
- 负载均衡专属集群提交创建后，百度智能云工作人员会在1-3个工作日内审核您提交的配置信息，审核通过后，则进入负载均衡专属集群搭建环节，完成搭建后，您将收到短信提示。

BLB专属集群

7天即将到期 自动续费

+ 创建专属集群 续费 释放

集群名称 请输入集群名称进行搜索 下载全部

☐ 集群名称/ID	状态	集群类型	可用区	集群性能	创建时间	到期时间	描述	操作
☐ example bgw_group-2b...	可用	4层	zoneA zoneB	4 CCU	2021-09-01 14:21:38	2021-12-30 16:49:09	-	集群升级

升级专属集群

1. 需要升级的负载均衡专属集群，点击“集群升级”按钮。选择需要变更的性能容量配置，点击“下一步”确认订单并支付。
2. 完成支付后百度智能云工作人员将在1-3个工作日内审核您提交的变配信息，审核通过后，则进入负载均衡专属集群升级环节，完成升级后，您将收到短信提示。

BLB专属集群									
7天即将到期 自动续费									
+ 创建专属集群 续费 释放									
集群名称 请输入集群名称进行搜索 下载全部									
☐	集群名称/ID	状态	集群类型	可用区	集群性能	创建时间	到期时间	描述	操作
☐	example bgw_group-2b...	可用	4层	zoneA zoneB	4 CCU	2021-09-01 14:21:38	2021-12-30 16:49:09	-	集群升级

管理BLB实例

BLB实例选择所在集群

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB>BLB实例”，进入BLB实例列表页面。
2. 点击“创建负载均衡”，进入创建负载均衡界面。
3. 选择所在集群和主备模式。

说明：

- 若4层、7层的所在集群均选择了专属集群，则BLB实例不再收费。

百度智能云 控制台总览 华北 - 北京

返回实例列表 创建实例

1 基本配置 2 配置公网IP 3 确认订单 4 在线支付 5 创建成功

付费及地域

付费方式: 后付费

当前地域: 华北 - 北京

基础信息

名称: 请输入实例名称

实例类型: 普通型实例 应用型实例 IPv6实例

4层所在集群: 专属集群 共享集群

example/bgw_group-2bc5...

主备模式: 双主 主备

7层所在集群: 专属集群 共享集群

l7/nginx_group-a7aacab

主备模式: 双主 主备

所在网络: test_nat_cfw (192.168.0.0/16) cfw01 (192.168.1.0/24)

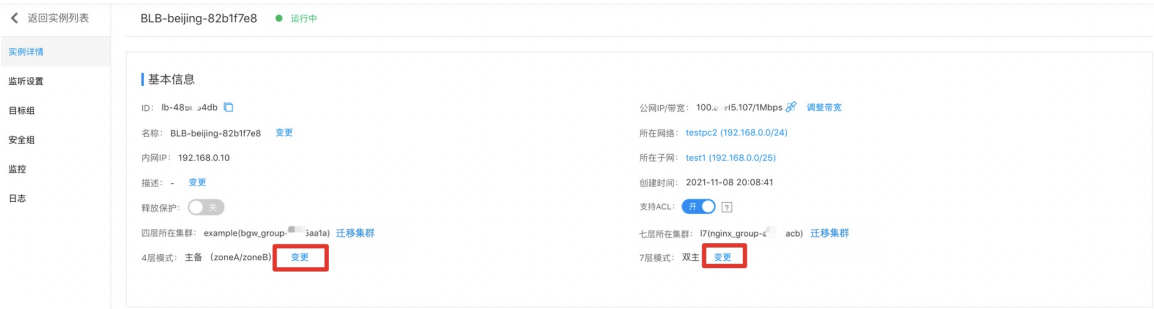
IP地址: 自动分配 指定

安全组: 请选择

BLB实例变更主备模式

点击BLB实例名称进入详情页面，在BLB详情页面基本信息中可点击变更主备模式，支持主备和双主相互变更。

说明：变更主备模式时，对BLB的访问长短链接均有感，有感时长约为对应监听器的健康检查间隔*健康阈值，后续发起重试可恢复，若希望减少迁移受损时间，可减小以上两个参数设置。



BLB实例迁移所在集群

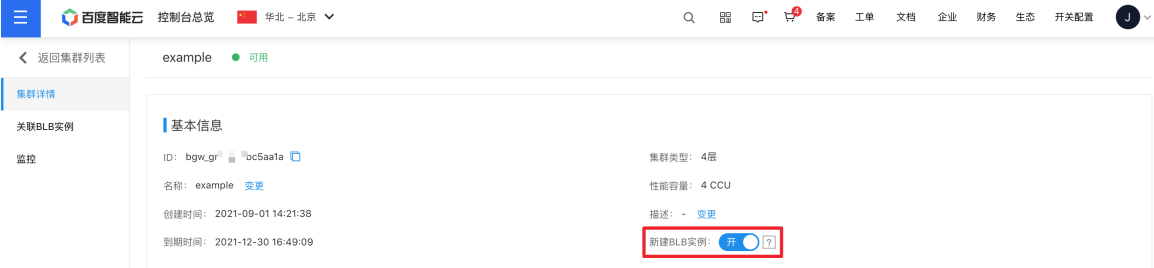
点击BLB实例名称进入详情页面，在BLB详情页面基本信息中可点击“迁移集群”，支持共享集群迁移专属集群，支持专属集群迁移到共享集群，也支持从一个专属集群迁移到另一个专属集群。

注意：
迁移过程有短暂的断流，请谨慎操作。



新建BLB实例开关

点击集群名称进入详情页面，可配置新建BLB实例开关。默认开启，若关闭，则该负载均衡专属集群不允许再新建BLB实例。



查看关联BLB实例

点击集群名称进入详情页面，在左侧导航栏选择“关联BLB实例”进入关联BLB实例列表页面，可查看该专属集群关联的BLB实例信息。

返回集群列表

集群详情

关联BLB实例

监控

example运行中

关联BLB实例列表

实例名称/ID	状态	实例类型	公网IP/线路类型/带宽	内网IP	IPv6 IP	所在网络	所在子网
BLB-beijing-82b... lb-489864db	运行中	应用型实例	100.88.10.107/1Mbps	192.168.0.10	-	testpc2	test1
fsa lb-a67620ee	运行中	普通型实例	100.88.9.1/1Mbps	192.168.0.9	-	testpc2	test1

查看监控数据

1. 点击集群名称进入详情页面，在左侧导航栏选择监控进入监控页面，可查看专属集群监控以及水位信息。
2. 在监控页面点击“报警详情”进入报警策略配置页面，可以管理负载均衡专属集群的报警策略，详细操作步骤请见[BCM管理报警](#)。

🔗 负载均衡专属集群支持TOPN统计

TOPN: 您可在选择时间范围、统计方式、监控项后，开启显示TOP功能并点击最左侧的刷新按钮，此时系统根据您的选择，自动展示流量占比排名前10的BLB实例ID。

统计方式：平均值、最大值、最小值、和值。

- 四层负载均衡监控项：健康检查（个）、网络流量（字节）、网络数据包（个数）、并发连接数（个数）、丢弃流量（字节/分钟）、丢弃数据包（个/分钟）、丢弃连接数（个/分钟）、新建连接数（个/分钟）、网络输入带宽（bps）、网络输出带宽（bps）、限速流量（字节）、限速数据包（个数）
- 七层负载均衡监控项：健康检查（个）、请求数（个数/分钟）、网络流量（字节）、网络数据包（个数）、并发连接数（个数）、丢弃流量（字节/分钟）、丢弃数据包（个/分钟）、丢弃连接数（个/分钟）、新建连接数（个/分钟）、响应时间（ms）、响应状态码（个/分钟）、响应状态码比例（百分比）、后台服务器响应状态码（个/分钟）、后端服务器响应时间（ms）、网络输入带宽（bps）、网络输出带宽（bps）、每秒查询数（个数）、限速流量（字节）、限速数据包（个数）、QPS使用率（百分比）

登录百度智能云【管理控制台】，选择【产品服务】--【网络】--【负载均衡 BLB】，在左侧导航栏中选择BLB专属集群，点击集群名称，左侧导航栏选择监控，并查看“BLB实例监控信息”。

说明:

- 目前TOPN功能处于公测阶段，若需使用请提[工单](#)申请。
- BLB专属集群实例状态为“可用”；
- 必须有网络流量经过BLB专属集群；
- 点刷新按钮后，流量图有延时，需等待大概3分钟左右后显示。
- 3天之内必须有数据，最大可以查询时间范围为30天。
- 系统支持以下三种时间选择方式：1、查询时间范围小于1小时（1-59分钟），例如：20:00发起一个查询任务，只能查18:56-19:55数据。2、查询范围大于1小时，需要按1h整数倍数查，查询范围为1-12h。例如：20:00发起一个查询任务，查时间范围为1-12小时，只能查19:00前的数据。3、查询范围大于1小时，且非整数倍的，可以结合查询时间范围小于1小时（1-59分钟）来解决查询1小时内的数据。

服务发布点

服务发布点支持用户使用BLB构建的服务在百度云内网开放给指定用户访问。用户利用服务发布点，无需在公网暴露IP地址和端口，所有访问流量通过百度云内网安全访问，服务使用者可以通过[服务网卡](#)挂载服务发布点生成的服务域名，实现在百度云内网安全访问服务。

配置前您需要了解：

- 每个地域可以创建10个服务发布点；
- 一个BLB可以被多个服务发布点绑定；
- 一个服务发布点可挂载1000个服务网卡；
- 服务发布点权限条目个数上限50。

🔗 创建服务发布点

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB>服务发布点”，进入服务发布点实例页面。
2. 点击“创建服务发布点”，出现创建服务发布点弹框。

3. 填写下列配置信息：

配置项	说明
发布点名称	必填，用户自定义服务发布点名称
服务名称	必填，用于构建域名，必须为英文大小写、数字，一旦创建发布不可更改
关联BLB	必选，选择关联的BLB实例
发布权限	必填，默认所有用户允许，可自定义权限
描述	编辑服务发布点相关描述信息

说明：

- 服务发布点具备地域属性，发布点仅能绑定与自身所属同一个地域的BLB。
- 处于"服务中"状态的服务发布点不可释放。若用户需要释放发布点，需操作发布点解绑BLB，发布点状态变为"不可用"后触发释放；或者需将所有已关联的服务网卡与其解关联，发布点状态变为"可用"后触发释放。

4. 点击“确定”，服务发布点创建完成。

🔗 管理发布权限

在服务发布点实例右侧操作中，点击“管理发布权限“，进入发布权限页面。

1. 点击“添加权限”，出现添加权限弹框，可以自定义授权。
2. 点击修改按钮，可以修改该权限条目。
3. 点击删除，可以删除该权限条目。

说明：

- 支持自定义权限，可对指定或所有用户设置允许或拒绝权限。若所有用户权限与单用户权限冲突，则拒绝优先级高于允许。
- 单用户若重复添加权限策略，将覆盖之前的权限策略，以用户最新添加的为准。
- 本用户默认允许关联自身用户服务发布点。如果没有配置任何权限，则默认仅允许本用户服务网卡关联自身用户服务发布点。
- 若取消或者拒绝(除本用户外)其他用户的授权，将自动解除该用户关联的服务网卡。
- 服务发布点所有者不受到服务发布点上权限配置的影响，允许本用户服务网卡关联自身用户服务发布点。即使配置针对本用户的拒绝权限，该权限规则并不生效。

🔗 查看关联服务网卡

在服务发布点实例右侧操作中，点击“查看关联服务网卡“，进入关联记录页面，可以查看服务中的服务网卡，以及关联的历史记录。

说明：

- 若某用户被取消授权，则该用户下的服务网卡将自动解除关联，并在关联历史中记录。

标签管理

概述

百度智能云提供标签管理功能，通过给每个云资源添加标签，从而快速分类和识别管理这些资源。

使用标签前，您需要了解：

- 标签：每个标签由键和值两部分组成，标签（键+值）唯一；
- 单个和批量：支持为单个资源设置标签，也可以批量为云资源创建标签；
- 排序和搜索：支持按照标签（键和值）进行资源搜索和排序；
- 跨产品标记：不同产品可以添加同一个标签。

限制

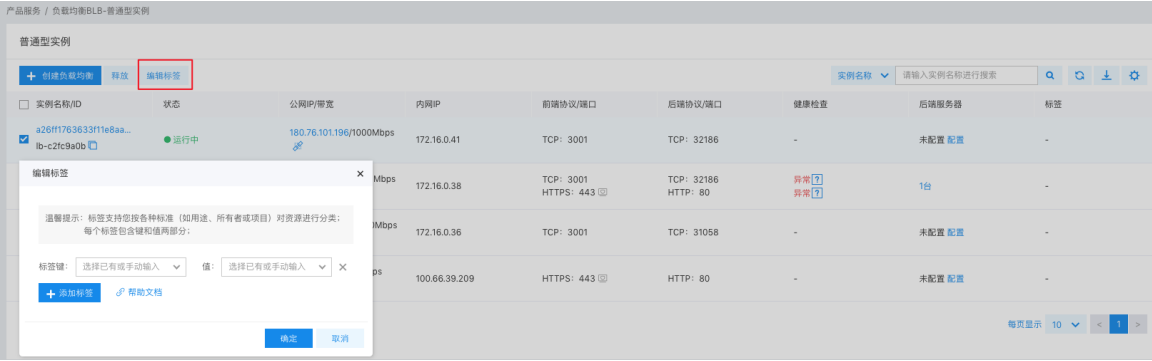
每个用户最多可以创建200个标签。

添加标签

拥有多个BLB实例的用户可以根据项目和场景需求，为BLB实例添加标签，便于对实例进行分类和识别管理。

操作步骤：

1. 登录管理控制台，进入“产品服务 > 网络 > 负载均衡 BLB”。
2. 点击左侧导航栏“普通型实例”，进入实例列表页。
3. 勾选一个或多个实例，在操作栏点击“编辑标签”。
4. 在弹出的对话框中，输入自定义标签键和标签值，注意“键”必须唯一，值可以为空，不填写。
5. 点击“确定”，完成标签创建。

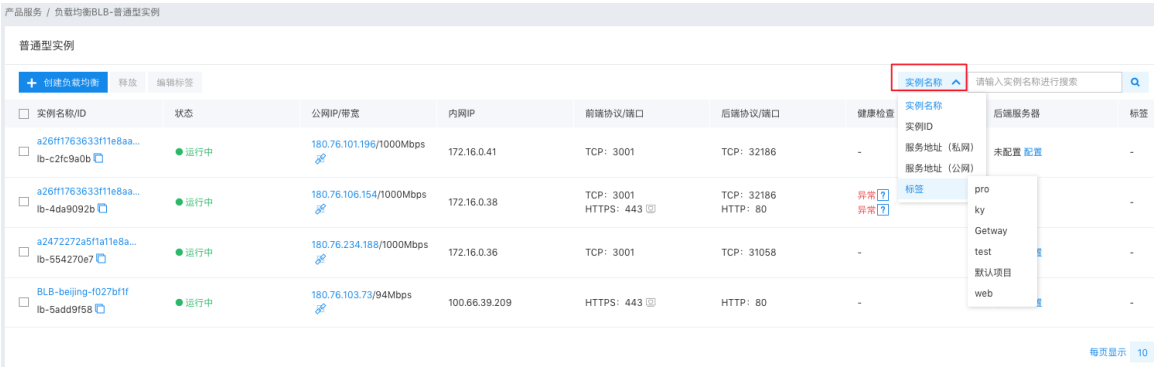


说明：批量创建实例标签时，无法查看已添加的单个实例标签。

标签搜索

标签搜索

通过组合筛选条件执行标签搜索，从而获得精确的BLB，您可以按标签键和值进行筛选：



删除标签

如果已创建的标签不再使用，您可以将标签删除。

- 1. 在左侧导航栏点击"标签管理"，进入标签列表页，勾选一个标签名称，操作栏点击"删除"。
- 2. 弹出对话框中，点击确定即可删除该标签，标签删除后无法恢复，系统自动解绑之前的资源。

资源账单

资源账单是记录每个资源的生命周期与消费相关的数据的功能，方便用户以资源为维度查看资源及产品的消费数据。

操作步骤：

- 1. 登录管理控制台，进入“产品服务 > 网络 > 负载均衡 BLB”。
- 2. 点击左侧导航栏"标签管理"，进入标签列表页。
- 3. 勾选一个标签，操作栏点击“查看账单”。
- 4. 进入资源账单页面，选择需要查看账单的月份，会显示当前标签资源的消费信息。还可以通过组合筛选条件执行标签搜索，按标签键和值进行选择其他标签的消费详情，并下载相关信息。

说明：还支持以实例ID进行检索，可以查看实例的消费信息。



- 5. 点击查看详情，可以查看实例ID、产品名称、账单金额。



说明：后付费是一直持续记账，消费信息会从创建月显示到使用时间段内显示。

多用户访问控制

介绍

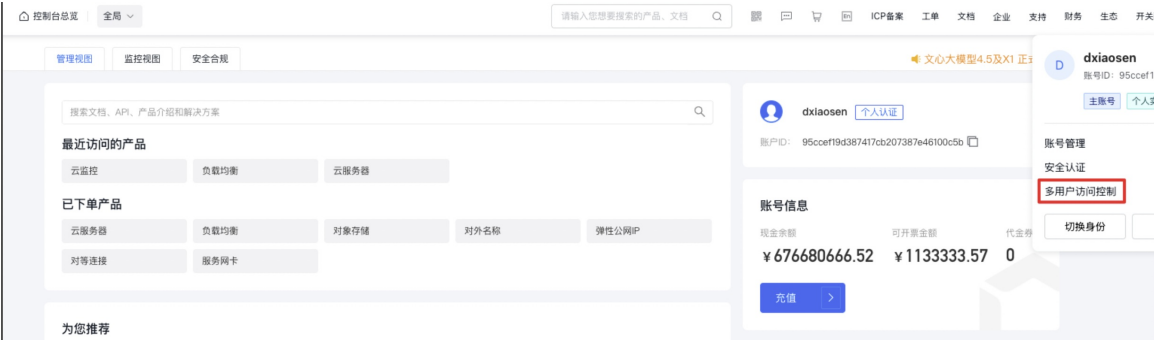
多用户访问控制，主要用于帮助用户管理云账户下资源的访问权限，适用于企业内的不同角色，可以对不同的工作人员赋予使用产品的不同权限，当您的企业存在多用户协同操作资源时，推荐您使用多用户访问控制。

适用于下列使用场景：

- 中大型企业客户：对公司内多个员工授权管理；
- 偏技术型vendor或SAAS的平台商：对代理客户进行资源和权限管理；
- 中小开发者或小企业：添加项目成员或协作者，进行资源管理。

创建用户

1. 主账号用户登录后在控制台选择“多用户访问控制”进入用户管理页面。



2. 在左侧导航栏点击“用户管理”，在“子用户”页，点击“新建子用户”。



3. 在弹出的“创建子用户”对话框中，完成填写“用户名”并点击确认，返回“子用户管理列表”区可以查看到刚刚创建的子用户。

创建子用户X

* 用户名：

长度1-64位的字母、数字、-、.或_

0 / 64

备注：

0 / 64

访问方式：

☐ 编程访问 自动生成AccessKey，子用户通过API或SDK工具访问

☐ 控制台密码访问 子用户使用账号密码登录云控制台

快速授权：

☐ 系统管理员

(其他权限可创建后添加)

取消

确定

配置策略

BLB和LBDC支持系统策略和用户自定义策略两种，分别实现BLB和LBDC的产品级权限和实例级权限控制。

- 系统策略：百度智能云系统为管理资源而预定义的权限集，这类策略可直接为子用户授权，用户只能使用而不能修改；
- 自定义策略：由用户自己创建，更细化的管理资源的权限集，可以针对单个实例配置权限，更加灵活的满足账户对不同用户的差异化权限管理。

系统策略

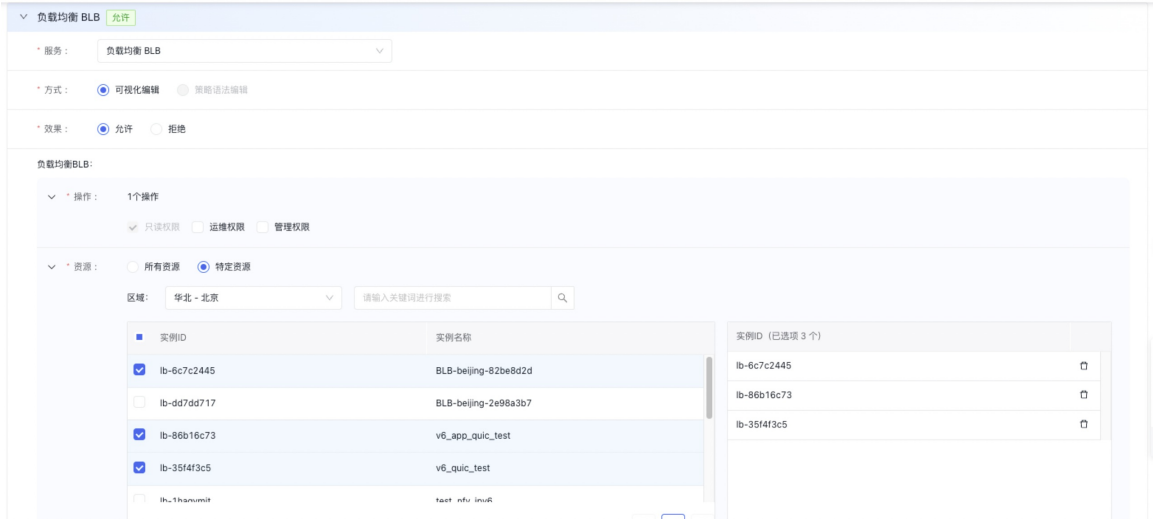
系统策略包含管理权限、运维权限和只读权限3种策略，权限范围详细如下：

策略名称	权限说明	权限范围
BLBFullControlAccessPolicy	完全控制管理负载均衡实例(BLB)的权限	查看实例列表、查看实例详情、修改BLB名称及描述、监听器配置、后端服务器配置、实例监控、实例报警、创建BLB实例、释放BLB实例
BIBOperateAccessPolicy	运维操作负载均衡（BLB）的权限	查看实例列表、查看实例详情、修改BLB名称及描述、监听器配置、后端服务器配置、实例监控、实例报警
BIBReadOnlyAccessPolicy	只读访问负载均衡（BLB）的权限	查看实例列表、查看实例详情
LBDCFullControlAccessPolicy	完全控制管理负载均衡专属集群（LBDC）的权限	查询专属集群列表、查看专属集群详情、查看关联的BLB实例、专属集群监控、专属集群报警、创建专属集群、释放专属集群、升级专属集群
LBDCOperateAccessPolicy	运维操作负载均衡专属集群（LBDC）的权限	查询专属集群列表、查看专属集群详情、查看关联的BLB实例、专属集群监控、专属集群报警
LBDCReadOnlyAccessPolicy	只读访问负载均衡专属集群（LBDC）的权限	查询专属集群列表、查看专属集群详情、查看关联的BLB实例

自定义策略

自定义策略是从实例维度进行授权，与系统策略不同，只对选定的实例生效。

子用户先在左侧导航栏点击"策略管理"，然后点击"创建策略"并点击"按策略生成器创建"，用户填写"策略名称"并选择服务类型为负载均衡BLB，其中策略生成方式默认为策略生成器，不需要修改。



子用户

先通过左侧导航栏进入【策略管理】，然后点击“创建策略”，用户填写策略名称并选择服务类型为负载均衡专属集群LBDC，其中策略生成方式默认为策略生成器，不需要修改。

权限配置

选择服务：

负载均衡专属集群 LBDC

配置方式：

策略生成器 编辑策略文件

权限效力：

☒ 允许 ☐ 拒绝

负载均衡LBDC：

☐ 只读权限 ☐ 运维权限 ☒ 管理权限 [权限说明](#)

选择资源：

区域：

华北 - 北京

ccq

已选择 1 个资源

清空

实例id	实例名称	实例id	
☒ bgw_group-8b0cca94	lbdc_4layer_ccq	bgw_group-8b0cca94	删除
☐ bgw_group-4f8c7c89	ccqTestJersey		
☐ nginx_group-99140754	ccqNginx		
☐ bgw_group-f303caeb	ccq0826layer4		

限制条件：

+ 添加条件

确认

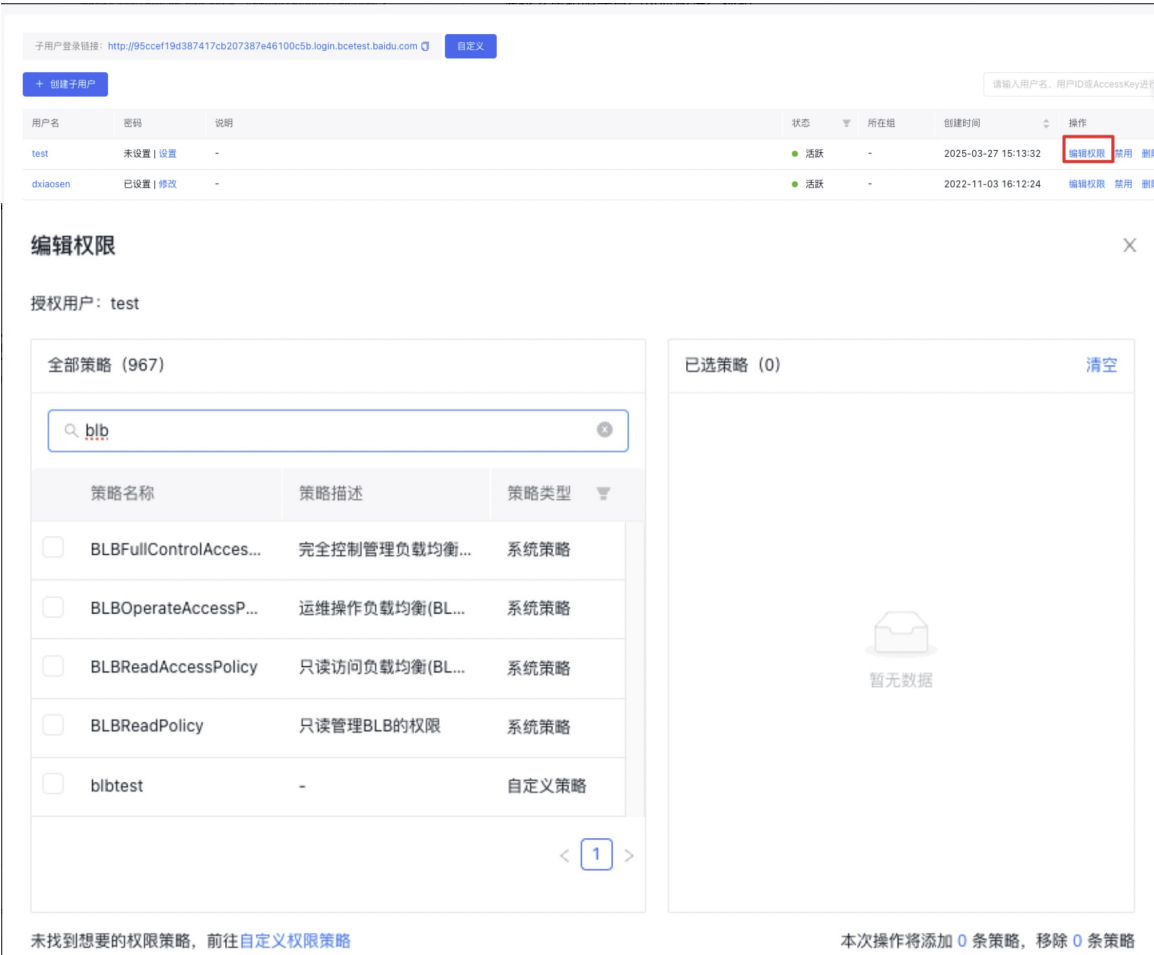
取消

自定义权限范围详细如下：

产品	只读权限	运维权限	管理权限
BLB	查看实例列表、查看实例详情	查看实例列表、查看实例详情、修改实例名称及描述、监听器配置、后端服务器配置、实例监控、实例报警	查看实例列表、查看实例详情、修改实例名称及描述、监听器配置、后端服务器配置、实例监控、实例报警、释放BLB
LBDC	查询专属集群列表、查看专属集群详情、查看关联的BLB实例	查询专属集群列表、查看专属集群详情、查看关联的BLB实例、专属集群监控、专属集群报警	查询专属集群列表、查看专属集群详情、查看关联的BLB实例、专属集群监控、专属集群报警、创建专属集群、释放专属集群、升级专属集群

🔗 用户授权

在“用户管理->子用户”的对应子用户的“操作”列选择“添加权限”，并为用户选择系统权限或自定义策略进行授权



说明：如果在不修改已有策略规则的情况下修改某子用户的权限，只能通过删除已有的策略并添加新的策略来实现，不能取消勾选已经添加过的策略权限。

子用户登录

主账号完成对子用户的授权后，可以将链接发送给子用户；子用户可以通过IAM用户登录链接登录主账号的管理控制台，根据被授权的策略对主账号资源进行操作和查看。



其他详细操作参考：[多用户访问控制](#)。

BLB监控项说明

BLB监控项说明

注意：负载均衡监控项不会包含经由服务网卡进入的流量，这部分流量不会影响负载均衡实例限速

🔗 普通型BLB

🔗 普通型BLB 端口监控

Metric name	英文名称	中文名称	采集维度	单位	含义	采集频率
NetworkInBytes	NetworkInBytes	网络输入流量	BlbId、 BlbPortType、 BlbPort	Bytes	网络输入流量	60s
NetworkOutBytes	NetworkOutBytes	网络输出流量	BlbId、 BlbPortType、 BlbPort	Bytes	网络输出流量	60s
ActiveConnCount	ActiveConnCount	活跃连接数	BlbId、 BlbPortType、 BlbPort	个	活跃连接数	60s
NetworkInPkgCount	NetworkInPkgCount	输入数据包数	BlbId、 BlbPortType、 BlbPort	个	输入数据包数	60s
NetworkOutPkgCount	NetworkOutPkgCount	输出数据包数	BlbId、 BlbPortType、 BlbPort	个	输出数据包数	60s
DropInPkgCount	DropInPkgCount	丢弃入向数据包数	BlbId、 BlbPortType、 BlbPort	个/min	丢弃入向数据包数	60s
DropOutPkgCount	DropOutPkgCount	丢弃出向数据包数	BlbId、 BlbPortType、 BlbPort	个/min	丢弃出向数据包数	60s
DropInBytes	DropInBytes	丢弃入向数据量	BlbId、 BlbPortType、 BlbPort	Bytes/ min	丢弃入向数据量	60s
DropOutBytes	DropOutBytes	丢弃出向数据量	BlbId、 BlbPortType、 BlbPort	Bytes/ min	丢弃出向数据量	60s
DropConnectCount	DropConnectCount	丢弃连接数	BlbId、 BlbPortType、 BlbPort	个/min	丢弃连接数	60s
NewConnectCount	NewConnectCount	新建连接数	BlbId、 BlbPortType、 BlbPort	个/min	新建连接数	60s
AverageUpsResponseTime	AverageUpsResponseTime	平均后端服务器响应时间	BlbId、 BlbPortType、 BlbPort	ms	平均后端服务器响应时间	60s
AverageResponseTime	AverageResponseTime	平均响应时间	BlbId、 BlbPortType、 BlbPort	ms	平均响应时间	60s
Response2xxNum	Response2xxNum	返回2xx数量	BlbId、 BlbPortType、 BlbPort	个/min	返回2xx数量	60s

			BlbPort			
Response3xxNum	Response3xxNum	返回3xx数量	BlbId、 BlbPortType、 BlbPort	个/min	返回3xx数量	60s
Response4xxNum	Response4xxNum	返回4xx数量	BlbId、 BlbPortType、 BlbPort	个/min	返回4xx数量	60s
Response5xxNum	Response5xxNum	返回5xx数量	BlbId、 BlbPortType、 BlbPort	个/min	返回5xx数量	60s
ResponseOtherNum	ResponseOtherNum	返回其它状态码数量	BlbId、 BlbPortType、 BlbPort	个/min	返回其它状态码数量	60s
Response2xxPercent	Response2xxPercent	返回2xx占比	BlbId、 BlbPortType、 BlbPort	%	返回2xx占比	60s
Response3xxPercent	Response3xxPercent	返回3xx占比	BlbId、 BlbPortType、 BlbPort	%	返回3xx占比	60s
Response4xxPercent	Response4xxPercent	返回4xx占比	BlbId、 BlbPortType、 BlbPort	%	返回4xx占比	60s
Response5xxPercent	Response5xxPercent	返回5xx占比	BlbId、 BlbPortType、 BlbPort	%	返回5xx占比	60s
ResponseOtherPercent	ResponseOtherPercent	返回其它状态码占比	BlbId、 BlbPortType、 BlbPort	%	返回其它状态码占比	60s
ResponseUps4xxNum	ResponseUps4xxNum	后端服务器返回4xx状态码数量	BlbId、 BlbPortType、 BlbPort	个/min	后端服务器返回4xx状态码数量	60s
ResponseUps5xxNum	ResponseUps5xxNum	后端服务器返回5xx状态码数量	BlbId、 BlbPortType、 BlbPort	个/min	后端服务器返回5xx状态码数量	60s
UnhealthyCount	UnhealthyCount	异常实例数量	BlbId、 BlbPortType、 BlbPort	个	异常实例数量	60s
TotalCount	TotalCount	实例总数	BlbId、 BlbPortType、 BlbPort	个	实例总数	60s
HealthyCount	HealthyCount	正常实例数量	BlbId、 BlbPortType、 BlbPort	个	正常实例数量	60s
RequestCount	RequestCount	请求数	BlbId、 BlbPortType、 BlbPort	个	请求数	60s

WebInBytes	WebInBytes	主IP从外网进入的流量	BlbId、BlbPortType、BlbPort	Bytes	主IP从外网进入的流量	60s
WebOutBytes	WebOutBytes	主IP流向外网的流量	BlbId、BlbPortType、BlbPort	Bytes	主IP流向外网的流量	60s
WebInBitsPerSecond	WebInBitsPerSecond	主IP从外网进入的带宽	BlbId、BlbPortType、BlbPort	bps	主IP从外网进入的带宽	60s
WebOutBitsPerSecond	WebOutBitsPerSecond	主IP流向外网的带宽	BlbId、BlbPortType、BlbPort	bps	主IP流向外网的带宽	60s

普通型BLB 公网IP监控

Metric name	英文名称	中文名称	采集维度	单位	含义	采集频率
WebInBytes	WebInBytes	从外网进入的流量	InstanceId	Bytes	从外网进入的流量	60s
WebOutBytes	WebOutBytes	流向外网的流量	InstanceId	Bytes	流向外网的流量	60s
WebOutBitsPerSecond	WebOutBitsPerSecond	出口带宽	InstanceId	bps	出口带宽	60s
WebInBitsPerSecond	WebInBitsPerSecond	入口带宽	InstanceId	bps	入口带宽	60s

应用型BLB

应用型BLB 端口监控

Metric name	英文名称	中文名称	采集维度	单位	含义	采集频率
NetworkInBytes	NetworkInBytes	网络输入流量	BlbId、BlbPortType、BlbPort	Bytes	网络输入流量	60s
NetworkOutBytes	NetworkOutBytes	网络输出流量	BlbId、BlbPortType、BlbPort	Bytes	网络输出流量	60s
ActiveConnCount	ActiveConnCount	活跃连接数	BlbId、BlbPortType、BlbPort	个	活跃连接数	60s
NetworkInPkgCount	NetworkInPkgCount	输入数据包数	BlbId、BlbPortType、BlbPort	个	输入数据包数	60s
NetworkOutPkgCount	NetworkOutPkgCount	输出数据包数	BlbId、BlbPortType、BlbPort	个	输出数据包数	60s
DropInPkgCount	DropInPkgCount	丢弃入向数据包数	BlbId、BlbPortType、BlbPort	个/min	丢弃入向数据包数	60s
DropOutPkgCount	DropOutPkgCount	丢弃出向数据包数	BlbId、BlbPortType、BlbPort	个/min	丢弃出向数据包数	60s

DropInBytes	DropInBytes	丢弃入向数据量	BlbId、 BlbPortType、 BlbPort	Bytes/ min	丢弃入向数据量	60s
DropOutBytes	DropOutBytes	丢弃出向数据量	BlbId、 BlbPortType、 BlbPort	Bytes/ min	丢弃出向数据量	60s
DropConnectCount	DropConnectCount	丢弃连接数	BlbId、 BlbPortType、 BlbPort	个/min	丢弃连接数	60s
NewConnectCount	NewConnectCount	新建连接数	BlbId、 BlbPortType、 BlbPort	个/min	新建连接数	60s
AverageUpsResponseTime	AverageUpsResponseTime	平均后端服务器响应时间	BlbId、 BlbPortType、 BlbPort	ms	平均后端服务器响应时间	60s
AverageResponseTime	AverageResponseTime	平均响应时间	BlbId、 BlbPortType、 BlbPort	ms	平均响应时间	60s
Response2xxNum	Response2xxNum	返回2xx数量	BlbId、 BlbPortType、 BlbPort	个/min	返回2xx数量	60s
Response3xxNum	Response3xxNum	返回3xx数量	BlbId、 BlbPortType、 BlbPort	个/min	返回3xx数量	60s
Response4xxNum	Response4xxNum	返回4xx数量	BlbId、 BlbPortType、 BlbPort	个/min	返回4xx数量	60s
Response5xxNum	Response5xxNum	返回5xx数量	BlbId、 BlbPortType、 BlbPort	个/min	返回5xx数量	60s
ResponseOtherNum	ResponseOtherNum	返回其它状态码数量	BlbId、 BlbPortType、 BlbPort	个/min	返回其它状态码数量	60s
Response2xxPercent	Response2xxPercent	返回2xx占比	BlbId、 BlbPortType、 BlbPort	%	返回2xx占比	60s
Response3xxPercent	Response3xxPercent	返回3xx占比	BlbId、 BlbPortType、 BlbPort	%	返回3xx占比	60s
Response4xxPercent	Response4xxPercent	返回4xx占比	BlbId、 BlbPortType、 BlbPort	%	返回4xx占比	60s
Response5xxPercent	Response5xxPercent	返回5xx占比	BlbId、 BlbPortType、 BlbPort	%	返回5xx占比	60s
ResponseOtherPercent	ResponseOtherPercent	返回其它状态码占比	BlbId、 BlbPortType、 BlbPort	%	返回其它状态码占比	60s

cent	cent	后端服务器返回4xx状态码数量	BlbId、BlbPortType、BlbPort	个/min	后端服务器返回4xx状态码数量	60s
ResponseUps4xxNum	ResponseUps4xxNum	后端服务器返回5xx状态码数量	BlbId、BlbPortType、BlbPort	个/min	后端服务器返回5xx状态码数量	60s
ResponseUps5xxNum	ResponseUps5xxNum	异常实例数量	BlbId、BlbPortType、BlbPort	个	异常实例数量	60s
UnhealthyCount	UnhealthyCount	实例总数	BlbId、BlbPortType、BlbPort	个	实例总数	60s
TotalCount	TotalCount	正常实例数量	BlbId、BlbPortType、BlbPort	个	正常实例数量	60s
HealthyCount	HealthyCount	请求数	BlbId、BlbPortType、BlbPort	个	请求数	60s
RequestCount	RequestCount	主IP从外网进入的流量	BlbId、BlbPortType、BlbPort	Bytes	主IP从外网进入的流量	60s
WebInBytes	WebInBytes	主IP流向外网的流量	BlbId、BlbPortType、BlbPort	Bytes	主IP流向外网的流量	60s
WebOutBytes	WebOutBytes	主IP从外网进入的带宽	BlbId、BlbPortType、BlbPort	bps	主IP从外网进入的带宽	60s
WebInBitsPerSecond	WebInBitsPerSecond	主IP流向外网的带宽	BlbId、BlbPortType、BlbPort	bps	主IP流向外网的带宽	60s
WebOutBitsPerSecond	WebOutBitsPerSecond					

应用型BLB 服务器组监控

Metric name	英文名称	中文名称	采集维度	单位	含义	采集频率
UnhealthyCount	UnhealthyCount	异常实例数量	BlbId、sgId、sgPort	个	异常实例数量	60s
TotalCount	TotalCount	实例总数	BlbId、sgId、sgPort	个	实例总数	60s
HealthyCount	HealthyCount	正常实例数量	BlbId、sgId、sgPort	个	正常实例数量	60s

应用型BLB IP组监控

Metric name	英文名称	中文名称	采集维度	单位	含义	采集频率
UnhealthyCount	UnhealthyCount	异常实例数量	BlbId、sgId、sgPort	个	异常实例数量	60s
TotalCount	TotalCount	实例总数	BlbId、sgId、sgPort	个	实例总数	60s
HealthyCount	HealthyCount	正常实例数量	BlbId、sgId、sgPort	个	正常实例数量	60s

应用型BLB 公网IP监控

Metric name	英文名称	中文名称	采集维度	单位	含义	采集频率
WebInBytes	WebInBytes	从外网进入的流量	InstanceId	Bytes	从外网进入的流量	60s
WebOutBytes	WebOutBytes	流向外网的流量	InstanceId	Bytes	流向外网的流量	60s
WebOutBitsPerSecond	WebOutBitsPerSecond	出口带宽	InstanceId	bps	出口带宽	60s
WebInBitsPerSecond	WebInBitsPerSecond	入口带宽	InstanceId	bps	入口带宽	60s

普通型IPv6 BLB

普通型IPv6 BLB 端口监控

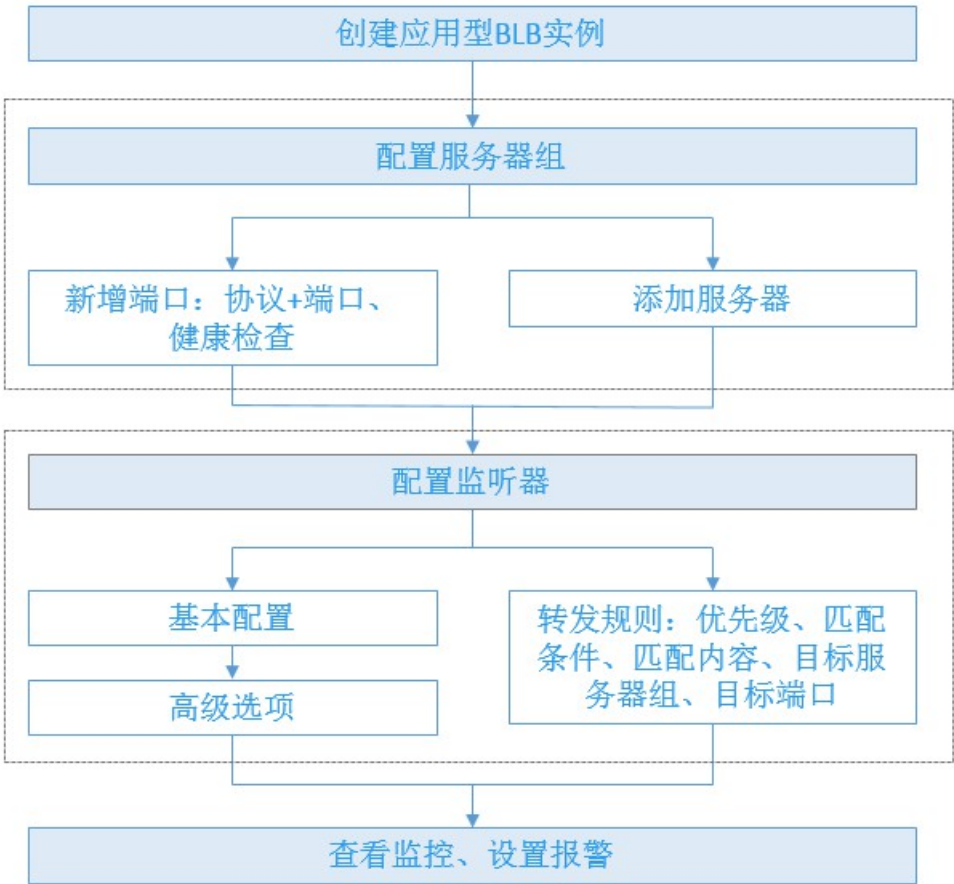
Metric name	英文名称	中文名称	采集维度	单位	含义	采集频率
NetworkInBytes	NetworkInBytes	网络输入流量	BlbId、 BlbPortType、 BlbPort	Bytes	网络输入流量	60s
NetworkOutBytes	NetworkOutBytes	网络输出流量	BlbId、 BlbPortType、 BlbPort	Bytes	网络输出流量	60s
ActiveConnCount	ActiveConnCount	活跃连接数	BlbId、 BlbPortType、 BlbPort	个	活跃连接数	60s
NetworkInPkgCount	NetworkInPkgCount	输入数据包数	BlbId、 BlbPortType、 BlbPort	个	输入数据包数	60s
NetworkOutPkgCount	NetworkOutPkgCount	输出数据包数	BlbId、 BlbPortType、 BlbPort	个	输出数据包数	60s
DropInPkgCount	DropInPkgCount	丢弃入向数据包数	BlbId、 BlbPortType、 BlbPort	个/min	丢弃入向数据包数	60s
DropOutPkgCount	DropOutPkgCount	丢弃出向数据包数	BlbId、 BlbPortType、 BlbPort	个/min	丢弃出向数据包数	60s
DropInBytes	DropInBytes	丢弃入向数据量	BlbId、 BlbPortType、 BlbPort	Bytes/ min	丢弃入向数据量	60s
DropOutBytes	DropOutBytes	丢弃出向数据量	BlbId、 BlbPortType、 BlbPort	Bytes/ min	丢弃出向数据量	60s
DropConnectCount	DropConnectCount	丢弃连接数	BlbId、 BlbPortType、 BlbPort	个/min	丢弃连接数	60s
NewConnectCount	NewConnectCount	新建连接数	BlbId、 BlbPortType、 BlbPort	个/min	新建连接数	60s
AverageUpsResponseTime	AverageUpsResponseTime	平均后端服务器响应时间	BlbId、 BlbPortType、 BlbPort	ms	平均后端服务器响应时间	60s

名称	名称	说明	单位	单位	单位	单位
AverageResponseTime	AverageResponseTime	平均响应时间	BlbId、BlbPortType、BlbPort	ms	平均响应时间	60s
Response2xxNum	Response2xxNum	返回2xx数量	BlbId、BlbPortType、BlbPort	个/min	返回2xx数量	60s
Response3xxNum	Response3xxNum	返回3xx数量	BlbId、BlbPortType、BlbPort	个/min	返回3xx数量	60s
Response4xxNum	Response4xxNum	返回4xx数量	BlbId、BlbPortType、BlbPort	个/min	返回4xx数量	60s
Response5xxNum	Response5xxNum	返回5xx数量	BlbId、BlbPortType、BlbPort	个/min	返回5xx数量	60s
ResponseOtherNum	ResponseOtherNum	返回其它状态码数量	BlbId、BlbPortType、BlbPort	个/min	返回其它状态码数量	60s
Response2xxPercent	Response2xxPercent	返回2xx占比	BlbId、BlbPortType、BlbPort	%	返回2xx占比	60s
Response3xxPercent	Response3xxPercent	返回3xx占比	BlbId、BlbPortType、BlbPort	%	返回3xx占比	60s
Response4xxPercent	Response4xxPercent	返回4xx占比	BlbId、BlbPortType、BlbPort	%	返回4xx占比	60s
Response5xxPercent	Response5xxPercent	返回5xx占比	BlbId、BlbPortType、BlbPort	%	返回5xx占比	60s
ResponseOtherPercent	ResponseOtherPercent	返回其它状态码占比	BlbId、BlbPortType、BlbPort	%	返回其它状态码占比	60s
ResponseUps4xxNum	ResponseUps4xxNum	后端服务器返回4xx状态码数量	BlbId、BlbPortType、BlbPort	个/min	后端服务器返回4xx状态码数量	60s
ResponseUps5xxNum	ResponseUps5xxNum	后端服务器返回5xx状态码数量	BlbId、BlbPortType、BlbPort	个/min	后端服务器返回5xx状态码数量	60s
UnhealthyCount	UnhealthyCount	异常实例数量	BlbId、BlbPortType、BlbPort	个	异常实例数量	60s
TotalCount	TotalCount	实例总数	BlbId、BlbPortType、BlbPort	个	实例总数	60s

HealthyCount	HealthyCount	正常实例数量	BlbId、BlbPortType、BlbPort	个	正常实例数量	60s
RequestCount	RequestCount	请求数	BlbId、BlbPortType、BlbPort	个	请求数	60s
WebInBytes	WebInBytes	主IP从外网进入的流量	BlbId、BlbPortType、BlbPort	Bytes	主IP从外网进入的流量	60s
WebOutBytes	WebOutBytes	主IP流向外网的流量	BlbId、BlbPortType、BlbPort	Bytes	主IP流向外网的流量	60s
WebInBitsPerSecond	WebInBitsPerSecond	主IP从外网进入的带宽	BlbId、BlbPortType、BlbPort	bps	主IP从外网进入的带宽	60s
WebOutBitsPerSecond	WebOutBitsPerSecond	主IP流向外网的带宽	BlbId、BlbPortType、BlbPort	bps	主IP流向外网的带宽	60s

应用型IPv6实例

用户使用应用型IPv6 BLB实例的流程，与应用型BLB实例基本相同。应用型实例配置流程图如下：



应用型BLB配置流程图

分为以下四个步骤：

第一步：创建应用型IPv6BLB实例；

- 第二步：配置服务器组，主要包括端口配置、健康检查和添加服务器；
- 第三步：配置监听器；
- 第四步：查看监控、设置报警。

创建应用型IPv6BLB实例

1. 登录百度智能云平台，选择“产品服务>负载均衡BLB>应用型IPv6实例”，进入“实例列表页面”。
2. 点击“创建负载均衡”，进入创建实例界面。

说明：

创建BLB实例之前，应先创建BCC实例，否则无法成功创建BLB实例。

3. 选择配置信息。
- 基础信息：可选择区域和设置实例名称。所在网络，指当前创建的 BLB 和相应 BCC 等云资源所属的私有网络(VPC)。

说明： 用户在创建BLB实例时需要指定分配了IPv6的子网。

1 基本配置2 配置公网访问3 确认订单

付费及地域

付费方式：

包年包月-预付费
先付费后使用，价格更低廉

按量付费-后付费
先使用后付费，按需开通

当前地域：

华北 - 北京

华北 - 保定

华北 - 阳泉

西南 - 成都

华南 - 广州

华南 - 度小满金融专区

华东 - 苏州

华东 - 上海

华东 - 南京

金融华中 - 武汉

中国香港

基础信息

实例类型：

普通实例

应用实例

普通型IPv6 实例

应用型IPv6 实例

名称：

请输入实例名称0/255

实例性能：

性能共享

性能规格

创建的BLB实例在共享集群，多租户性能共享，在极端情况下出现资源争抢时，性能可能会受到限制。若需要超高性能和独占资源，建议您使用[负载均衡专属集群](#)。

*4层所在集群：

专属集群

共享集群

*7层所在集群：

专属集群

共享集群

所在网络：

online_test_vpc (240c:4084:1:4300::)

online_test (240c:4084:1:4300::/64)

当前可用区A，可用IP共251个；如需创建子网，您可以到[私有网络-子网](#)去创建

分配IPv6 IP：

自动分配

指定

安全组：

普通安全组

企业安全组

请选择

105

描述：

请输入

0/200

标签

绑定标签：

请选择或手动输入标签键

请选择或手动输入标签值

+ 添加标签

标签包含键和值两部分，支持按用途、所有者或项目等对资源进行分类，重复的标签会自动去重处理。[帮助文档](#)

资源分组

资源分组：

默认分组

如需创建新的资源分组，可前往资源管理 [去创建](#)。更多内容请查看 [帮助文档](#)

购买信息

购买数量：

-

1

+

① 最多可开通500个BLB实例，还可开通492个

自动释放：

关闭

开启

下一步

配置费用：

¥0.00036/分钟

(预计¥0.51 /天 ¥ 15.55 /月)

已选配置：

[查看详情](#)

4. 确认配置信息并根据系统提示，完成订单确认及支付。创建成功后，用户将收到短信提示。

🔗 配置服务器组

配置服务器组步骤主要包括：

- 1. 服务器组基本信息填写
- 2. 配置开放端口；
- 3. 添加服务器组；

注意：一个服务器组可以对应多个开放端口，例如：A服务器组，可以同时开放 TCP 80和HTTP 80两个端口。

🔗 服务器组基本信息填写

操作步骤

- 1. 选择“产品服务>负载均衡BLB>应用型IPv6实例”，进入应用型IPv6BLB实例列表。
- 2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
- 3. 选择左侧导航栏中的“目标组“，详情页面选择“服务器组”页签，点击”添加服务器组“，进入添加服务器组配置页面,完成服务器组名称和描述填写。

🔗 配置TCP协议开放端口

操作步骤

- 1.进入添加服务器组配置页面,点击"新增端口"进行监听器的基本设置和健康检查设置,开放协议[端口]选择TCP。
- 2.设置配置信息。

项目	描述
开放协议 [端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1~65535间的整数。默认“80”。

- 3.设置健康检查。

106

项目	描述
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。
健康检查间隔	对后端服务器进行检查的时间间隔。
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。

4.完成上述配置后，点击“确认”按钮完成TCP协议开放端口的配置。

配置UDP协议开放端口

操作步骤

- 1.进入添加服务器组配置页面,点击"新增端口"进行监听器的基本设置和健康检查设置,开放协议[端口]选择UDP。
- 2.设置配置信息。

项目	描述
开放协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1～65535间的整数。默认“80”。

3.设置健康检查。

项目	描述
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。
检查字符串	BLB采用给后端服务器发送带有用户指定字符串的UDP包实现健康检查。 您需要配置应用程序来完成响应。对于后端服务器是DNS server的情况，您可以配置DNS解析请求作为健康检查包，默认值为对baidu.com的DNS解析请求 （00\01\01\00\01\00\00\00\00\00\00\05baidu\03com\00\00\01\00\01）。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。 输入范围为1～60间的整数，建议设置为3秒。
健康检查间隔	对后端服务器进行检查的时间间隔。输入范围为1～10间的整数，建议设置为3秒
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。

4.完成上述配置后，点击“确认”按钮完成UDP协议开放端口的配置。

配置HTTP协议开放端口

- 1.进入添加服务器组配置页面,点击"新增端口"进行监听器的基本设置和健康检查设置,开放协议[端口]选择HTTP。
- 2.设置配置信息。

项目	描述
开放协议[端口]	指定后端服务器提供服务的协议和端口。BLB服务转发请求至目标服务，目标服务监听的端口，端口输入范围为1～65535间的整数。默认“80”。

3.设置健康检查信息。

项目	描述
健康检查协议	指定检查后端服务器健康状态时使用的协议类型。可选择“HTTP”或“TCP”
Host头域	发往后端服务器的健康检查请求的Host头域，默认为空。
检查端口	指定后端服务器提供服务的端口。本字段只对HTTP健康检查有效。
检查路径	用于健康检查页面文件的URI，建议对静态页面进行检查。 长度限制为1-80个字符，只能使用字母、数字、‘-’、‘/’、‘.’、‘%’、‘?’、‘#’、‘&’这些字符。本字段只对HTTP健康检查有效。
响应超时时间	如果在指定时间内没有收到后端服务器的响应，则认为后端服务器响应超时。
健康检查间隔	对后端服务器进行检查的时间间隔。
不健康阈值	如果后端服务器连续健康检查失败的次数超过阈值，则认为该服务器不健康。
健康阈值	对于不健康的服务器，如果连续健康检查成功的次数超过阈值，则认为该服务器健康。
正常码状态	健康检查正常的http状态码，可选“http_2xx”、“http_3xx”、“http_4xx”、“http_5xx”，本字段只对HTTP健康检查有效

4.完成上述配置后，点击“确认”按钮完成HTTP协议服务器组配置。

添加服务器组

在服务器组详情页面点击“添加服务器”，选择需要添加进入该服务器组的服务器名称，然后选择后点击“下一步”，进行服务器权重设置，最后点击“确认”，完成服务器组的添加步骤。

注意：

- 添加服务器过程中，所有的服务器都需要和应用型IPv6 BLB实例在同一个VPC下。需要注意的是，对于IPv6 实例，只能挂载分配了IPv6 IP的云服务器。
- 配置后端服务器权重。权重是用来针对不同的后端服务器实例设置不同的请求转发比例。例如有两台配置不同的后端服务器实例，那么对于高配置的实例可以选择设置更高的权重来接收更多的外部访问请求。
- 权重值是一个相对值，如果设置成相同数值，其转发比例是一样的。
- 支持设置0-100的权重值，权重越高，接收的请求越多。
- 权重对加权轮询外的转发规则不生效，如果权重设置为“0”，则不再接收新的请求。

服务器安全组设置

应用型IPv6 BLB实例的服务器安全组设置可以参考[普通型BLB实例的安全组设置](#)

编辑端口

在“开放端口”页签，点击<编辑>，跳入编辑界面，编辑后点击<确定>。

删除端口

1.

在“开放端口”页签，勾选要删除的监听设置，点击<删除>按钮。

-
2. 在弹出的界面，点击<确定>，完成释放。

完成服务器组基本信息填写、添加服务器和配置开放端口后，点击页面下侧“确认”，完成服务器组的创建步骤。

配置IP组

配置IP组步骤主要包括：

1. IP组基本信息填写
2. 配置开放协议；
3. 添加IP。

说明： IPv6类型IP组暂不支持跨Region经由专线、VPN互访，仅支持同Region。

IP组基本信息填写

操作步骤

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 选择左侧导航栏中的“目标组”，详情页面选择“IP组”页签，点击“添加IP组”，进入添加IP组配置页面,完成IP组名称和描述填写。

配置TCP开放协议

操作步骤

- 1.进入添加IP组配置页面,点击"新增协议"进行监听器的基本设置和健康检查设置,开放协议选择TCP。
- 2.设置配置信息。

项目	描述
开放协议	指定后端成员提供服务的协议。BLB服务转发请求至后端成员。

- 3.设置健康检查。

项目	描述
健康检查协议	指定检查后端成员健康状态时使用的协议类型。
响应超时时间	如果在指定时间内没有收到后端成员的响应，则认为后端成员响应超时。
健康检查间隔	对后端成员进行检查的时间间隔。
不健康阈值	如果后端成员连续健康检查失败的次数超过阈值，则认为该后端成员不健康。
健康阈值	对于不健康的后端成员，如果连续健康检查成功的次数超过阈值，则认为该后端成员健康。

- 4.完成上述配置后，点击“确认”按钮完成TCP开放协议的配置。

配置UDP开放协议

操作步骤

- 1.进入添加IP组配置页面,点击"新增协议"进行监听器的基本设置和健康检查设置,开放协议选择UDP。
- 2.设置配置信息。

项目	描述
开放协议	指定后端成员提供服务的协议和端口。BLB服务转发请求至后端成员。

3.设置健康检查。

项目	描述
健康检查协议	指定检查后端成员健康状态时使用的协议类型。
检查字符串	BLB采用给后端成员发送带有用户指定字符串的UDP包实现健康检查。 您需要配置应用程序来完成响应。对于后端服务器是DNS server的情况，您可以配置DNS解析请求作为健康检查包，默认值为对baidu.com的DNS解析请求（\00\01\01\00\00\01\00\00\00\00\00\05baidu\03com\00\00\01\00\01）。
响应超时时间	如果在指定时间内没有收到后端成员的响应，则认为后端成员响应超时。 输入范围为1～60间的整数，建议设置为3秒。
健康检查间隔	对后端成员进行检查的时间间隔。输入范围为1～10间的整数，建议设置为3秒
不健康阈值	如果后端成员连续健康检查失败的次数超过阈值，则认为该后端成员不健康。
健康阈值	对于不健康的后端成员，如果连续健康检查成功的次数超过阈值，则认为该后端成员健康。

4.完成上述配置后，点击“确认”按钮完成UDP开放协议的配置。

配置HTTP开放协议

- 1.进入添加IP组配置页面,点击"新增协议"进行监听器的基本设置和健康检查设置,开放协议选择HTTP。
- 2.设置配置信息。

项目	描述
开放协议[端口]	指定后端成员提供服务的协议。BLB服务转发请求至后端成员。

3.设置健康检查信息。

项目	描述
健康检查协议	指定检查后端成员健康状态时使用的协议类型。可选择“HTTP”。
Host头域	发往后端服务器的健康检查请求的Host头域，默认为空。
检查端口	指定后端成员提供服务的端口。本字段只对HTTP健康检查有效。
检查路径	用于健康检查页面文件的URI，建议对静态页面进行检查。 长度限制为1-80个字符，只能使用字母、数字、‘-’、‘/’、‘.’、‘%’、‘?’、‘#’、‘&’这些字符。本字段只对HTTP健康检查有效。
响应超时时间	如果在指定时间内没有收到后端成员的响应，则认为后端成员响应超时。
健康检查间隔	对后端成员进行检查的时间间隔。
不健康阈值	如果后端成员连续健康检查失败的次数超过阈值，则认为该后端成员不健康。
健康阈值	对于不健康的后端成员，如果连续健康检查成功的次数超过阈值，则认为该后端成员健康。
正常码状态	健康检查正常的http状态码，可选“http_2xx”、“http_3xx”、“http_4xx”、“http_5xx”，本字段只对HTTP健康检查有效

4.完成上述配置后，点击“确认”按钮完成HTTP开发协议的配置。

添加IP组

在服务器组详情页面点击“添加IP”，进入“添加IP”菜单栏，填写需要添加进入该IP组的IP地址+端口，并设置权重，最后点击“确认”，完成IP的添加。

- 说明：
- 添加IP过程中，IP可以跟应用型IPv6 BLB实例在同一个VPC下；IPv6类型IP组暂不支持跨Region经由专线、VPN互访，仅支持同地域。
 - 配置后端成员即IP+端口的权重。权重是用来针对不同的后端成员设置不同的请求转发比例。例如有两个配置不同的后端成员，那么对于高配置的成员可以选择设置更高的权重来接收更多的外部访问请求。
 - 权重值是一个相对值，如果设置成相同数值，其转发比例是一样的。
 - 支持设置0-100的权重值，权重越高，接收的请求越多。
 - 权重对加权轮询外的转发规则不生效，如果权重设置为“0”，则不再接收新的请求。

配置应用型BLB实例监听器

主要包括TCP、UDP、SSL、HTTP和HTTPS五种监听器的配置

TCP监听器

场景简介

TCP协议常见的应用场景有：注重可靠性、对数据准确性要求高和速度可以相对较慢的场景。

操作步骤

- 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
- 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
- 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“TCP”。
- 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1～65535间的整数。默认“80”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP算法为TCP监听器提供会话保持机制。此算法下，服务器权重不生效。

5. 高级选项

项目	描述
超时时间	指定TCP连接的超时时间，输入范围为10～4000间的整数，默认900秒。

6. 设置转发规则。

项目	描述
目标组	把匹配规则的流量转发至指定的服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：TCP监听器只有一条默认转发规则，且该规则不允许删除。例如：将该监听器接收到的流量转发到A服务器组的TCP 80端口。则在目标服务器组中，选择A服务器组，在目标端口中，选择TCP 80。

7. 完成上述配置后，点击“确认”按钮完成TCP监听器添加。

UDP监听器

场景简介

UDP协议常见的应用场景有：DNS服务、物联网及一些对数据传输可靠性要求不高的场景。

操作步骤

开启健康检查后，BLB将自动屏蔽不健康的后端成员。

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“UDP”。
4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“53”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP：将请求的源IP进行hash运算后派发请求至某匹配的服务器，这可以保证同一个客户端IP的请求始终被派发至某特定的服务器。源IP算法为UDP监听器提供会话保持机制。此算法下，服务器权重不生效。

5. 高级选项

项目	描述
超时时间	指定TCP连接的超时时间，输入范围为10~4000间的整数，默认900秒。

6. 设置转发规则。

项目	描述
目标组	把匹配规则的流量转发至指定服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：UDP监听器只有一条默认转发规则，且该规则不允许删除。例如：将该监听器接收到的流量转发到A服务器组的UDP 53端口。则在目标服务器组中，选择A服务器组，在目标端口中，选择UDP 53。

7. 完成上述配置后，点击“确认”按钮完成UDP监听器添加。

🔗 SSL监听器

场景简介

SSL协议常见应用场景有：需要超高性能和大规模TLS卸载的场景。

操作步骤

开启健康检查后，BLB将自动屏蔽不健康的后端成员。

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“SSL”。
4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1～65535间的整数。默认“443”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。 源IP：将请求的源IP进行hash运算后派发请求至某匹配的服务器，这可以保证同一个客户端IP的请求始终被派发至某特定的服务器。源IP算法为UDP监听器提供会话保持机制。此算法下，服务器权重不生效。
服务器证书	在配置SSL监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看 证书管理 。 说明：专属集群的BLB实例支持国际标准、国密（SM2）标准证书；共享集群的BLB实例只支持国际标准证书。
双向认证	选择是否开启双向认证。
加密选项	安全策略包含HTTPS可选的TLS协议版本和配套的加密算法套件。

5. 高级选项

项目	描述
超时时间	等待后端成员响应的超时时间；如果后端成员需较长时间处理请求，可适当调大该值，默认30秒。

6. 设置转发规则。

项目	描述
目标组	把匹配规则的流量转发至指定服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：SSL监听器只有一条默认转发规则，且该规则不允许删除。

7. 完成上述配置后，点击“确认”按钮完成SSL监听器添加。

🔗 HTTP监听器

场景简介

HTTP协议常见应用场景有：需要对数据内容进行识别的应用场景。

操作步骤

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“HTTP”。
4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“80”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。
超时时间	等待后端成员响应的超时时间；如果后端成员需较长时间处理请求，可适当调大该值，默认30秒。

5. 高级选项

项目	描述
会话保持	TCP监听器可以通过源IP转发规则实现会话保持；7层监听器（HTTP、HTTPS）通过插入/修改cookie实现会话保持。可选择“开启”或“关闭”。
附加HTTP头部	在HTTP请求中附加头字段，在后端成员处理报文时获取客户端真实IP或监听协议信息。 获取真实IP：开启该功能后，可将用户IP地址记录在X-Forwarded-For头域中； 获取监听协议：开启该功能后，可将监听协议记录在X-Forwarded-Proto头域中。
启用HTTPS重定向	将此监听器的HTTP请求重定向至此BLB的HTTPS监听器。请确保相应端口的HTTPS监听器工作正常。可选择“开启”或“关闭”。默认“关闭”。
HTTP请求允许非ASCII字符	默认关闭。开启此选项后如果HTTP请求中含有非ASCII字符，BLB会放行并转发给后端服务器处理。开启此选项可能带来安全风险，非特殊使用场景建议保持关闭。
数据压缩	开启将对特定文件类型进行压缩；关闭则不会对任何文件类型进行压缩，默认关闭
定制配置	BLB 7层监听支持定制配置，配置项和说明可参见 配置参数详情 。

6. 设置转发规则。

项目	描述
优先级	优先级数值越小，则优先级越高，优先级不允许重复
匹配条件	可以设置域名、URL和匹配全部的转发规则
目标组	把匹配规则的流量转发至指定服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：

- 单条规则同时出现域名和URL条件时，则需要同时满足条件，才算匹配成功

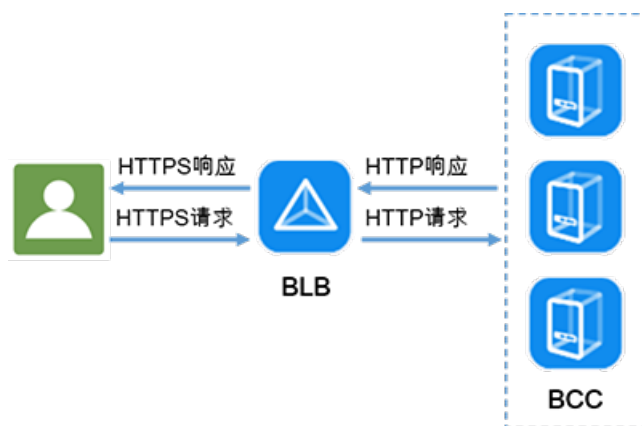
- 域名转发规则匹配内容限制：
 - 限制长度<100
 - 通配符仅支持*
 - 通配符*可出现0次或1次
 - 通配符*可匹配0-n个字符
 - 仅支持前缀匹配，即*只可出现在域名的开头，如：*.baidu.com
- URL转发规则匹配内容限制：
 - 限制长度<100
 - 通配符仅支持*
 - 通配符*可出现0次或1次
 - 通配符*可匹配0-n个字符
 - 支持前、中、后缀匹配，即*可出现在匹配规则的任意位置

例如：将该监听器接收到的流量转发到A服务器组的HTTP 80端口。则在目标服务器组中，选择A服务器组，在目标端口中，选择HTTP 80。

7. 完成上述配置后，点击“确认”按钮完成HTTP监听器添加。

🔗 HTTPS监听器

为了保证信息安全，用户可以配置BLB监听器的协议为HTTPS。BLB对于用户的HTTPS的请求进行解密，然后发送至后端成员；后端成员处理完请求后的返回包首先发送至BLB，由BLB进行加密后，再传回用户侧。整个过程如下图所示：



前置任务

在配置HTTPS监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看[证书管理](#)。

操作步骤

1. 选择“产品服务>负载均衡BLB>应用型实例”，进入应用型实例列表。
2. 点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
3. 点击左侧导航栏的“监听设置>添加监听设置”，进行监听器的基本设置。监听协议[端口]选择“HTTPS”。
4. 设置配置信息。

项目	描述
监听协议[端口]	指定BLB监听器监听的协议和端口。端口输入范围为1~65535间的整数。默认“443”。
调度方法	加权轮询：依据后端成员的权重，将请求轮流发送给后端成员，常用于短连接服务，例如HTTP服务。有关权重的配置方法，请参看“配置后端服务器”章节。 加权最小连接数：除了根据每台后端服务器设定的权重值来进行轮询，同时还考虑后端服务器的连接数。当权重值相同时，优先将请求发给拥有最少连接数的后端服务器，常用于长连接服务，例如数据库连接等服务。
服务器证书	在配置HTTPS监听器之前，应先申请服务器证书并将证书上传至证书管理模块，具体操作请参看 证书管理 。 说明： 专属集群的BLB实例支持国际标准、国密（SM2）标准证书；共享集群的BLB实例只支持国际标准证书。
扩展域名	选择是否开启扩展域名。
双向认证	选择是否开启双向认证。
加密选项	安全策略包含HTTPS可选的TLS协议版本和配套的加密算法套件。
超时时间	等待后端成员响应的超时时间；如果后端成员需较长时间处理请求，可适当调大该值，默认30秒。

5. 高级选项

项目	描述
会话保持	TCP监听器可以通过源IP转发规则实现会话保持；7层监听器（HTTP、HTTPS）通过插入/修改cookie实现会话保持。可选择“开启”或“关闭”。默认“关闭”。
附加HTTP头部	在HTTP请求中附加头字段，在后端成员处理报文时获取客户端真实IP或监听协议信息。 获取真实IP：开启该功能后，可将用户IP地址记录在X-Forwarded-For头域中； 获取监听协议：开启该功能后，可将监听协议记录在X-Forwarded-Proto头域中。
HTTP请求允许非ASCII字符	开启此选项后如果HTTP请求中含有非ASCII字符，BLB会放行并转发给后端成员处理。开启此选项可能带来安全风险，非特殊使用场景建议保持关闭。默认“关闭”。
数据压缩	开启将对特定文件类型进行压缩；关闭则不会对任何文件类型进行压缩，默认关闭
定制配置	BLB 7层监听支持定制配置，配置项和说明可参见 配置参数详情 。

6. 设置转发规则。

项目	描述
优先级	优先级数值越小，则优先级越高，优先级不允许重复
转发规则	可以设置域名、HOST和两者合集的转发规则
目标组	把匹配规则的流量转发至指定服务器组或IP组
目标端口	将匹配规则的流量转发至指定服务器组的指定开放端口

注意：

- 单条规则同时出现域名和URL条件时，则需要同时满足条件，才算匹配成功
- 域名转发规则匹配内容限制：
 - 限制长度<100
 - 通配符仅支持*

- 通配符*可出现0次或1次
- 通配符*可匹配0-n个字符
- 仅支持前缀匹配，即 * 只可出现在域名的开头，如： *.baidu.com
- URL转发规则匹配内容限制：
 - 限制长度<100
 - 通配符仅支持*
 - 通配符*可出现0次或1次
 - 通配符*可匹配0-n个字符 - 支持前、中、后缀匹配，即*可出现在匹配规则的任意位置

例如：将该监听器接收到的流量转发到A服务器组的HTTP 80端口。则在目标服务器组中，选择A服务器组，在目标端口中，选择HTTP 80。

7. 完成上述配置后，点击“确认”按钮完成HTTPS监听器添加。

🔗 编辑监听

在“监听设置”页签，点击<编辑>，跳入编辑界面，编辑后点击<确定>。

🔗 释放监听

1. 在“监听设置”页签，勾选要释放的监听设置，点击<释放>按钮。
2. 在弹出的界面，点击<确定>，完成释放。

IP组使用指南

🔗 简介

IP 目标组（IP Target Group）是一种使用IP作为目标的负载均衡服务器组类型。它是负载均衡服务的一部分，用于将传入的网络请求流量分发给指定的目标IP，将流量路由到基于 IP 地址的目标，例如 BCC实例、容器、IDC内IP 地址等。它允许您根据 IP 地址来定义目标，并将流量动态地分发到这些目标之一。

🔗 优点

IP组允许您自定义目标，允许跨VPC的目标和IDC内的目标等，更为灵活易用，支持更多组网场景。

🔗 操作

🔗 创建IP组

- 在应用型BLB实例列表页点击BLB实例名称进入BLB实例详情页。
- 在详情页点击左侧导航栏：目标组，在上方标签选择IP组。
- 点击：添加IP组。

🔗 IP组类型的选择

当前IP组支持以下五种类型，同地域-同VPC 同地域-跨VPC 跨地域 混合云 IP自定义。 其中：各类型支持的场景如下：

IP组类型	网络连接环境	后端服务
同地域-同VPC	VPC内网环境	· 同VPC环境下的云服务器。 · 同VPC环境下的Pod服务器。 · 同VPC环境下的弹性网卡。 · 同VPC环境下的服务网卡。 · 同VPC环境下的负载均衡。
同地域-跨VPC	· 对等连接 · CSN · VPN	· 云服务器。 · Pod服务器。 · 弹性网卡。 · 负载均衡。
跨地域	· 对等连接 · CSN · VPN	· 云服务器。 · Pod服务器。 · 弹性网卡。 · 负载均衡。
混合云	· 专线 · VPN	· 用户IDC内部署的服务。 · 百度内部IDC内部署的服务。 · 挂载公共服务的服务网卡。
IP自定义	支持以上四种场景下的IP类型组合在同一组中。	支持以上所有类型的后端服务。

🔗 开放协议

- 在【开放协议】栏选择新增协议并在此处添加健康检查设置。
- 开放协议指负载均衡对前端开放的访问协议方式。对于IP组而言，客户端访问将被打到IP管理中IP条目定义的端口上。
- 对于健康检查请求，如果未配置健康检查端口，则健康检查请求同样将被打到IP管理中IP条目定义的端口上。如果用户定义了健康检查端口，则健康检查请求会被打到所有已在IP管理中添加的IP的【用户定义的健康检查端口】上。

🔗 添加IP

在下方的【IP管理】处点击添加希望添加的后端服务的IP。支持添加单条或多条IP。

- 选择添加多条IP时，将一次性注册这些IP的同一端口到IP组中。
- 如果选择的IP组类型是IP自定义，您将需要根据IP地址的实际情况选择这些IP条目的类型。
- 子网内第一个IP地址为系统预留地址，不支持添加格式为.0.1结尾的IP。

点击确认后可以成功创建IP组。

🔗 将IP组绑定在监听上

在对应的负载均衡实例中创建一个监听，然后将这个IP组绑定在这个监听上。注意，监听的协议需要匹配IP组的开放协议，如果不满足可以在IP组中新增开放协议。

🔗 配置路由

如果您配置的IP为 同地域-跨VPC 跨地域 混合云 IP自定义 场景，您还需要根据对应的网络环境配置相应的路由。包括负载均衡访问后端服务的路由和后端服务回源路由。

可以根据你使用的VPN，CSN，对等连接或专线产品的操作文档来具体配置。

注意：同region跨vpc、跨region跨vpc采用对等链接打通时，均需要工单联系配置自定义路由。

🔗 管理IP组

在目标组列表页可以管理IP组。

- 如果希望新增开放协议，则点击操作列的新增协议。
- 如果希望管理已存在的开放协议或健康检查，可以在【协议/规则绑定数量】列，点击图标删除或修改已有的协议。
- 如果希望管理IP组中的IP条目则点击操作列的管理IP组，可以新增或修改删除已存在的IP条目。

IP组跨VPC场景访问协议使用限制

注意：同region跨vpc、跨region跨vpc采用对等链接打通时，均需要工单联系配置自定义路由。

当前，使用IP组进行VPC到VPC的跨VPC访问的场景支持情况如下：

IP版本	是否跨地域	连接环境	负载均衡协议	是否支持
后端为v4	同地域	VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
		CSN	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
		对等连接	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
	跨地域	VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
		CSN	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持
		对等连接	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
后端为v6	同地域	专线	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
		VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
		对等连接	TCP/UDP	支持
			HTTP/HTTPS/SSL	不支持
	跨地域	VPN	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持
		CSN	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持
		对等连接	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持
	IDC	专线	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
		VPN	TCP/UDP	支持
			HTTP/HTTPS/SSL	支持
		对等连接	TCP/UDP	不支持
			HTTP/HTTPS/SSL	不支持

🔗 负载均衡级联挂载说明

当后端挂载的IP实际为负载均衡的前端IP时，是负载均衡级联挂载。请避免负载均衡互相挂载作为彼此的后端服务的行为，这种行为会形成网络循环回路，影响业务正常运行。

负载均衡前后端使用协议对应关系

🔗 网络协议说明

网络协议是计算机网络中用于通信和数据交换的规则集合。以下是负载均衡相关网络协议的简要介绍：

🔗 TCP（Transmission Control Protocol，传输控制协议）：

TCP是一种面向连接的、可靠的协议，用于在网络上传输数据。 它提供数据包的排序、重传机制、流量控制和拥塞控制等功能，确保数据的可靠传输。 TCP通过三次握手建立连接，然后进行数据传输，最后通过四次挥手断开连接。

🔗 UDP（User Datagram Protocol，用户数据报协议）：

UDP是一种无连接的、不可靠的协议，适用于需要快速传输但不需要可靠性保证的应用。 UDP简单高效，适用于音频、视频流等实时应用，因为它不需要建立连接和维护状态。

🔗 HTTP（Hypertext Transfer Protocol，超文本传输协议）：

HTTP是一种用于传输超文本数据的应用层协议，通常用于在Web浏览器和Web服务器之间传输数据。 HTTP是无状态的，每个请求-响应周期都是独立的，不保留会话状态。 HTTP的通信是明文的，不具备加密功能，因此传输的数据容易被窃听和篡改。

🔗 HTTPS（Hypertext Transfer Protocol Secure，安全超文本传输协议）：

HTTPS是在HTTP基础上加入了加密机制的安全传输协议，通过SSL/TLS协议对数据进行加密。 HTTPS通过数字证书验证服务器身份，并加密传输数据，提供了更高的安全性，防止数据被窃听和篡改。 HTTPS通常用于安全性要求较高的网站，如银行、电子商务等，以保护用户数据的安全。

🔗 ICMP（Internet Control Message Protocol）：

ICMP是用于在IP网络上发送控制消息和错误报告的协议，位于网络层。 ICMP用于网络设备间的通信，提供诊断网络问题、报告错误、执行网络探测等功能。 ICMP常用于Ping测试、Traceroute工具、路由器通知等网络操作。

🔗 注意：

- HTTP和HTTPS协议都是基于TCP协议的，因为它们都需要TCP提供可靠的数据传输和连接管理。
- TCP协议作为传输层协议，为HTTP和HTTPS协议提供了可靠的数据传输服务，确保数据的完整性和顺序性。
- HTTPS在HTTP的基础上加入了加密机制，通过TCP协议提供的连接保证传输的数据在网络上传输时的安全性。
- UDP协议发起访问时，本身不存在“连接”的概念，但是负载均衡会保持UDP访问端的Session信息，业内通常把UDP Session的超时时间定义为UDP的“连接超时时间”。

🔗 前后端对应关系说明

在使用负载均衡产品时，需要关注的协议具体有：

- 前端接收协议： 指负载均衡转发客户端请求时，客户端所使用的协议。
- 后端服务协议： 指后端服务器提供服务的协议。
- 健康检查协议： 负载均衡向后端服务器发起健康检查请求时，使用的协议。

对于不同的监听和后端服务器协议，存在以下几种搭配：

监听协议	可搭配的后端服务协议	可搭配的健康检查协议
TCP	TCP	TCP
SSL	TCP	TCP
UDP	UDP	UDP ICMP
HTTP	HTTP	HTTP(仅对HTTP后端服务生效) TCP
HTTPS	HTTP HTTPS	TCP HTTP(仅对HTTP、HTTPS后端服务生效) HTTPS(仅对HTTPS后端服务生效)

🔗 HTTPS访问说明

🔗 通用的HTTPS握手过程

🔗 1.ClientHello :

客户端向服务器发送一个ClientHello消息，其中包含支持的加密算法、协议版本和其他必要信息。

🔗 2.ServerHello :

服务器接收到ClientHello消息后，选择一个加密算法和协议版本，并向客户端发送一个ServerHello消息，包含服务器证书以及其他必要信息。

🔗 3.认证 :

客户端收到服务器的证书后，会验证证书的有效性，包括检查证书是否由受信任的证书颁发机构签发，证书是否过期等。

🔗 4.协商加密密钥 :

如果证书验证成功，客户端生成一个随机的对称密钥，并使用服务器的公钥加密后发送给服务器，用于后续数据传输的加密和解密。

🔗 5.握手完成 :

服务器收到客户端发送的加密密钥后，使用自己的私钥解密，获取客户端生成的对称密钥。此后，客户端和服务端之间的通信将使用这个对称密钥进行加密和解密。

🔗 6.数据传输 :

握手完成后，客户端和服务端之间建立了安全连接，数据传输将通过这个安全通道进行加密传输，确保数据的机密性和完整性。

🔗 HTTPS服务可选搭配说明

🔗 1. HTTPS监听器搭配HTTP后端协议

负载均衡HTTPS监听器会将协商加密密钥的过程卸载到负载均衡实例处理，所以用户端->负载均衡实例->后端服务器的过程中，第一段使用负载均衡实例上配置的证书进行加密协商，第二段负载均衡实例作为请求方使用HTTP协议进行请求，不进行加密。

🔗 2. TCP监听器搭配TCP后端协议

在这种配置下，当用户发起HTTPS访问时，负载均衡实例上只对TCP数据包进行NAT和转发操作，不会修改TCP Payload部分，用户实际与后端服务器做加密协商。

🔗 3. HTTPS监听器搭配HTTPS后端协议

负载均衡HTTPS监听器会将协商加密密钥的过程卸载到负载均衡实例处理，所以用户端->负载均衡实例->后端服务器的过程中，第一段使用负载均衡实例上配置的证书进行加密协商，第二段负载均衡实例作为请求方，与后端服务器使用后端服务器上部署的证书再次进行加密协商。

🔗 4. SSL监听器搭配TCP后端协议

在这种配置下，当用户发起HTTPS访问时，负载均衡实例上处理SSL连接的加密协商，所以用户端->负载均衡实例->后端服务器的过程中，第一段使用负载均衡实例上配置的证书进行SSL加密协商，第二段负载均衡实例作为请求方使用TCP协议进行请求，不进行加密。

使用VPC流日志记录访问日志

🔗 使用场景

在负载均衡开通对外服务后，用户需要对访问进行记录。在这种情况下您可以开通负载均衡的7层访问日志能力，该能力会将访问记录的数据存储到VPC的流日志中。

说明：

当前负载均衡4层访问日志记录到流日志的功能仅对专属集群上的实例生效。负载均衡7层访问日志记录到流日志的功能对共享集群上和专属集群上的实例都生效。

🔗 准备工作

🔗 服务开通

除负载均衡外，您还需要开通以下服务

1. VPC流日志
2. BLS服务 或 BOS服务

VPC流日志可以选择将负载均衡实例所产生的访问数据打入BLS服务或者BOS服务中，所以对于BLS服务和BOS服务二选一即可。

🔗 创建BLS

请参考[BLS操作文档](#)创建BLS日志集。

🔗 创建BOS

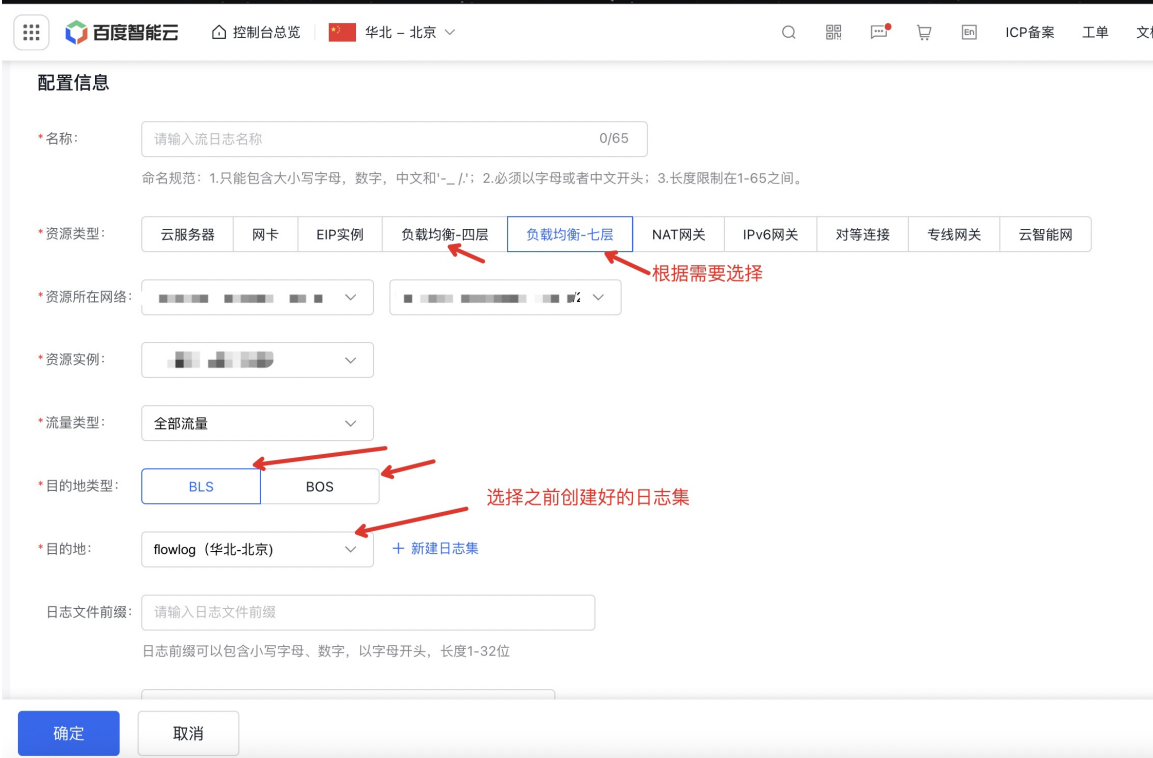
请参考[BOS操作文档](#)创建BOS存储桶。

🔗 为负载均衡开启记录访问日志功能

1. 您需要有一个已创建好的负载均衡实例，并且在这个实例上已经部署了7层的监听（HTTP/HTTPS）。
2. 在负载均衡列表页，点击负载均衡实例ID进入负载均衡详情页。在左侧列表导航栏中选择流日志



击：七层流日志->创建流日志，跳转到VPC侧流日志创建页。根据需要选择对应的参数，并且选择刚创建好的BLS或者BOS的存储日志集或存储桶。



确定，之后回到负载均衡实例。



可以

看到当前负载均衡实例已开启了访问日志功能，并关联到了VPC流日志中。

注意：日志信息记录有2~3分钟的延迟

🔗 日志记录的字段

字段	类型	说明
time_local	string	记录访问日志的时间
vip	string	BLB IP
vport	int	BLB监听端口
response_status	int	BLB应答报文的状态码
vdum_ups_id	string	流量访问的服务器组ID
upstream_addr	string	流量访问的后端服务器地址
upstream_status	string	流量访问后端服务器的响应码
remote_addr	string	与BLB建连的远端client IP，如果BGW不开启cip_forward，则为BGW BIP
cip	string	与BLB建连的远端client IP
cport	int	与BLB建连的远端client 端口
server_addr	string	访问的BLB的虚拟服务器IP
server_port	int	访问的BLB的虚拟服务器端口
scheme	string	协议类型（HTTP/HTTPS）
host	string	域名
request	string	请求行信息
request_length	int	请求长度
body_bytes_sent	int	发送的响应的字节数
http_referer	string	BLB收到请求报文中的refer header内容
http_user_agent	string	BLB收到请求报文中的user-agent header内容
http_x_forwarded_for	string	BLB收到请求报文中的x-forward-for内容

upstream_connect_time	float	BLB与后端服务器的建连时间
upstream_response_time	float	BLB从转发请求到后端服务器到收到后端服务器的响应的时间
request_time	float	BLB从接收到client的请求到返回响应给client的时间
ssl_handshake_time	float	SSL握手时间
ssl_session_reused	string	SSL会话是否复用，复用该字段为r，否则为.
vdum_rs_proto	string	后端服务器协议类型
upstream_local_addr	string	BLB业务节点内部使用ip，用户无需关注
extension_log	string	BLB实例id & BLB实例限速状态，0表示不限速，1表示限速 & BLB实例在当前业务节点上的qps，不限速时显示为0 & BLB实例在当前业务节点上的qps令牌个数，不限速时显示为0

典型实践

通过应用型BLB实现多站点托管

概览

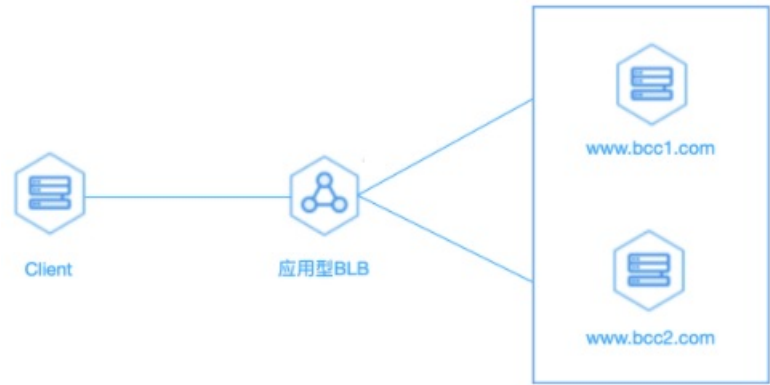
本文主要介绍如何配置应用型BLB，实现将访问不同域名的请求分发至后端不同的服务器上。

需求场景

你 用户经常会遇到多站点访问的场景，既多个域名均解析到了同一个EIP上，而后端则由不同服务器来提供不同站点的服务；此时我们就可以通过应用型BLB来实现多域名转发访问的需求。

方案概述

如下图所示，在同一VPC内创建三台BCC，其中一台作为测试 client，另外两台作为提供网站服务的WebServer，并在同VPC下创建应用型BLB实例用于流量负载。



配置步骤

环境准备

- 1.创建了VPC。创建VPC的具体步骤可参考[创建VPC](#)。
- 2.在VPC内创建子网。创建子网的具体步骤可参考[创建子网](#)。

- 3.在子网中创建三台BCC，分别为bcc1、bcc2和bcc3。创建BCC的具体步骤可参考[创建BCC实例](#)。
- 4.在其中两台BCC服务器上分别部署并安装Nginx，随后启动对应的网站服务，网站地址分别为www.bcc1testsdkhfahsdfh.com和www.bcc2testsdkhfahsdfh.com。CentOS中部署Nginx可参考[Centos-7.2部署LNMP环境](#)。
- 5.在子网中购买应用型BLB实例。购买应用型BLB实例的具体步骤可参考[应用型BLB实例](#)。

添加目标组

- 1.选择“产品服务>负载均衡BLB”，进入负载均衡BLB列表。
- 2.点击“负载均衡ID/名称”列的ID链接，进入实例详情页。
- 3.点击“目标组”页签，选择“添加服务器组”，在弹出的页面中配置服务器组的名称、描述、端口、服务器等。这里我们分别将BCC1和BCC2添加到两个目标组BCC_GROUP1和BCC_GROUP2中，配置目标组的具体操作请参考[配置目标组](#)。

目标组名称(中文)	状态	BCC ID(目标组后端服务器ID)	监听端口	描述	操作
BCC_GROUP1 目标组名称(中文)	可用	HTTP-8080	1	描述: BCC1 监听端口: www.bcc1.com	新建目标组 管理目标组
BCC_GROUP2 目标组名称(中文)	可用	HTTP-8080	1	描述: BCC2 监听端口: www.bcc2.com	新建目标组 管理目标组

添加HTTP监听器

- 配置监听包括“配置信息”和“健康检查设置”两部分。
- 1.选择“产品服务>负载均衡BLB”，点击应用型实例进入负载均衡BLB列表。
- 2.点击“负载均衡ID/名称”列的ID链接，进入实例详情页。点击“监听设置”页签，进入监听器，点击创建监听器。
- 3.监听器的BLB协议处选择HTTP，端口选择Nginx服务对的端口，这里以默认的80端口为例。

* BLB协议[端口]:

HTTP

▼

80

?

调度方法:

加权轮询

▼

?

超时时间:

30

秒

?

描述:

0/200

高级选项

会话保持:

☐ 开启

☒ 关闭

?

附加HTTP头部:

☒ 开启

☐ 关闭

?

☒ 获取真实IPX-Forwarded-For方式获取客户端真实IP

☐ 获取监听协议X-Forwarded-Proto方式获取监听协议

启用HTTPS重定向:

☐ 开启

☒ 关闭

?

HTTP请求允许非

配置转发规则

在转发规则中，点击添加规则，创建两条规则：

规则1: 匹配条件选择"域名", 匹配内容写"*.bcc1testsdkhfahsdfh.com", 目标组选择BCC1所在的目标组, 端口选择80

规则2: 匹配条件选择"域名", 匹配内容写"*.bcc1testsdkhfahsdhf.com", 目标组选择BCC2所在的目标组, 端口选择80

转发规则						
优先级	匹配条件	匹配内容	目标组	目标端口	描述	操作
1	域名 新增匹配条件	*.bbc1.cor	BCC_GROUP1 (sg-f9f8ac6c)	HTTP:80		删除
2	域名 新增匹配条件	*.bbc2.cor	BCC_GROUP2 (sg-fcdfea5f)	HTTP:80		删除

关于应用型BLB监听器及转发规则的详细说明，可以参考配置[应用型BLB实例监听器](#)。

访问测试

在客户端中配置下域名解析（模拟真实访问的场景），将www.bcc1testsdkhfahsdfh.com和www.bcc2testsdkhfahsdfh.com都解

析到BLB的前端IP上。

```
[root@11ccc ~]# cat /etc/hosts
192.168.0.7 www.bcc1.com
192.168.0.7 www.bcc2.com
```

分别访问www.bcc1testsdkhfahsdfh.com和www.bcc2testsdkhfahsdfh.com，可以看到请求被正确的分发到了对应的BCC上。

```
[root@b1b0003 ~]# curl www.bcc1.com
This is BCC1
[root@b1b0003 ~]# curl www.bcc2.com
This is BCC2
```

相关产品

[云服务器BCC](#)、[弹性公网IP](#)

通过Terraform管理BLB

概览

本文介绍了通过Terraform管理BLB的方式，实现对BLB资源的快速创建、管理等功能。

需求场景

简化资源运维，利用代码即基础设施的思想，通过编写Terraform相关代码文件来实现对BLB生命周期的管理。

方案概述

- 下载Terraform工具，准备基础环境
- 编写.tf文件，配置BLB的基础属性
- 通过Terraform相关命令完成对BLB的生命周期管理

配置步骤

环境准备

- 准备一台可以连接到Internet的PC或者笔记本电脑
- 登录[Terraform官网](#)
- 选择对应系统下载，解压后是一个2进制可执行文件，需要进行环境变量配置
- 进入配置文件 vi ~/.bash_profile
- 添加环境变量 export PATH=\${PATH}:/path (path:二进制文件位置)
- 配置生效 source ~/.profile
- 检查Terraform是否可以正常运行：执行 terraform -v 如果可以正常显示版本号则证明安装成功

Terraform管理BLB示例

编辑相关tf文件，用于管理BLB实例配置

- 编辑main.tf：用于配置要下载的provider版本及后续需要的ak sk信息

```
terraform {
  required_providers {
    baiducloud = {
      source = "baidubce/baiducloud" #源地址
      version = "1.12.0" #版本号
    }
  }
}

provider "baiducloud" {
  access_key = "xxxxx" #用户uid对应的ak
  secret_key = "xxxx"  #用户uid对应的sk
  region     = "bj"     #待创建实例所在地域
}
```

- 编辑.tf文件创建vpc

```
resource "baiducloud_vpc" "default" {
  provider = baiducloud.bj
  name     = "terra-test-vpc"
  description = "baiducloud vpc created by terraform"
  cidr     = "192.168.0.0/16"
}
```

- 编辑.tf文件创建subnet

```
resource "baiducloud_subnet" "default" {
  name       = "terra-subnet"
  zone_name = "cn-bj-a"
  cidr      = "192.168.3.0/24"
  vpc_id    = "${baiducloud_vpc.default.id}" # 上一步创建好的vpcid
}
```

- 编辑.tf文件创建BLB实例

```
resource "baiducloud_blb" "default" {
  name       = "terratestLoadBalance-changed"
  description = "this is a test LoadBalance instance"
  vpc_id     = "${baiducloud_vpc.default.id}" # 上一步创建好的vpcid
  subnet_id  = "${baiducloud_subnet.default.id}" # 上一步创建好的subnetid
}
```

- 完整配置信息如下：

```

resource "baiducloud_vpc" "default" {
  name = "terra-test-vpc"
  description = "baiducloud vpc created by terraform"
  cidr = "192.168.0.0/16"
}

resource "baiducloud_subnet" "default" {
  name = "terra-subnet"
  zone_name = "cn-bj-a"
  cidr = "192.168.3.0/24"
  vpc_id = "${baiducloud_vpc.default.id}"
}

resource "baiducloud_blb" "default" {
  name      = "terratestLoadBalance-changed"
  description = "this is a test LoadBalance instance"
  vpc_id    = "${baiducloud_vpc.default.id}"
  subnet_id = "${baiducloud_subnet.default.id}"
}

```

执行Terraform命令创建BLB实例

- terraform init : 用于下载或更新百度provider，出现以下信息代表下载成功。

Initializing the backend...

Initializing provider plugins...

- Finding baidubce/baiducloud versions matching "1.12.0"...
- Installing baidubce/baiducloud v1.12.0...
- Installed baidubce/baiducloud v1.12.0 (unauthenticated)

Terraform has created a lock file **.terraform.lock.hcl** to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.

- terraform plan : 确认资源变更计划
- terraform apply : 执行资源创建，出现以下信息代表下载成功

Plan: 3 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```

baiducloud_vpc.default: Creating...
baiducloud_vpc.default: Creation complete after 1s [id=vpc-ikfgnxq88j19]
baiducloud_subnet.default: Creating...
baiducloud_subnet.default: Creation complete after 0s [id=sbn-k3hfgnca5i57]
baiducloud_blb.default: Creating...
baiducloud_blb.default: Still creating... [10s elapsed]
baiducloud_blb.default: Still creating... [20s elapsed]
baiducloud_blb.default: Creation complete after 24s [id=lb-c4460def]

```

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

🔗 资源销毁

- 执行terraform destroy 命令可销毁上述资源

Plan: 0 to add, 0 to change, 3 to destroy.

Do you really want to destroy all resources?

Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

```
baiducloud_blb.default: Destroying... [id=lb-c4460def]
baiducloud_blb.default: Destruction complete after 1s
baiducloud_subnet.default: Destroying... [id=sbn-k3hfgnca5i57]
baiducloud_subnet.default: Destruction complete after 0s
baiducloud_vpc.default: Destroying... [id=vpc-ikfgnxq88j19]
baiducloud_vpc.default: Destruction complete after 2s
```

Destroy complete! Resources: 3 destroyed.

🔗 测试验证

- 执行terraform show 查看是否显示如下信息


```

[~] ~$ cd /root/.ssh/050B000000007175D resourblb % terraform show
# baiduccloud_blb.default:
resource "baiduccloud_blb" "default" {
  address      = "192.168.3.2"
  cidr         = "192.168.0.0/16"
  create_time  = "2022-10-18T07:22:36Z"
  description  = "this is a test LoadBalance instance"
  id           = "lb-c4460def"
  listener     = []
  name         = "terratestLoadBalance-changed"
  status       = "available"
  subnet_id    = "sbn-k3hfgnca5i57"
  tags         = {
    "product_name" = "terra-test"
  }
  vpc_id       = "vpc-ikfgnxq88j19"
  vpc_name     = "terra-test-vpc"
}

# baiduccloud_subnet.default:
resource "baiduccloud_subnet" "default" {
  cidr      = "192.168.3.0/24"
  id        = "sbn-k3hfgnca5i57"
  name      = "terra-subnet"
  subnet_type = "BCC"
  vpc_id    = "vpc-ikfgnxq88j19"
  zone_name = "cn-bj-a"
}

# baiduccloud_vpc.default:
resource "baiduccloud_vpc" "default" {
  cidr            = "192.168.0.0/16"
  description     = "baiduccloud vpc created by terraform"
  id              = "vpc-ikfgnxq88j19"
  name            = "terra-test-vpc"
  route_table_id = "rt-i4c7qjugtuvs"
  secondary_cidrs = []
  tags            = {
    "product_name" = "terraform-test"
  }
}

```

- 登录百度智能云，查看是否成功购买



相关产品

[负载均衡BLB](#)

通过Terraform实现BLB关联监听器、后端服务器、安全组等

概览

本文介绍了通过Terraform实现BLB关联监听器、后端服务器、安全组等功能

需求场景

简化资源运维，通过编写Terraform相关代码文件来实现BLB关联监听器、后端服务器、安全组等功能

方案概述

- 下载Terraform工具，准备基础环境
- 编写.tf文件，管理基础配置
- 通过Terraform相关命令完成BLB实例关联监听器、后端服务器、安全组等功能

配置步骤

环境准备

- 准备一台可以连接到Internet的PC或者笔记本电脑
- 登录[Terraform官网](#)
- 选择对应系统进行下载，解压后是一个2进制可执行文件，需要进行环境变量配置

- 进入配置文件 vi ~/.bash_profile
- 添加环境变量 export PATH=\${PATH}:/path (path:二进制文件位置)
- 配置生效 source ~/.profile
- 检查Terraform是否可以正常运行：执行 terraform -v 如果可以正常显示版本号则证明安装成功

🔗 Terraform实现BLB关联监听器、后端服务器、安全组等功能示例

编辑相关tf文件，用于管理BLB、监听器、后端服务器、安全组等配置

- 编辑main.tf：用于配置要下载的provider版本及后续需要的ak sk信息

```
terraform {
  required_providers {
    baiducloud = {
      source = "baidubce/baiducloud" #源地址
      version = "1.12.0" #版本号
    }
  }
}

provider "baiducloud" {
  access_key = "xxxxx" #用户uid对应的ak
  secret_key = "xxxx"  #用户uid对应的sk
  region     = "bj"     #待创建实例所在地域
}
```

- 编辑blb.tf管理blb示例，可参考[通过Terraform管理BLB](#)
- 编辑blb_listener.tf文件关联listener

```
resource "baiducloud_blb_listener" "TCP" { # 关联TCP
  blb_id      = "${baiducloud_blb.default.id}" # 上一步创建好的blbid
  listener_port = 124                          # 前端监听端口
  backend_port = 1240                          # 后端服务端口
  protocol     = "TCP"                        # 监听协议类型
  #scheduler   = "LeastConnection"           # 负载均衡算法（转发规则）
  scheduler    = "RoundRobin"
}

resource "baiducloud_blb_listener" "UDP" { # 关联UDP
  blb_id      = "${baiducloud_blb.default.id}" # 上一步创建好的blbid
  listener_port = 125
  backend_port = 125
  protocol     = "UDP"
  #scheduler   = "LeastConnection"
  scheduler    = "RoundRobin"
  health_check_string = "healthy"             #UDP健康检查字符串（udp协议特有）
}

resource "baiducloud_blb_listener" "HTTP" { # 关联HTTP
  blb_id      = "${baiducloud_blb.default.id}" # 上一步创建好的blbid
  listener_port = 126
  backend_port = 126
  protocol     = "HTTP"
  #scheduler   = "LeastConnection"
  scheduler    = "RoundRobin"
}
```

- 编辑blb_server.tf文件关联安全组

```

data "baiducloud_images" "default" {                # 查询镜像
  image_type = "System"
  name_regex = "8.4 aarch"
  os_name    = "CentOS"
}

resource "baiducloud_instance" "default1" {         # 创建bcc实例
  billing = {
    payment_timing = "Postpaid"
  }
  instance_spec = "bcc.gr1.c1m4"
  image_id      = data.baiducloud_images.default.images.0.id # 上一步查出的镜像id
  availability_zone = "cn-bj-d"
}

resource "baiducloud_instance" "default2" {         # 创建bcc实例
  billing = {
    payment_timing = "Postpaid"
  }
  instance_spec = "bcc.gr1.c1m4"
  image_id      = data.baiducloud_images.default.images.0.id # 上一步查出的镜像id
  availability_zone = "cn-bj-d"
}

resource "baiducloud_blb_backend_server" "default" { # 关联后端服务器
  blb_id      = "${baiducloud_blb.default.id}"
  backend_server_list {
    instance_id = "${baiducloud_instance.default1.id}" # 上一步创建出的实例id
    weight      = 39                                     # 设置权重
  }
  backend_server_list {
    instance_id = "${baiducloud_instance.default2.id}"
    weight      = 5
  }
}

```

- 编辑blb_security.tf文件关联安全组

```

resource "baiducloud_vpc" "default" {               # 创建vpc
  name = "terra-test-vpc"
  description = "baiducloud vpc created by terraform"
  cidr = "192.168.0.0/16"
}

resource "baiducloud_security_group" "default1" {   # 创建安全组
  name      = "terra-security-group-1"
  description = "created by terraform"
  vpc_id    = "${baiducloud_vpc.default.id}"        # 上一步创建好的vpcid
}

resource "baiducloud_blb_securitygroup" "blb_default" { # 关联安全组
  blb_id    = "${baiducloud_blb.default.id}"
  security_group_ids = ["${baiducloud_security_group.default1.id}"]
}

```

执行Terraform命令完成创建

- terraform init ：用于下载或更新百度provider，出现以下信息代表下载成功

Initializing the backend...

Initializing provider plugins...

- Finding baidubce/baiducloud versions matching "1.12.0"...
- Installing baidubce/baiducloud v1.12.0...
- Installed baidubce/baiducloud v1.12.0 (unauthenticated)

Terraform has created a lock file **.terraform.lock.hcl** to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.

- terraform plan：确认资源变更计划
- terraform apply：执行资源创建，出现“Apply complete”代表成功

🔗 资源销毁

- 执行terraform destroy命令可销毁上述资源
- 出现“Destroy complete”代表销毁成功

🔗 测试验证

- 执行terraform show 查看本地是否可以正确输出购买结果
- 登录百度智能云，查看BLB实例是否已关联监听器、安全组、后端服务器

🔗 相关产品

[负载均衡BLB](#)

健康检查异常排查

场景介绍

负载均衡的健康检查是一种机制，用于定期检测后端服务器或服务状态，来确保负载均衡器只将流量分配给那些处于正常工作状态的服务器。通过健康检查，负载均衡器可以实时了解后端服务的可用性，快速检测并隔离故障节点，确保系统的稳定性和性能。在有些时候，我们会遇到不明原因的健康检查异常情况，本文将介绍几种可能的健康检查异常原因以及对应的排查方式。

🔗 处理步骤

🔗 检查RS安全组配置

通常情况下，后端服务器会默认放通来自BLB实例的健康检查流量，在少数不支持此能力的机型，或默认放通被关闭的情况下，健康检查流量可能会被后端服务器的安全组拦截，您需要在后端服务器所在安全组中放通入向来源段为：

100.64.0.0/10（IPv4）或2403:ed40:f200::/40（IPv6）段，入向协议为健康检查所选择的协议，入向端口为配置的健康检查端口，如果BLB侧健康检查端口未配置时，默认情况下健康检查端口为后端服务器提供服务的端口。之后，后端服务器可以正常放通这个范围的健康检查流量。

🔗 检查Host配置（针对http/https监听器）

在一些后端服务器配置中，为了规避异常请求，会屏蔽HTTP Header中Host为空的情况。在配置HTTP和HTTPS的后端开放协议

时，将“Host头域”这一参数配置为预期的域名，将可以解决因这一情况导致的健康检查异常。

🔗 检查后端服务器端口服务配置

当健康检查端口为单独指定的情况下，可能出现健康检查端口与后端服务器实际所提供的服务的端口不一致。在这种情况下，需要在后端服务器上对应健康检查的端口起对应的服务，目的是为了能够正确响应BLB实例的健康检查请求。

- 该服务需要开放在0.0.0.0，可以使用 `netstat -antup(Linux)` 或 `netstat -ano(Windows)`命令进行确认。
- 该服务需要正确响应对应的健康检查请求协议或方法。
- 如已配置了预期响应的条件，服务返回的响应需要能够匹配该响应条件，才可以被视为成功。

🔗 其他可能情况

如非以上问题导致的健康检查提示异常时，需要按以下步骤排查

- 检查后端服务器已开启了相应的服务端口。
- 检查后端服务器内部是否有防火墙之类的防护软件，拦截健康检查报文。
- 检查后端服务器上有没有做 iptables 规则限制。
- 检查客户端直接访问后端服务器应用服务是否可以成功。
- 检查负载均衡健康检查参数设置是否正确。
- 负载均衡七层(http/https)监听器建议使用静态页面来健康检查。
- 检查后端服务器是否有高负载导致服务对外响应慢。

通过应用型BLB实例获取客户端信息

🔗 场景介绍

在负载均衡中分析客户端信息是为了更智能地分配请求，提高系统的性能、可靠性和用户体验，能够实现：帮助负载均衡器选择合适的后端服务器来处理请求。例如，可以基于客户端IP地址、用户会话信息或地理位置等因素进行路由决策，将请求分配给最适合处理的服务器；增强安全性。例如，可以根据客户端的地理位置或IP地址进行访问控制，防范恶意行为或DDoS攻击等能力。本文将介绍如何使用BLB自带的功能分析用户信息。

🔗 基本原理

由于BLB7层监听器(http/https)在转发时会改写客户端源IP信息，后端服务器收到请求的实际源IP为BLB自留100.64.0.0/16网段地址，负载均衡的7层监听器(HTTP/HTTPS)监听器支持在转发请求时将客户端真实IP写到header:X-Forwarded-For处，支持将监听器协议写到header:X-Forwarded-Proto处。这样，后端在接收到请求时，可以根据这两个头信息记录和分析来访客户端的特征。

本文将基于CentOS BCC上配置Nginx服务器为例，演示如何获取客户端信息。

🔗 准备工作

您需要先创建好应用型BLB实例，配置好7层监听器（HTTP或HTTPS），在监听器上配置好对应的服务器组或IP组，详见：[应用型BLB指导文档](#)

🔗 配置服务器

1. 在与BLB实例同VPC环境下，购买BCC实例，此处选择CentOS 7.6镜像，详见：[快速配置Linux云服务器](#)
2. 将BCC实例添加为BLB实例的后端服务器，详见：[应用型BLB指导文档](#)
3. 将此服务器所在服务器组与监听器关联，并开启

4. 登录BCC实例，执行以下命令配置Nginx和http_realip_module `yum -y install gcc pcre pcre-devel zlib zlib-devel openssl`

```
openssl-devel
```

```
wget http://nginx.org/download/nginx-1.17.0.tar.gz
```

```
tar zxvf nginx-1.17.0.tar.gz
```

```
cd nginx-1.17.0
```

```
./configure --prefix=/path/server/nginx --with-http_stub_status_module --without-http-cache --with-http_ssl_module --
```

```
with-http_realip_module
```

```
make
```

```
make install
```

说明：下载Nginx相关包需要公网环境

配置服务端获取客户端信息

执行以下命令，打开nginx.conf文件。 `vi /path/server/nginx/conf/nginx.conf`

- 按i进入编辑模式，在以下 server 配置信息后添加新的配置字段和信息。

```
server {
    listen 80;
    server_name localhost;
}
```

1. 获取源IP

在Server字块中需要添加的配置字段和信息：

```
``NGINX
set_real_ip_from 100.64.0.0/10;
real_ip_header X-Forwarded-For;
``
```

在location / 字块中需要添加的配置字段和信息：

```
``NGINX
location / {
    # 获取真实IP
    set $real_ip $remote_addr;

    # 如果启用了realip模块，可以使用如下变量
    # set $real_ip $realip_remote_addr;

    # 在这里可以使用$real_ip进行进一步的处理
    # 例如，将其传递给后端服务器或记录日志等
}
``
```

- 说明

- 您可以将代理服务器的网段添加到 `set_real_ip_from <IP_cidr>`，以便获取代理服务器的IP地址。例如，添加 应用型 BLB 的IP地址段（100.64.0.0/10是 应用型BLB的保留地址，其他用户无法分配到该网段内，不会存在安全风险）。
- 如为应用型IPv6BLB,此处网段为：2403:ed40:f200::/40
- listen 80 如果监听端口非80，请替换为后端服务器的实际端口。
- 多个IP地址段之间使用逗号分隔。

2. 获取协议信息

在 location / 字块中需要添加的配置字段和信息：

端口设置

温馨提示：普通安全组是“允许访问”的规则集合，每添加一个规则，代表允许一个source访问某个或某个范围的端口。若安全组中未添加任何规则，表示所关联BCC实例的所有端口都不能被外界访问

入站

出站

允许访问所有端口：☐

+ 添加规则

删除

☐	类型	协议	端口	目的	备注	更新时间
☐	IPv6	全部协议	1-65535	目的IP: :/0	全部协议	2024-03-04 16:30:24
☐	IPv4	全部协议	1-65535	目的IP: 0.0.0.0/0	全部协议	2024-03-04 16:30:24

2.将负载均衡与安全组关联

点击BLB实例名称，在左导栏中选择安全组，在“关联的安全组列表”中选择“关联安全组”，即可在弹窗中选择已创建的安全组关联。

实例详情

监听设置

目标组

安全组

监控

日志

启用默认放通：☒

当前应用的安全组规则

入站

出站

协议	端口范围	Source	描述
暂无数据			

关联的安全组列表

关联安全组

安全组名称	安全组类型	所在网络	描述
暂无数据			

说明：启动默认放通开关开启后，BLB和BCC之间默认放通；关闭后，来自BLB的流量将受到BCC安全组的管控。

3. 设置VPC ACL

除安全组外，可以设置VPC ACL对流入子网流量进行整体控制。点击VPC名称进入VPC详情页面，再点击私有网络标签，选中对应实例名称，在包含资源-网络资源 中找到ACL，点击右侧蓝色数字。

基本信息

名称: create_from_api_test_vpc_中文_./10fWdzouNZPh4S3DJayyB qTDm_FiiwA5_1

ID: vpc-hxs6szhqrt97

域名: bj.baidu.internal

网段: 172.16.144.0/20 (主)

创建时间: 2024-02-29 16:51:23

域名服务器: - 变更

描述: 中文测试_Ab_!@#%*^&*()_+{}|~<>?12-=[];.,/ a g n test

包含资源

基础云资源

BCC0

DCC0

BBC0

BCI0

网络资源

子网1

路由表1

弹性网卡0

服务网卡0

VPN网关0

安全组9

ACL1

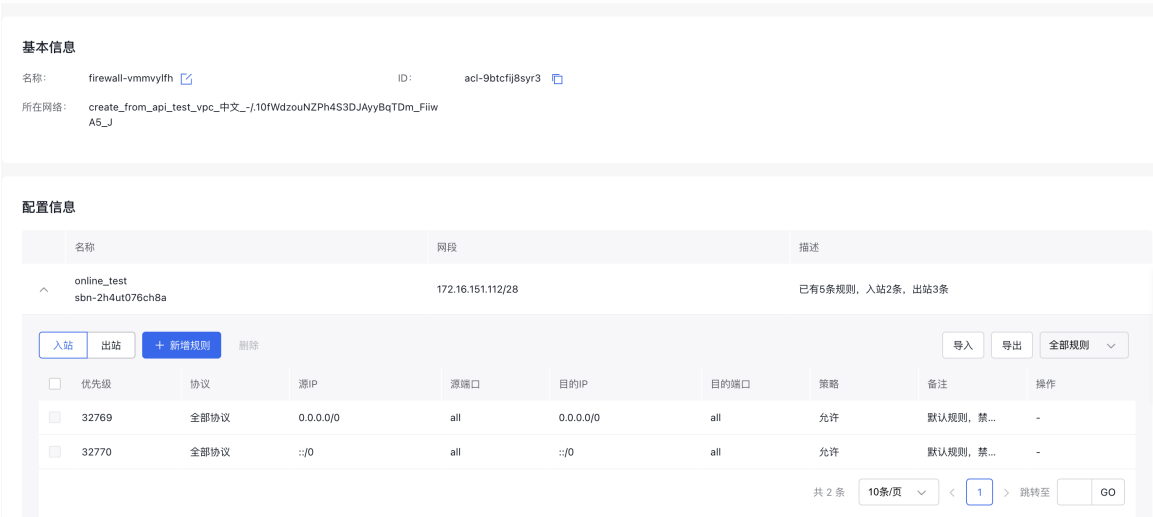
NAT网关0

对等连接0

专线网关1

进入ACL

详情页，点击左侧下拉箭头，配置出入站规则。



之后，

出入BLB流量将经过所在VPC的ACL和BLB所关联的安全组的两道校验。这将极大地提高BLB实例的安全性。

配置HTTPS访问

HTTPS（Hypertext Transfer Protocol Secure）是一种安全的超文本传输协议，它是HTTP的安全版本。HTTPS通过使用TLS（Transport Layer Security）或其前身SSL（Secure Sockets Layer）协议来保护数据的传输安全性。

当前，负载均衡支持配置HTTPS协议的监听端口，能够保护数据在客户端和服务端之间的传输，防止中间人攻击和窃听。

配置HTTPS监听器需要完成以下步骤：

1. 证书获取

可以在[证书管理](#)中上传或新购SSL证书，具体添加证书指引见[购买证书指导文档](#)和[上传证书指导文档](#) 完成以上流程后，您可以在[证书管理](#)中看到您的已签发或上传的SSL证书。,详细说明参考[证书管理指导文档](#)。 之后，您可以在应用型BLB HTTPS监听器创建时，在下拉列表中选择以上SSL证书。

注意：仅支持在监听器配置服务器类型的证书

2.HTTPS监听器配置

启用HTTPS协议的监听器，可以将数据加解密交由负载均衡器处理，后端服务只需要关注业务本身。使用上面配置好的证书，您可以配置HTTPS监听器，以提高服务的安全性。 [配置HTTPS监听端口指导文档](#) 示例配置如下：

基本配置

BLB协议[端口]:

HTTPS

443

?

调度方法:

加权轮询

?

大搜泛域名证书:

开启

关闭

服务器证书:

blb.sin.baidubce.com/blb.sin.baidubce.com

添加证书

管理我的证书

申购SSL证书

扩展域名:

开启

关闭

双向认证:

开启

关闭

加密选项:

预定义的安全策略

自定义安全策略

?

tls_cipher_policy_default

加密协议:

TLSv1.0协议

TLSv1.1协议

TLSv1.2协议

加密套件:

全部 (19项)

超时时间:

30

秒

?

描述:

3.启用双向认证（可选）

双向认证，也被称为双向SSL认证或客户端证书认证，是一种加强网络通信安全性的方法。在传统的SSL/TLS连接中，通常只有服务器需要提供证书来验证其身份，而客户端不需要验证。而双向认证要求客户端也提供有效的证书，从而确保双方都经过身份验证。最显著的好处是降低了中间人攻击和伪装的风险。

3.1 上传CA证书 可以在**证书管理**中上传CA证书，上传时选择类型为CA证书。

3.2 在HTTPS监听器中配置双向认证 在配置HTTPS监听器时选择开启双向认证，并选择刚刚上传的CA证书

基本配置

BLB协议[端口]:

HTTPS

443

?

调度方法:

加权轮询

?

大搜泛域名证书:

开启

关闭

服务器证书:

blb.sin.baidubce.com/blb.sin.baidubce.com

添加证书

管理我的证书

申购SSL证书

扩展域名:

开启

关闭

双向认证:

开启

关闭

客户端CA证书:

client_cert_id/TrustAsia TLS RSA CA

添加证书

管理我的证书

请确保

对负载均衡发起访问的客户端已正确配置同一CA证书并拥有证书私钥。

使用应用防火墙 WAF 提供访问安全性

Web应用防火墙（Web Application Firewall，简称WAF）是百度智能云面向用户提供的Web安全防护产品，能够有效防护各种Web攻击，协助用户定制访问规则，提升网站等业务的安全。独创的全新WAF技术架构体系，能够将WAF实例灵活部署在各个

143

Web业务入口，避免了传统云WAF架构下黑客绕过代理直攻源站的尴尬。云端安全大数据能力的集成也让WAF能更有效更快捷的帮助客户提升网站安全性与可用性。 通过在应用型负载均衡实例上配置绑定WAF，能够提高服务入口的安全防护能力。

1. 创建WAF实例

在WAF实例列表点击购买WAF资源。 选择BLB类型进行新建。 配置示例：

实例配置

类型：

云原生 WAF

类别：

BLB

CDN

地域：

华北 - 北京

华南 - 广州

华东 - 南京

中国香港

华东 - 苏州

华东 - 上海

华北 - 保定

金融华中 - 武汉

华北 - 阳泉

西南 - 成都

配置：

最大保护的根域名数：1个

最大防护的子域名数：10个

支持协议：HTTP、HTTPS

Web安全防护：各种常见Web攻击、Oday攻击规则自动更新

最大的自定义规则数：20个

付费方式

付费方式：

预付费

后付费

下一步

取消

已选配置：

查看详情

购买数量：

1

2. 将WAF与BLB实例绑定

完成WAF的实例购买后，需要对其进行配置，才能够实现WAF的防护能力，配置步骤如下：

- 选择“产品服务>应用防火墙服务 WAF”，进入BLB WAF 列表页，点击主域名栏下的配置，进入配置详情页面。
- 基本配置填写，填写需要防护的”根域名”、”子域名”，并选择绑定的负载均衡BLB实例。

只能绑定与WAF实例所在同一区域的BLB实例，仅支持HTTP/HTTPS 协议，如没有符合条件的BLB实例，请前往控制台购买或重新配置BLB实例。

WAF-beijing-be165cfb ● 运行中

基本配置

根域名：

baidu.com

保护的子域名，一个WAF实例仅可保护1个子域名

子域名：

1

baidu.com

+ 子域名

负载均衡绑定：

lb-10487148/BLBWAF

lb-10487148/BLBWAF

协议/端口

功能配置

Web防护：

ON

 开启

中级策略集（默认）

拦截

自定义访问控制：

OFF

 关闭

144

- 开启Web防护，选择防护策略等级。
 - 默认开启中级策略集，越严格安全防护效果越好。高级策略集，表示开启严格的防护策略，但可能出现误拦截情况；中级表示具备中和低两种策略集；低级表示防护策略宽松。
 - 每种防护策略都具备『拦截』和『观察』功能，拦截模式发现攻击请求立即阻断；观察模式发现攻击请求立即记录但不拦截。
- （可选）开启自定义访问控制，点击『添加』按键，通过自定义规则实现对您自己的业务控制和过滤。

参数	说明
名称	自定义访问控制规则名称
匹配项	http请求的“来源IP”、“URL地址”、“Referer”字段、“User-Agent”字段等内容的匹配处理。
匹配模式	选择匹配模式：前缀、包含或后缀。
匹配字符串	输入需要访问控制的字符串。
执行动作	将字符串列为黑名单或白名单
模式	拦截：发现攻击请求立即阻断；观察：发现攻击立即记录但不拦截。

- 点击『确认生效』，完成绑定BLB操作。

🔗 攻击排查

对于异常访问请求，可在BLB上开启访问日志，转存到BOS和BLS产品，结合BLB的云监控能力实现对异常访问请求的分析和排查。**注意：**访问日志转存在BOS和BLS产品的能力，共享集群上的实例只在7层监听器支持。

🔗 1. 云监控

云监控将展示BLB实例接收流量的情况。可以在LB实例的监控页面查看。

- 点击BLB名称进入BLB实例详情页，点击左导栏中的“监控”进入当前BLB实例的监控页，在[负载均衡服务端口]下拉选择想要查看的端口。可以着重注意**网络输入带宽**和**网络输出带宽**是否有异常激增情况。

🔗 2. 访问日志

访问日志是记录访问BLB实例的请求详情的能力。已开启访问日志的BLB实例，可以在流量异常时同步排查访问日志。开启访问日志前，您需要先创建BOS存储桶，用来存储BLB访问日志功能记录的访问详情。

🔗 2.1 创建BOS Bucket

在[BOS首页](#)点击左导栏“Bucket列表”进入后点击上方“+”创建Bucket。配置示例：

创建存储桶



基本信息

* Bucket名称:

只能包含小写字母、数字和“-”，开头结尾为小写字母和数字，长度在4-63之间；存储桶名称创建后不可修改

* 所属地域:

华北 - 北京

不同区域内的云产品内网互不相通，存储桶创建成功后地域不可修改

请求域名:

创建完成后，您可以使用该域名对存储桶进行访问

* 多AZ特性:

单AZ

多AZ

推荐开启多AZ，开启多AZ（可用区）存储后，BOS会将用户的数据分散存放在同一地域下的多个可用区内；多AZ存储可以提供强大的容灾能力，当某个可用区不可用时仍能保障数据的正常访问，从而保证业务的连续性。

* 默认存储类型:

标准存储

低频存储

冷存储

归档存储

上传文件时如果未指定存储类型，则使用该默认存储类型 [了解存储类别差异](#)；不同存储类型收费标准不同 [计费文档](#)

* 读写权限:

私有

公共读

公共读写

只有用户本人可读写该Bucket中的数据

创建完成后可在[Bucket设置-权限设置](#)里进行更灵活的权限管理。

高级配置

CDN加速:

关闭

开启

开启CDN加速可使用官方域名对文件进行加速访问，开启后建议您及时前往CDN配置缓存规则，可减少对BOS不必要的访问，从而降低CDN回源BOS的流量费用。

2.2 开启访问日志功能

- 点击BLB名称进入BLB实例详情页，点击左导栏中的“日志”进入当前BLB实例的日志详情页，开启日志启用并存储到上面创建出来的存储桶中。

百度智能云

控制台总览

华北 - 北京

请输入您想要搜索的产品

返回实例列表

实例详情

监听设置

后端服务器

安全组

监控

日志

访问日志

日志设置

当前支持7层访问日志收集，开启访问日志服务后，我们会将您的原始访问日志推送至您选择的bucket，计费参照BOS计费规则，BOS存储可以保证您的日志永久保存。

日志启用:

开

推送频率:

小时

* 存储到:

bucket必须存在，如不存在，请新建;folder可不存在，如不存在，系统会自动创建

日志前缀:

日志前缀可以包含小写字母、数字，以字母开头，长度1-32位

保存

后，在访问日志保持开启的状态下，可以在所选Bucket中看到访问信息。

BLB选型指南

146

概览

本文将帮助您选择合适的BLB类型，部署模式，和计费方式。

产品概述

产品类型

BLB分为如下几种类型：

实例类型	说明	区别	适用场景
普通型 BLB实例	提供了负载均衡常规功能组件，能够满足用户对负载均衡的通用需求，可以将来自互联网或内网的IPv6和IPv4流量分发至多台IPv4后端服务器，具备轻量化、配置简单、价格便宜等特点。	按实例维度组织后端服务器，每个监听器下所有后端服务器监听端口相同。流量转发方式：监听器统一调度支持加权轮询、加权最小连接数、基于源IP等调度算法转发	IPv4或IPv6的客户端访问 IPv4后端服务器，支持NAT44和NAT64转换，HTTP/HTTPS监听不支持配置复杂的转发策略
应用型 BLB实例	针对用户复杂应用部署架构，特别是大型网站架构。使用基于策略的网络管理框架构建，可以将来自互联网或内网的IPv6和IPv4流量分发至多台IPv4后端服务器，实现业务驱动流量负载均衡。	后端服务支持服务器组或IP组，可以按监听器维度组织后端服务器或后端服务IP，每个监听器下所有后端服务器或后端服务IP监听端口可以不同。流量转发方式：HTTP/HTTPS监听按策略转发支持根据域名/URL等做流量转发	IPv4或IPv6的客户端访问 IPv4后端服务器，支持NAT44和NAT64转换，HTTP/HTTPS监听支持配置复杂的转发策略
普通型 IPv6 BLB实例	提供了负载均衡常规功能组件，能够满足用户对负载均衡的通用需求，可以将来自互联网或内网的IPv6流量分发至多台IPv6后端服务器，具备轻量化、配置简单、价格便宜等特点。	按实例维度组织后端服务器，每个监听器下所有后端服务器监听端口相同。流量转发方式：监听器统一调度支持加权轮询、加权最小连接数、基于源IP等调度算法转发	IPv6客户端访问IPv6后端服务器，支持NAT66转换，HTTP/HTTPS监听不支持配置复杂的转发策略
应用型 IPv6BLB实例	针对用户复杂应用部署架构，特别是大型网站架构。使用基于策略的网络管理框架构建，可以将来自互联网或内网的IPv6流量分发至多台IPv6后端服务器，实现业务驱动流量负载均衡。	后端服务支持服务器组或IP组，可以按监听器维度组织后端服务器或后端服务IP，每个监听器下所有后端服务器或后端服务IP监听端口可以不同。流量转发方式：HTTP/HTTPS监听按策略转发支持根据域名/URL等做流量转发	IPv6客户端访问IPv6后端服务器，支持NAT66转换，HTTP/HTTPS监听支持配置复杂的转发策略

🔗 产品部署模式

所在集群	性能类型	性能限制
专属集群	性能共享型	不限速，在流量高峰期会与专属集群内的其他实例互相性能挤占，专属集群保障用户在当前集群下所有BLB实例的整体性能。
专属集群	性能规格型	限速，在流量高峰期会保证实例性能，不会与专属集群内的其他实例互相性能挤占。
共享集群	不限速	不限速，在流量高峰期会与共享集群内的其他实例互相性能挤占，不保障用户整体性能。
共享集群	性能规格型	限速，在流量高峰期会保证实例性能，不会与共享集群内的其他实例互相性能挤占。

可用区模式	适用场景
主备可用区	容灾要求较高的场景。
单可用区	仅支持单可用区的网络结构场景，容灾能力较低。

🔗 产品计费模式

BLB产品具有如下几种收费模式

计费模式	性能限制	适用场景
包年包月计费	根据所选择的规格限制性能，不同规格收费不同	长期，业务波动相对较小的场景
后付费-按使用时长计费	根据所选择的规格限制性能，不同规格收费不同	短期，业务波动相对较小的场景
后付费-按使用量计费	根据所选择的规格限制性能，或不限制性能，根据实际使用量统一收费	业务波动相对较大的场景

使用IP组功能在同地域跨VPC挂载后端服务器

🔗 场景介绍

IP组是应用型BLB的一种后端服务器组类型，指的是使用固定的IP挂载后端服务器。在一些情况下，使用 IP 组跨 VPC 挂载后端服务器可以提供更灵活、安全和可管理的网络架构，适用于复杂的多 VPC 环境和多租户应用场景。 本文档将通过具体的实施案例引领您使用IP组功能，使用对等连接或CSN打通同地域下的两个独立VPC环境。

🔗 限制

此功能仅支持TCP和UDP监听。

🔗 环境准备

- 两个已规划子网的VPC
- 已购买的BLB
- 与BLB同VPC内后端服务器（作为client端访问）
- 已部署完毕的同地域其他VPC内后端服务器
- 已经购买并部署完成的对等连接/CSN产品

假设已部署的环境为如下IP：

资源类型	所处VPC	IP
VPC	VPC1	fwh_192_168_0 (192.168.0.0/24)
VPC	VPC2	默认私有网络 (172.18.0.0/16)
应用型负载均衡实例	VPC1	192.168.0.2
云服务器1（client端）	VPC1	192.168.0.3
云服务器2（后端服务器）	VPC2	172.18.0.6

- 客户端client信息

实例名称/ID	状态	支付方式	创建时间	到期时间	IPv6 IP	IP地址/带宽	弹性网卡	配置/类型	OS
Instance-56k7sior i-t8RC06W	运行中	后付费	2023-05-18 14:24:15	-	240c:4088:0:b801::2	192.168.0.3(内) -/(公)	0	2核/8GB/20GB/通用型g5	

- 后端服务器信息 *在此场景中后端服务器需要选择普四(g4)及以上虚机实例

instance-server	bcc.g5.c2m8	运行中	后付费	2023-05-18 14:15:09	-	-	172.18.0.6(内) +(-公)	0	2核/8GB/20G	详情 VNC连接 更多

● 负载均衡lb信息

实例名称/ID	状态	公网IP/线路类型/带宽	内网IP	IPv6 IP	所在网络	所在子网	服务器组监听协议 / 属
fw_h_two_vpc lb-4fcbde19	运行中	- / - / -	192.168.0.2	-	fw_h_192_168_0	sub_192_168_0	未配置 配置

🔗 挂载后端服务器到负载均衡后端

● 在负载均衡下创建IP组类型的目标组

返回实例列表

实例详情

监听设置

目标组

安全组

监控

日志

fw_h_two_vpc 运行中

服务器组

IP组

+ 添加IP组

删除

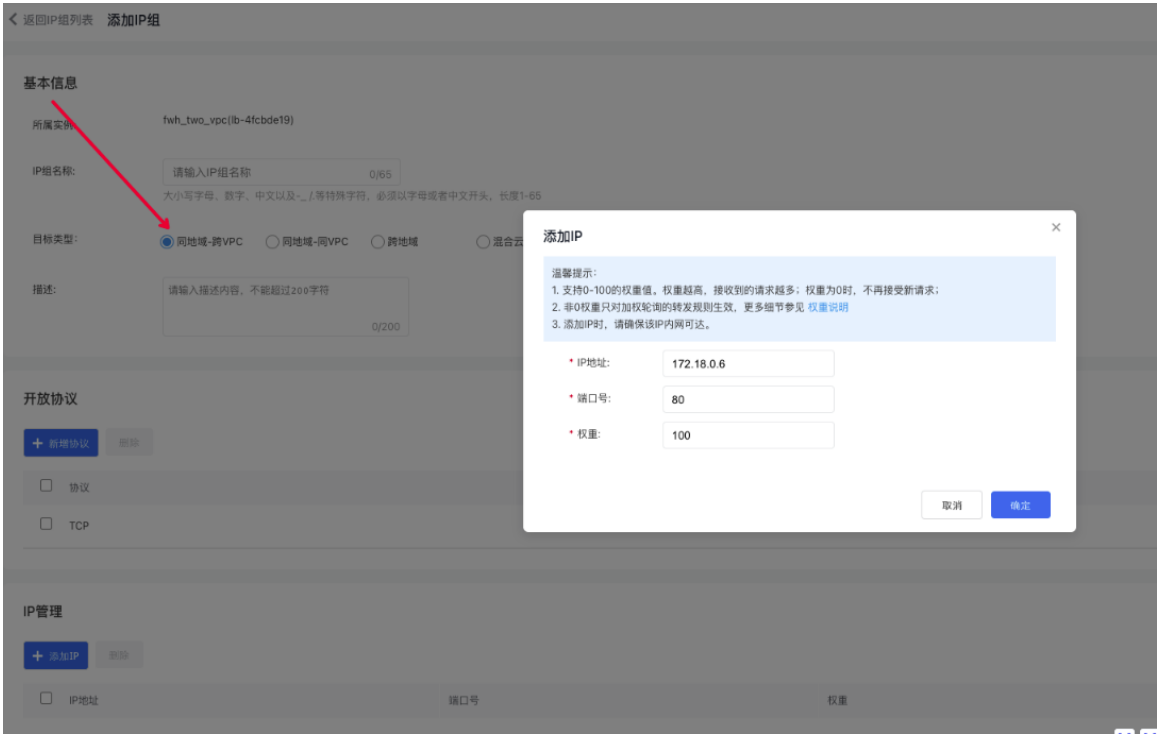
IP组名称/ID	协议/规则绑定数量	IP+端口数量
IG-8cJrZs2y	TCP-1	1
ip_group-654960b9		

已选中0条 / 共1条

● 注：各类型所代表的含义如下。

类型	使用场景
同地域-同VPC：	IP组内添加的IP地址与BLB属于同一个VPC。例如，可以挂载同VPC内的BCC或BBC服务器
同地域-跨VPC：	IP组内添加的IP地址与BLB属于不同的VPC，两个VPC之间能过对等连接等方式互联。例如，可以挂载跨VPC内的BCC或BBC服务器
跨地域：	IP组内添加的IP地址与BLB属于不同的地域，两个VPC之间能过对等连接等方式互联，且IP地址是BCC或BBC类型的计算类型服务器
混合云：	IP组内添加的IP地址与BLB属于同一地域或者不同的地域，例如，可以挂载线下IDC内的服务器，或云上跨地域的服务器IP地址、跨地域有IP地址的网元、跨云可访问的IP地址等
IP自定义：	若选择IP自定义类型，则IP组内的IP成员可以分别配置以上各目标类型

- 挂载172.18.0.6作为IP组成员，目标类型为同地域-跨VPC，并新增开放协议类型为TCP或UDP



- 在负载均衡中创建对应的TCP或UDP监听，并挂载上一步创建的IP组。

使用对等连接或CSN打通VPC1和VPC2环境

对等链接和CSN产品二选一即可，推荐采用CSN产品，减少自定义路由配置。更多关于CSN的使用说明请见：[CSN说明文档](#) 更多关于对等连接的使用说明请见：[对等连接说明文档](#)

使用CSN

通过CSN产品打通fwh_192_168_0 (192.168.0.0/24) 和 默认私有网络 (172.18.0.0/16) 两个vpc网络。



使用对等连接

- 第一步，LB侧VPC需要开启中继路由功能。
- 第二步，第二步，需在BLB所在VPC1的路由配置加上自定义路由：**100.64.0.0/10->X.X.X.X/32**(后端服务器IP地址，或者后端服务器所在子网网段)。比如配置100.64.0.0/10->172.18.0.6/32, 打通健康检查路由连通性。
- 第三步，第三步，检查对等连接中BLB所在VPC1 子网路由配置是否与VPC2子网建立路由条目。如果没有，可以参考["对等连接说明文档"](#)通过对等连接产品中配置添加路由fwh_192_168_0 (192.168.0.0/24) 和 默认私有网络 (172.18.0.0/16) 连接两个VPC网络。

付费及地域

付费方式: 预付费 后付费 [对等连接计费详情>](#)

当前地域: 金融华中 - 武汉

配置信息

所在网络: fwh_192_168_0 (192.168.0.0/24)

本端端口名称:

大小写字母、数字以及-、_、特殊字符, 必须以字母开头, 长度1-65

连接类型: 本帐号 跨帐号

对端地域: 金融华中 - 武汉

对端网络: 默认私有网络

对端端口名称:

大小写字母、数字以及-、_、特殊字符, 必须以字母开头, 长度1-65

计费方式: 按使用流量计费 按使用带宽计费

带宽上限:

1Mbps

5000Mbps

10000Mbps

10000

Mbps

备注:

测试连通性

使用client端访问负载均衡IP 192.168.0.2:80

The image shows a terminal window with two parts. The top part shows a series of curl requests from 192.168.0.3 to 192.168.0.2:80, all returning 200 status codes. The bottom part shows a terminal session on a server (root@instance-56k7sior wrk-master) where the user runs 'ip addr show dev eth0' and 'date; curl 192.168.0.2:80; date'. The output of 'ip addr show dev eth0' shows the interface is up and has an IP of 192.168.0.3/25. The output of the curl command shows a successful GET request to 192.168.0.2:80, returning a 200 status code. A red arrow points from the text above to the curl output in the terminal.

同region跨vpc数据流可通，源IP地址(192.168.0.3)在服务器RS(172.18.0.6)侧可见。

注意事项

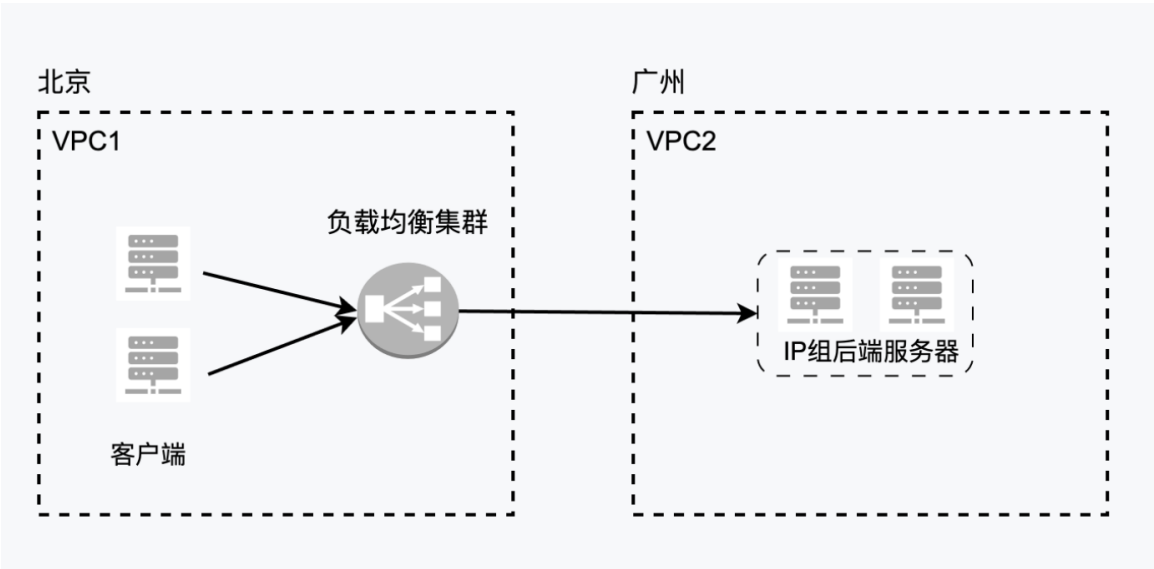
- 路由规则和负载均衡器配置需要遵循客户实际环境网段进行配置，确保可通。
- 各BLB以及后端服务器上的安全组和ACL需要正确配置，避免不必要的拦截丢包，导致访问不通。

使用IP组功能在跨地域VPC挂载后端服务器

场景介绍

IP组是应用型BLB的一种后端服务器组类型，指的是使用固定的IP挂载后端服务器。在一些情况下，使用 IP 组跨 VPC 挂载后端服务器可以提供更灵活、安全和可管理的网络架构，适用于复杂的多 VPC 环境和多租户应用场景。 本文档将通过具体的实施案例引领您使用IP组功能，使用对等连接打通跨地域下的两个独立VPC环境。

151



前提

- 当前仅支持通过对等连接打通的跨地域VPC环境。
- 三个以上VPC通过对等连接互访场景需要提工单支持多个VPC中继路由开放功能。

环境准备

- 两个已规划子网的VPC，一个在北京，一个在广州
- 已购买的BLB
- 已部署完毕的跨其他VPC内后端服务器
- 已经购买并部署完成的对等连接

假设已部署的环境为如下IP：

资源类型	所处VPC	IP
北京VPC	VPC1	bj_vpc_192 (192.168.0.0/16)
广州VPC	VPC2	gz_vpc_10 (10.0.0.0/16)
北京应用型负载均衡实例	VPC1	192.168.0.8
云服务器1 (广州后端服务器1)	VPC2	10.100.100.5
云服务器2 (广州后端服务器2)	VPC2	10.100.200.4

负载均衡实例和虚机

- 负载均衡lb信息

BLB-beijing-48b604f7 ● 运行中

基本信息

ID: lb-09f2bfb6

名称: BLB-beijing-48b604f7 变更

内网IP: 192.168.0.8

描述: - 变更

释放保护: 关

支持ACL: 关

公网IP/带宽: -/Mbps

所在网络: bj_vpc_192 (192.168.0.0/16)

所在子网: 系统预定义子网 (192.168.0.0/20)

创建时间: 2021-04-20 16:06:56

运行状态信息

运行状态: ● 运行中

自动释放时间: - 设置

● 广州后端服务器虚拟机信息

*在此场景中后端服务器需要选择普四(g4)及以上虚拟机实例

+ 创建实例 开启 重启 停止 续费 批量操作

下载选中 下载全部

☐	实例名称/ID	状态	到期时间	IPv6 IP	IP地址/带宽	配置/类型	操作
☐	instance-i7onuhg0 i-ZREJa5Vn	● 运行中 空 启动	2021-09-02	-	10.100.100.5(内) 106.13.192.242/1Mbps(公)	1核/4GB/20GB/通用型g3	详情 VNC远程 更多
☐	instance-2v6h3l58 i-pObMejsj	● 运行中 空 启动	2021-09-02	-	10.100.200.4(内) 106.12.134.236/1Mbps(公)	2核/8GB/20GB/通用型g4	详情 VNC远程 更多

对等连接准备

● 北京端对等连接：

< blb ● 可用

实例详情

监控

基本信息

对等连接ID: peerconn-hgq1jyudykkrr

连接类型: 本帐号

创建时间: 2021-08-02 20:01:14

备注:

本端配置信息

本端地域: 华北 - 北京

本端接口名称: blb

对端配置信息

对端地域: 华南 - 广州

对端账户ID: 516081f68bb3489691834fba5904f477

状态: ● 可用

带宽上限: 1Mbps 带宽升级

连接角色: 发起端

本端VPC ID: vpc-it3tp066cc85

本端接口ID: qpif-mfdw6kwz8229

对端VPC ID: vpc-crflpggi10gy

● 广州端对等连接：

153

<

bcc

● 可用

实例详情

监控

基本信息

对等连接ID：peerconn-hgq1jyudykkr

连接类型：本帐号

创建时间：2021-08-02 20:01:14

备注：[🔗](#)

本端配置信息

本端地域：华南 - 广州

本端接口名称：bcc [🔗](#)

对端配置信息

对端地域：华北 - 北京

对端账户ID：516081f68bb3489691834fba5904f477

状态：● 可用

带宽上限：1Mbps

连接角色：接受端

本端VPC ID：vpc-crfjpggi10gy

本端接口ID：qpif-frn41nezpm3u [🔗](#)

对端VPC ID：vpc-it3tp066cc85

🔄

🔗

🔗 挂载后端服务器到负载均衡后端

- 在负载均衡下创建IP组类型的目标组

☰

百度智能云

控制台总览

🇨🇳 金融华中 - 武汉

< 返回实例列表

fwh_two_vpc ● 运行中

实例详情

监听设置

目标组

安全组

监控

日志

服务器组

IP组

+ 添加IP组

删除

☐ IP组名称/ID	协议/规则绑定数量	IP+端口数量
☐ IG-8cJrZs2y 🔗	TCP-1 🔗 ×	1
☐ ip_group-654960b9 🔗		

已选中0条/ 共1条

< 返回IP组列表

添加IP组

基本信息

所属实例：BLB-beijing-66cdf288(lb-845b1539)

IP组名称： 0/65

大小写字母、数字、中文以及_./等特殊字符，必须以字母或者中文开头，长度1-65

目标类型：

☐ 同地域-同VPC

☐ 同地域-跨VPC

☐ 跨地域

☒ 混合云

☐ IP自定义 [?](#)

描述：

- 注：各类型所代表的含义如下。

类型	使用场景
同地域-同VPC：	IP组内添加的IP地址与BLB属于同一个VPC。例如，可以挂载同VPC内的BCC或BBC服务器
同地域-跨VPC：	IP组内添加的IP地址与BLB属于不同的VPC，两个VPC之间能过对等连接等方式互联。例如，可以挂载跨VPC内的BCC或BBC服务器
跨地域：	IP组内添加的IP地址与BLB属于不同的地域，两个VPC之间能过对等连接等方式互联，且IP地址是BCC或BBC类型的计算类型服务器
混合云：	IP组内添加的IP地址与BLB属于同一地域或者不同的地域，例如，可以挂载线下IDC内的服务器，或云上跨地域的服务器IP地址、跨地域有IP地址的网元、跨云可访问的IP地址等
IP自定义：	若选择IP自定义类型，则IP组内的IP成员可以分别配置以上各目标类型

- 挂载10.100.100.5和10.100.200.4作为IP组成员，目标类型为跨地域，并新增开放协议。

添加IP

温馨提示：

1. 支持0-100的权重值。权重越高，接收到的请求越多；权重为0时，不再接受新请求；

2. 非0权重只对加权轮询的转发规则生效，更多细节参见 [权重说明](#)

3. 添加IP时，请确保该IP内网可达。

4. 暂不支持添加HAVIP的内网IP地址。

* 地址选择:

☐ 单条

☒ 多条

* IP地址:

10.100.200.4

10.100.100.5

* 端口号:

80

* 权重:

100

取消

确定

- 在负载均衡中创建对应的TCP或UDP监听，并挂载上一步创建的IP组。

使用对等连接打通VPC1和VPC2环境并配置路由

更多关于对等连接的使用说明请见：[对等连接说明文档](#) 创建好对应的对等连接后，我们还需要进行如下配置：

1.VPC中继路由功能

BLB所在VPC1需要开启中继路由功能。

2.BLB所在VPC1路由配置

需在BLB所在VPC1的路由配置加上自定义路由: 100.64.0.0/10->X.X.X.X/32(后端服务器IP地址，或者网段)。 在此示例中，我们配置100.64.0.0/10->10.100.100.5/32 和100.64.0.0/10->10.100.200.4/32, 路由指向对等连接端，打通健康检查路由连

155

通性。

<

dvr-route-cdmpozh3

● 可用

基本信息

名称: dvr-route-cdmpozh3 [变更](#)

ID: rt-fbkuufnxas7c [🔗](#)

所在网络: b

路由条目

+ 添加路由

源网段	目标网段	下一跳	路由类型	VPC状态	主路径抢占	CSN发
100.64.0.0/10	10.100.200.4/32	blb	对等连接	● 可用	-	-
100.64.0.0/10	10.100.100.5/32	blb	对等连接	● 可用	-	-
0.0.0.0/0	192.168.0.0/20	local	系统	● 可用	-	-
0.0.0.0/0	192.168.16.0/20	local	系统	● 可用	-	-
0.0.0.0/0	192.168.32.0/20	local	系统	● 可用	-	-
0.0.0.0/0	192.168.48.0/20	local	系统	● 可用	-	-
0.0.0.0/0	192.168.64.0/20	local	系统	● 可用	-	-
0.0.0.0/0	192.168.80.0/20	local	系统	● 可用	-	-

3.后端服务器所在VPC2路由配置

后端服务器所在VPC2内的，后端服务器所使用的路由表添加源网段为VPC2地址10.100.100.0/24和10.100.200.0/24（后端服务器所在地址段），目标网段为负载均衡保留段(100.64.0.0/10), 路由指向对等连接端。

<

default

● 可用

基本信息

名称: default [变更](#)

ID: rt-u79uezkpi8kq [🔗](#)

路由条目

+ 添加路由

源网段	目标网段	下一跳	路由类型	VPC状态	主路径
subnet_200(10.100....	100.64.0.0/10	bcc	对等连接	● 可用	-
gz_vpc_10_subnet(...	100.64.0.0/10	bcc	对等连接	● 可用	-
0.0.0.0/0	10.100.100.0/24	local	系统	● 可用	-
0.0.0.0/0	10.100.200.0/24	local	系统	● 可用	-

备注：

若VPC2不仅与VPC1通过对等连接实现互通，还与位于其他地域的VPC（例如VPC3）建立了连接，并且同时作为这两个不同地域负载均衡（BLB）的后端服务器，那么在配置目标负载均衡地址网段时，需要更精细的路由策略。如有此需求，可以通过提交工单来获取相应的支持。

测试连通性

在VPC1内使用客户端访问负载均衡IP 192.168.0.8:80


```
</body>
</html>
[root@instance-2v6h3l58 ~]# tail -f /var/log/nginx/access.log
100.82.37.245 - - [03/Aug/2021:15:53:50 +0800] "GET / HTTP/1.1" 200 4833 "-" "curl/7.61.1" "-"
100.82.38.249 - - [03/Aug/2021:15:53:52 +0800] "GET / HTTP/1.1" 200 4833 "-" "curl/7.61.1" "-"
100.82.37.244 - - [03/Aug/2021:15:53:53 +0800] "GET / HTTP/1.1" 200 4833 "-" "curl/7.61.1" "-"
127.0.0.1 - - [03/Aug/2021:16:44:26 +0800] "GET / HTTP/1.1" 200 4833 "-" "curl/7.29.0" "-"
192.168.48.5 - - [03/Aug/2021:16:44:47 +0800] "GET / HTTP/1.1" 200 4833 "-" "curl/7.61.1" "-"
```

跨地域跨vpc数据流可通。

如何获取客户端源IP（CIP）

- TCP/UDP类监听在同地域-同VPC、同地域-跨VPC和跨地域场景下获取CIP直接看报文头即可。
- 混合云类型跨地域TCP/UDP类监听获取CIP，如BLB挂挂载线下IDC内的服务器，或云上跨地域的服务器IP地址、跨地域有IP地址的网元、跨云可访问的IP地址等场景，客户端IP地址因多次NAT转换默认不可见。如有此需求，可以通过提交工单来获取相应的支持。
- 目前TCP/UDP/TCPSSL类监听支持 ProxyProtocol 协议，用户可在BLB上启用该选项，同时在后端服务器上解析 ProxyProtocol 标准协议的方式，来获取 CIP。
- HTTP/HTTPS类型的监听，目前暂不支持 CIP 透传，用户可开启『获取真实IP X-Forwarded-For方式获取客户端真实IP』选项，在后端服务器上通过解析 HTTP header 头来获取 CIP。

注意事项

- 在使用专线等回源的过程中，确保两端的VPC路由中继不会产生环路。
- 路由规则和负载均衡器配置需要遵循客户实际环境网段进行配置，确保可通。
- 各BLB以及后端服务器上的安全组和ACL需要正确配置，避免不必要的拦截丢包，导致访问不通。

API参考

简介

百度负载均衡BLB（Baidu Load Balance）是百度智能云对外提供的稳定、安全、高效的负载均衡服务。可以均衡应用流量，实现故障自动切换，消除故障节点，提高业务可用性。BLB对外提供Restful API。本文档对BLB开放的API进行详细描述。目前BLB提供创建/查询/删除类API，之后会发布修改类API。

如果您是初次调用百度智能云产品的API，可以观看[API入门视频指南](#)，快速掌握调用API的能力。

API服务域名

百度智能云目前开通了多区域支持，目前支持“北京”、“广州”、“苏州”、“香港”、“武汉”、“保定”、“上海”、“新加坡”等区域。各区域采用不同的域名，具体域名为blb.{region}.baidubce.com。BLB API服务域名支持HTTP和HTTPS两种调用方式。为了提升数据的安全性，建议通过HTTPS调用。

区域	服务域名	协议
北京	blb.bj.baidubce.com	HTTP/HTTPS
广州	blb.gz.baidubce.com	HTTP/HTTPS
苏州	blb.su.baidubce.com	HTTP/HTTPS
香港	blb.hkg.baidubce.com	HTTP/HTTPS
武汉	blb.fwh.baidubce.com	HTTP/HTTPS
保定	blb.bd.baidubce.com	HTTP/HTTPS
上海	blb.fsh.baidubce.com	HTTP/HTTPS
新加坡	blb.sin.baidubce.com	HTTP/HTTPS

访问控制

百度智能云采用统一的API鉴权认证机制，详情请见[鉴权认证](#)。

接口规范

数据交换格式为JSON，所有request/response body内容均采用UTF-8编码。

说明：所有创建删除等修改操作，后端需要一定时间处理，所以修改类操作设计为异步请求。只要参数合法，后端会直接返回成功，同时BLB实例的状态会变成updating，处于updating状态的BLB实例不能进行其他修改操作，查询操作不受影响。您可以通过调用DescribeLoadBalancers查看实例状态，当实例状态更新为available时可以继续发送后续修改操作。

请求参数

请求参数包括如下4种：

参数类型	说明
URI	通常用于指明操作实体，如:PUT /v1/blb/{blb_id}
Query String	URL中携带的请求参数
Header	通过HTTP头域传入，如x-bce-date
Request Body	通过JSON格式组织的请求数据体

返回值说明

返回值分为两种形式：

返回内容	说明
HTTP Status Code	如200,400,403,404等
Response Body	JSON格式组织的响应数据体

公共请求头

下表列出了所有BLB API所携带的公共头域。HTTP协议的标准头域不在此处列出

头域 (HEADER)	类型	是否必须	说明
Authorization	String	是	签名字符串，生成签名字符串的方法请参考 鉴权认证
Content-Type	String	是	application/json; charset=utf-8
x-bce-date	String	否	签名日期

🔗 公共响应头

下表列出了所有BLB API的公共响应头域。HTTP协议的标准响应头域不在此处列出

头域 (HEADER)	说明
Content-Type	application/json; charset=utf-8
x-bce-request-id	本次请求的requestId

🔗 日期与时间

日期与时间的表示有多种方式。为统一起见，除非是约定俗成或者有相应规范的，凡是HTTP标准中规定的表示日期和时间字段用GMT，其他日期时间表示的地方一律采用UTC时间，遵循ISO 8601，并做以下约束：

- 表示日期一律采用YYYY-MM-DD方式，例如2014-06-01表示2014年6月1日。
- 表示时间一律采用hh:mm:ss方式，并在最后加一个大写字母Z表示UTC时间。例如 23:00:10Z表示UTC时间23点0分10秒。
- 凡涉及日期和时间合并表示时，在两者中间加大写字母T，例如2014-06-01T23:00:10Z表示UTC时间2014年6月1日23点0分10秒。

🔗 幂等性

在调用API时，很容易出现由于网络等问题导致客户端没收到响应连接就中断的情况。此时客户端无法得知服务器端是否收到了请求，重试又可能导致问题。例如一个创建实例的请求被多次发送就可能出现重复创建。对此，应加入幂等性的机制来加以应对。

幂等性的意思是无论同一个请求被重复发送多次，其结果都和发送一次一样。

BCE API采用clientToken机制来保证API调用的幂等性。

🔗 clientToken

clientToken是一个长度不超过64位的ASCII字符串，通常放在query string里面，如http://rds.bj.baidubce.com/v1/instance?clientToken=be31b98c-5e41-4838-9830-9be700de5a20。

clientToken的唯一性与服务及用户相关。不同用户的clientToken互不相关，因此用户无需关注和其他用户的clientToken冲突问题。对于允许匿名调用的API，所有匿名用户视为同一个用户。匿名用户要保证clientToken的唯一性时，应采用随机生成长token的方式，将冲突概率降到足够小。不同服务的clientToken也互不相关。同时一个非全局唯一的服务的不同region之间也可以重复使用clientToken。

clientToken的有效期为至少24小时，以服务端最后一次收到该clientToken为准。也就是说，如果客户端不断发送同一个clientToken，那么该clientToken将长期有效。

🔗 服务端逻辑

当服务端收到带有clientToken的请求时，首先检查发起调用的用户是否曾经发送过同一个clientToken。如是，则应检查API的参数是否完全一致。当不一致时，返回IdempotentParameterMismatch。如完全一致，应返回正常结果。

例如创建一个实例时，通常会返回实例状态。如果clientToken相同，则不再重复创建实例，直接返回对应实例当前状态。

这里的API参数特指会对结果产生影响的url、query string和header等。某些不产生影响的部分如Authorization不包含在内。

🔗 结果分页

查询列表类接口通常会有分页需求。当API返回结果过多时，应限制返回数量，采用分页机制来获取全列表。
DescribeTCPListeners接口。所有分页统一采用marker机制。

- maxKeys：每页所包含查询结果的最大数量。
- marker：查询请求里设置的查询标志位，用来标记查询的起始位置。第一次查询请求时无需设置marker，如果发现一页没列举完，再把服务端返回的nextMarker设为下次查询的marker。
- isTruncated：true表示后面还有数据，false表示已经是最后一页。
- nextMarker：获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现。

错误码

当用户访问BLB出现错误时，BLB会返回给用户相应的错误码和错误信息，便于用户定位问题，并做出适当的处理。 请求发生错误时通过Response Body返回详细错误信息，遵循如下格式：

参数名	类型	说明
code	String	错误码
message	String	错误描述
requestId	String	本次请求的requestId

示例：

```
{
  "requestId": "ae2225f7-1c2e-427a-a1ad-5413b762957d",
  "code": "NoSuchKey",
  "message": "The resource you requested does not exist"
}
```

🔗 BCE公共错误码

错误码	错误描述	HTTP状态码	语义
AccessDenied	Access denied.	403 Forbidden	无权限访问对应的资源
InappropriateJSON	The JSON you provided was well-formed and valid, but not appropriate for this operation.	400 Bad Request	请求中的JSON格式正确，但语义上不符合要求。如缺少某个必需项，或者值类型不匹配等。出于兼容性考虑，对于所有无法识别的项应直接忽略，不应该返回这个错误。
InternalServerError	We encountered an internal error. Please try again.	500 Internal Server Error	所有未定义的其他错误。在有明确对应的其他类型的错误时（包括通用的和服务自定义的）不应该使用。
InvalidAccessKeyId	The Access Key ID you provided does not exist in our records.	403 Forbidden	Access Key ID不存在
InvalidHTTPAuth	The HTTP authorization header is invalid. Consult the service documentation for details.	400 Bad Request	Authorization头域格式错误

Header		t	
InvalidHTTRequest	There was an error in the body of your HTTP request.	400 Bad Request	HTTP body格式错误。例如不符合指定的Encoding等
InvalidURI	Could not parse the specified URI.	400 Bad Request	URI形式不正确。例如一些服务定义的关键词不匹配等。对于ID不匹配等问题，应定义更加具体的错误码，例如NoSuchKey。
MalformedJSON	The JSON you provided was not well-formed.	400 Bad Request	JSON格式不合法
InvalidVersion	The API version specified was invalid.	404 Not Found	URI的版本号不合法
OptInRequired	A subscription for the service is required.	403 Forbidden	没有开通对应的服务
PreconditionFailed	The specified If-Match header doesn't match the ETag header.	412 Precondition Failed	ETag不匹配
RequestExpired	Request has expired. Timestamp date is XXX.	400 Bad Request	请求超时
IdempotentParameterMismatch	The request uses the same client token as a previous, but non-identical request.	403 Forbidden	clientToken对应的API参数不一样。
SignatureDoesNotMatch	The request signature we calculated does not match the signature you provided. Check your Secret Access Key and signing method. Consult the service documentation for details.	400 Bad Request	Authorization头域中附带的签名和服务端验证不一致

 BLB业务错误码

错误码	错误描述	HTTP状态码	语义
InstanceNotFound	The specified BLB instance does not exist.	404	指定的LoadBalancer实例不存在
ListenerNotFound	The specified listener does not exist.	404	指定的监听器不存在
BackendServerNotFou nd	The specified backend server does not exist.	404	指定的后端服务器不存在
ListenerAlreadyExist	Listener already exists.	400	要创建的监听器已存在
BackendServerAlready Exist	Backend server already exists.	400	要绑定的后端服务器已存在
LastOperationNotFinis hed	Last operation has not finished.	409	上一个LoadBalancer请求还未处理完成
RealNameAuthenticati onRequired	You need to pass real-name authentication.	403	当前用户未通过实名认证
QuotaExceeded	You have exceeded current quota.	413	BLB数量超过用户配额限制
ServerRequired	You must have at least one bcc, dcc or bbc to create BLB.	404	创建BLB时必须先拥有BCC，DCC或BBC实例
ListenerExceeded	You can create at most 20 listeners per BLB instance.	413	每个BLB实例最多可以创建20个监听器
ServiceBlocked	Your BLB service is blocked, please contact technical support for assistance.	403	BLB服务被封禁，请联系技术支持
InstanceCreationFaile d	Fail to create blb instance.	400	创建BLB实例失败，通常是由于资源不足，用户金额不足等
MissingParameter	A required parameter 'parameterName' is not supplied.	400	该请求缺少必要的参数"parameterName"
InvalidParameter	The parameter 'parameterName' is invalid.	400	"parameterName"参数不合法
EipUnbindFailed	Fail to unbind EIP	400	解绑EIP失败
RedirectPortNotFound	The specified redirect port does not exist.	404	指定的HTTPS监听器不存在
CertificateNotFound	The specified certificate does not exist.	404	指定的证书不存在

普通型BLB接口

普通型BLB实例相关接口

CreateLoadBalancer创建实例

描述

- 创建一个LoadBalancer，返回分配的服务地址及实例ID。
- 付费方式为后付费。
- 需要实名认证，若未通过实名认证可以前往百度智能云官网[控制台](#)中的安全认证下的实名认证中进行认证。

请求结构

```
POST /v{version}/blb?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name": name,
  "desc": desc,
  "vpclId": vpclId,
  "subnetId": subnetId,
  "address": address,
  "type": type,
  "eip": eip,
  "tags": [
    {
      "tagKey": tagKey,
      "tagValue": tagValue
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
name	String	否	RequestBody参数	LoadBalancer的名称，方便记忆。长度1~65个字节，字母开头，可包含字母数字_/.字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	LoadBalancer实例的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文，默认为空
vpclId	String	是	RequestBody参数	LoadBalancer实例所属VPC的ID
subnetId	String	是	RequestBody参数	LoadBalancer实例所属子网的ID
addresses	String	否	RequestBody参数	指定负载均衡实例的私网IP地址，该地址必须包含在子网网段下。
type	String	否	RequestBody参数	指定负载均衡实例的类型，不传默认为普通型，传"ipv6"代表创建ipv6类型的BLB。
eip	String	否	RequestBody参数	绑定已有的eip。取值为eip的IP地址。

			数	
tags	List<TagModel>	否	RequestBody参数	待创建的标签键值对列表
billing	BillingFormCreate	否	RequestBody参数	订单相关信息，购买预付费blb的时候必填
performanceLevel	String	否	RequestBody参数	性能规格参数，默认为共享型。取值如下："small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型1，"large1"超大型1，"large2"超大型2，"large3"超大型3,如果是后付费按使用量计费的时候还可以传"unlimited"不限速
autoRenewLength	int	否	RequestBody参数	支持创建预付费BLB同时开通自动续费，根据autoRenewTimeUnit的取值有不同的范围，month 为1到9，year 为1到3
autoRenewTimeUnit	String	否	RequestBody参数	支持创建BLB同时开通自动续费，取值为 month 获 year （默认 month）
resourceGroupId	String	否	RequestBody参数	支持创建BLB同时绑定资源分组id
allowDelete	Boolean	否	RequestBody参数	是否允许删除。缺省值为true，代表允许删除
allocateIpv6	Boolean	否	RequestBody参数	是否分配ipv6地址。true代表分配ipv6地址，false代表不分配ipv6地址

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
address	string	分配的内网服务地址IP，通过这个IP即能通过内网访问该实例
name	string	LoadBalancer的名称
blbId	string	LoadBalancer的ID。后续针对该实例的操作，均需要在请求中带上此ID
desc	string	LoadBalancer的描述

请求示例


```
POST /v1/blb?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization: authorization string
```

创建后付费blb实例

```
{
  "name": "blb-for-test",
  "desc": "myblb",
  "vpclId": "vpc-fy6vdykpwkqb",
  "subnetId": "sbn-a34td0pxcucd",
  "address": "192.168.0.24",
  "eip": "100.88.8.139"
}
```

创建预付费blb实例

```
{
  "vpclId": "vpc-fbsxgfs9ty60",
  "subnetId": "sbn-3317rkjfq8j9",
  "name": "ccq20240223_ccq",
  "billing": {
    "paymentTiming": "Prepaid",
    "reservation": {
      "reservationLength": 2
    }
  },
  "performanceLevel": "small1"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "blbId": "lb-BLuOPSLZ",
  "name": "blb-for-test",
  "address": "192.168.0.24",
  "desc": "myblb"
}
```

DescribeLoadBalancers 查询已有的BLB实例

描述

- 查询用户账户下所有LoadBalancer的信息
- 支持按LoadBalancer的Name、Address进行模糊查询
- 支持按LoadBalancer的ID进行精确查询
- 支持按bcclId进行查询，目前只支持单个bcclId查询，并且仅支持精确查询。查询结果为包含该后端服务器的负载均衡实例。
- 返回结果是多重查询条件的交集
- 如果BLB实例开启了公网访问，返回结果将会带上公网IP
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/blb?address={address}&name={name}&blbld={blbld}&bccld={bccld}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
address	String	否	Query参数	要查询的LoadBalancer内网服务地址，格式为点分十进制
name	String	否	Query参数	要查询的LoadBalancer名称
blbld	String	否	Query参数	要查询的LoadBalancer ID（只支持精确查询）
bccld	String	否	Query参数	绑定该BCC的LoadBalancer（只支持精确查询）
exactlyMatch	Boolean	否	Query参数	为true表示查询条件为精准匹配，否则为模糊匹配。缺省值为false
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000
type	String	否	Query参数	指定负载均衡实例的类型，不传默认为普通型，传"ipv6"代表查询ipv6类型的BLB。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
blbList	List< BLBModel >	包含查询结果的列表
marker	string	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/blb?name=test&maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "blbList": [
    {
      "blbId": "lb-a7e5zPPk",
      "status": "available",
      "name": "test-blb",
      "desc": "desc",
      "address": "192.168.0.23"
    },
    {
      "blbId": "lb-gj5gVpeq",
      "status": "available",
      "name": "nametest",
      "desc": "desc",
      "address": "192.168.0.24",
      "publicIp": "123.2.3.4"
    }
  ],
  "marker": "blb-0A20F971",
  "nextMarker": "blb-0A20FB09",
  "isTruncated": true,
  "maxKeys": 2
}
```

DescribeLoadBalancerDetail 查询BLB实例详情

描述

- 按ID查询用户账户下所有LoadBalancer的详细信息
- 包含LoadBalancer所有的监听器端口信息

请求结构

```
GET /v{version}/blb/{blbId}?type={type} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	要查询的LoadBalancer的标识符
type	String	否	Query参数	指定查询负载均衡实例的类型，不传默认为普通型，传"ipv6"代表查询ipv6类型的BLB。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
blbId	String	LoadBalancer的标识符
status	blbStatus	BLB状态
name	String	LoadBalancer的名称
desc	String	LoadBalancer的描述
address	String	LoadBalancer的内网地址
publicIp	String	LoadBalancer的公网地址
cidr	String	LoadBalancer所在网络cidr
vpcName	String	LoadBalancer所属vpc名称
vpId	String	LoadBalancer所属vpId
createTime	String	LoadBalancer创建时间
releaseTime	String	LoadBalancer自动释放时间
listener	List<ListenerModel>	LoadBalancer下挂载监听器列表
tags	List<TagModel>	标签键值对列表
allowDelete	Boolean	是否允许删除
paymentTiming	String	付款时间，预支付（Prepaid）或后支付（Postpaid）
billingMethod	String	计费方式，"ByCapacityUnit" 按使用量 "BySpec" 按固定规格
performanceLevel	String	性能规格。"small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型1，"large1"超大型1，"large2"超大型2，"large3"超大型3
expireTime	String	预付费实例的过期时间
eipRouteType	String	EIP线路类型
publicIpv6	String	如果LoadBalancer绑定过EIPv6，则显示该项，否则不显示
eipV6RouteType	String	EIPv6线路类型

请求示例

```
GET /v1/blb/lb-7fb1e926 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "blbId": "lb-gj5gVpeq",
  "status": "available",
  "name": "nametest",
  "desc": "",
  "address": "192.168.0.2",
  "publicIp": "123.2.3.4",
  "cidr": "192.168.0.0/16",
  "vpcName": "test",
  "createTime": "2019-03-07T02:35:31Z",
  "releaseTime": "2019-03-07T02:35:31Z",
  "paymentTiming": "Postpaid",
  "billingMethod": "ByCapacityUnit",
  "listener": [
    {
      "port": 80,
      "backendPort": 90,
      "type": "TCP",
      "healthCheckType": "TCP",
      "healthCheckStatus": "Alive"
    }
  ]
}
```

UpdateLoadBalancer更新实例

描述

更新指定的单个LoadBalancer配置信息，包括LoadBalancer实例的名称和描述。

请求结构

```
PUT /v{version}/blb/{blbId}?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name": name,
  "desc": desc
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	待更新的LoadBalancer的ID
name	String	否	RequestBody参数	LoadBalancer的名称，方便记忆。长度1~65个字节，字母开头，可包含字母数字-_/。字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	LoadBalancer实例的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空
allowDelete	Boolean	否	RequestBody参数	是否允许删除。缺省值为true，代表允许删除
allocateIpv6	Boolean	否	RequestBody参数	是否分配ipv6地址。true代表分配ipv6地址，false代表不分配ipv6地址

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "name": "blb-for-test",
  "desc": "用于测试"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

 DeleteLoadBalancer释放BLB实例

描述

- 释放指定LoadBalancer，被释放的LoadBalancer无法找回

请求结构

```
DELETE /v{version}/blb/{blbId}?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	待释放的LoadBalancer的ID
clientToken	String	是	Query参数	幂等性Token，长度不超过64位

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
DELETE /v1/blb/lb-j8UqjYf?clientToken=aabbccdd HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 07:09:45 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateLoadBalancerAcl更新实例ACL功能

描述

更新指定的LoadBalancer是否支持ACL功能，支持更新类型（普通型、应用型、IPv6型）

请求结构

```
PUT /v{version}/blb/acl/{blbId}?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "supportAcl": supportAcl
}
```


请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	待更新的LoadBalancer的ID
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
supportAcl	Boolean	否	RequestBody参数	是否支持ACL，缺省值为true，代表支持

返回状态码

成功返回200，失败返回见[错误码](#)。

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/acl/lb-9f59e197?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2021-09-10T10:00:00Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "supportAcl":true
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 10 Sep 2021 10:00:10 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

普通型BLB监听器相关接口

CreateTCPListener创建TCP监听器

描述

- 在指定LoadBalancer下，创建一个基于TCP协议的监听器，监听一个前端端口，将发往该端口的所有TCP流量，转发到后端服务器监听的后端端口上。TCP监听器只支持TCP协议的健康检查，检查端口和后端服务器监听端口一致。

请求结构

```
POST /v{version}/blb/{blbld}/TCPListener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "backendPort":backendPort,
  "scheduler":scheduler,
  "tcpSessionTimeout":tcpSessionTimeout,
  "healthCheckType":healthCheckType,
  "healthCheckTimeoutInSeconds":healthCheckTimeoutInSeconds,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
backendPort	int	是	RequestBody参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"WeightLeastConn"代表加权最小连接、"Hash"代表根据源ip哈希
tcpSessionTimeout	int	否	RequestBody参数	tcp设置连接超时时间（单位：秒），默认900，需为10-4000间的整数
healthCheckType	String	否	RequestBody参数	健康检查协议类型，默认"TCP"
healthCheckTimeoutInSeconds	int	否	RequestBody参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckInterval	int	否	RequestBody参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	RequestBody参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	RequestBody参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/blb/lb-j8UqjYf/TCPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "backendPort":95,
  "scheduler":"LeastConnection",
  "tcpSessionTimeout":900,
  "healthCheckType":"TCP",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "healthyThreshold":3
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateUDPListener创建UDP监听器

描述

- 在指定LoadBalancer下，创建一个基于UDP协议的监听器，监听一个前端端口，将发往该端口的所有UDP流量，转发到后端服务器监听的后端端口上。UDP监听器支持UDP、ICMP协议的健康检查，检查端口和后端服务器监听端口一致。

请求结构

```
POST /v{version}/blb/{blbId}/UDPlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "backendPort":backendPort,
  "scheduler":scheduler,
  "healthCheckType":healthCheckType,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold,
  "healthCheckString":healthCheck,
  "udpSessionTimeout":udpSessionTimeout
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
backendPort	int	是	RequestBody参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"WeightLeastConn"代表加权最小连接、"Hash"代表根据源ip哈希
healthCheckType	String	否	RequestBody参数	健康检查协议类型，支持"UDP"/"ICMP"，默认"UDP"
healthCheckString	String	否	RequestBody参数	健康发送的请求字符串，后端服务器收到后需要进行应答，支持16进制\00-\xFF和标准ASCII字符串，最大长度1299，健康检查协议为"UDP"时必须传
healthCheckTimeoutInSecond	int	否	RequestBody参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckInterval	int	否	RequestBody参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	RequestBody参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	RequestBody参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
udpSessionTimeout	int	否	RequestBody参数	udp会话超时时间。默认为90，需为5-4000间的整数，单位秒

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/blb/lb-j8UqijYf/UDPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckType":"UDP",
  "healthCheckString":"healthCheck",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "healthyThreshold":3,
  "udpSessionTimeout":90
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateHTTPListener创建HTTP监听器

描述

- 在指定LoadBalancer下，创建一个基于HTTP协议的监听器，监听一个前端端口，将发往该端口的所有HTTP请求，转发到后端服务器监听的后端端口上

请求结构

```
POST /v{version}/blb/{blbld}/HTTPlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "backendPort":backendPort,
  "scheduler":scheduler,
  "keepSession":keepSession,
  "keepSessionType":keepSessionType,
  "keepSessionDuration":keepSessionDuration,
  "keepSessionCookieName":keepSessionCookieName,
  "xForwardFor":xForwardFor,
  "xForwardedProto":xForwardedProto
  "healthCheckType":healthCheckType,
  "healthCheckPort":healthCheckPort,
  "healthCheckURI":healthCheckURI,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold,
  "healthCheckNormalStatus":healthCheckNormalStatus,
  "healthCheckHost":healthCheckHost,
  "serverTimeout":serverTimeout,
  "redirectPort":redirectPort
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	是	Request Body参数	监听器的监听端口，需为1-65535间的整数
backendPort	int	是	Request Body参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	是	Request Body参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	bool	否	Request Body参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	Request Body参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionDur	int	否	Request Body参	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默

ation			数	认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	Request Body参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardFor	bool	否	Request Body参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	bool	否	Request Body参数	将监听使用的协议通过 x-forwarded-proto HTTP Header 转发给后端服务器
healthCheckType	String	否	Request Body参数	健康检查协议，支持"HTTP"/"TCP"，默认"HTTP"
healthCheckPort	int	否	Request Body参数	健康检查端口，默认为backendPort
healthCheckURI	String	否	Request Body参数	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckTimeoutInSecond	int	否	Request Body参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckInterval	int	否	Request Body参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	Request Body参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	Request Body参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	否	Request Body参数	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckHost	String	否	Request Body参数	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
serverTimeout	int	否	Request Body参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
redirectPort	int	否	Request Body参数	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/blb/lb-j8UqijYf/HTTPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":95,
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckTimeoutInSeconds":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "xForwardFor":true,
  "xForwardedProto":true,
  "healthCheckType":"HTTP",
  "serverTimeout":1800,
  "healthCheckNormalStatus":"http_1xx|http_2xx",
  "healthCheckHost":"localhost",
  "redirectPort":80
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateHTTPSListener创建HTTPS监听器

描述

在指定LoadBalancer下，创建一个基于HTTPS协议的监听器，监听一个前端端口，将发往该端口的所有HTTPS请求，先通过SSL卸载转换为HTTP请求后，再转发到后端服务器监听的后端端口上。

请求结构

```
POST /v{version}/blb/{blbld}/HTTPSListener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "backendPort":backendPort,
  "scheduler":scheduler,
  "keepSession":keepSession,
  "keepSessionType":keepSessionType,
  "keepSessionDuration":keepSessionDuration,
  "keepSessionCookieName":keepSessionCookieName,
  "xForwardFor":xForwardFor,
  "xForwardedProto":xForwardedProto
  "healthCheckType":healthCheckType,
  "healthCheckPort":healthCheckPort,
  "healthCheckURI":healthCheckURI,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold,
  "healthCheckNormalStatus":healthCheckNormalStatus,
  "healthCheckHost":healthCheckHost,
  "serverTimeout":serverTimeout,
  "certIds":[certId],
  "encryptionType":encryptionType,
  "encryptionProtocols":[protocol1, protocol2],
  "dualAuth":false,
  "clientCertIds":[clientCertId],
  "additionalCertDomains":[{"certId":certId,"host":host}]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
backendPort	int	是	RequestBody参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数

keepSession	boolean	否	RequestBody参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	RequestBody参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionDuration	int	否	RequestBody参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	int	否	RequestBody参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardFor	boolean	否	RequestBody参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	RequestBody参数	将监听使用的协议通过x-forwarded-proto HTTP Header 转发给后端服务器
healthCheckType	String	否	RequestBody参数	健康检查协议，支持"HTTP"/"TCP"，默认"HTTP"
healthCheckPort	int	否	RequestBody参数	健康检查端口，默认为backendPort
healthCheckURI	String	否	RequestBody参数	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckTimeoutInSecond	int	否	RequestBody参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckInterval	int	否	RequestBody参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	RequestBody参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	RequestBody参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	否	RequestBody参数	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckHost	String	否	RequestBody参数	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
serverTime		否	RequestBody参数	后端服务器是否支持返回时间戳，默认为false，需为true或false的布尔值

out	int	否	tBody参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
certIds	List<String>	是	RequestBody参数	监听器要加载的证书,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
encryptionType	String	否	RequestBody参数	加密选项，默认tls_cipher_policy_default。支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCertIds	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链，当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
additionalCertDomains	List< AdditionalCertDomain >	否	RequestBody参数	扩展域名

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

```
POST /v1/blb/lb-j8UqijYf/HTTPSlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "healthyThreshold":3,
  "xForwardFor":true,
  "xForwardedProto": true,
  "healthCheckType":"HTTP",
  "serverTimeout":1800,
  "healthCheckNormalStatus":"http_1xx|http_2xx",
  "healthCheckHost":"localhost",
  "certIds":["cert-zfj2ey2z4kmm"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateSSLListener创建SSL监听器

描述

- 在指定LoadBalancer下，创建一个基于SSL协议的监听器，监听一个前端端口，将发往该端口的所有SSL流量，通过SSL卸载转换为TCP请求后转发到后端服务器监听的后端端口上。SSL监听器只支持TCP协议的健康检查，检查端口和后端服务器监听端口一致。

请求结构

```
POST /v{version}/blb/{blbld}/SSLlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "backendPort":backendPort,
  "scheduler":scheduler,
  "healthCheckType":healthCheckType,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold,
  "certIds":[certId],
  "encryptionType":encryptionType,
  "encryptionProtocols":[protocol1, protacol2],
  "dualAuth":false,
  "clientCertIds":[clientCertId]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
backendPort	int	是	RequestBody参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
healthCheckType	String	否	RequestBody参数	健康检查协议类型，默认"TCP"
tcpSessionTimeout	int	否	RequestBody参数	tcp设置连接超时时间（单位：秒），默认900，需为10-4000间的整数
healthCheckTimeoutInSecond	int	否	RequestBody参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数

healthCheckInterval	int	否	RequestBody参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	RequestBody参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	RequestBody参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
certIds	List<String>	是	RequestBody参数	监听器要加载的证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
encryptionType	String	否	RequestBody参数	加密选项，默认tls_cipher_policy_default。支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCertIds	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/blb/lb-j8UqijYf/SSLlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckType":"TCP",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "healthyThreshold":3,
  "certIds":["cert-zfj2ey2z4kmm"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeTCPListeners查询TCP监听器

描述

- 查询指定LoadBalancer下所有TCP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/blb/{blbId}/TCPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< TCPLListenerModel >	包含查询结果的列表
marker	string	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/blb/lb-j8UqijYf/TCPlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "listenerList": [
    {
      "listenerPort": 80,
      "backendPort": 80,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3
    },
    {
      "listenerPort": 88,
      "backendPort": 88,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 2,
      "healthCheckInterval": 4,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeUDPListeners查询UDP监听器

描述

- 查询指定LoadBalancer下所有UDP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/blb/{blbId}/UDPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< UDPListenerModel >	包含查询结果的列表
marker	string	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/blb/lb-j8UqijYf/UDPlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "listenerList": [
    {
      "listenerPort": 80,
      "backendPort": 80,
      "scheduler": "RoundRobin",
      "healthCheckType": "UDP",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3,
      "healthCheckString": "healthCheck",
      "udpSessionTimeout": 90
    },
    {
      "listenerPort": 88,
      "backendPort": 88,
      "scheduler": "RoundRobin",
      "healthCheckType": "UDP",
      "healthCheckTimeoutInSecond": 2,
      "healthCheckInterval": 4,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3,
      "healthCheckString": "healthCheck",
      "udpSessionTimeout": 90
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeHTTPListeners查询HTTP监听器

描述

- 查询指定LoadBalancer下所有HTTP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/blb/{blbId}/HTTPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< HTTPListenerModel >	包含查询结果的列表
marker	string	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/blb/lb-j8UqijYf/HTTPlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

响应示例

```

HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

```

```

{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 90,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_2xx|http_3xx",
      "serverTimeout": 30,
      "redirectPort": 80
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 95,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_1xx|http_2xx",
      "serverTimeout": 1800,
      "redirectPort": 80
    }
  ],
  "marker": "listener-005A",
  "nextMarker": "listener-0322",
  "isTruncated": true,
  "maxKeys": 2
}

```

DescribeHTTPSListeners 查询HTTPS监听器

描述

- 查询指定LoadBalancer下所有HTTPS监听器的信息

- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/blb/{blbId}/HTTPSListener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< HTTPSListenerModel >	包含查询结果的列表
marker	string	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/blb/lb-j8UqijYf/HTTPSListener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

响应示例

HTTP/1.1 200 OK x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09 Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked Content-Type: application/json;charset=UTF-8 Server: BWS

```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 90,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_2xx|http_3xx",
      "serverTimeout": 30,
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 95,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_1xx|http_2xx",
      "serverTimeout": 1800,
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-005A",
  "nextMarker": "listener-0322",
  "isTruncated": true,
  "maxKeys": 2
}
```

DescribeSSLListeners查询SSL监听器

描述

- 查询指定LoadBalancer下所有SSL监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/blb/{blbId}/SSLlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< SSLListenerModel >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/blb/lb-j8UqijYf/SSLlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例


```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "listenerList": [
    {
      "listenerPort": 80,
      "backendPort": 80,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3
    },
    {
      "listenerPort": 88,
      "backendPort": 88,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 2,
      "healthCheckInterval": 4,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

 DescribeAllListeners查询所有监听

请求结构

```
GET /v{version}/blb/{blbId}/listener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量通常不超过1000，缺省值为1000

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
marker	String	标记查询的起始位置
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量
listenerList	List<AllListenerModel>	包含查询的监听列表

请求示例

```
GET /v1/blb/lb-a7942975/listener HTTP/1.1
Host: blb.bj.baidubce.com
ContentType: application/json
Authorization: bce auth v1/f81d3b34e48048fbb2634dc7882d7e21/2017 03
11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7 4ad5 451d 9215 71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content Type: application/json;charset=UTF 8
Server: BWS

{
  "nextMarker": "",
  "marker": "",
  "maxKeys": 1,
  "isTruncated": true,
  "listenerList": [
    {
      "listenerPort": 53,
      "backendPort": 53,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3,
      "getBblp": false,
      "listenerType": "UDP",
      "udpSessionTimeout": 90,
      "healthCheckString":
"\00\01\01\00\00\01\00\00\00\00\00\00\05baidu\03com\00\00\01\00\01",
      "healthCheckType": "UDP"
    },
    {
      "listenerPort": 82,
      "backendPort": 80,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3,
      "getBblp": false,
```

```

    "listenerType": "TCP",
    "tcpSessionTimeout": 900
  },
  {
    "listenerPort": 87,
    "backendPort": 80,
    "scheduler": "RoundRobin",
    "healthCheckTimeoutInSecond": 3,
    "healthCheckInterval": 3,
    "unhealthyThreshold": 3,
    "healthyThreshold": 3,
    "getBlbIp": false,
    "listenerType": "HTTP",
    "keepSession": false,
    "xForwardFor": true,
    "healthCheckType": "HTTP",
    "healthCheckPort": 80,
    "healthCheckURI": "/",
    "healthCheckNormalStatus": "http_2xx|http_3xx",
    "healthCheckHost": "",
    "serverTimeout": 30,
    "redirectPort": 0
  },
  {
    "listenerPort": 443,
    "backendPort": 80,
    "scheduler": "RoundRobin",
    "healthCheckTimeoutInSecond": 3,
    "healthCheckInterval": 3,
    "unhealthyThreshold": 3,
    "healthyThreshold": 3,
    "getBlbIp": false,
    "listenerType": "HTTPS",
    "keepSession": false,
    "xForwardFor": true,
    "healthCheckType": "HTTP",
    "healthCheckPort": 80,
    "healthCheckURI": "/",
    "healthCheckNormalStatus": "http_2xx|http_3xx",
    "healthCheckHost": "",
    "serverTimeout": 30,
    "certIds": [
      "cert-gs8bktrm7drp"
    ],
    "dualAuth": false,
    "encryptionType": "tls_cipher_policy_1_2_secure",
    "encryptionProtocols": [
      "tls12"
    ],
    "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA"
  },
  {
    "listenerPort": 446,
    "backendPort": 80,
    "scheduler": "RoundRobin",
    "healthCheckTimeoutInSecond": 3,
    "healthCheckInterval": 3,
    "unhealthyThreshold": 3,
    "healthyThreshold": 3,

```

```
"getBlbIp": false,
"listenerType": "SSL",
"certIds": [
  "cert-gs8bktrm7drp"
],
"dualAuth": true,
"clientCertIds": [
  "cert-0x037gwe4fmg"
],
"encryptionType": "tls_cipher_policy_default",
"encryptionProtocols": [
  "tlsv10",
  "tlsv11",
  "tlsv12"
],
"appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA"
}
]
}
```

UpdateTCPListener更新TCP监听器

描述

更新指定LoadBalancer下的TCP监听器参数，请求参数中所有指定的字段都会被更新，未指定的字段保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/blb/{blbId}/TCPListener?listenerPort={listenerPort}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendPort":backendPort,
  "scheduler":scheduler,
  "healthCheckType":healthCheckType,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
backendPort	int	否	RequestBody参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	否	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
healthCheckType	String	否	RequestBody参数	健康检查协议类型，支持"TCP"
healthCheckTimeoutInSeconds	int	否	RequestBody参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckInterval	int	否	RequestBody参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	RequestBody参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	RequestBody参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/TCPlistener?listenerPort=80&clientToken=be31b98c-5e41-4838-9830-9be700de5a20
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateUDPListener更新UDP监听器

描述

- 更新指定LoadBalancer下的UDP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/blb/{blbId}/UDPListener?listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendPort":backendPort,
  "scheduler":scheduler,
  "healthCheckType":healthCheckType,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold,
  "healthCheckString":"healthCheck"
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	是	Query参数	监听器的监听端口，需为1-65535间的整数
backendPort	int	否	RequestBody参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	否	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
healthCheckType	String	否	RequestBody参数	健康检查协议类型，支持"UDP"/"ICMP"
healthCheckTimeoutInSeconds	int	否	RequestBody参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckInterval	int	否	RequestBody参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	RequestBody参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	RequestBody参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckString	String	否	RequestBody参数	健康发送的请求字符串，后端服务器收到后需要进行应答，支持标准转义如\00、\xf2，方便配置二进制格式请求，健康检查协议为"UDP"时必传
udpSessionTimeout	int	否	RequestBody参数	udp会话超时时间。默认为90，需为5-4000间的整数，单位秒

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/blb/lb-j8UqijYf/UDPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "healthCheckString":"healthCheck",
  "udpSessionTimeout": 90
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateHTTPListener更新HTTP监听器

描述

更新指定LoadBalancer下的HTTP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/blb/{blbId}/HTTPListener?listenerPort={listenerPort}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendPort":backendPort,
  "scheduler":scheduler,
  "keepSession":keepSession,
  "keepSessionType":keepSessionType,
  "keepSessionDuration":keepSessionDuration,
  "keepSessionCookieName":keepSessionCookieName,
  "xForwardFor":xForwardFor,
  "xForwardedProto":xForwardedProto,
  "healthCheckType":healthCheckType,
  "healthCheckPort":healthCheckPort,
  "healthCheckURI":healthCheckURI,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold,
  "healthCheckNormalStatus":healthCheckNormalStatus,
  "healthCheckHost":healthCheckHost,
  "serverTimeout":serverTimeout,
  "redirectPort":redirectPort
}
```


请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
backendPort	int	否	Request Body参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	否	Request Body参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	bool	否	Request Body参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	Request Body参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionDuration	int	否	Request Body参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	Request Body参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardFor	bool	否	Request Body参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	bool	否	Request Body参数	将监听使用的协议通过x-forwarded-proto HTTP Header 转发给后端服务器
healthCheckType	String	否	Request Body参数	健康检查协议，支持"HTTP"/"TCP"
healthCheckPort	int	否	Request Body参数	健康检查端口，默认为backendPort
healthCheckURI	String	否	Request Body参数	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckTimeoutInSecond	int	否	Request Body参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数

			xx	
healthCheckInterval	int	否	Request Body参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	Request Body参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	Request Body参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	否	Request Body参数	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckHost	String	否	Request Body参数	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
serverTimeout	int	否	Request Body参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
redirectPort	int	否	Request Body参数	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/HTTPListener?listenerPort=8080&clientToken=be31b98c-5e41-4838-9830-9be700de5a20
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "xForwardFor":true,
  "xForwardedProto":true,
  "healthCheckType":"HTTP",
  "serverTimeout":1800,
  "healthCheckNormalStatus":"http_1xx|http_2xx",
  "healthCheckHost":"localhost"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateHTTPSListener更新HTTPS监听器

描述

更新指定LoadBalancer下的HTTPS监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/blb/{blbId}/HTTPSListener?listenerPort={listenerPort}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendPort":backendPort,
  "scheduler":scheduler,
  "keepSession":keepSession,
  "keepSessionType":keepSessionType,
  "keepSessionDuration":keepSessionDuration,
  "keepSessionCookieName":keepSessionCookieName,
  "xForwardFor":xForwardFor,
  "xForwardedProto":xForwardedProto,
  "healthCheckType":healthCheckType,
  "healthCheckPort":healthCheckPort,
  "healthCheckURI":healthCheckURI,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold,
  "healthCheckNormalStatus":healthCheckNormalStatus,
  "healthCheckHost":healthCheckHost,
  "serverTimeout":serverTimeout,
  "certIds":[certId],
  "additionalCertDomains":[{"certId":certId,"host":host}]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
backendPort	int	否	RequestBody参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	否	RequestBody参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	bool	否	RequestBody参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	RequestBody参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
			RequestBody参数	会话保持的cookie名称，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"

keepSessionDuration	int	否	tBody参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	RequestBody参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardFor	bool	否	RequestBody参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	bool	否	RequestBody参数	将监听使用的协议通过x-forwarded-proto HTTP Header 转发给后端服务器
healthCheckType	String	否	RequestBody参数	健康检查协议，支持"HTTP"/"TCP"
healthCheckPort	int	否	RequestBody参数	健康检查端口，默认为backendPort
healthCheckURI	String	否	RequestBody参数	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckTimeoutInSecond	int	否	RequestBody参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckInterval	int	否	RequestBody参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
unhealthyThreshold	int	否	RequestBody参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	RequestBody参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	否	RequestBody参数	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckHost	String	否	RequestBody参数	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
serverTimeout	int	否	RequestBody参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
certIds	List<int>	否	RequestBody参数	监听器要加载的证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
additionalCertDomains	List<AdditionalCertDomain>	否	RequestBody参数	扩展域名

contains	ain>		数	
encryptionType	String	否	RequestBody参数	加密选项。支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/HTTPSlistener?listenerPort=88&clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "xForwardFor":true,
  "healthCheckType":"HTTP",
  "serverTimeout":1800,
  "healthCheckNormalStatus":"http_1xx|http_2xx",
  "healthCheckHost":"localhost"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 UpdateSSLListener更新SSL监听器

描述

- 更新指定LoadBalancer下的SSL监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/blb/{blbId}/SSLlistener?listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendPort":backendPort,
  "scheduler":scheduler,
  "healthCheckType":healthCheckType,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckInterval":healthCheckInterval,
  "unhealthyThreshold":unhealthyThreshold,
  "healthyThreshold":healthyThreshold,
  "certIds":[certId],
  "encryptionType":encryptionType,
  "encryptionProtocols":[protocol1, protocol2],
  "dualAuth":false,
  "clientCertIds":[clientCertId]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
backendPort	int	否	RequestBody参数	后端服务器的监听端口，需为1-65535间的整数
scheduler	String	否	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
healthCheckType	String	否	RequestBody参数	健康检查协议类型，支持"TCP"
healthCheckTimeoutInSecond	int	否	RequestBody参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckInterval	int	否	RequestBody参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数

unhealthyThreshold	int	否	RequestBody参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthyThreshold	int	否	RequestBody参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
certIds	List<int>	否	RequestBody参数	监听器要加载的证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
encryptionType	String	否	RequestBody参数	加密选项，支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCertIds	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/blb/lb-j8UqijYf/SSLlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "backendPort":95,
  "scheduler":"LeastConnection",
  "healthCheckTimeoutInSecond":4,
  "healthCheckInterval":5,
  "unhealthyThreshold":2,
  "certIds":["cert-zfj2ey2z4kmm"]
}
```


响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DeleteListeners 释放监听器

描述

- 释放指定LoadBalancer下的监听器，监听器通过监听端口来指定
- 支持批量释放

请求结构

```
PUT /v{version}/blb/{blbId}/listener?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

```
{
  "portList":[port1,port2,...]
}
```

或者

```
{
  "portTypeList":[
    {
      "port":port,
      "type":"type"
    },
    {
      "port":port,
      "type":"type"
    },
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	URL参数	batchdelete
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
portList	List<int>	否	Request Body参数	所有待释放的监听器的端口，一起组成一个数组
portTypeList	List<PortTypeModel>	否	Request Body参数	所有待释放的监听器的端口和协议，一起组成一个数组，如果开通了同端口多协议，并且只想删除其中一个协议，使用这个参数。portList和portTypeList至少传一个。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqjYf/listener?batchdelete&clientToken=clienttokentest
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "portList":[80,87]
}
或者
{
  "portTypeList":[
    {
      "port":80,
      "type":"TCP"
    },
    {
      "port":80,
      "type":"UDP"
    },
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

普通型BLB后端服务器相关接口

 AddBackendServers添加后端服务器

描述

- 为指定LoadBalancer添加后端服务器，支持批量添加

请求结构

```
POST /v{version}/blb/{blbld}/backendserver?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendServerList":[
    {
      "instanceId":instanceId,
      "weight":weight
    },...
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
backendServerList	List< BackendServerModel >	是	RequestBody参数	待添加的后端服务器列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/blb/lb-j8UqijYf/backendserver?clientToken=clienttokentest
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendServerList":[
    {
      "instanceId":"i-YfAibl4A",
      "weight":"50"
    },
    {
      "instanceId":"i-jBXABQ8D",
      "weight":"60"
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeHealthStatus查询后端服务器健康状态

描述

- 查询指定LoadBalancer下的后端服务器信息，需要指定监听端口，结果会返回后端服务器的健康状态
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/blb/{blbId}/backendserver?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	是	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
backendServerList	List<BackendServerStatus>	后端服务器状态列表
type	String	监听器类型，值为"TCP"/"HTTP"
listenerPort	int	监听器的监听端口
backendPort	int	后端服务器的监听端口
marker	string	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/blb/lb-j8UqijY/backendserver?listenerPort=88 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "backendServerList": [
    {
      "instanceId": "i-YfAibI4A",
      "weight": 50,
      "status": "Alive"
    },
    {
      "instanceId": "i-jBXABQ8D",
      "weight": 60,
      "status": "Dead"
    }
  ],
  "type": "TCP",
  "listenerPort": 88,
  "backendPort": 88,
  "marker": "rs-0A6BE9BB",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

描述

- 查询指定LoadBalancer下的后端服务器列表
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/blb/{blbId}/backendserver?marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
backendServerList	List< BackendServerModel >	后端服务器列表
marker	string	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/blb/lb-j8UqjY/backendserver HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

响应示例

HTTP/1.1 200 OK x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09 Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked Content-Type: application/json;charset=UTF-8 Server: BWS

```
{
  "backendServerList": [
    {
      "instanceId": "i-YfAib14A",
      "weight": 50,
    }
  ],
  "marker": "rs-0A6BE9BB",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

UpdateBackendServers更新后端服务器

描述

更新指定LoadBalancer下的后端服务器权重，支持批量修改。

请求结构

```
PUT /v{version}/blb/{blbId}/backendserver?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendServerList":[
    {
      "instanceId":instanceId,
      "weight":weight
    },...
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为update
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
backendServerList	List< BackendServerModel >	是	RequestBody参数	待更新的后端服务器列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/backendserver?update&clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "backendServerList":[
    {
      "instanceId":"i-YfAibI4A",
      "weight":"50"
    },
    {
      "instanceId":"i-jBXABQ8D",
      "weight":"60"
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

RemoveBackendServers 释放后端服务器

描述

- 释放指定LoadBalancer下的后端服务器，后端服务器通过其标识符指定
- 支持批量释放

请求结构

```
PUT /v{version}/blb/{blbId}/backendserver?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendServerList":[backendServer1,backendServer2,...]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
backendServerList	List<String>	是	RequestBody参数	所有待释放的后端服务器标识符，一起组成一个数组

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijY/backendserver?clientToken=shfkjanajsabvhj HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "backendServerList":["i-jBXABQ8D"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

普通型BLB安全组相关接口

 BindSecurityGroups绑定普通安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）绑定普通安全组，支持批量绑定

请求结构

```
PUT /v{version}/blb/{blbId}/securitygroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "securityGroupIds": securityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值bind
securityGroupIds	List	是	RequestBody参数	绑定的普通安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/securitygroup?bind&clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16

{
  "securityGroupIds": ["g-4NxWoxeq", "g-4NxWoxea"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef474
```

 UnbindSecurityGroups解绑普通安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）解绑普通安全组，支持批量解绑

请求结构

```
PUT /v{version}/blb/{blbld}/securitygroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "securityGroupIds": securityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值unbind
securityGroupIds	List	是	RequestBody参数	解绑的普通安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/securitygroup?unbind&clientToken=be31b98c-5e41-4838-9830-9be700de5a21 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf17

{
  "securityGroupIds": ["g-4NxWoxeq", "g-4NxWoxea"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef478
```

 DescribeSecurityGroups查询BLB实例普通安全组列表

描述

按ID查询用户账户下指定LoadBalancer的普通安全组信息

请求结构

```
GET /v{version}/blb/{blbId}/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	要查询的LoadBalancer的标识符

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
securityGroups	List< BlbSecurityGroupModel >	普通安全组信息，由BlbSecurityGroupModel组成的集合

请求示例

```
GET /v1/blb/lb-j8UqijYf/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: dfd5d01f-a869-49eb-aba5-d853a3350428

{
  "blbSecurityGroups": [
    {
      "securityGroupDesc": "",
      "securityGroupId": "g-4NxWoxeq",
      "securityGroupName": "securitygroup-name",
      "vpcName": "vpc-name",
      "securityGroupRules": [
        {
          "destGroupId": "",
          "destIp": "all",
          "direction": "egress",
          "ethertype": "IPv4",
          "portRange": "1-65535",
          "protocol": "all",
          "securityGroupRuleId": "r-gkv8yupumvx2"
        }
      ]
    }
  ]
}
```

BindEnterpriseSecurityGroups绑定企业安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）绑定企业安全组（目前每个BLB实例最多支持绑定1个企业安全组）

请求结构

```
PUT /v{version}/blb/{blbld}/enterprise/securitygroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "enterpriseSecurityGroupIds": enterpriseSecurityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值bind
enterpriseSecurityGroupIds	List	是	RequestBody参数	绑定的企业安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/enterprise/securitygroup?bind&clientToken=be31b98c-5e41-4838-9830-9be700de5a20
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16

{
  "enterpriseSecurityGroupIds": ["esg-g8z4qfj0m0bu"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef474
```

UnbindEnterpriseSecurityGroups解绑企业安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）解绑企业安全组，支持批量解绑

请求结构

```
PUT /v{version}/blb/{blbld}/enterprise/securitygroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "enterpriseSecurityGroupIds": enterpriseSecurityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值unbind
enterpriseSecurityGroupIds	List	是	RequestBody参数	解绑的企业安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/enterprise/securitygroup?unbind&clientToken=be31b98c-5e41-4838-9830-9be700de5a21
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf17

{
  "enterpriseSecurityGroupIds": ["esg-gzsqygs59npx"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef478
```

DescribeEnterpriseSecurityGroups查询BLB实例企业安全组列表

描述

按ID查询用户账户下指定LoadBalancer的企业安全组信息

请求结构

```
GET /v{version}/blb/{blbld}/enterprise/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	要查询的LoadBalancer的标识符

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
enterpriseSecurityGroupIds	List< BlbEnterpriseSecurityGroupModel >	企业安全组信息，由BlbEnterpriseSecurityGroupModel组成的集合

请求示例

```
GET /v1/blb/lb-j8UqijYf/enterprise/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: dfd5d01f-a869-49eb-aba5-d853a3350428
```

```
{
  "enterpriseSecurityGroups": [{
    "enterpriseSecurityGroupId": "esg-g8z4qfj0m0bu",
    "enterpriseSecurityGroupName": "test0627",
    "enterpriseSecurityGroupDesc": "",
    "enterpriseSecurityGroupRules": [{
      "remark": "全部协议",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv6",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu",
      "protocol": "all"
    },
    {
      "remark": "",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9je09z6bd9wv",
      "protocol": "tcp"
    },
    {
      "remark": "",
      "direction": "ingress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "2",
      "sourceIp": "all",
      "destIp": "",
      "enterpriseSecurityGroupRuleId": "esgr-jcej4juqbh49",
      "protocol": "tcp"
    }
  ]
}
```

应用型BLB接口

应用型BLB实例相关接口

CreateLoadBalancer创建实例

描述

- 创建一个LoadBalancer，返回分配的服务地址及实例ID。
- 付费方式为后付费。
- 需要实名认证，若未通过实名认证可以前往[百度开放云官网控制台](#)中的安全认证下的实名认证中进行认证。

请求结构

```
POST /v{version}/appblb?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name": name,
  "type": type,
  "desc": desc ,
  "subnetId": subnetShortId,
  "vpcId": vpcShortId,
  "address": address,
  "eip": eip,
  "tags":[
    {
      "tagKey": tagKey,
      "tagValue": tagValue
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
name	String	否	RequestBody参数	LoadBalancer的名称，方便记忆。长度1~65个字节，字母开头，可包含字母数字_./:字符。若不传该参数，会自动生成
type	String	否	RequestBody参数	LoaBalancer的类型，此处取值"application"(默认)
desc	String	否	RequestBody参数	LoadBalancer实例的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空
subnetId	String	是	RequestBody参数	LoadBalancer实例所属子网

			~~	
vpcId	String	是	RequestBody参数	LoadBalancer实例vip所属VPC的vpcShortId
addresses	String	否	RequestBody参数	指定负载均衡实例的私网IP地址，该地址必须包含在子网网段下。
eip	String	否	RequestBody参数	绑定已有的eip。取值为eip的IP地址
tags	List<TagModel>	否	RequestBody参数	待创建的标签键值对列表
billing	BillingFormCreate	否	RequestBody参数	订单相关信息，购买预付费blb的时候必填
performanceLevel	String	否	RequestBody参数	性能规格参数，默认为共享型。取值如下："small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型1，"large1"超大型1，"large2"超大型2，"large3"超大型3，如果是后付费按使用量计费的时候还可以传"unlimited"不限速
autoRenewLength	int	否	RequestBody参数	支持创建预付费BLB同时开通自动续费，根据autoRenewTimeUnit的取值有不同的范围，month 为1到9，year 为1到3
autoRenewTimeUnit	String	否	RequestBody参数	支持创建BLB同时开通自动续费，取值为 month 获 year （默认 month）
resourceGroupId	String	否	RequestBody参数	支持创建BLB同时绑定资源分组id
allowDelete	Boolean	否	RequestBody参数	是否允许删除。缺省值为true，代表允许删除
allocateIpv6	Boolean	否	RequestBody参数	是否分配ipv6地址。true代表分配ipv6地址，false代表不分配ipv6地址

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
address	String	分配的服务地址IP，通过这个IP即能访问该实例
name	String	LoadBalancer的名称
desc	String	LoadBalancer的描述
blbld	String	LoadBalancer的ID。后续针对该实例的操作，均需要在请求中带上此ID

请求示例

```
POST /v1/appblb?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

创建后付费blb实例
{
  "name": "blb-for-test",
  "subnetId": "sbn-hj9k8v413tqm",
  "vpcId": "vpc-tuf4sqw82i5m",
  "address": "192.168.0.24",
  "eip": "100.88.8.139"
}

创建预付费blb实例
{
  "vpcId": "vpc-fbsxgfs9ty60",
  "subnetId": "sbn-3317rkjfq8j9",
  "name": "ccq20240319",
  "billing": {
    "paymentTiming": "Prepaid",
    "reservation": {
      "reservationLength": 2
    }
  },
  "performanceLevel": "small1"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "blbld": "lb-BLuOPSLZ",
  "name": "blb-for-test",
  "desc": "",
  "address": "192.168.0.24"
}
```

 UpdateLoadBalancer更新实例

描述

- 更新一个LoadBalancer的配置信息。

请求结构

```
PUT /v{version}/appblb/{blbld}?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name": name
  "desc": desc
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	待更新的LoadBalancer的ID
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
name	String	否	RequestBody参数	LoadBalancer的名称，方便记忆。长度1~65个字节，字母开头，可包含字母数字_./字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	LoadBalancer实例的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空
allowDelete	Boolean	否	RequestBody参数	是否允许删除。缺省值为true，代表允许删除
allocateIpv6	Boolean	否	RequestBody参数	是否分配ipv6地址。true代表分配ipv6地址，false代表不分配ipv6地址

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-9f59e197?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "name": "blb-for-test",
  "desc": "用于测试环境"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeLoadBalancers查询已有的BLB实例

描述

- 查询用户账户下所有LoadBalancer的信息
- 支持按LoadBalancer的Name、Address进行模糊查询
- 支持按LoadBalancer的ID进行精确查询
- 支持查找绑定指定BCC的LoadBalancer，通过bccId参数指定
- 返回结果是多重查询条件的交集
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/apblb?address={address}&name={name}&blbId={blbId}&marker={marker}&maxKeys={maxKeys}
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
address	String	否	Query参数	要查询的LoadBalancer地址，格式为点分十进制的IP
name	String	否	Query参数	要查询的LoadBalancer名称
blbId	String	否	Query参数	要查询的LoadBalancer ID（只支持精确查询）
bccId	String	否	Query参数	绑定该BCC的LoadBalancer（只支持精确查询）
exactlyMatch	boolean	否	Query参数	为true表示查询条件为精准匹配，否则为模糊匹配。缺省值为false
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
blbList	List<AppBLBModel>	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb?name=test&maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "blbList": [
    {
      "blbId": "lb-a7e5zPPk",
      "status": available,
      "name": "test-blb",
      "desc": "用于生产环境",
      "vpId": "vpc-fy6vdykpwkqb",
      "address": "192.168.0.24"
    },
    {
      "blbId": "lb-gj5gVpeq",
      "status": available,
      "name": "nametest",
      "desc": "用于测试环境",
      "vpId": "vpc-a8n5p6kybbx4",
      "address": "192.168.0.25"
    }
  ],
  "marker": "blb-0A20F971",
  "nextMarker": "blb-0A20FB09",
  "isTruncated": true,
  "maxKeys": 2
}
```

 DescribeLoadBalancerDetail查询BLB实例详情

描述

- 按id查询用户账户下所有LoadBalancer的详细信息
- 包含LoadBalancer所有的监听器端口信息

请求结构

```
GET /v{version}/appblb/{blbld} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	要查询的LoadBalancer的标识符（必须输入完整的blbld）

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
blbId	String	LoadBalancer的标识符
status	blbStatus	BLB状态
desc	String	LoadBalancer的描述
address	String	LoadBalancer的内网地址
publicIp	String	LoadBalancer的公网地址
cidr	String	LoadBalancer所在网络cidr
vpcName	String	LoadBalancer所属vpc名称
subnetCidr	String	LoadBalancer所属子网cidr
subnetName	String	LoadBalancer所属子网名称
createTime	String	LoadBalancer创建时间
releaseTime	String	LoadBalancer自动释放时间
listener	List<ListenerModel>	LoadBalancer下挂载监听器列表
tags	List<TagModel>	标签键值对列表
allowDelete	Boolean	是否允许删除
paymentTiming	String	付款时间，预支付（Prepaid）或后支付（Postpaid）
paymentTiming	String	计费方式，"ByCapacityUnit" 按使用量 "BySpec" 按固定规格
performanceLevel	String	性能规格。"small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型1，"large1"超大型1，"large2"超大型2，"large3"超大型3
expireTime	String	预付费实例的过期时间
eipRouteType	String	EIP线路类型
publicIpv6	String	如果LoadBalancer绑定过EIPv6，则显示该项，否则不显示
eipV6RouteType	String	EIPv6线路类型

请求示例

```
GET /v1/appblb/lb-7fb1e926 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例


```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "blbId": "lb-gj5gVpeq",
  "status": "available",
  "name": "nametest",
  "desc": "",
  "address": "192.168.0.2",
  "publicIp": "123.2.3.4",
  "cidr": "192.168.0.0/16",
  "vpcName": "test",
  "subnetName": "系统预定义子网",
  "subnetCidr": "192.168.0.0/20",
  "createTime": "2019-03-07T02:35:31Z",
  "releaseTime": "2019-03-07T02:35:31Z",
  "paymentTiming": "Postpaid",
  "billingMethod": "BySpec",
  "listener": [
    {
      "port": 80,
      "type": "TCP"
    }
  ]
}
```

🔗 DeleteLoadBalancer释放BLB实例

描述

- 释放指定LoadBalancer，被释放的LoadBalancer无法找回

请求结构

```
DELETE /v{version}/appblb/{blbId} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	待释放的LoadBalancer的ID

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
DELETE /v1/appblb/lb-j8UqijYf HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T07:09:25Z/1800/host/fa13c6057d34b0152a0959541f49bea9ae4eece2317ac3e371dda33c27f5f76
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 07:09:45 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

应用型BLB监听器相关接口

CreateAppTCPListener创建TCP监听器

- 在指定LoadBalancer下，创建一个基于TCP协议的应用型blb监听器，监听一个前端端口，将发往该端口的所有TCP流量，根据策略进行转发。

请求结构

```
POST /v{version}/appblb/{blbId}/TCPListener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort": listenerPort,
  "scheduler": scheduler,
  "tcpSessionTimeout": tcpSessionTimeout,
  "description": description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"WeightLeastConn"代表加权最小连接数、"Hash"代表根据源ip哈希
tcpSessionTimeout	int	否	RequestBody参数	tcp设置连接超时时间（单位：秒），默认900，需为10-4000间的整数
description	String	否	RequestBody参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/TCPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "description":"description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppUDPListener创建UDP监听器

- 在指定LoadBalancer下，创建一个基于UDP协议的应用型监听器，监听一个前端端口，将发往该端口的所有UDP流量，根据策略进行转发。

请求结构

```
POST /v{version}/appblb/{blbld}/UDPListener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "udpSessionTimeout":udpSessionTimeout,
  "description":description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"WeightLeastConn"代表加权最小连接数、"Hash"代表根据源ip哈希
udpSessionTimeout	int	否	RequestBody参数	udp会话超时时间。默认为90，需为5-4000间的整数，单位秒
description	String	否	RequestBody参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/UDPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "udpSessionTimeout":90,
  "description":"description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppHTTPListener创建HTTP监听器

- 在指定LoadBalancer下，创建一个基于HTTP协议的应用型监听器，监听一个前端端口，将发往该端口的所有HTTP请求，根据策略转发到后端服务器监听的后端端口上

请求结构

```
POST /v{version}/appblb/{blbld}/HTTPlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "keepSession":keepSession,
  "keepSessionType":keepSessionType,
  "keepSessionTimeout":keepSessionTimeout,
  "keepSessionCookieName":keepSessionCookieName,
  "xForwardedFor":xForwardedFor,
  "xForwardedProto":xForwardedProto,
  "serverTimeout":serverTimeout,
  "redirectPort":redirectPort,
  "description":description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	否	RequestBody参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	RequestBody参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionTimeout	int	否	RequestBody参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	RequestBody参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	否	RequestBody参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	RequestBody参数	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议，默认关闭
serverTimeout	int	否	RequestBody参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
redirectPort	int	否	RequestBody参数	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定
description	String	否	RequestBody参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/HTTPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "xForwardedFor":true,
  "serverTimeout":1800,
  "redirectPort":0,
  "description":"description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppHTTPSListener创建HTTPS监听器

- 在指定LoadBalancer下，创建一个基于HTTPS协议的应用型监听器，监听一个前端端口，将发往该端口的所有HTTPS请求，先通过SSL卸载转换为HTTP请求后，再根据策略转发到后端服务器监听的后端端口上

请求结构

```
POST /v{version}/appblb/{blbld}/HTTPSlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "keepSession":keepSession,
  "keepSessionType":keepSessionType,
  "keepSessionTimeout":keepSessionTimeout,
  "keepSessionCookieName":keepSessionCookieName,
  "xForwardedFor":xForwardedFor,
  "xForwardedProto":xForwardedProto,
  "serverTimeout":serverTimeout,
  "certIds":[certId],
  "encryptionType":encryptionType,
  "encryptionProtocols":[protocol1, protacol2],
  "dualAuth":false,
  "clientCertIds":[clientCertId],
  "additionalCertDomains":[{"certId":certId,"host":host}],
  "description":description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

		是否必需	数据类型	

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	否	RequestBody参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	RequestBody参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionTimeout	int	否	RequestBody参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	RequestBody参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	否	RequestBody参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	RequestBody参数	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议，默认关闭
serverTimeout	int	否	RequestBody参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
certIds	List<String>	是	RequestBody参数	监听器要加载的证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
encryptionType	String	否	RequestBody参数	加密选项，默认tls_cipher_policy_default。支持：tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tls10"、"tls11"、"tls12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开

			数	
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCerts	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
additionalCertDomains	List<AdditionalCertificateDomain>	否	RequestBody参数	扩展域名
description	String	否	RequestBody参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqjYf/HTTPSListener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "xForwardedFor":true,
  "serverTimeout":1800,
  "certIds":["cert-zfj2ey2z4kmm"],
  "description":"description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppSSLListener创建SSL监听器

- 在指定LoadBalancer下，创建一个基于SSL协议的应用型监听器，监听一个前端端口，将发往该端口的所有SSL流量，根据策略进行转发。

请求结构

```
POST /v{version}/appblb/{blbId}/SSLlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "certIds":[certId],
  "encryptionType":encryptionType,
  "encryptionProtocols":[protocol1, protocol2],
  "dualAuth":false,
  "clientCertIds":[clientCertId],
  "description":description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
certIds	List<String>	是	RequestBody参数	监听器要加载的证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
encryptionType	String	否	RequestBody参数	加密选项，默认tls_cipher_policy_default。支持：tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocol	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCertificateIds	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
description	String	否	RequestBody参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/SSLlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "certIds":["cert-zfj2ey2z4kmm"],
  "description":"description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateAppTCPLListener更新TCP监听器

- 更新指定LoadBalancer下的TCP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/appblb/{blbld}/TCPLlistener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler":scheduler,
  "tcpSessionTimeout":tcpSessionTimeout
  "description":description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Query参数	监听器端口号
scheduler	String	否	Request Body参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
tcpSessionTimeout	int	否	Request Body参数	tcp设置连接超时时间（单位：秒），默认900，需为10-4000间的整数
description	String	否	Request Body参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqjYf/TCPlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "description": "update description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

- 更新指定LoadBalancer下的UDP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/appblb/{blbld}/UDPlistener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler":scheduler,
  "udpSessionTimeout":udpSessionTimeout,
  "description":description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	URL参数	监听器的端口号
scheduler	String	是	Request Body参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
udpSessionTimeout	int	否	Request Body参数	udp会话超时时间。默认为90，需为5-4000间的整数，单位秒
description	String	否	Request Body参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/UDPlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "udpSessionTimeout": 90,
  "description": "update description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateAppHTTPListener更新HTTP监听器

- 更新指定LoadBalancer下的HTTP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定

请求结构

```
PUT /v{version}/appblb/{blbld}/HTTPListener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler": scheduler,
  "keepSession": keepSession,
  "keepSessionType": keepSessionType,
  "keepSessionTimeout": keepSessionTimeout,
  "keepSessionCookieName": keepSessionCookieName,
  "xForwardedFor": xForwardedFor,
  "xForwardedProto": xForwardedProto,
  "serverTimeout": serverTimeout,
  "redirectPort": redirectPort,
  "description": description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	否	RequestBody参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	RequestBody参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionTimeout	int	否	RequestBody参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	RequestBody参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	否	RequestBody参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	RequestBody参数	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议，默认关闭
serverTimeout	int	否	RequestBody参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
redirectPort	int	否	RequestBody参数	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定
description	String	否	RequestBody参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例


```
PUT /v1/appblb/lb-j8UqijYf/HTTPlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "xForwardedFor": true,
  "serverTimeout": 1800,
  "description": "update description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 UpdateAppHTTPSListener更新HTTPS监听器

- 更新指定LoadBalancer下的HTTPS监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定

请求结构

```
PUT /v{version}/appblb/{blbld}/HTTPSListener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler": scheduler,
  "keepSession": keepSession,
  "keepSessionType": keepSessionType,
  "keepSessionDuration": keepSessionDuration,
  "keepSessionCookieName": keepSessionCookieName,
  "xForwardedFor": xForwardedFor,
  "xForwardedProto": xForwardedProto,
  "serverTimeout": serverTimeout,
  "certIds": [certId],
  "encryptionType": encryptionType,
  "encryptionProtocols": [protocol1, protocol2],
  "dualAuth": false,
  "clientCertIds": [clientCertId],
  "additionalCertDomains": [{"certId": certId, "host": host}],
  "description": description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
			URI 参	

version	String	是	URL参 数	API版本号，当前取值1
blbld	String	是	URL参 数	所属LoadBalancer的标识符
clientToken	String	是	Query 参数	幂等性Token，长度不超过64位
listenerPort	int	是	Query 参数	要更新的监听器端口，需为1-65535间的整数
scheduler	String	是	Reques tBody参 数	负载均衡算法，支持2种方 式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、 "LeastConnection"代表最小连接数
keepSession	boolean	否	Reques tBody参 数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务 器，默认关闭
keepSessionType	String	否	Reques tBody参 数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支 持"insert"/"rewrite"，默认"insert"
keepSessionTimeout	int	否	Reques tBody参 数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认 3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	Reques tBody参 数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且 keepSessionType="rewrite"时有效
xForwardedFor	boolean	否	Reques tBody参 数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个 HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	Reques tBody参 数	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个 HTTP Header来获得监听协议，默认关闭
serverTimeout	int	否	Reques tBody参 数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
certIds	List<String>	是	Reques tBody参 数	监听器要加载的证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后 一个生效
encryptionType	String	否	Reques tBody参 数	加密选项。支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher _policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	Reques tBody参 数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、 "tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	Reques tBody参 数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	Reques tBody参 数	是否开启双向认证，默认为关闭

			请求体	
clientCertIds	List<String>	否	RequestBody参数数	当dualAuth为true时，加载的客户端证书链,当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
additionalCertDomains	List<AdditionalCertDomain>	否	RequestBody参数数	扩展域名
description	String	否	RequestBody参数数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/HTTPSlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "xForwardedFor": true,
  "serverTimeout": 1800,
  "certIds": ["cert-zfj2ey2z4kmm"],
  "description": "update description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 UpdateAppSSLListener更新SSL监听器

- 更新指定LoadBalancer下的SSL监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/appblb/{blbld}/SSLlistener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler":scheduler,
  "certIds":[certId],
  "encryptionType":encryptionType,
  "encryptionProtocols":[protocol1, protocol2],
  "dualAuth":false,
  "clientCertIds":[clientCertId],
  "description":description
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
scheduler	String	否	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
certIds	List<String>	是	RequestBody参数	监听器要加载的证书链;当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
encryptionType	String	否	RequestBody参数	加密选项。支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCertIds	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链;当前仅允许传入1个证书ID，如果传入多个，则只有最后一个生效
description	String	否	RequestBody参数	描述信息，长度不超过200个字符。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/TCPlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "certIds": ["cert-zfj2ey2z4kmm"],
  "description": "update description"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeAppTCPListeners查询应用型TCP监听器

- 查询指定LoadBalancer下所有TCP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/TCPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List<AppTCPListenerModel>	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/TCPlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin"
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin"
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeAppUDPListeners查询UDP监听器

- 查询指定LoadBalancer下所有UDP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/UDPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< AppUDPLListenerModel >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/UDPLlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例


```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin",
      "udpSessionTimeout":90
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin",
      "udpSessionTimeout":90
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeAppHTTPListeners查询HTTP监听器

- 查询指定LoadBalancer下所有HTTP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/HTTPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys}
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List<AppHTTPListenerModel>	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/HTTPlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 30,
      "redirectPort": 80
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 1800,
      "redirectPort": 80
    }
  ],
  "marker": "listener-005A",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeAppHTTPSListeners查询HTTPS监听器

- 查询指定LoadBalancer下所有HTTPS监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/HTTPSlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys}
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< AppHTTPSListenerModel >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/HTTPSlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 30,
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 1800,
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-005A",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeAppSSLListeners查询应用型SSL监听器

- 查询指定LoadBalancer下所有SSL监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/SSLlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< AppSSLListenerModel >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/SSLlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin",
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin",
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

 DescribeAppAllListeners查询所有监听

请求结构

```
GET /v{version}/appblb/{blbld}/listener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量通常不超过1000，缺省值为1000

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
marker	String	标记查询的起始位置
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量
listenerList	List<AppListenerModel>	包含查询的监听列表

请求示例

```
GET /v1/appblb/lb-a7942975/listener HTTP/1.1
Host: blb.bj.baidubce.com
ContentType: application/json
Authorization: bce auth v1/f81d3b34e48048fbb2634dc7882d7e21/2017 03
11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7 4ad5 451d 9215 71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content Type: application/json;charset=UTF 8
Server: BWS

{
  "nextMarker": "",
  "marker": "",
  "maxKeys": 1,
  "isTruncated": true,
  "listenerList": [
    {
      "listenerPort": 53,
      "scheduler": "RoundRobin",
      "listenerType": "UDP"
    },
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin",
      "listenerType": "HTTP",
      "keepSession": false,
      "xForwardFor": true,
      "xForwardedProto": false,
      "serverTimeout": 30,
      "redirectPort": 0
    },
    {
      "listenerPort": 81,
      "scheduler": "RoundRobin",
      "listenerType": "TCP",
      "tcpSessionTimeout": 900
    },
    {
      "listenerPort": 443,
      "scheduler": "RoundRobin",
      "listenerType": "SSL",
      "certIds": [
        "cert-gs8bktrm7drp"
      ]
    }
  ]
}
```



```

    },
    "dualAuth": false,
    "encryptionType": "tls_cipher_policy_ie",
    "encryptionProtocols": [
        "sslv3",
        "tlsv10",
        "tlsv11",
        "tlsv12"
    ],
    "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-ECDSA-RC4-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-RSA-RC4-SHA:ECDHE-RSA-DES-CBC3-SHA:RC4-SHA:AES128-GCM-SHA256:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES256-SHA:AES128-SHA256:DES-CBC3-SHA"
  },
  {
    "listenerPort": 444,
    "scheduler": "RoundRobin",
    "listenerType": "HTTPS",
    "keepSession": false,
    "xForwardFor": false,
    "xForwardedProto": false,
    "serverTimeout": 30,
    "certIds": [
        "cert-gs8bktrm7drp"
    ],
    "dualAuth": false,
    "encryptionType": "tls_cipher_policy_1_1",
    "encryptionProtocols": [
        "tlsv11",
        "tlsv12"
    ],
    "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA"
  }
]
}

```

🔗 DeleteAppListeners释放监听器

- 释放指定LoadBalancer下的监听器，监听器通过监听端口来指定
- 支持批量释放

请求结构

```
PUT /v{version}/appblb/{blbld}/listener?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "portList":[port1,port2]
}
或者
{
  "portTypeList":[
    {
      "port":port,
      "type":"type"
    },
    {
      "port":port,
      "type":"type"
    },
  ],
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为batchdelete
clientToken	String	是	Query参数	幂等性Token
portList	List<int>	否	Request Body参数	所有待释放的监听器的端口，一起组成一个数组
portTypeList	List<PortTypeModel>	否	Request Body参数	所有待释放的监听器的端口和协议，一起组成一个数组，如果开通了同端口多协议，并且只想删除其中一个协议，使用这个参数。portList和portTypeList至少传一个。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/listener?batchdelete&clientToken=abc
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "portList":[80,87]
}
或者
{
  "portTypeList":[
    {
      "port":80,
      "type":"TCP"
    },
    {
      "port":80,
      "type":"UDP"
    },
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreatePolicys创建策略

- 在指定应用型BLB监听器端口下创建策略

请求结构

```
POST /v{version}/appblb/{blbld}/policys?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort": listenerPort,
  "appPolicyVos": [
    {
      "appServerGroupId": appServerGroupId,
      "backendPort": backendPort,
      "priority": priority,
      "desc": desc,
      "ruleList": [
        {
          "key": key,
          "value": value
        }
      ]
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token
listenerPort	int	是	RequestBody参数	对应所在BLB下监听器端口号
appPolicyVos	List<CreateAppPolicy>	是	RequestBody参数	监听器绑定策略列表，当监听器类型为TCP、UDP、SSL时，只能创建一条且只支持全匹配；当监听器类型为http、https监听器支持多个policy
type	String	否	RequestBody参数	当监听器端口下有多个协议时，type必传

返回状态码

成功返回200，失败返回见[错误码](#) 返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/policys?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort": 98,
  "type": "HTTP",
  "appPolicyVos": [
    {
      "appServerGroupId": "sg-db3975dc",
      "backendPort": 83,
      "priority": 100,
      "desc": "aa",
      "ruleList": [
        {
          "key": "*",
          "value": "*"
        }
      ]
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

 DescribePolicys查询对应BLB端口下策略信息

- 查询指定LoadBalancer下所有服务器组的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/policys?marker={marker}&maxKeys={maxKeys}&port={port} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
port	int	是	Query参数	BLB下监听器端口号
type	String	否	Query参数	当监听器端口下有多个协议时，type必传
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
policyList	List< AppPolicy >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/policys?maxKeys=2&port=80&type=HTTP HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "marker": "rs-012310",
  "isTruncated": true,
  "maxKeys": 1000,
  "nextMarker": "rs-013214",
  "policyList": [
    {
      "id": "policy_id",
      "appServerGroup": "sg_id",
      "appServerGroupName": "sg_name",
      "frontendPort": 80,
      "type": "TCP",
      "backendPort": 98,
      "portType": "TCP",
      "priority": 100,
      "desc": "aa",
      "groupType": "Server",
      "ruleList": [
        {
          "key": "host",
          "value": "*.baidu.com"
        }
      ]
    }
  ]
}
```

UpdatePolicies 批量修改策略

- 批量修改 BLB 监听器指定端口下转发策略的优先级和权重。

请求结构

```
PUT /v{version}/appblb/{blbld}/policies?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
ContentType: application/json;charset=UTF-8
Authorization: authorization string

{
  "port": 8080,
  "type": "HTTP",
  "policyList": [
    {
      "policyId": "policyId-1",
      "priority": 100,
      "description": "description"
    },
    {
      "policyId": "policyId-2",
      "priority": 101,
      "description": "description"
    }
  ]
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
clientToken	String	否	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	取值为"batchupdate"
port	Integer	是	ReuquestBody参数	监听端口
type	String	是	ReuquestBody参数	监听的端口所使用的协议
policyList	List<AppPolicyForUpdate>	是	RequestBody参数	要修改的监听策略列表

返回头域

除公共头域，无其它特殊头域。

返回参数

无。

请求示例

```
PUT /v1/appblb/lb-9afscms1/policys?batchupdate&clientToken=e97a78a9-4855-4ada-9a24-cecc9bd66878 HTTP/1.1
Host: blb.bj.baidubce.com
ContentType: application/json;charset=UTF-8
Authorization: authorization string
{
  "port": 80,
  "type": "TCP",
  "policyList": [
    {
      "policyId": "policy-5152f520",
      "priority": 100,
      "description": "desc"
    }
  ]
}
```

返回示例

```
HTTP/1.1 200 OK
Date: Wed, 03 Dec 2014 06:42:19 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 DeletePolicys批量删除策略

- 批量删除对应BLB端口下的策略

请求结构


```
PUT /v{version}/appblb/{blbld}/policys?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "port": port,
  "policyIdList": ["policyId1", "policyId2", "policyId3"]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为batchdelete
clientToken	String	是	Query参数	幂等性Token
port	int	是	RequestBody参数	要删除策略前端端口
policyIdList	List<String>	是	RequestBody参数	所有待释放的策略标识符，一起组成一个数组
type	String	否	RequestBody参数	当监听器端口下有多个协议时，type必传

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/policys?batchdelete&clientToken=your_clientToken
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "port":90,
  "policyIdList": ["ab48bbe", "2520741", "a78e04"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

应用型BLB服务器组相关接口

CreateAppServerGroup创建应用型服务器组

- 在指定应用型BLB下，创建一个服务器组，用来绑定后端服务器，以及为监听器开放相应的端口

请求结构

```
POST /v{version}/appblb/{blbld}/appservergroup?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name":name,
  "desc":desc,
  "backendServerList": [
    {
      "instanceId": instanceId,
      "weight": weight
    },
    {
      "instanceId": instanceId,
      "weight": weight
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
name	String	否	RequestBody参数	服务器组的名称，方便记忆。长度1~65个字节，字母开头，_可包含字母数字/.字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	服务器组的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空。
backendServerList	List<AppBackendServer>	否	RequestBody参数	服务器组绑定的后端服务器列表

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
id	String	服务器组id
name	String	服务器组的名称
desc	String	服务器组的描述
status	String	服务器组状态，详见 blbStatus

请求示例

```
POST /v1/appblb/lb-j8UqijYf/appservergroup?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "name": "name1",
  "desc": "aaa",
  "backendServerList":[
    {
      "instanceId": "BCC_UUID",
      "weight": 20
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
{
  "id": "sg_short_id",
  "name": "name",
  "desc": "aaa",
  "status": "available"
}
```

🔗 UpdateAppServerGroup更新服务器组

- 更新指定LoadBalancer下的TCP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/appblb/{blbld}/appservergroup?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgld":sgld,
  "name":name,
  "desc":desc
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	RequestBody参数	要更新的服务器组的id
name	String	否	RequestBody参数	服务器组的名称，方便记忆。长度1~65个字节，字母开头，可包含字母数字_/.字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	服务器组的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/appservergroup?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgld": "sgld",
  "name": "name1",
  "desc": "aaa"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeAppServerGroup查询服务器组

- 查询指定LoadBalancer下所有服务器组的信息
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/appservergroup?name={name}&marker={marker}&maxKeys={maxKeys}&exactlyMatch={exactlyMatch} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
name	String	否	Query参数	要查询的服务器组名称
exactlyMatch	boolean	否	Query参数	设置name是否全局匹配
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
appServerGroupList	List< AppServerGroup >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/appservergroup?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "appServerGroupList": [
    {
      "id": "sg_short_id",
      "name": "name",
      "desc": "bbb",
      "status": "available" ,
      "portList": [
        {
          "id": "port_id"
          "port": BACKEND_PORT,
          "type": "TCP",
          "healthCheck": "TCP",
          "healthCheckNormalStatus": "^[1-5][0-9]*$",
          "healthCheckPort": 90,
          "healthCheckTimeoutInSecond": 2,
          "healthCheckDownRetry": 3,
          "healthCheckUpRetry": 3,
          "healthCheckIntervalInSecond": 1,
          "healthCheckUrlPath": "HEALTH_CHECK_URL_PATH",
          "healthCheckHost": www.xxx,
          "status": "available"
        }
      ]
    }
  ],
  "marker": "sg-012310",
  "isTruncated": false,
  "maxKeys": 1000,
  "nextMarker": "sg-013214"
}
```

DeleteAppServerGroup 删除服务器组

- 删除服务器组，通过服务器组id指定

请求结构

```
PUT /v{version}/appblb/{blbld}/appservergroup?action={action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为delete
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgId	String	是	RequestBody参数	要删除的服务器组id

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/appservergroup?delete&clientToken=be31b98c-5e41-4838-9830-9be700de5a20
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "sgId":"abc"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppServerGroupPort创建应用型服务器组端口

- 在指定应用型BLB下，创建一个服务器组后端端口，将发往该端口的所有流量按权重轮询分发到其绑定的对应服务器列表中的服务器

请求结构

```
POST /v{version}/appblb/{blbld}/appservergroupport?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgld":sgld,
  "port":port,
  "type":type,
  "enableHealthCheck":enableHealthCheck,
  "healthCheck":healthCheck,
  "healthCheckPort":healthCheckPort,
  "healthCheckUriPath":healthCheckUriPath,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckIntervalInSecond":healthCheckIntervalInSecond,
  "healthCheckDownRetry":healthCheckDownRetry,
  "healthCheckUpRetry":healthCheckUpRetry,
  "healthCheckNormalStatus":healthCheckNormalStatus,
  "healthCheckHost":healthCheckHost,
  "udpHealthCheckString":udpHealthCheckString
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	Request Body参数	端口所属服务器组标识符
port	int	是	Request Body参数	端口号，需为1-65535间的整数
type	String	是	Request Body参数	监听端口的协议类型，"TCP"/"UDP"/"HTTP"
enableHealthCheck	Boolean	否	Request Body参数	是否开启健康检查，默认true
healthCheck	String	否	Request Body参数	健康检查协议，支持"HTTP"/"TCP"/"UDP"/"ICMP"， TCP默认TCP，UDP默认UDP，HTTP默认HTTP
healthCheckPort	int	否	Request Body参数	健康检查端口，允许范围为1-65535，默认为port的取值，当健康检查协议为"HTTP"时生效
healthCheckUriPath			Request	

healthCheckPath	String	否	Body参数	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckTimeoutInSecond	int	否	Request Body参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckIntervalInSecond	int	否	Request Body参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
healthCheckDownRetry	int	否	Request Body参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthCheckUpRetry	int	否	Request Body参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	否	Request Body参数	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckHost	String	否	Request Body参数	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
udpHealthCheckString	String	否	Request Body参数	UDP服务的健康检查字段。当健康检查协议为"UDP"时，则必传

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
id	String	服务器组端口id
status	String	服务器组端口状态，详见 blbStatus

请求示例

```
POST /v1/appblb/lb-j8UqijYf/appservergroupport?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgId": "sgId",
  "type": "TCP",
  "port": 80,
  "enableHealthCheck": true,
  "healthCheck": "TCP",
  "healthCheckTimeoutInSecond": 2,
  "healthCheckDownRetry": 3,
  "healthCheckUpRetry": 3,
  "healthCheckIntervalInSecond": 3
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "id": "sg_port_id",
  "status": "available"
}
```

UpdateAppServerGroupPort更新服务器组端口

- 根据id更新服务器组端口。

请求结构

```
PUT /v{version}/appblb/{blbId}/appservergroupport?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId,
  "portId": portId,
  "enableHealthCheck": enableHealthCheck,
  "healthCheck": healthCheck,
  "healthCheckPort": healthCheckPort,
  "healthCheckUriPath": healthCheckUriPath,
  "healthCheckTimeoutInSecond": healthCheckTimeoutInSecond,
  "healthCheckIntervalInSecond": healthCheckIntervalInSecond,
  "healthCheckDownRetry": healthCheckDownRetry,
  "healthCheckUpRetry": healthCheckUpRetry,
  "healthCheckNormalStatus": healthCheckNormalStatus,
  "healthCheckHost": healthCheckHost,
  "udpHealthCheckString": udpHealthCheckString
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	Request Body参数	端口所属服务器组标识符
portld	String	是	Request Body参数	要更新的服务器组端口的标识符
enableHealthCheck	Boolean	否	Request Body参数	是否开启健康检查，默认true
healthCheck	String	否	Request Body参数	健康检查协议，支持"HTTP"/"TCP"/"UDP"/"ICMP"，TCP默认TCP，UDP默认UDP，HTTP默认HTTP
healthCheckPort	int	否	Request Body参数	健康检查端口，默认为port的取值，当健康检查协议为"HTTP"时生效
healthCheckUriPath	String	否	Request Body参数	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckTimeoutInSecond	int	否	Request Body参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckIntervalInSecond	int	否	Request Body参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
healthCheckDownRetry	int	否	Request Body参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthCheckUpRetry	int	否	Request Body参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	否	Request Body参数	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckHost	String	否	Request Body参数	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
udpHealthCheckString	String	否	Request Body参数	UDP服务的健康检查字段。当健康检查协议为"UDP"时，则必传

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/appservergroupport?clientToken=abc HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgld": "abc",
  "portld": "dbd",
  "enableHealthCheck": true,
  "healthCheck": "TCP",
  "healthCheckTimeoutInSecond": 2,
  "healthCheckDownRetry": 3,
  "healthCheckUpRetry": 3,
  "healthCheckIntervalInSecond": 1
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DeleteAppServerGroupPort删除服务器组端口

- 删除服务器组端口，通过服务器组id指定

请求结构

```
PUT /v{version}/appblb/{blbld}/appservergroupport?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgld": sgld,
  "portldList": [portid1, portid2]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为batchdelete
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	RequestBody参数	要删除的端口所在服务器组id
portIdList	List<String>	是	RequestBody参数	所有待释放的端口id，一起组成一个数组

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/appservergroupport?batchdelete&clientToken=asdfg
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "sgld": "abc",
  "portIdList": ["a", "b"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateBlbRs添加应用型BLB后端RS

- 在指定应用型BLB和服务器组下绑定后端服务器RS

请求结构

```
POST /v{version}/appblb/{blbld}/blbrs?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgld":sgld,
  "backendServerList": [
    {
      "instanceId": instanceId,
      "weight": weight
    },
    {
      "instanceId": instanceId,
      "weight": weight
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	RequestBody参数	RS所属服务器组id
backendServerList	List<AppBackendServer>	是	RequestBody参数	后端服务器列表

成功返回200，失败返回见[错误码](#)

返回状态码

成功返回200，失败返回见[错误码](#) 返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/blbrs?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgId": "sg-db3975dc",
  "backendServerList":[
    {
      "instanceId": "i-b3Dq3ea3",
      "weight": 20
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateBlbRs更新服务器组下挂载的RS权重

- 更新指定服务器组下的RS信息

请求结构

```
PUT /v{version}/appblb/{blbld}/blbrs?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId,
  "backendServerList":[
    {
      "instanceId": instanceId,
      "weight": weight
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgId	String	是	RequestBody参数	RS所属服务器组id
backendServerList	List<AppBackendServer>	是	RequestBody参数	后端服务器列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/blbrs?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgId": "sg-db3975dc",
  "backendServerList":[
    {
      "instanceId": "i-b3Dq3ea3",
      "weight": 20
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeBlbRs查询服务器组下的RS列表信息

- 查询指定LoadBalancer下所有服务器组的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbId}/blbrs?marker={marker}&maxKeys={maxKeys}&sgId={sgId} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
sgId	String	是	Query参数	RS所属服务器组id
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
backendServerList	List< AppBackendServer >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/blbrs?maxKeys=2&sgId=sg-db3975dc HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```

HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

```

```

{
  "marker": "rs-012310",
  "isTruncated": false,
  "maxKeys": 1000,
  "nextMarker": "rs-013214",
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "privateIp": "192.168.0.1",
      "weight": "WEIGHT",
      "portList": [
        {
          "listenerPort": "LISTENER_PORT1",
          "backendPort": "BACKEND_PORT1",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "portId": "port_xxx",
          "policyId": "policy_xxx",
          "status": "Alive"
        },
        {
          "listenerPort": "LISTENER_PORT2",
          "backendPort": "BACKEND_PORT2",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "portId": "port_xxx",
          "policyId": "policy_xxx",
          "status": "Alive"
        }
      ]
    }
  ]
}

```

🔗 DeleteBlbRs删除服务器组下挂载的rs

- 删除服务器组，通过服务器组id指定

请求结构

```

PUT /v{version}/appblb/{blbId}/blbrs?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId,
  "backendServerIdList": ["Serverid1", "Serverid2", "Serverid3"]
}

```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	描述要进行的操作，本接口取batchdelete
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	RequestBody参数	要删除的RS所属服务器组id
backendServerIdList	List<String>	是	RequestBody参数	所有待释放的后端服务器标识符，一起组成一个数组

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/blbrs?batchdelete&clientToken=be31b98c-5e41-4838-9830-9be700de5a20
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "sgld":"abc",
  "backendServerIdList": ["ab48bbe", "2520741", "a78e04"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

 DescribeRsMount查询服务器组下绑定的server

- 查询服务器组下绑定的server

请求结构

```
GET /v{version}/appblb/{blbld}/blbrsmount?sgld={sgld} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
sgId	String	是	Query参数	RS所属服务器组id

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
backendServerList	List< AppBackendServer >	包含查询结果的列表

请求示例

```
GET /v1/appblb/lb-j8UqijYf/blbrsmount?sgId=abc HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "privateIp": "192.168.0.1",
      "weight": "WEIGHT",
      "portList": [
        {
          "listenerPort": "LISTENER_PORT1",
          "backendPort": "BACKEND_PORT1",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "portId": "port_xxx",
          "policyId": "policy_xxx",
          "status": "Alive"
        },
        {
          "listenerPort": "LISTENER_PORT2",
          "backendPort": "BACKEND_PORT2",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "portId": "port_xxx",
          "policyId": "policy_xxx",
          "status": "Alive"
        }
      ]
    }
  ]
}
```

 DescribeRsUnMount查询服务器组下未绑定的RS

- 查询服务器组下未绑定的RS

请求结构

```
GET /v{version}/appblb/{blbld}/blbrsunmount?sgld={sgld} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
sgld	String	是	Query参数	RS所属服务器组id

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
backendServerList	List< AppBackendServer >	包含查询结果的列表

请求示例

```
GET /v1/appblb/lb-j8UqijYf/blbrsunmount?sgld=abc HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "privateIp": null,
      "weight": null,
      "portList": []
    }
  ]
}
```

应用型BLB安全组相关接口

 BindSecurityGroups绑定普通安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）绑定普通安全组，支持批量绑定

请求结构

```
PUT /v{version}/blb/{blbld}/securitygroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "securityGroupIds": securityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值bind
securityGroupIds	List	是	RequestBody参数	绑定的普通安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/securitygroup?bind&clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16

{
  "securityGroupIds": ["g-4NxWoxeq", "g-4NxWoxea"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef474
```

🔗 UnbindSecurityGroups解绑普通安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）解绑普通安全组，支持批量解绑

请求结构

```
PUT /v{version}/blb/{blbId}/securitygroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "securityGroupIds": securityGroupIds
}
```


请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值unbind
securityGroupIds	List	是	RequestBody参数	解绑的普通安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/securitygroup?unbind&clientToken=be31b98c-5e41-4838-9830-9be700de5a21 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdf17

{
  "securityGroupIds": ["g-4NxWoxeq", "g-4NxWoxea"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef478
```

 DescribeSecurityGroups查询BLB实例普通安全组列表

描述

按ID查询用户账户下指定LoadBalancer的安全组信息

请求结构

```
GET /v{version}/blb/{blbId}/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	要查询的LoadBalancer的标识符

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
securityGroups	List< BlbSecurityGroupModel >	普通安全组信息，由BlbSecurityGroupModel组成的集合

请求示例

```
GET /v1/blb/lb-j8UqijYf/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: dfd5d01f-a869-49eb-aba5-d853a3350428

{
  "blbSecurityGroups": [
    {
      "securityGroupDesc": "",
      "securityGroupId": "g-4NxWoxeq",
      "securityGroupName": "securitygroup-name",
      "vpcName": "vpc-name",
      "securityGroupRules": [
        {
          "destGroupId": "",
          "destIp": "all",
          "direction": "egress",
          "ethertype": "IPv4",
          "portRange": "1-65535",
          "protocol": "all",
          "securityGroupRuleId": "r-gkv8yupumvx2"
        }
      ]
    }
  ]
}
```

BindEnterpriseSecurityGroups绑定企业安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）绑定企业安全组（目前每个BLB实例最多支持绑定1个企业安全组）

请求结构

```
PUT /v{version}/blb/{blbld}/enterprise/securitygroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "enterpriseSecurityGroupIds": enterpriseSecurityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值bind
enterpriseSecurityGroupIds	List	是	RequestBody参数	绑定的企业安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/enterprise/securitygroup?bind&clientToken=be31b98c-5e41-4838-9830-9be700de5a20
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16

{
  "enterpriseSecurityGroupIds": ["esg-g8z4qfj0m0bu"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef474
```

UnbindEnterpriseSecurityGroups解绑企业安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）解绑企业安全组，支持批量解绑

请求结构

```
PUT /v{version}/blb/{blbld}/enterprise/securitygroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "enterpriseSecurityGroupIds": enterpriseSecurityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值unbind
enterpriseSecurityGroupIds	List	是	RequestBody参数	解绑的企业安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/enterprise/securitygroup?unbind&clientToken=be31b98c-5e41-4838-9830-9be700de5a21
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf17

{
  "enterpriseSecurityGroupIds": ["esg-gzsqygs59nxp"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef478
```

DescribeEnterpriseSecurityGroups查询BLB实例企业安全组列表

描述

按ID查询用户账户下指定LoadBalancer的企业安全组信息

请求结构

```
GET /v{version}/blb/{blbld}/enterprise/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	要查询的LoadBalancer的标识符

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
enterpriseSecurityGroupIds	List< BlbEnterpriseSecurityGroupModel >	企业安全组信息，由BlbEnterpriseSecurityGroupModel组成的集合

请求示例

```
GET /v1/blb/lb-j8UqijYf/enterprise/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: dfd5d01f-a869-49eb-aba5-d853a3350428
```

```
{
  "enterpriseSecurityGroups": [{
    "enterpriseSecurityGroupId": "esg-g8z4qfj0m0bu",
    "enterpriseSecurityGroupName": "test0627",
    "enterpriseSecurityGroupDesc": "",
    "enterpriseSecurityGroupRules": [{
      "remark": "全部协议",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv6",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu",
      "protocol": "all"
    },
    {
      "remark": "",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9je09z6bd9wv",
      "protocol": "tcp"
    },
    {
      "remark": "",
      "direction": "ingress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "2",
      "sourceIp": "all",
      "destIp": "",
      "enterpriseSecurityGroupRuleId": "esgr-jcej4juqbh49",
      "protocol": "tcp"
    }
  ]
}
```

普通型IPv6BLB接口

普通型IPv6BLB相关接口

接口跟普通型BLB接口一致，除了创建和查询BLB列表接口需要增加type:"ipv6"。

🔗 创建普通型IPv6BLB

请求结构

```
POST /v{version}/blb?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "type": "ipv6",
  "vpclId": "vpclId",
  ...
}
```

🔗 查询已有的普通型IPv6BLB列表

请求结构

```
GET /v{version}/blb?type=ipv6&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

说明：

- 建议优先使用应用型ipv6实例。

应用型IPv6BLB接口

应用型IPv6BLB实例相关接口

🔗 CreateLoadBalancer创建实例

描述

- 创建一个LoadBalancer，返回分配的服务地址及实例ID。
- 付费方式为后付费。
- 需要实名认证，若未通过实名认证可以前往[百度开放云官网控制台](#)中的安全认证下的实名认证中进行认证。

请求结构

```
POST /v{version}/appblb?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "type": type,
  "subnetId": subnetShortId,
  "vpcId": vpcShortId,
  "name": name,
  "desc": desc ,
  "address": address,
  "eip": eip,
  "tags":[
    {
      "tagKey": tagKey,
      "tagValue": tagValue
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
type	String	是	RequestBody参数	LoadBalancer的类型，这里需要传入"ipv6Application"
subnetId	String	是	RequestBody参数	LoadBalancer实例所属子网
vpcId	String	是	RequestBody参数	LoadBalancer实例vip所属VPC的vpcShortId
name	String	否	RequestBody参数	LoadBalancer的名称，方便记忆。长度1~65个字节，字母开头，可包含字母数字_/.字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	LoadBalancer实例的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空
addresses	String	否	RequestBody参数	指定负载均衡实例的私网IP地址，该地址必须包含在子网网段下。
eip	String	否	RequestBody参数	绑定已有的eipV6。

			数	
tags	List<TagModel>	否	RequestBody参数	待创建的标签键值对列表
billing	BillingFormCreate	否	RequestBody参数	订单相关信息，购买预付费blb的时候必填
performanceLevel	String	否	RequestBody参数	性能规格参数，默认为共享型。取值如下："small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型1，"large1"超大型1，"large2"超大型2，"large3"超大型3，如果是后付费按使用量计费的时候还可以传"unlimited"不限速
autoRenewLength	int	否	RequestBody参数	支持创建预付费BLB同时开通自动续费，根据autoRenewTimeUnit的取值有不同的范围，month 为1到9，year 为1到3
autoRenewTimeUnit	String	否	RequestBody参数	支持创建BLB同时开通自动续费，取值为 month 或 year （默认 month）
resourceGroupId	String	否	RequestBody参数	支持创建BLB同时绑定资源分组id
allowDelete	Boolean	否	RequestBody参数	是否允许删除。缺省值为true，代表允许删除
allocateIpv6	Boolean	否	RequestBody参数	是否分配ipv6地址。true代表分配ipv6地址，false代表不分配ipv6地址

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
address	String	分配的服务地址IP，通过这个IP即能访问该实例
name	String	LoadBalancer的名称
desc	String	LoadBalancer的描述
blbId	String	LoadBalancer的ID。后续针对该实例的操作，均需要在请求中带上此ID

请求示例

```
POST /v1/appblb?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

创建后付费blb实例

```
{
  "type": "ipv6Application",
  "name": "blb-for-test",
  "subnetId": "sbn-hj9k8v413tqm",
  "vpclId": "vpc-tuf4sqw82i5m",
  "address": "192.168.0.24",
  "eip": "2400:da00:e003:2011:0:1:0:da"
}
```

创建预付费blb实例

```
{
  "type": "ipv6Application",
  "vpclId": "vpc-fbsxgfs9ty60",
  "subnetId": "sbn-3317rkjfq8j9",
  "name": "ccq20240319",
  "billing": {
    "paymentTiming": "Prepaid",
    "reservation": {
      "reservationLength": 2
    }
  },
  "performanceLevel": "small1"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "blbId": "lb-BLuOPSLZ",
  "name": "blb-for-test",
  "desc": "",
  "address": "240c:4086:0:4501::2"
}
```

UpdateLoadBalancer更新实例

描述

- 更新一个LoadBalancer的配置信息。

请求结构

```
PUT /v{version}/appblb/{blbld}?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name": name
  "desc": desc
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	待更新的LoadBalancer的ID
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
name	String	否	RequestBody参数	LoadBalancer的名称，方便记忆。长度1~65个字节，字母开头，可包含字母数字_./字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	LoadBalancer实例的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空
allowDelete	Boolean	否	RequestBody参数	是否允许删除。缺省值为true，代表允许删除
allocateIpv6	Boolean	否	RequestBody参数	是否分配ipv6地址。true代表分配ipv6地址，false代表不分配ipv6地址

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-9f59e197?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "name": "blb-for-test",
  "desc": "用于测试环境"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeLoadBalancers查询已有的BLB实例

描述

- 查询用户账户下所有LoadBalancer的信息
- 支持按LoadBalancer的Name、Address进行模糊查询
- 支持按LoadBalancer的ID进行精确查询
- 支持查找绑定指定BCC的LoadBalancer，通过bccId参数指定
- 返回结果是多重查询条件的交集
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/apblb?type="ipv6Application"&address={address}&name={name}&blbId={blbId}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
type	String	是	Query参数	要查询的LoadBalancer的类型，这里需要传入"ipv6Application"
address	String	否	Query参数	要查询的LoadBalancer地址，格式为IPv6地址
name	String	否	Query参数	要查询的LoadBalancer名称
blbId	String	否	Query参数	要查询的LoadBalancer ID（只支持精确查询）
bccId	String	否	Query参数	绑定该BCC的LoadBalancer（只支持精确查询）
exactlyMatch	boolean	否	Query参数	为true表示查询条件为精准匹配，否则为模糊匹配。缺省值为false
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
blbList	List<AppBLBModel>	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb?type="ipv6Application"&name=test&maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "blbList": [
    {
      "blbId": "lb-a7e5zPPk",
      "status": available,
      "name": "test-blb",
      "desc": "用于生产环境",
      "vpId": "vpc-fy6vdykpwkqb",
      "address": "240c:4086:0:2c03::2"
    },
    {
      "blbId": "lb-gj5gVpeq",
      "status": available,
      "name": "nametest",
      "desc": "用于测试环境",
      "vpId": "vpc-a8n5p6kybbx4",
      "address": "240c:4086:0:2c03::2"
    }
  ],
  "marker": "blb-0A20F971",
  "nextMarker": "blb-0A20FB09",
  "isTruncated": true,
  "maxKeys": 2
}
```

 DescribeLoadBalancerDetail查询BLB实例详情

描述

- 按id查询用户账户下所有LoadBalancer的详细信息
- 包含LoadBalancer所有的监听器端口信息

请求结构

```
GET /v{version}/appblb/{blbld} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	要查询的LoadBalancer的标识符（必须输入完整的blbld）

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
blbId	String	LoadBalancer的标识符
status	blbStatus	BLB状态
desc	String	LoadBalancer的描述
address	String	LoadBalancer的内网地址
publicIp	String	LoadBalancer的公网地址
cidr	String	LoadBalancer所在网络cidr
vpcName	String	LoadBalancer所属vpc名称
subnetCidr	String	LoadBalancer所属子网cidr
subnetName	String	LoadBalancer所属子网名称
createTime	String	LoadBalancer创建时间
releaseTime	String	LoadBalancer自动释放时间
listener	List<ListenerModel>	LoadBalancer下挂载监听器列表
tags	List<TagModel>	标签键值对列表
allowDelete	Boolean	是否允许删除
paymentTiming	String	付款时间，预支付（Prepaid）或后支付（Postpaid）
billingMethod	String	计费方式，"ByCapacityUnit" 按使用量 "BySpec" 按固定规格
performanceLevel	String	性能规格。"small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型1，"large1"超大型1，"large2"超大型2，"large3"超大型3
expireTime	String	预付费实例的过期时间
publicIp	String	如果LoadBalancer绑定过EIPV6，则显示该项，否则不显示
eipRouteType	String	EIPV6线路类型

请求示例

```
GET /v1/appblb/lb-7fb1e926 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "blbId": "lb-gj5gVpeq",
  "status": "available",
  "name": "nametest",
  "desc": "",
  "address": "240c:4086:0:4501::2",
  "publicIp": "240c:4086:ff00::1:0:3ca",
  "cidr": "192.168.0.0/16",
  "vpcName": "test",
  "subnetName": "系统预定义子网",
  "subnetCidr": "192.168.0.0/20",
  "createTime": "2019-03-07T02:35:31Z",
  "releaseTime": "2019-03-07T02:35:31Z",
  "paymentTiming": "Prepaid",
  "billingMethod": "BySpec",
  "listener": [
    {
      "port": 80,
      "type": "TCP"
    }
  ]
}
```

🔗 DeleteLoadBalancer释放BLB实例

描述

- 释放指定LoadBalancer，被释放的LoadBalancer无法找回

请求结构

```
DELETE /v{version}/appblb/{blbId} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	待释放的LoadBalancer的ID

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
DELETE /v1/appblb/lb-j8UqijYf HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T07:09:25Z/1800/host/fa13c6057d34b0152a0959541f49bea9ae4eeece2317ac3e371dda33c27f5f76
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 07:09:45 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

应用型IPv6BLB监听器相关接口

CreateAppTCPListener创建TCP监听器

- 在指定LoadBalancer下，创建一个基于TCP协议的应用型blb监听器，监听一个前端端口，将发往该端口的所有TCP流量，根据策略进行转发。

请求结构

```
POST /v{version}/appblb/{blbId}/TCPListener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "tcpSessionTimeout":tcpSessionTimeout
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Request Body参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	Request Body参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
tcpSessionTimeout	int	否	Request Body参数	tcp设置连接超时时间（单位：秒），默认900，需为10-4000间的整数

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/TCPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppUDPListener创建UDP监听器

- 在指定LoadBalancer下，创建一个基于UDP协议的应用型监听器，监听一个前端端口，将发往该端口的所有UDP流量，根据

策略进行转发。

请求结构

```
POST /v{version}/appblb/{blbld}/UDPlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "udpSessionTimeout":udpSessionTimeout
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Request Body参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	Request Body参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
udpSessionTimeout	int	否	Request Body参数	udp会话超时时间。默认为90，需为5-4000间的整数，单位秒

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/UDPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "udpSessionTimeout":90
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppHTTPListener创建HTTP监听器

- 在指定LoadBalancer下，创建一个基于HTTP协议的应用型监听器，监听一个前端端口，将发往该端口的所有HTTP请求，根据策略转发到后端服务器监听的后端端口上

请求结构

```
POST /v{version}/appblb/{blbId}/HTTPlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "keepSession":keepSession,
  "keepSessionType":keepSessionType,
  "keepSessionTimeout":keepSessionTimeout,
  "keepSessionCookieName":keepSessionCookieName,
  "xForwardedFor":xForwardedFor,
  "xForwardedProto":xForwardedProto,
  "serverTimeout":serverTimeout,
  "redirectPort":redirectPort
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	否	RequestBody参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	RequestBody参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionTimeout	int	否	RequestBody参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	RequestBody参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	否	RequestBody参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	RequestBody参数	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议，默认关闭
serverTimeout	int	否	RequestBody参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
redirectPort	int	否	RequestBody参数	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/HTTPlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "xForwardedFor":true,
  "serverTimeout":1800,
  "redirectPort":0
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppHTTPSListener创建HTTPS监听器

- 在指定LoadBalancer下，创建一个基于HTTPS协议的应用型监听器，监听一个前端端口，将发往该端口的所有HTTPS请求，先通过SSL卸载转换为HTTP请求后，再根据策略转发到后端服务器监听的后端端口上

请求结构

```
POST /v{version}/appblb/{blbld}/HTTPSlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "keepSession":keepSession,
  "keepSessionType":keepSessionType,
  "keepSessionTimeout":keepSessionTimeout,
  "keepSessionCookieName":keepSessionCookieName,
  "xForwardedFor":xForwardedFor,
  "xForwardedProto":xForwardedProto,
  "serverTimeout":serverTimeout,
  "certIds":[id1, id2],
  "encryptionType":encryptionType,
  "encryptionProtocols":[protocol1, protacol2],
  "dualAuth":false,
  "clientCertIds":[id1, id2]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
			URI 参	

version	String	是	URL参 数	API版本号，当前取值1
blbld	String	是	URL参 数	所属LoadBalancer的标识符
clientToken	String	是	Query 参数	幂等性Token，长度不超过64位
listenerPort	int	是	Reques tBody参 数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	Reques tBody参 数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	否	Reques tBody参 数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	Reques tBody参 数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionTimeout	int	否	Reques tBody参 数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	Reques tBody参 数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	否	Reques tBody参 数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	Reques tBody参 数	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议，默认关闭
serverTimeout	int	否	Reques tBody参 数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
certIds	List<String>	是	Reques tBody参 数	监听器要加载的证书链
encryptionType	String	否	Reques tBody参 数	加密选项，默认tls_cipher_policy_default。支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	Reques tBody参 数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	Reques tBody参 数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	Reques tBody参 数	是否开启双向认证，默认为关闭

listenerPort	监听端口	否	body参数	监听端口监听地址，默认监听80
clientCertificates	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/HTTPSListener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "xForwardedFor":true,
  "serverTimeout":1800,
  "certIds":["cert-zfj2ey2z4kmm"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppSSLListener创建SSL监听器

- 在指定LoadBalancer下，创建一个基于SSL协议的应用型监听器，监听一个前端端口，将发往该端口的所有SSL流量，根据策略进行转发。

请求结构


```
POST /v{version}/appblb/{blbId}/SSLlistener?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort":listenerPort,
  "scheduler":scheduler,
  "certIds":[id1, id2],
  "encryptionType":encryptionType,
  "encryptionProtocols":[protocol1, protocol2],
  "dualAuth":false,
  "clientCertIds":[id1, id2]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	RequestBody参数	监听器的监听端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
certIds	List<String>	是	RequestBody参数	监听器要加载的证书链
encryptionType	String	否	RequestBody参数	加密选项，默认tls_cipher_policy_default。支持：tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocol	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCertificates	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/SSLlistener?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort":95,
  "scheduler":"LeastConnection",
  "certIds":["cert-zfj2ey2z4kmm"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateAppTCPLListener更新TCP监听器

- 更新指定LoadBalancer下的TCP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/appblb/{blbld}/TCPlistener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler":scheduler,
  "tcpSessionTimeout":tcpSessionTimeout
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Query参数	监听器端口号
scheduler	String	否	Request Body参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
tcpSessionTimeout	int	否	Request Body参数	tcp设置连接超时时间（单位：秒），默认900，需为10-4000间的整数

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/TCPlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 UpdateAppUDPListener更新UDP监听器

- 更新指定LoadBalancer下的UDP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/appblb/{blbld}/UDPlistener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler":scheduler,
  "udpSessionTimeout":udpSessionTimeout
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	URL参数	监听器的端口号
scheduler	String	是	Request Body参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
udpSessionTimeout	int	否	Request Body参数	udp会话超时时间。默认为90，需为5-4000间的整数，单位秒

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/UDPlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "udpSessionTimeout": 90
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateAppHTTPListener更新HTTP监听器

- 更新指定LoadBalancer下的HTTP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定

请求结构

```
PUT /v{version}/appblb/{blbld}/HTTPlistener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler": scheduler,
  "keepSession": keepSession,
  "keepSessionType": keepSessionType,
  "keepSessionTimeout": keepSessionTimeout,
  "keepSessionCookieName": keepSessionCookieName,
  "xForwardedFor": xForwardedFor,
  "xForwardedProto": xForwardedProto,
  "serverTimeout": serverTimeout,
  "redirectPort": redirectPort
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	否	RequestBody参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	RequestBody参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionTimeout	int	否	RequestBody参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	RequestBody参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	否	RequestBody参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	RequestBody参数	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议，默认关闭
serverTimeout	int	否	RequestBody参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
redirectPort	int	否	RequestBody参数	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqjYf/HTTPlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "xForwardedFor": true,
  "serverTimeout": 1800,
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 UpdateAppHTTPSListener更新HTTPS监听器

- 更新指定LoadBalancer下的HTTPS监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定

请求结构

```
PUT /v{version}/appblb/{blbld}/HTTPSListener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler": scheduler,
  "keepSession": keepSession,
  "keepSessionType": keepSessionType,
  "keepSessionDuration": keepSessionDuration,
  "keepSessionCookieName": keepSessionCookieName,
  "xForwardedFor": xForwardedFor,
  "xForwardedProto": xForwardedProto,
  "serverTimeout": serverTimeout,
  "certIds": [id1, id2],
  "encryptionType": encryptionType,
  "encryptionProtocols": [protocol1, protocol2],
  "dualAuth": false,
  "clientCertIds": [id1, id2]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
			URL参	

blbld	String	是	数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
scheduler	String	是	RequestBody参数	负载均衡算法，支持2种方式："RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	否	RequestBody参数	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器，默认关闭
keepSessionType	String	否	RequestBody参数	会话保持的cookie处理方式，当且仅当开启会话保持时有效，支持"insert"/"rewrite"，默认"insert"
keepSessionTimeout	int	否	RequestBody参数	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效，默认3600s，需为1-15552000间的整数
keepSessionCookieName	String	否	RequestBody参数	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	否	RequestBody参数	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址，默认关闭
xForwardedProto	boolean	否	RequestBody参数	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议，默认关闭
serverTimeout	int	否	RequestBody参数	后端服务器最大超时（单位：秒），默认30s，需为1-3600间的整数
certIds	List<String>	是	RequestBody参数	监听器要加载的证书链
encryptionType	String	否	RequestBody参数	加密选项。支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCertId	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链

s	ing>		数	

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/HTTPSlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "xForwardedFor": true,
  "serverTimeout": 1800,
  "certIds": ["cert-zfj2ey2z4kmm"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 UpdateAppSSLListener更新SSL监听器

- 更新指定LoadBalancer下的SSL监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/appblb/{blbId}/SSLlistener?clientToken={clientToken}&listenerPort={listenerPort} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "scheduler": scheduler,
  "certIds": [id1, id2],
  "encryptionType": encryptionType,
  "encryptionProtocols": [protocol1, protocol2],
  "dualAuth": false,
  "clientCertIds": [id1, id2]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
listenerPort	int	是	Query参数	要更新的监听器端口，需为1-65535间的整数
scheduler	String	否	RequestBody参数	负载均衡算法，支持3种方式："RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
certIds	List<String>	是	RequestBody参数	监听器要加载的证书链
encryptionType	String	否	RequestBody参数	加密选项。支持： tls_cipher_policy_default/tls_cipher_policy_1_1/tls_cipher_policy_1_2/tls_cipher_policy_1_2_secure/userDefind，对应的加密协议和加密套件参见： encryption
encryptionProtocols	List<String>	否	RequestBody参数	当encryptionType值为userDefind时协议类型列表，是以"tlsv10"、"tlsv11"、"tlsv12"三种协议组合形成的字符串列表
appliedCiphers	String	否	RequestBody参数	加密套件参见： encryption ，不同加密套件用冒号":"隔开
dualAuth	boolean	否	RequestBody参数	是否开启双向认证，默认为关闭
clientCertIds	List<String>	否	RequestBody参数	当dualAuth为true时，加载的客户端证书链

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/TCPlistener?clientToken=asdfg&listenerPort=80 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "scheduler": "LeastConnection",
  "certIds": ["cert-zfj2ey2z4kmm"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeAppTCPListeners查询应用型TCP监听器

- 查询指定LoadBalancer下所有TCP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/TCPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List<AppTCPListenerModel>	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/TCPlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin"
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin"
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

 DescribeAppUDPListeners查询UDP监听器

- 查询指定LoadBalancer下所有UDP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/UDPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< AppUDPLListenerModel >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/UDPLlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin",
      "udpSessionTimeout":90
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin",
      "udpSessionTimeout":90
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeAppHTTPListeners查询HTTP监听器

- 查询指定LoadBalancer下所有HTTP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/HTTPlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys}
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List<AppHTTPListenerModel>	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/HTTPlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例


```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 30,
      "redirectPort": 80
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 1800,
      "redirectPort": 80
    }
  ],
  "marker": "listener-005A",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeAppHTTPSListeners查询HTTPS监听器

- 查询指定LoadBalancer下所有HTTPS监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/HTTPSlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys}
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< AppHTTPSListenerModel >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/HTTPSlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 30,
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 1800,
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-005A",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

DescribeAppSSLListeners 查询应用型SSL监听器

- 查询指定LoadBalancer下所有SSL监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/SSLlistener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
listenerList	List< AppSSLListenerModel >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/SSLlistener?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin",
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin",
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

 DescribeAppAllListeners查询所有监听

请求结构

```
GET /v{version}/appblb/{blbld}/listener?listenerPort={listenerPort}&marker={marker}&maxKeys={maxKeys} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
listenerPort	int	否	Query参数	要查询的监听器端口
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量通常不超过1000，缺省值为1000

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
marker	String	标记查询的起始位置
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量
listenerList	List<AppListenerModel>	包含查询的监听列表

请求示例

```
GET /v1/appblb/lb-a7942975/listener HTTP/1.1
Host: blb.bj.baidubce.com
ContentType: application/json
Authorization: bce auth v1/f81d3b34e48048fbb2634dc7882d7e21/2017 03
11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7 4ad5 451d 9215 71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content Type: application/json;charset=UTF 8
Server: BWS

{
  "nextMarker": "",
  "marker": "",
  "maxKeys": 1,
  "isTruncated": true,
  "listenerList": [
    {
      "listenerPort": 53,
      "scheduler": "RoundRobin",
      "listenerType": "UDP"
    },
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin",
      "listenerType": "HTTP",
      "keepSession": false,
      "xForwardFor": true,
      "xForwardedProto": false,
      "serverTimeout": 30,
      "redirectPort": 0
    },
    {
      "listenerPort": 81,
      "scheduler": "RoundRobin",
      "listenerType": "TCP",
      "tcpSessionTimeout": 900
    },
    {
      "listenerPort": 443,
      "scheduler": "RoundRobin",
      "listenerType": "SSL",
      "certIds": [
        "cert-gs8bktrm7drp"
      ]
    }
  ]
}
```

```

    },
    "dualAuth": false,
    "encryptionType": "tls_cipher_policy_ie",
    "encryptionProtocols": [
        "ssl3",
        "tlsv10",
        "tlsv11",
        "tlsv12"
    ],
    "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-ECDSA-RC4-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-RSA-RC4-SHA:ECDHE-RSA-DES-CBC3-SHA:RC4-SHA:AES128-GCM-SHA256:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES256-SHA:AES128-SHA256:DES-CBC3-SHA"
  },
  {
    "listenerPort": 444,
    "scheduler": "RoundRobin",
    "listenerType": "HTTPS",
    "keepSession": false,
    "xForwardFor": false,
    "xForwardedProto": false,
    "serverTimeout": 30,
    "certIds": [
      "cert-gs8bktrm7drp"
    ],
    "dualAuth": false,
    "encryptionType": "tls_cipher_policy_1_1",
    "encryptionProtocols": [
      "tlsv11",
      "tlsv12"
    ],
    "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA"
  }
]
}

```

🔗 DeleteAppListeners释放监听器

- 释放指定LoadBalancer下的监听器，监听器通过监听端口来指定
- 支持批量释放

请求结构

```
PUT /v{version}/appblb/{blbld}/listener?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "portList":[port1,port2]
}
或者
{
  "portTypeList":[
    {
      "port":port,
      "type":"type"
    },
    {
      "port":port,
      "type":"type"
    },
  ],
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为batchdelete
clientToken	String	是	Query参数	幂等性Token
portList	List<int>	否	Request Body参数	所有待释放的监听器的端口，一起组成一个数组
portTypeList	List<PortTypeModel>	否	Request Body参数	所有待释放的监听器的端口和协议，一起组成一个数组，如果开通了同端口多协议，并且只想删除其中一个协议，使用这个参数。portList和portTypeList至少传一个。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/listener?batchdelete&clientToken=abc
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "portList":[80,87]
}
或者
{
  "portTypeList":[
    {
      "port":80,
      "type":"TCP"
    },
    {
      "port":80,
      "type":"UDP"
    },
  ],
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreatePolicys创建策略

- 在指定应用型BLB监听器端口下创建策略

请求结构

```
POST /v{version}/appblb/{blbld}/policys?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "listenerPort": listenerPort,
  "appPolicyVos": [
    {
      "appServerGroupId": appServerGroupId,
      "backendPort": backendPort,
      "priority": priority,
      "desc": desc,
      "ruleList": [
        {
          "key": key,
          "value": value
        }
      ]
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token
listenerPort	int	是	RequestBody参数	对应所在BLB下监听器端口号
appPolicyVos	List<CreateAppPolicy>	是	RequestBody参数	监听器绑定策略列表，当监听器类型为TCP、UDP、SSL时，只能创建一条且只支持全匹配；当监听器类型为http、https监听器支持多个policy
type	String	否	RequestBody参数	当监听器端口下有多个协议时，type必传

返回状态码

成功返回200，失败返回见[错误码](#) 返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/policys?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "listenerPort": 98,
  "type": "HTTP",
  "appPolicyVos": [
    {
      "appServerGroupId": "sg-db3975dc",
      "backendPort": 83,
      "priority": 100,
      "desc": "aa",
      "ruleList": [
        {
          "key": "*",
          "value": "*"
        }
      ]
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

 DescribePolicys查询对应BLB端口下策略信息

- 查询指定LoadBalancer下所有服务器组的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/policys?marker={marker}&maxKeys={maxKeys}&port={port} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
port	int	是	Query参数	BLB下监听器端口号
type	String	否	Query参数	当监听器端口下有多个协议时，type必传
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
policyList	List< AppPolicy >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/policys?maxKeys=2&port=80&type=HTTP HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "marker": "rs-012310",
  "isTruncated": true,
  "maxKeys": 1000,
  "nextMarker": "rs-013214",
  "policyList": [
    {
      "id": "policy_id",
      "appServerGroupid": "sg_id",
      "appServerGroupName": "sg_name",
      "frontendPort": 80,
      "type": "TCP",
      "backendPort": 98,
      "portType": "TCP",
      "priority": 100,
      "desc": "aa",
      "groupType": "Server",
      "ruleList": [
        {
          "key": "host",
          "value": "*.baidu.com"
        }
      ]
    }
  ]
}
```

DeletePolicys批量删除策略

- 批量删除对应BLB端口下的策略

请求结构

```
PUT /v{version}/appblb/{blbld}/policys?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

```
{
  "port": port,
  "policyIdList": ["policyId1", "policyId2", "policyId3"]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为batchdelete
clientToken	String	是	Query参数	幂等性Token
port	int	是	RequestBody参数	要删除策略前端端口
policyIdList	List<String>	是	RequestBody参数	所有待释放的策略标识符，一起组成一个数组
type	String	否	RequestBody参数	当监听器端口下有多个协议时，type必传

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/policys?batchdelete&clientToken=your_clientToken
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "port":90,
  "policyIdList":["ab48bbe", "2520741", "a78e04"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

应用型IPv6BLB服务器组相关接口

CreateAppServerGroup创建应用型服务器组

- 在指定应用型BLB下，创建一个服务器组，用来绑定后端服务器，以及为监听器开放相应的端口

请求结构

```
POST /v{version}/appblb/{blbld}/appservergroup?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name":name,
  "desc":desc,
  "backendServerList": [
    {
      "instanceId": instanceId,
      "weight": weight
    },
    {
      "instanceId": instanceId,
      "weight": weight
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
name	String	否	RequestBody参数	服务器组的名称，方便记忆。长度1~65个字节，字母开头，_可包含字母数字/.字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	服务器组的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空。
backendServerList	List<AppBackendServer>	否	RequestBody参数	服务器组绑定的后端服务器列表

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
id	String	服务器组id
name	String	服务器组的名称
desc	String	服务器组的描述
status	String	服务器组状态，详见 blbStatus

请求示例

```
POST /v1/appblb/lb-j8UqijYf/appservergroup?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "name": "name1",
  "desc": "aaa",
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "weight": 20
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "id": "sg_short_id",
  "name": "name",
  "desc": "aaa",
  "status": "available"
}
```

UpdateAppServerGroup更新服务器组

- 更新指定LoadBalancer下的TCP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

请求结构

```
PUT /v{version}/appblb/{blbld}/appservergroup?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId,
  "name": name,
  "desc": desc
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgId	String	是	RequestBody参数	要更新的服务器组的id
name	String	否	RequestBody参数	服务器组的名称，方便记忆。长度1~65个字节，字母开头，可包含字母数字_./:字符。若不传该参数，会自动生成
desc	String	否	RequestBody参数	服务器组的描述，便于用户添加更详细的描述信息。长度0~450个字节，支持中文。默认为空

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/appservergroup?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgId": "sgId",
  "name": "name1",
  "desc": "aaa"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

 DescribeAppServerGroup查询服务器组

- 查询指定LoadBalancer下所有服务器组的信息
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbld}/appservergroup?name={name}&marker={marker}&maxKeys={maxKeys}&exactlyMatch={exactlyMatch} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
name	String	否	Query参数	要查询的服务器组名称
exactlyMatch	boolean	否	Query参数	设置name是否全局匹配
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
appServerGroupList	List< AppServerGroup >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/appservergroup?maxKeys=2 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "appServerGroupList":[
    {
      "id": "sg_short_id",
      "name": "name",
      "desc": "bbb",
      "status": "available" ,
      "portList":
      [{
        "id": "port_id"
        "port": BACKEND_PORT,
        "type": "TCP",
        "healthCheck": "TCP",
        "healthCheckNormalStatus": "^[1-5][0-9]*$",
        "healthCheckPort": 90,
        "healthCheckTimeoutInSecond": 2,
        "healthCheckDownRetry": 3,
        "healthCheckUpRetry": 3,
        "healthCheckIntervalInSecond": 1,
        "healthCheckUriPath": "HEALTH_CHECK_URL_PATH",
        "healthCheckHost": www.xxx,
        "status": "available"
      }]
    }
  ],
  "marker": "sg-012310",
  "isTruncated": false,
  "maxKeys": 1000,
  "nextMarker": "sg-013214"
}
```

DeleteAppServerGroup删除服务器组

- 删除服务器组，通过服务器组id指定

请求结构

```
PUT /v{version}/appblb/{blbld}/appservergroup?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为delete
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgId	String	是	RequestBody参数	要删除的服务器组id

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/appservergroup?delete&clientToken=be31b98c-5e41-4838-9830-9be700de5a20
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "sgId": "abc"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateAppServerGroupPort创建应用型服务器组端口

- 在指定应用型BLB下，创建一个服务器组后端端口，将发往该端口的所有流量按权重轮询分发到其绑定的对应服务器列表中的服务器

请求结构

```
POST /v{version}/appblb/{blbld}/appservergroupport?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgld":sgld,
  "port":port,
  "type":type,
  "enableHealthCheck":enableHealthCheck,
  "healthCheck":healthCheck,
  "healthCheckPort":healthCheckPort,
  "healthCheckUriPath":healthCheckUriPath,
  "healthCheckTimeoutInSecond":healthCheckTimeoutInSecond,
  "healthCheckIntervalInSecond":healthCheckIntervalInSecond,
  "healthCheckDownRetry":healthCheckDownRetry,
  "healthCheckUpRetry":healthCheckUpRetry,
  "healthCheckNormalStatus":healthCheckNormalStatus,
  "healthCheckHost":healthCheckHost,
  "udpHealthCheckString":udpHealthCheckString
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	Request Body参数	端口所属服务器组标识符
port	int	是	Request Body参数	端口号，需为1-65535间的整数
type	String	是	Request Body参数	监听端口的协议类型，"TCP"/"UDP"/"HTTP"
enableHealthCheck	Boolean	否	Request Body参数	是否开启健康检查，默认true
healthCheck	String	否	Request Body参数	健康检查协议，支持"HTTP"/"TCP"/"UDP"/"ICMP"， TCP默认TCP，UDP默认UDP，HTTP默认HTTP
healthCheckPort	int	否	Request Body参数	健康检查端口，允许范围为1-65535，默认为port的取值，当健康检查协议为"HTTP"时生效
healthCheckUriPath			Request	

healthCheckPath	String	否	Body参数	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckTimeoutInSecond	int	否	Request Body参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckIntervalInSecond	int	否	Request Body参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
healthCheckDownRetry	int	否	Request Body参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthCheckUpRetry	int	否	Request Body参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	否	Request Body参数	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckHost	String	否	Request Body参数	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
udpHealthCheckString	String	否	Request Body参数	UDP服务的健康检查字段。当健康检查协议为"UDP"时，则必传

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
id	String	服务器组端口id
status	String	服务器组端口状态，详见 blbStatus

请求示例

```
POST /v1/appblb/lb-j8UqijYf/appservergroupport?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgId": "sgId",
  "type": "TCP",
  "port": 80,
  "enableHealthCheck": true,
  "healthCheck": "TCP",
  "healthCheckTimeoutInSecond": 2,
  "healthCheckDownRetry": 3,
  "healthCheckUpRetry": 3,
  "healthCheckIntervalInSecond": 3
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "id": "sg_port_id",
  "status": "available"
}
```

UpdateAppServerGroupPort更新服务器组端口

- 根据id更新服务器组端口。

请求结构

```
PUT /v{version}/appblb/{blbId}/appservergroupport?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId,
  "portId": portId,
  "enableHealthCheck": enableHealthCheck,
  "healthCheck": healthCheck,
  "healthCheckPort": healthCheckPort,
  "healthCheckUriPath": healthCheckUriPath,
  "healthCheckTimeoutInSecond": healthCheckTimeoutInSecond,
  "healthCheckIntervalInSecond": healthCheckIntervalInSecond,
  "healthCheckDownRetry": healthCheckDownRetry,
  "healthCheckUpRetry": healthCheckUpRetry,
  "healthCheckNormalStatus": healthCheckNormalStatus,
  "healthCheckHost": healthCheckHost,
  "udpHealthCheckString": udpHealthCheckString
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	Request Body参数	端口所属服务器组标识符
portld	String	是	Request Body参数	要更新的服务器组端口的标识符
enableHealthCheck	Boolean	否	Request Body参数	是否开启健康检查，默认true
healthCheck	String	否	Request Body参数	健康检查协议，支持"HTTP"/"TCP"/"UDP"/"ICMP"，TCP默认TCP，UDP默认UDP，HTTP默认HTTP
healthCheckPort	int	否	Request Body参数	健康检查端口，默认为port的取值，当健康检查协议为"HTTP"时生效
healthCheckUriPath	String	否	Request Body参数	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckTimeoutInSecond	int	否	Request Body参数	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckIntervalInSecond	int	否	Request Body参数	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
healthCheckDownRetry	int	否	Request Body参数	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthCheckUpRetry	int	否	Request Body参数	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	否	Request Body参数	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckHost	String	否	Request Body参数	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
udpHealthCheckString	String	否	Request Body参数	UDP服务的健康检查字段。当健康检查协议为"UDP"时，则必传

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/appservergroupport?clientToken=abc HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgId": "abc",
  "portId": "dbd",
  "enableHealthCheck": true,
  "healthCheck": "TCP",
  "healthCheckTimeoutInSecond": 2,
  "healthCheckDownRetry": 3,
  "healthCheckUpRetry": 3,
  "healthCheckIntervalInSecond": 1
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DeleteAppServerGroupPort删除服务器组端口

- 删除服务器组端口，通过服务器组id指定

请求结构

```
PUT /v{version}/appblb/{blbId}/appservergroupport?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId,
  "portIdList": [portid1, portid2]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	必须为batchdelete
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	RequestBody参数	要删除的端口所在服务器组id
portIdList	List<String>	是	RequestBody参数	所有待释放的端口id，一起组成一个数组

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/appservergroupport?batchdelete&clientToken=asdfg
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "sgld": "abc",
  "portIdList": ["a", "b"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

CreateBlbRs添加应用型BLB后端RS

- 在指定应用型BLB和服务器组下绑定后端服务器RS

请求结构

```
POST /v{version}/appblb/{blbld}/blbrs?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgld":sgld,
  "backendServerList": [
    {
      "instanceId": instanceId,
      "weight": weight
    },
    {
      "instanceId": instanceId,
      "weight": weight
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	RequestBody参数	RS所属服务器组id
backendServerList	List<AppBackendServer>	是	RequestBody参数	后端服务器列表

成功返回200，失败返回见[错误码](#)

返回状态码

成功返回200，失败返回见[错误码](#) 返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/appblb/lb-j8UqijYf/blbrs?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgId": "sg-db3975dc",
  "backendServerList":[
    {
      "instanceId": "i-b3Dq3ea3",
      "weight": 20
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

UpdateBlbRs更新服务器组下挂载的RS权重

- 更新指定服务器组下的RS信息

请求结构

```
PUT /v{version}/appblb/{blbld}/blbrs?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId,
  "backendServerList":[
    {
      "instanceId": instanceId,
      "weight": weight
    }
  ]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgId	String	是	RequestBody参数	RS所属服务器组id
backendServerList	List<AppBackendServer>	是	RequestBody参数	后端服务器列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/blbrs?clientToken=asdfg HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/5e5a8adf11ae475ba95f1bd38228b44f/2016-04-10T08:26:52Z/1800/host;x-bce-date/ec3c0069f9abb1e247773a62707224124b2b31b4c171133677f9042969791f02

{
  "sgId": "sg-db3975dc",
  "backendServerList":[
    {
      "instanceId": "i-b3Dq3ea3",
      "weight": 20
    }
  ]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2016 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

DescribeBlbRs查询服务器组下的RS列表信息

- 查询指定LoadBalancer下所有服务器组的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

请求结构

```
GET /v{version}/appblb/{blbId}/blbrs?marker={marker}&maxKeys={maxKeys}&sgId={sgId} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
sgId	String	是	Query参数	RS所属服务器组id
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	int	否	Query参数	每页包含的最大数量，最大数量不超过1000。缺省值为1000

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
backendServerList	List< AppBackendServer >	包含查询结果的列表
marker	String	标记查询的起始位置，若结果列表为空，此项不存在
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量

请求示例

```
GET /v1/appblb/lb-j8UqijYf/blbrs?maxKeys=2&sgId=sg-db3975dc HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "marker": "rs-012310",
  "isTruncated": false,
  "maxKeys": 1000,
  "nextMarker": "rs-013214",
  "backendServerList": [
    {
      "instanceId": "i-7WrgmIY",
      "privateIp": "240c:4086:0:4501::5",
      "weight": "WEIGHT",
      "portList": [
        {
          "listenerPort": 80,
          "backendPort": "80",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "policyId": "policy-260061dc",
          "status": "Alive"
        }
      ]
    }
  ]
}
```

 DeleteBlbRs删除服务器组下挂载的rs

- 删除服务器组，通过服务器组id指定

请求结构

```
PUT /v{version}/appblb/{blbId}/blbrs?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "sgId": sgId,
  "backendServerIdList": ["Serverid1", "Serverid2", "Serverid3"]
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
action	String	是	Query参数	描述要进行的操作，本接口取batchdelete
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
sgld	String	是	RequestBody参数	要删除的RS所属服务器组id
backendServerIdList	List<String>	是	RequestBody参数	所有待释放的后端服务器标识符，一起组成一个数组

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/appblb/lb-j8UqijYf/blbrs?batchdelete&clientToken=be31b98c-5e41-4838-9830-9be700de5a20
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T03:26:45Z/1800/host/c863c8b9f75ff307eb1a3c564307b0833acb9d857c65de108258c3edd4cd0118

{
  "sgld": "abc",
  "backendServerIdList": ["ab48bbe", "2520741", "a78e04"]
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 03:54:29 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

 DescribeRsMount查询服务器组下绑定的server

- 查询服务器组下绑定的server

请求结构

```
GET /v{version}/appblb/{blbld}/blbrsmount?sgld={sgld} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
sgId	String	是	Query参数	RS所属服务器组id

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
backendServerList	List< AppBackendServer >	包含查询结果的列表

请求示例

```
GET /v1/appblb/lb-j8UqijYf/blbrsmount?sgId=abc HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "privateIp": "240c:4086:0:4501::5",
      "weight": "WEIGHT",
      "portList": [
        {
          "listenerPort": "LISTENER_PORT1",
          "backendPort": "BACKEND_PORT1",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "portId": "port_xxx",
          "policyId": "policy_xxx",
          "status": "Alive"
        },
        {
          "listenerPort": "LISTENER_PORT2",
          "backendPort": "BACKEND_PORT2",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "portId": "port_xxx",
          "policyId": "policy_xxx",
          "status": "Alive"
        }
      ]
    }
  ]
}
```

DescribeRsUnMount查询服务器组下未绑定的RS

- 查询服务器组下未绑定的RS

请求结构

```
GET /v{version}/appblb/{blbld}/blbrsunmount?sgld={sgld} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
sgld	String	是	Query参数	RS所属服务器组id

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
backendServerList	List< AppBackendServer >	包含查询结果的列表

请求示例

```
GET /v1/appblb/lb-j8UqijYf/blbrsunmount?sgld=abc HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/ae4780f344774daa832f34caadd4d7de/2016-06-03T06:42:31Z/1800/host/3f26e5a322d41fa0f3315e906342c2691fe5ee35d21c7e66e425b4ee0f67ced3
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Fri, 03 Jun 2016 06:42:44 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "privateIp": "240c:4086:0:4501::5",
      "weight": null,
      "portList": []
    }
  ]
}
```

应用型IPv6BLB安全组相关接口

 BindSecurityGroups绑定普通安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）绑定普通安全组，支持批量绑定

请求结构

```
PUT /v{version}/blb/{blbld}/securitygroup?action={action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "securityGroupIds": securityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值bind
securityGroupIds	List	是	RequestBody参数	绑定的普通安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/securitygroup?bind&clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16

{
  "securityGroupIds": ["g-4NxWoxeq", "g-4NxWoxea"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef474
```

🔗 UnbindSecurityGroups解绑普通安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）解绑普通安全组，支持批量解绑

请求结构

```
PUT /v{version}/blb/{blbId}/securitygroup?action={action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "securityGroupIds": securityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值unbind
securityGroupIds	List	是	RequestBody参数	解绑的普通安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/securitygroup?unbind&clientToken=be31b98c-5e41-4838-9830-9be700de5a21 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf17

{
  "securityGroupIds": ["g-4NxWoxeq", "g-4NxWoxea"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef478
```

 DescribeSecurityGroups查询BLB实例普通安全组列表

描述

按ID查询用户账户下指定LoadBalancer的安全组信息

请求结构

```
GET /v{version}/blb/{blbId}/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	要查询的LoadBalancer的标识符

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
securityGroups	List< BlbSecurityGroupModel >	普通安全组信息，由BlbSecurityGroupModel组成的集合

请求示例

```
GET /v1/blb/lb-j8UqijYf/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: dfd5d01f-a869-49eb-aba5-d853a3350428

{
  "blbSecurityGroups": [
    {
      "securityGroupDesc": "",
      "securityGroupId": "g-4NxWoxeq",
      "securityGroupName": "securitygroup-name",
      "vpcName": "vpc-name",
      "securityGroupRules": [
        {
          "destGroupId": "",
          "destIp": "all",
          "direction": "egress",
          "ethertype": "IPv6",
          "portRange": "1-65535",
          "protocol": "all",
          "securityGroupRuleId": "r-gkv8yupumvx2"
        }
      ]
    }
  ]
}
```

BindEnterpriseSecurityGroups绑定企业安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）绑定企业安全组（目前每个BLB实例最多支持绑定1个企业安全组）

请求结构

```
PUT /v{version}/blb/{blbId}/enterprise/securitygroup?action={action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "enterpriseSecurityGroupIds": enterpriseSecurityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值bind
enterpriseSecurityGroupIds	List	是	RequestBody参数	绑定的企业安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/enterprise/securitygroup?bind&clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdf16

{
  "enterpriseSecurityGroupIds": ["esg-g8z4qfj0m0bu"]
}
```

响应示例


```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef474
```

UnbindEnterpriseSecurityGroups解绑企业安全组

描述

为指定LoadBalancer（普通型、应用型、IPv6型）解绑企业安全组，支持批量解绑

请求结构

```
PUT /v{version}/blb/{blbld}/enterprise/securitygroup?action={action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "enterpriseSecurityGroupIds": enterpriseSecurityGroupIds
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	所属LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	本接口中该参数取值unbind
enterpriseSecurityGroupIds	List	是	RequestBody参数	解绑的企业安全组ID列表

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/lb-j8UqijYf/enterprise/securitygroup?unbind&clientToken=be31b98c-5e41-4838-9830-9be700de5a21
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf17

{
  "enterpriseSecurityGroupIds": ["esg-gzsqygs59nxp"]
}
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: 70b6324a-7b10-4736-8872-4c29be8ef478
```

DescribeEnterpriseSecurityGroups查询BLB实例企业安全组列表

描述

按ID查询用户账户下指定LoadBalancer的企业安全组信息

请求结构

```
GET /v{version}/blb/{blbld}/enterprise/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	要查询的LoadBalancer的标识符

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
enterpriseSecurityGroupIds	List< BlbEnterpriseSecurityGroupModel >	企业安全组信息，由BlbEnterpriseSecurityGroupModel组成的集合

请求示例

```
GET /v1/blb/lb-j8UqijYf/enterprise/securitygroup HTTP/1.1
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16
```

响应示例

```
HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Thu, 28 Apr 2022 11:06:00 GMT
Content-Type: application/json; charset=UTF-8
X-Bce-Request-Id: dfd5d01f-a869-49eb-aba5-d853a3350428

{
  "enterpriseSecurityGroups": [{
    "enterpriseSecurityGroupId": "esg-g8z4qfj0m0bu",
    "enterpriseSecurityGroupName": "test0627",
    "enterpriseSecurityGroupDesc": "",
    "enterpriseSecurityGroupRules": [{
      "remark": "全部协议",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv6",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu",
      "protocol": "all"
    }]
  }]
}
```

负载均衡专属集群LBDC接口

LBDC实例相关接口

 创建LBDC

请求结构

```
POST /v{version}/lbdc?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "name": name,
  "type": type,
  "ccuCount": ccuCount,
  "desc": desc,
  "billing": {
    "paymentTiming": paymentTiming,
  "reservation" : {
    "reservationLength": reservationLength
  }
},
"renewReservation":{
  "reservationLength": reservationLength
},
"tags": [
  {
    "tagKey": "hikwnf",
    "tagValue": "nwklwmflk"
  }
]
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
name	String	是	RequestBody参数	集群名称，长度1~65个字节，字母开头，_可包含字母数字-/.字符
type	String	是	RequestBody参数	集群类型，取值为"4Layer"或者"7Layer"
ccuCount	int	是	RequestBody参数	集群性能容量单位CCU（Cluster Capacity Unit）是用来衡量BLB集群处理流量时涉及各个指标。
desc	String	否	RequestBody参数	LBDC的描述，最大支持200字符
billing	BillingForCreate	是	RequestBody参数	订单信息
renewReservation	ReservationForCreate	否	RequestBody参数	自动续费信息，是否开启自动续费任务
tags	List< TagModel >	否	RequestBody参数	标签

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
id	String	集群id
type	String	集群类型
desc	String	描述信息

请求示例

```
POST /v1/lbdc?clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2020-07-14T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863fffac5190fdec1ec24b7c03eb5d67d2e1de
{
  "name": "abc",
  "type": "4Layer",
  "ccuCount": 1,
  "desc": "",
  "billing": {
    "paymentTiming": "Prepaid",
    "reservation" : {
      "reservationLength": 1
    }
  },
  "renewReservation":{
    "reservationLength": 1
  },
  "tags": [
    {
      "tagKey": "hikwnf",
      "tagValue": "nwklwmflk"
    }
  ]
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Tus, 14 Jul 2020 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "id": "bgw_group-a3t786p1",
  "name": "abc",
  "type": "4Layer",
  "desc":""
}
```

 升级LBDC

请求结构

```
PUT /v{version}/lbdc/{id}?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "ccuCount": ccuCount
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
id	String	是	URL参数	升级的集群id
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
action	String	是	Query参数	本接口中该参数取值resize
ccuCount	int	是	RequestBody参数	集群性能容量单位CCU（Cluster Capacity Unit）是用来衡量BLB集群处理流量时涉及的各项指标。

返回头域

除公共头域，无其它特殊头域。

返回参数

无特殊返回参数

请求示例

```
PUT /v1/lbdc/bgw_group-a3t786p1?resize&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2020-07-14T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863fffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "ccuCount": 1
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Tus, 14 Jul 2020 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

 续费LBDC

请求结构

```
PUT /v{version}/lbdc/{id}?{action}&clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "billing":{
    "reservation":{
      "reservationLength": reservationLength
    }
  }
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
id	String	是	URL参数	续费的集群id
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
action	String	是	Query参数	本接口中该参数取值purchaseReserved
billing	BillingForRenew	是	RequestBody参数	订单信息

返回头域

除公共头域，无其它特殊头域。

返回参数

无特殊返回参数

请求示例

```
PUT /v1/lbdc/bgw_group-a3t786p1?purchaseReserved&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2020-07-14T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "billing":{
    "reservation":{
      "reservationLength":1
    }
  }
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Tus, 14 Jul 2020 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 LBDC列表

请求结构

```
GET /v{version}/lbdc?id={id}&name={name} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
id	String	否	Query参数	要查询的集群id
name	String	否	Query参数	要查询的集群名称

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
marker	String	标记查询的起始位置
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量
clusterList	List< ClusterModel >	包含查询结果的LBDC列表

请求示例

```
GET /v1/lbdc?id=bgw_group-a3t786p1 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2020-07-14T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例


```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Tus, 14 Jul 2020 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "clusterList": [
    {
      "id": "bgw_group-a3t786p1",
      "name": "abc",
      "type": "4Layer",
      "status": "available",
      "ccuCount": 1,
      "createTime": "2021-08-25T06:13:09Z",
      "expireTime": "2021-09-25T06:13:09Z",
      "desc": ""
    }
  ],
  "marker": "bgw_group-a3t786p1",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

[🔗 LBDC详情](#)

请求结构

```
GET /v{version}/lbdc/{id} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
id	String	是	URL参数	要查询的集群id

返回头域

除公共头域，无其它特殊头域。

4层返回参数

参数名称	类型	描述
id	String	集群id
name	String	集群名称
type	String	集群类型
status	String	集群状态
ccuCount	String	集群性能容量
createTime	String	集群创建时间
expireTime	String	集群失效时间
totalConnectCount	long	并发连接数
newConnectCps	long	新建连接速度
networkInBps	long	网络输入带宽
networkOutBps	long	网络输出带宽
tags	List<TagModel>	标签

7层返回参数

参数名称	类型	描述
id	String	集群id
name	String	集群名称
type	String	集群类型
status	String	集群状态
ccuCount	String	集群性能容量
createTime	String	集群创建时间
expireTime	String	集群失效时间
totalConnectCount	long	并发连接数
networkInBps	long	网络输入带宽
networkOutBps	long	网络输出带宽
httpsQps	long	https的qps
httpQps	long	http的qps
httpNewConnectCps	long	http新建速度
httpsNewConnectCps	long	https新建速度
sslNewConnectCps	long	ssl新建速度
tags	List<TagModel>	标签

请求示例

```
GET /v1/lbdc/bgw_group-a3t786p1 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2020-07-14T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863fffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Tus, 14 Jul 2020 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "id": "bgw_group-a3t786p1",
  "name": "abc",
  "type": "4Layer",
  "status": "available",
  "ccuCount": 1,
  "createTime": "2021-08-25T06:13:09Z",
  "expireTime": "2021-09-25T06:13:09Z",
  "desc": "",
  "totalConnectCount":100,
  "newConnectCps":100,
  "networkInBps":100,
  "networkOutBps":100,
  "tags": [
    {
      "tagKey": "hikwnf",
      "tagValue": "nwklwmflk"
    }
  ]
}
```

更新LBDC

请求结构

```
PUT /v{version}/lbdc/{id}?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name":name,
  "desc":desc
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
id	String	是	URL参数	集群id
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
name	String	否	RequestBody参数	集群名称。长度1~65个字节，字母开头，_可包含字母数字-/.字符。
desc	String	否	RequestBody参数	集群描述，最大支持200字符

返回头域

除公共头域，无其它特殊头域。

返回参数

无返回参数

请求示例

```
PUT /v1/lbdc/bgw_group-a3t786p1?clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2017-03-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "name": "abc",
  "desc": ""
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 LBDC关联的BLB列表

请求结构

```
GET /v{version}/lbdc/{id}/blb HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
id	String	否	URL参数	要查询的集群id

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
blbList	List< AssociateBlbModel >	包含查询结果的BLB列表

请求示例

```
GET /v1/lbdc/bgw_group-a3t786p1/blb HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2020-07-14T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Tus, 14 Jul 2020 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

```
{
  "blbList": [
    {
      "blbId": "lb-1c791997",
      "name": "abc",
      "status": "available",
      "blbType": "normal",
      "publicIp": "10.10.10.10",
      "eipRouteType": "bgp",
      "bandwidth": 20,
      "address": "192.168.0.7",
      "ipv6": "2400:da00:e003:1820::4",
      "vpcId": "vpc-0n1hhh8759b0",
      "subnetId": "sbn-p09p0ihp7498"
    }
  ]
}
```

服务发布点接口

服务发布点相关接口

创建服务发布点

描述

创建服务发布点

请求结构

```
POST /v{version}/service?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "name": name,
  "description": description,
  "serviceName": serviceName,
  "instanceId": instanceId,
  "authList": [
    {
      "uid": uid,
      "auth": auth
    }
  ]
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
name	String	是	Request Body参数	服务发布点的名称，大小写字母、数字以及-_/特殊字符，必须以字母开头，长度1-65
description	String	否	Request Body参数	服务发布点的描述，最大支持200字符
serviceName	String	是	Request Body参数	对应服务名称,大小写字母、数字长度1-65
instanceId	String	是	Request Body参数	绑定实例id，当前只支持绑定blb
authList	List<Auth>	否	Request Body参数	用户授权列表，默认拒绝所有

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
service	String	服务发布点的域名，使用此域名绑定服务网卡

请求示例

```
POST /v1/service?clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2015-08-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "name": "name",
  "description": "",
  "serviceName": "serviceName",
  "instanceId": "lb-d3e2cf75",
  "authList": [
    {
      "uid": "*",
      "auth": "allow"
    }
  ]
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "service": "xxx.baidubce.com"
}
```

更新服务发布点

描述

更新服务发布点，name、description不能同时为空

请求结构

```
PUT /v{version}/service/{service}?{action}&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "name":name,
  "description":description
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
service	String	是	URL参数	服务发布点的域名
action	String	是	Query参数	执行动作，当前取值为modifyAttribute
name	String	否	RequestBody参数	服务发布点名称,大小写字母、数字以及-_/特殊字符，必须以字母开头，长度1-65
description	String	否	RequestBody参数	服务发布点描述，不超过200字符

返回头域

除公共头域，无其它特殊头域。

返回参数

无返回参数

请求示例

```
PUT /v1/service/xxx.baidubce.com?modifyAttribute&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2017-03-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "name": "name",
  "description": "description"
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 服务发布点绑定实例

描述

服务发布点绑定服务

请求结构

```
PUT /v{version}/service/{service}?{action}&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "instanceId": instanceId
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
service	String	是	URL参数	服务发布点的域名
action	String	是	Query参数	执行动作，当前取值为bind
instanceId	String	是	Request Body参数	绑定实例id

返回头域

除公共头域，无其它特殊头域。

返回参数

无返回参数

请求示例


```
PUT /v1/service/xxx.baidubce.com?bind&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2017-03-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863fffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "instanceId": "lb-95d3c305"
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 服务发布点解绑实例

描述

服务发布点解绑服务

请求结构

```
PUT /v{version}/service/{service}?{action}&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
service	String	是	URL参数	服务发布点的域名
action	String	是	Query参数	执行动作，当前取值为unbind

返回头域

除公共头域，无其它特殊头域。

返回参数

无返回参数

请求示例

```
PUT /v1/service/xxx.baidubce.com?unbind&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2017-03-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863fffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 新增服务发布点鉴权信息

描述

新增服务发布点鉴权信息，如果用户授权已存在则覆盖之前的授权信息

请求结构

```
PUT /v{version}/service/{service}?{action}&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "authList": [
    {
      "uid": uid,
      "auth" : auth
    }
  ]
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
service	String	是	URL参数	服务发布点的域名
action	String	是	Query参数	执行动作，当前取值为addAuth
authList	List<Auth>	是	Request Body参数	用户授权列表

返回头域

除公共头域，无其它特殊头域。

返回参数

无返回参数

请求示例

```
PUT /v1/service/xxx.baidubce.com?addAuth&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2017-03-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "authList": [
    {
      "uid": "e8deba4d81d3435bb4591ce97288f296",
      "auth" : "deny"
    }
  ]
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

修改服务发布点鉴权信息

描述

修改服务发布点鉴权信息

请求结构

```
PUT /v{version}/service/{service}?{action}&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "authList": [
    {
      "uid": uid,
      "auth" : auth
    }
  ]
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
service	String	是	URL参数	服务发布点的域名
action	String	是	Query参数	执行动作，当前取值为editAuth
authList	List<Auth>	是	Request Body参数	用户授权列表

返回头域

除公共头域，无其它特殊头域。

返回参数

无返回参数

请求示例

```
PUT /v1/service/xxx.baidubce.com?editAuth&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2017-03-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "authList": [
    {
      "uid": "e8deba4d81d3435bb4591ce97288f296",
      "auth" : "deny"
    }
  ]
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 删除服务发布点鉴权信息

描述

新增服务发布点鉴权信息

请求结构

```
PUT /v{version}/service/{service}?{action}&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: authorization string

{
  "uidList": [
    "uid"
  ]
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
service	String	是	URL参数	服务发布点的域名
action	String	是	Query参数	执行动作，当前取值为removeAuth
uidList	List<String>	是	Request Body参数	用户id列表，所有人使用"*"

返回头域

除公共头域，无其它特殊头域。

返回参数

无返回参数

请求示例

```
PUT /v1/service/xxx.baidubce.com?removeAuth&clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2017-03-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de

{
  "uidList": [
    "e8deba4d81d3435bb4591ce97288f296",
    "f8deba4d81d8908bb4591ce97288f212",
  ]
}
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

🔗 查询服务发布点列表

描述

查询服务发布点列表。

请求结构

```
GET /v{version}/service?maxKeys=1 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
marker	String	否	Query参数	批量获取列表的查询的起始位置，是一个由系统生成的字符串
maxKeys	Integer	否	Query参数	每页包含的最大数量，最大数量通常不超过1000，缺省值为1000

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
marker	String	标记查询的起始位置
isTruncated	boolean	true表示后面还有数据，false表示已经是最后一页
nextMarker	String	获取下一页所需要传递的marker值。当isTruncated为false时，该域不出现
maxKeys	int	每页包含的最大数量
services	List< Service >	服务发布点列表

请求示例

```
GET /v{version}/service?marker=uservice-a223vc&maxKeys=1 HTTP/1.1
Host: blb.bj.baidubce.com
ContentType: application/json
Authorization: bce auth v1/f81d3b34e48048fbb2634dc7882d7e21/2017 03
11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863ffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7 4ad5 451d 9215 71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content Type: application/json;charset=UTF 8
Server: BWS
```

```
{
  "nextMarker": "uservice-a223vc",
  "marker": "uservice-a032va",
  "maxKeys": 1,
  "isTruncated": true,
  "services": [
    {
      "serviceId": "uservice-a223vc",
      "name": "name",
      "description": "",
      "serviceName": "serviceName",
      "bindType": "blb",
      "instanceId": "lb-d3e2cf75",
      "status": "inService",
      "service": "xxx.baidubce.com",
      "createTime": "2019-03-07T02:35:31Z",
      "endpointCount": 1,
      "endpointList": [
        {
          "endpointId": "endpoint-a223vc",
          "uid": "e8deba4d81d3435bb4591ce97288f296",
          "attachTime": "2019-03-07T02:35:31Z",
        }
      ],
      "authList": [
        {
          "uid": "*",
          "auth": "allow"
        }
      ]
    }
  ]
}
```

 查询服务发布点详情

描述

查询服务发布点详情

请求结构

```
GET /v{version}/service/{service} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
service	String	是	URL参数	服务发布点的域名

返回头域

除公共头域，无其它特殊头域。

返回参数

参数名称	类型	描述
serviceId	String	服务发布点的id
name	String	服务发布点的名称
description	String	描述
serviceName	String	服务名称
bindType	String	绑定服务类型，目前仅支持绑定BLB实例
instanceId	String	绑定实例ID
status	String	发布点状态，取值范围inService/available/unavailable/dead/free，分别表示：服务中/可用/不可用/故障/未绑定
service	String	服务发布点唯一对应域名
createTime	String	创建时间
endpointCount	int	关联的服务网卡数量
endpointList	List<RelatedEndpoint>	关联的服务网卡列表
authList	List<Auth>	授权列表

请求示例

```
GET /v1/service/xxx.baidubce.com HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

返回示例


```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7 4ad5 451d 9215 71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content Type: application/json;charset=UTF 8
Server: BWS

{
  "serviceId": "userice-a223vc",
  "name": "name",
  "description": "",
  "serviceName": "serviceName",
  "bindType": "blb",
  "instanceId": "lb-d3e2cf75",
  "status": "inService",
  "service": "xxx.baidubce.com",
  "createTime": "2019-03-07T02:35:31Z",
  "endpointCount": 1,
  "endpointList": [
    {
      "endpointId": "endpoint-a223vc",
      "uid": "e8deba4d81d3435bb4591ce97288f296",
      "attachTime": "2019-03-07T02:35:31Z",
    }
  ],
  "authList": [
    {
      "uid": "*",
      "auth": "allow"
    }
  ]
}
```

🔗 删除服务发布点

描述

删除服务发布点

请求结构

```
DELETE /v{version}/service/{service}?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必需	参数位置	描述
version	String	是	URL参数	API版本号
clientToken	String	否	Query参数	幂等性Token，是一个长度不超过64位的ASCII字符串
service	String	是	URL参数	服务发布点的域名

返回头域

除公共头域，无其它特殊头域。

返回参数 无返回参数

请求示例

```
DELETE /v1/service/xxx.baidubce.com?clientToken=1849ef6b-f73c-4f5d-b88b-259ca47af849 HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: bce-auth-v1/f81d3b34e48048fbb2634dc7882d7e21/2015-08-11T04:17:29Z/3600/host/74c506f68c65e26c633bfa104c863fffac5190fdec1ec24b7c03eb5d67d2e1de
```

返回示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Thu, 16 Mar 2017 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

BLB订单相关接口

BLB退订接口

🔗 预付费BLB退订接口

请求结构

```
PUT /v{version}/blb/refund/{blbId}?clientToken={clientToken} HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbId	String	否	URL参数	待更新的LoadBalancer的ID
clientToken	String	否	Query参数	幂等性Token，长度不超过64位

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
PUT /v1/blb/refund?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2024 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
```

BLB询价

 BLB支持询价接口

请求结构

```
POST /v{version}/blb/price HTTP/1.1
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "blbType": "ipv6Application",
  "performanceLevel": "small1",
  "count": 1,
  "billing": {
    "paymentTiming": "Prepaid",
    "reservation": {
      "reservationLength": 1
    }
  }
}
```

请求头域

除公共头域外，无其他特殊头域

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbType	String	否	RequestBody参数	blb类型，不传默认是普通型blb。取值可以为"normal"代表普通型，"application"代表应用型，"ipv6"代表普通型IPv6，"ipv6Application"代表应用型IPv6
performanceLevel	String	否	RequestBody参数	性能规格参数，不传默认为共享型。取值如下："small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型1，"large1"超大型1，"large2"超大型2，"large3"超大型3
count	Integer	否	RequestBody参数	购买数量，不传默认是1
billing	Billing	否	RequestBody参数	订单相关信息

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
prices	List< Price >	返回的价格信息

预付费请求示例

```
PUT /v1/blb/price?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization: authorization string
{
  "blbType": "ipv6Application",
  "performanceLevel": "small1",
  "count": 1,
  "billing": {
    "paymentTiming": "Prepaid",
    "reservation": {
      "reservationLength": 1
    }
  }
}
```

预付费响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2024 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
{
  "prices": [
    {
      "chargeItem": "instance",
      "originalPrice": "190",
      "discountPrice": "20.90"
    }
  ]
}
```

后付费请求示例

```
PUT /v1/blb/price?clientToken=be31b98c-5e41-4838-9830-9be700de5a20 HTTP/1.1
HOST blb.bj.baidubce.com
Authorization: authorization string
{
  "blbType": "ipv6Application",
  "performanceLevel": "small1",
  "count": 1,
  "billing": {
    "paymentTiming": "Postpaid",
    "billingMethod": "ByCapacityUnit"
  }
}
```

后付费响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 946002ee-cb4f-4aad-b686-5be55df27f09
Date: Wed, 10 Apr 2024 08:26:52 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Server: BWS
{
  "prices": [
    {
      "chargeItem": "instance",
      "originalPrice": "0.01056",
      "discountPrice": "0.00116",
      "chargeUnit": "minute"
    },
    {
      "chargeItem": "lcu",
      "originalPrice": "0.04",
      "discountPrice": "0.04",
      "chargeUnit": "lcu"
    }
  ]
}
```

BLB计费变更-预付费转后付费

 预付费转后付费接口

描述

BLB实例付费方式与计费方式变更。可进行以下场景变更。

- 预付费变更为后付费。可同时指定后付费的计费方式和性能规格。若后付费为按使用量计费("ByCapacityUnit")此时性能规格还支持不限速("unlimited")。
- 后付费按规格变更为后付费按使用量。
- 后付费按使用量变更为后付费按规格。

请求结构

```
POST /v{version}/blb/{blbld}/charge?action=TO_POSTPAY&clientToken={clientToken}
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "billingMethod": "BySpec",
  "performanceLevel": "small2"
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	当前取值"TO_POSTPAY"
billingMethod	String	否	Request Body参数	计费方式。"BySpec"表示按固定规格计费(默认值)，"ByCapacityUnit"表示按使用量计费。
performanceLevel	String	否	Request Body参数	性能规格参数，默认为当前实例的性能规格。取值如下："small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型2，"large1"超大型1，"large2"超大型2，"large3"超大型3。仅后付费-按使用量支持"unlimited"不限速。
effectively	Boolean	否	RequestBody参数	是否立即生效，默认false。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域 除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
orderId	String	预付费转后付费订单ID

请求示例

```
POST /v1/blb/lb-6x7atqxl/charge?action=TO_POSTPAY&clientToken=0fc2497e-6216-4702-a841-e5f2f6e366ce
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "billingMethod": "BySpec",
  "performanceLevel": "small2"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Fri, 16 Aug 2024 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
{
  "orderId": "38f903d90ae84fd28a71d70e47fb6406"
}
```

BLB-变配

变配接口

描述

变配接口可对LB实例的性能规格进行调整。

- 预付费实例支持升级配置。
- 后付费实例可以任意变配。仅后付费按使用量类型的性能规格支持变配为不限速("unlimited")。

请求结构

```
POST /v{version}/blb/{blbId}?action=RESIZE&clientToken={clientToken}
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "performanceLevel":"medium1"
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	当前取值"RESIZE"
performanceLevel	String	否	Request Body参数	性能规格。取值如下："small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型2，"large1"超大型1，"large2"超大型2，"large3"超大型3。仅后付费-按使用量支持变配为"unlimited"不限速。

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
orderId	String	变配订单ID

请求示例

```
POST /v1/blb/lb-6x7atqxl/charge?action=RESIZE&clientToken=0fc2497e-6216-4702-a841-e5f2f6e366ce
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "performanceLevel":"medium1"
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Fri, 16 Aug 2024 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
{
  "orderId": "05182695159a4924a473d1c21be921c7"
}
```

BLB计费变更-取消预付费转后付费

取消后付费转预付费接口

描述

当触发预付费转后付费操作后，需等待预付费实例到期后才会转换为后付费。在此期间无法对实例进行变配等其他订单相关操作。可通过该接口取消预付费转后付费。

请求结构

```
POST /v{version}/blb/{blbld}/charge?action=CANCEL_TO_POSTPAY&clientToken={clientToken}
Host: blb.bj.baidubce.com
Authorization: authorization string
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	当前取值"CANCEL_TO_POSTPAY"

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

无

请求示例

```
POST /v1/blb/lb-6x7atqxl/charge?action=CANCEL_TO_POSTPAY&clientToken=0fc2497e-6216-4702-a841-e5f2f6e366ce
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdf16
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Fri, 16 Aug 2024 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS
```

BLB计费变更-后付费转预付费

🔗 后付费转预付费接口

描述

当前接口可将BLB实例从后付费变更为预付费。付费方式转换时可同时指定“performanceLevel”参数进行实例配置变更，同时必须指定预付费时长。

请求结构

```
POST /v{version}/blb/{blbld}/charge?action=TO_PREPAY&clientToken={clientToken}
Host: blb.bj.baidubce.com
Authorization: authorization string
{
  "billingMethod": "BySpec",
  "performanceLevel": "medium1",
  "reservationLength": 1
}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

参数名称	类型	是否必选	参数位置	描述
version	String	是	URL参数	API版本号，当前取值1
blbld	String	是	URL参数	LoadBalancer的标识符
clientToken	String	是	Query参数	幂等性Token，长度不超过64位
action	String	是	Query参数	当前取值"TO_REPAY"
billingMethod	String	否	Request Body参数	计费类型。当前只支持默认值"BySpec"。
performanceLevel	String	否	Request Body参数	性能规格。不填表示不进行配置变更。取值如下："small1"标准型1，"small2"标准型2，"medium1"增强型1，"medium2"增强型2，"large1"超大型1，"large2"超大型2，"large3"超大型3。注意：预付费不支持"unlimited"不限速
reservationLength	Int	是	Request Body参数	购买月份时长，[1,2,3,4,5,6,7,8,9,12,24,36]

返回状态码

成功返回200，失败返回见[错误码](#)

返回头域

除公共头域外，无其他特殊头域

返回参数

参数名称	类型	描述
orderId	String	后付费转预付费订单ID

请求示例

```
POST /v1/blb/lb-6x7atqxl/charge?action=TO_REPAY&clientToken=0fc2497e-6216-4702-a841-e5f2f6e366ce
Host: blb.bj.baidubce.com
Authorization:bce-auth-v1/a43eea003308400587496785f7f5d9ee/2022-04-28T11:06:00Z/3600/host;x-bce-date/25ec3d4baba54ad7397cb31c96cb5e364088ff79f9b853ffb8dc6c11ebcdcf16

{
  "billingMethod": "BySpec",
  "performanceLevel":"medium1",
  "reservationLength": 2
}
```

响应示例

```
HTTP/1.1 200 OK
x-bce-request-id: 1214cca7-4ad5-451d-9215-71cb844c0a50
Date: Fri, 16 Aug 2024 06:29:48 GMT
Content-Type: application/json;charset=UTF-8
Server: BWS

{
  "orderId": "d79fa3041c6b49edb5b943618c9b5656"
}
```

附录

Model对象定义

BLBModel

参数名称	类型	描述
blbId	String	LoadBalancer的标识符
name	String	LoadBalancer的名称
desc	String	LoadBalancer的描述
address	String	分配的内网服务地址IP，通过这个IP即能通过内网访问该实例
status	blbStatus	BLB状态
vpId	String	vpc 的ID
subnetId	String	subnet 的ID
publicIp	String	如果LoadBalancer绑定过EIP，则显示该项，否则不显示
tags	List<TagModel>	标签键值对列表
allowDelete	Boolean	是否允许删除
eipRouteType	String	EIP线路类型
publicIpv6	String	如果LoadBalancer绑定过EIPv6，则显示该项，否则不显示
eipV6RouteType	String	EIPV6线路类型
paymentTiming	String	计费方式 取值："Postpaid" 后付费 "Prepaid" 预付费
billingMethod	String	计费方式 取值："ByCapacityUnit" 按使用量 "BySpec" 按固定规格

TCPLListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
backendPort	int	后端服务器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"WeightLeastConn"代表加权最小连接数、"Hash"代表根据源ip哈希
healthCheckTime outInSeconds	int	健康检查超时
healthCheckInterv al	int	健康检查间隔
unhealthyThresho ld	int	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器
healthyThreshold	int	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用
tcpSessionTimeo ut	int	tcp设置连接超时时间（单位：秒），默认为900，需为10-4000间的整数

UDPListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
backendPort	int	后端服务器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"WeightLeastConn"代表加权最小连接数、"Hash"代表根据源ip哈希
healthCheckType	String	健康检查协议，值为"UDP"/"ICMP"
healthCheckTime outInSeconds	int	健康检查超时
healthCheckInterv al	int	健康检查间隔
unhealthyThresho ld	int	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器
healthyThreshold	int	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用
healthCheckStrin g	String	健康发送的请求字符串，后端服务器收到后需要进行应答，支持标准转义如 \00、\xf2，方便配置二进制格式请求
udpSessionTimeo ut	int	udp会话超时时间。默认为90，需为5-4000间的整数，单位秒

HTTPListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
backendPort	int	后端服务器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"LeastConnection"
keepSession	bool	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器
keepSessionType	String	会话保持的cookie处理方式，当且仅当开启会话保持时有效，值为"insert"/"rewrite"
keepSessionDuration	int	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效
keepSessionCookieName	String	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardFor	bool	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址
healthCheckType	String	健康检查协议，值为"HTTP"/"TCP"
healthCheckPort	int	健康检查端口
healthCheckURI	String	健康检查URI
healthCheckTimeoutInSecond	int	健康检查超时
healthCheckInterval	int	健康检查间隔
unhealthyThreshold	int	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器
healthyThreshold	int	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用
healthCheckNormalStatus	String	健康检查正常时的HTTP状态码，支持5类状态码的组合，格式为http_1xx或http_2xx
healthCheckHost	String	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
serverTimeout	int	后端服务器最大超时（单位：秒）
redirectPort	int	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定

 [HTTPSListenerModel](#)

参数名称	类型	描述
listenerPort	int	监听器的监听端口
backendPort	int	后端服务器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"LeastConnection"
keepSession	bool	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器
keepSessionType	String	会话保持的cookie处理方式，当且仅当开启会话保持时有效，值为"insert"/"rewrite"
keepSessionDuration	int	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效
keepSessionCookieName	String	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardFor	bool	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址
healthCheckType	String	健康检查协议，值为"HTTP"/"TCP"
healthCheckPort	int	健康检查端口
healthCheckURI	String	健康检查URI
healthCheckTimeoutInSecond	int	健康检查超时
healthCheckInterval	int	健康检查间隔
unhealthyThreshold	int	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器
healthyThreshold	int	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用
healthCheckNormalStatus	String	健康检查正常时的HTTP状态码，支持5类状态码的组合，格式为http_1xx 或 http_2xx
healthCheckHost	String	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
serverTimeout	int	后端服务器最大超时（单位：秒）
certIds	List<String>	加载的SSL证书，目前HTTPS监听器只能绑定一个SSL证书
additionalCertDomains	List<AdditionalCertDomain>	扩展域名
dualAuth	bool	是否开启双向认证，默认为关闭
clientCertIds	List<String>	当dualAuth为true时，加载的客户端证书链
encryptionType	String	加密类型
encryptionProtocols	List<String>	加密协议
appliedCiphers	String	加密套件

SSLListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
backendPort	int	后端服务器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
healthCheckTime outInSeconds	int	健康检查超时
healthCheckInterval	int	健康检查间隔
unhealthyThreshold	int	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器
healthyThreshold	int	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用
certIds	List<String>	加载的SSL证书链
encryptionType	String	加密类型
encryptionProtocols	List<String>	加密协议
appliedCiphers	String	加密套件
dualAuth	boolean	是否开启双向认证，默认为关闭
clientCertIds	List<String>	当dualAuth为true时，加载的客户端证书链
serverTimeout	int	后端服务器最大超时（单位：秒），默认900s，需为10-4000间的整数

BackendServerModel

参数名称	类型	描述
instanceId	String	后端服务器标识符
weight	int	后端服务器权重，取值范围[0, 100]，权重为0表示不要把流量转发到该后端服务器上

BackendServerStatus

参数名称	类型	描述
instanceId	String	后端服务器标识符
weight	int	后端服务器权重
status	String	后端服务器健康状态，值为"Alive"/"Dead"/"Unknown"

AppBLBModel

参数名称	类型	描述
blbId	String	LoadBalancer的标识符
name	String	LoadBalancer的名称
desc	String	LoadBalancer的描述
address	String	LoadBalancer的地址，点分十进制表示
status	blbStatus	BLB状态
subnetId	String	subnet 的ID
vpcId	String	vpc短Id
publicIp	String	如果LoadBalancer绑定过EIP，则显示该项，否则不显示
tags	List	标签键值对列表
allowDelete	Boolean	是否允许删除
eipRouteType	String	EIP线路类型
publicIpv6	String	如果LoadBalancer绑定过EIPV6，则显示该项，否则不显示
eipV6RouteType	String	EIPV6线路类型
paymentTiming	String	计费方式 取值："Postpaid" 后付费 "Prepaid" 预付费
billingMethod	String	计费方式 取值："ByCapacityUnit" 按使用量 "BySpec" 按固定规格

TagModel

参数名称	类型	描述
tagKey	String	标签的键，可包含大小写字母、数字、中文以及_/.特殊字符，长度1-65
tagValue	String	标签的值，可包含大小写字母、数字、中文以及_/.特殊字符，长度0-65

ListenerModel

参数名称	类型	描述
port	String	监听器端口
type	String	监听器协议类型
description	String	描述信息，长度不超过200个字符。

AppTCPListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"WeightLeastConn"代表加权最小连接数、"Hash"代表根据源ip哈希
tcpSessionTimeout	int	tcp设置连接超时时间（单位：秒），默认为900，需为10-4000间的整数
description	String	描述信息，长度不超过200个字符。

AppUDPListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"WeightLeastConn"代表加权最小连接数、"Hash"代表根据源ip哈希
description	String	描述信息，长度不超过200个字符。

AppHTTPListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器
keepSessionType	String	会话保持的cookie处理方式，当且仅当开启会话保持时有效，值为"insert"/"rewrite"
keepSessionTimeout	int	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效
keepSessionCookieName	String	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址
xForwardedProto	boolean	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议
serverTimeout	int	后端服务器最大超时（单位：秒）
redirectPort	int	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定
description	String	描述信息，长度不超过200个字符。

AppHTTPSListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"LeastConnection"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数
keepSession	boolean	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器
keepSessionType	String	会话保持的cookie处理方式，当且仅当开启会话保持时有效，值为"insert"/"rewrite"
keepSessionTimeout	int	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效
keepSessionCookieName	String	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardedFor	boolean	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址
xForwardedProto	boolean	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议
serverTimeout	int	后端服务器最大超时（单位：秒）
certIds	List<String>	加载的SSL证书链
additionalCertDomains	List<AdditionalCertificateDomain>	扩展域名
encryptionType	String	加密类型
encryptionProtocols	List<String>	加密协议
appliedCiphers	String	加密套件
dualAuth	boolean	是否开启双向认证，默认为关闭
clientCertIds	List<String>	当dualAuth为true时，加载的客户端证书链
description	String	描述信息，长度不超过200个字符。

AppSSLListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"LeastConnection"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"Hash"代表根据源ip哈希
certIds	List<String>	加载的SSL证书链
encryptionType	String	加密类型
encryptionProtocols	List<String>	加密协议
appliedCiphers	String	加密套件
dualAuth	boolean	是否开启双向认证，默认为关闭
clientCertIds	List<String>	当dualAuth为true时，加载的客户端证书链
description	String	描述信息，长度不超过200个字符。

AppBackendServer

参数名称	类型	描述
instanceId	String	后端服务器标识符
weight	int	后端服务器权重，取值范围0-100
privateIp	String	查询时返回值，后端绑定的该服务器ip地址
portList	List<AppRsPortModel>	查询时返回值，设置了相应策略，RS开放的端口列表

AppRsPortModel

参数名称	类型	描述
listenerPort	int	监听器端口(后端端口关联的监听器端口)
backendPort	String	后端开放的端口
portType	String	端口协议类型(服务器组开放的端口类型，包含TCP、UDP、HTTP、HTTPS)
healthCheckPortType	String	健康检查端口协议类型(TCP、UDP、ICMP、HTTP、HTTPS)
status	String	端口状态，"Alive"/"Dead"/"Unknown"
portId	String	端口id
policyId	String	对应策略id

AppServerGroup

参数名称	类型	描述
id	String	后端服务器组标识符
name	String	后端服务器组名称
desc	String	后端服务器组描述
status	String	服务器组状态，详见 blbStatus
portList	List< AppServerGroupPort >	服务器组开放的端口列表

 AppServerGroupPort

参数名称	类型	描述
id	String	服务器组端口标识符
port	int	服务器组开放端口号
type	String	端口协议类型，"TCP"/"HTTP"
status	String	服务器组状态，详见 blbStatus
healthCheck	String	健康检查协议，支持"HTTP"/"TCP"
healthCheckPort	int	健康检查端口
healthCheckTimeoutInSecond	int	健康检查超时（单位：秒），默认为3，需为1-60间的整数
healthCheckIntervalInSecond	int	健康检查间隔（单位：秒），默认为3，需为1-10间的整数
healthCheckDownRetry	int	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器。默认为3，需为2-5间的整数
healthCheckUpRetry	int	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用。默认为3，需为2-5间的整数
healthCheckNormalStatus	String	健康检查正常时的HTTP状态码，支持5类状态码的组合，例如"http_1xx http_2xx"，默认"http_2xx http_3xx"。当健康检查协议为"HTTP"时生效
healthCheckUrlPath	String	健康检查URI，默认/。当健康检查协议为"HTTP"时生效
healthCheckHost	String	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
udpHealthCheckString	String	udp健康检查字符串，当端口类型为udp是有效

 AppPolicy

参数名称	类型	描述
id	String	策略标识符
desc	String	策略描述默认为空
appServerGroupIpd	String	策略绑定服务器组标识符
appServerGroupName	String	策略绑定服务器组名称
frontendPort	int	前端服务器的监听端口
type	String	前端服务器的监听端口协议
backendPort	int	目标端口号，当listenerPort对应监听器为TCP或SSL时需要传入对应服务器组（appServerGroupId）下开放的TCP端口号；当listenerPort对应监听器为HTTP或HTTPS时需要传入对应服务器组（appServerGroupId）下开放的HTTP端口号；当listenerPort对应监听器为UDP时需要传入对应服务器组（appServerGroupId）下开放的UDP端口号
portType	String	端口类型
priority	int	策略优先级，有效取值范围是1-32768
ruleList	List<AppRule>	策略规则列表
groupType	String	"Server" 表示后端绑定的是服务器组

AppPolicyForUpdate

参数名称	类型	是否必须	描述
policyId	String	是	转发策略id。
priority	Integer	否	优先级，取值范围1-32768，且不能与已有条目重复。数值越小，优先级越高，规则匹配顺序为按优先级由高到低匹配。priority和description不能同时为空。
description	String	否	描述信息。最大200字符。priority和description不能同时为空。

AppRule

参数名称	类型	描述
id	String	规则的标识符
key	String	规则的类型，host/uri/*
value	String	通配符匹配字符串，详见ValueExample

CreateAppPolicy

参数名称	类型	是否必需	描述
appServerGroupIpd	String	是	策略绑定服务器组标识符
backendPort	int	是	目标端口号，当listenerPort对应监听器为TCP或SSL时需要传入对应服务器组（appServerGroupIpd）下开放的TCP端口号；当listenerPort对应监听器为HTTP或HTTPS时需要传入对应服务器组（appServerGroupIpd）下开放的HTTP端口号；当listenerPort对应监听器为UDP时需要传入对应服务器组（appServerGroupIpd）下开放的UDP端口号
portType	String	否	目标端口类型，当目标组是服务器组时默认为目标端口号所使用的协议
priority	int	是	策略优先级，有效取值范围是1-32768
ruleList	List<CreateAppRule>	是	策略规则列表
desc	String	否	策略描述默认为空

CreateAppRule

参数名称	类型	是否必需	描述
key	String	是	规则的类型，host/uri/*
value	String	是	通配符匹配字符串，详见ValueExample

PortTypeModel

参数名称	类型	是否必需	描述
port	int	是	监听端口
type	String	是	监听端口的协议

BillingForCreate

参数名称	类型	是否必需	描述
paymentTiming	String	是	付款时间，预支付（Prepaid）和后支付（Postpaid），目前只支持预支付
billingMethod	String	否	计费方式，后付费的时候传，"BySpec"按固定规格、"ByCapacityUnit"按使用量
reservation	ReservationForCreate	是	保留信息，支付方式为后支付时不需要设置，预支付时必须设置

ReservationForCreate

参数名称	类型	是否必需	描述
reservationLength	int	是	购买月份时长，[1,2,3,4,5,6,7,8,9,12,24,36]

BillingForRenew

参数名称	类型	是否必需	描述
reservation	ReservationForCreate	是	保留信息，预支付时必须设置

Billing

参数名称	类型	是否必选	描述
paymentTiming	String	是	付款时间，预支付（Prepaid）和后支付（Postpaid）
billingMethod	String	否	计费方式，预付费的时候不要传，后付费的时候传，按固定规格计费（BySpec）和按使用量计费（ByCapacityUnit）
reservation	Reservation	否	保留信息，支付方式为后支付时不需要设置，预支付时必须设置

Reservation

参数名称	类型	是否必选	描述
reservationLength	Integer	是	购买月份时长，取值范围为：[1,2,3,4,5,6,7,8,9,12,24,36]

Price

参数名称	类型	描述
chargeItem	String	计费项，比如取值"instance"代表实例费用；比如"nettraffic"代表流量费用
discountPrice	String	折扣价，单位都是元。预付费的时候，返回值比如："150"，后付费的时候，返回值比如："0.00028"
originalPrice	String	原价
chargeUnit	String	计费单位。后付费的时候取值，比如"minute"。minute：表示计价单元是按每分钟来计算。GB：表示计价单元是按每GB来计算。"Icu"代表按每Icu单位计费

AdditionalCertDomain

参数名称	类型	是否必需	描述
certId	String	是	证书ID
Host	String	是	证书域名。若证书的产品类型为通配符域名版还支持通配符域名及其子域名

ClusterModel

参数名称	类型	描述
id	String	集群id
name	String	集群名称
type	String	集群类型
status	String	集群状态
ccuCount	int	集群性能容量
createTime	String	集群创建时间
expireTime	String	集群失效时间
desc	String	描述

AssociateBlbModel

参数名称	类型	描述
blbId	String	负载均衡id
name	String	blb名称
status	String	blb状态
blbType	String	blb类型
publicIp	String	公网ip
eipRouteType	String	eip线路类型
bandwidth	int	带宽
address	String	内网ip地址
ipv6	String	ipv6地址
vpId	String	vpId
subnetId	String	子网id

BlbSecurityGroupModel

参数名称	类型	描述
securityGroupId	String	普通安全组ID
securityGroupName	String	普通安全组名称
securityGroupDesc	String	普通安全组描述
vpcName	String	私有网络VPC名称
securityGroupRules	List<BlbSecurityGroupRuleModel>	普通安全组规则

BlbSecurityGroupRuleModel

参数名称	类型	描述
securityGroupRuleId	String	普通安全组规则ID
direction	String	入站/出站，取值ingress或egress
ethertype	String	网络类型，取值IPv4或IPv6，值为空时表示默认取值IPv4
portRange	String	端口范围，可以指定80等单个端口，值为空时默认取值1-65535
protocol	String	协议类型，tcp、udp或icmp，值为空时默认取值all
sourceGroupId	String	源安全组ID
sourceIp	String	源IP地址，与sourceGroupId不能同时设定值
destGroupId	String	目的安全组ID
destIp	String	目的IP地址，与destGroupId不能同时设定值

BlbEnterpriseSecurityGroupModel

参数名称	类型	描述
enterpriseSecurityGroupId	String	企业安全组ID
enterpriseSecurityGroupName	String	企业安全组名称
enterpriseSecurityGroupDesc	String	企业安全组描述
enterpriseSecurityGroupRules	List< BlbEnterpriseSecurityGroupRuleModel >	企业安全组规则

BlbEnterpriseSecurityGroupRuleModel

参数名称	类型	描述
remark	String	备注，长度1-255。
direction	String	入站/出站，取值ingress或egress。
ethertype	String	网络类型，取值IPv4或IPv6。值为空时表示默认取值IPv4。
portRange	String	端口范围，支持连续端口（比如“1-80”）和离散端口（比如“80,7000,8000”，英文逗号隔开，离散值数量不超过15个），值为空时默认取值1-65535。
protocol	String	协议类型，tcp、udp或icmp，值为空时默认取值all。
sourceIp	String	源IP地址，all表示全部。
destIp	String	目的IP地址，all表示全部。
action	String	允许/拒绝，取值allow或deny。
priority	Integer	优先级，取值范围1-1000。
enterpriseSecurityGroupRuleId	String	企业安全组规则ID

规则配置示例

ValueExample

key	value	描述
*	*	全匹配
host	*.baidu.com	匹配以“.baidu.com”结尾的域名
uri	/abc/def.*	匹配以“/abc/def.”开头的URI路径

状态编码定义

🔗 blbStatus

状态	描述
creating	创建中
available	运行中
updating	更新中
paused	已欠费
unavailable	暂不可用

🔗 加密类型、加密协议、加密套件

🔗 encryption

encrypti onType(加 密类 型)	encrypti onProto cols(加 密协议)	appliedCiphers(加密套件)
tls_ciph er_polic y_defau lt	["tlsv10", "tlsv11", "tlsv12"]	ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA
tls_ciph er_polic y_1_1	["tlsv11", "tlsv12"]	同tls_cipher_policy_default
tls_ciph er_polic y_1_2	["tlsv12"]	同tls_cipher_policy_default
tls_ciph er_polic y_1_2_ secure	["tlsv12"]	ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA

🔗 所有监听类型字段，具体返回值根据监听类型来返回

🔗 AllListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
listenerType	String	监听器的监听类型
backendPort	int	后端服务器的监听端口
scheduler	String	负载均衡算法，值为"RoundRobin"/"LeastConnection"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"WeightLeastConn"代表加权最小连接数,"Hash"代表根据源ip哈希
healthCheckTimeoutInSeco	int	健康检查超时时间

nd		
healthCheckInterval	int	健康检查间隔时间
healthyThreshold	int	健康阈值，即连续多少次健康检查成功后，重新将该后端服务器置为可用
unhealthyThreshold	int	不健康阈值，即连续多少次健康检查失败后，屏蔽该后端服务器
getBblp	bool	open-api作为隐形参数对特定有需求用户开放，目前只针对TCP类型可设置，其它类型默认为false，打开后4层数据流到达VM后，VM可通过tcp_option获取到原始blb的ip（vip/ovip）信息
tcpSessionTimeout	int	tcp设置连接超时时间（单位：秒）
udpSessionTimeout	int	udp会话超时时间（单位：秒）
healthCheckString	String	UDP健康检查发送的请求字符串
keepSession	bool	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器
keepSessionType	String	会话保持的cookie处理方式，当且仅当开启会话保持时有效
keepSessionDuration	int	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效
keepSessionCookieName	String	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardFor	bool	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址
healthCheckType	String	健康检查协议类型
healthCheckPort	int	健康检查端口
healthCheckURI	String	健康检查URI
healthCheckNormalStatus	String	健康检查正常时的HTTP状态码，支持5类状态码的组合，格式为http_1xx或http_2xx
healthCheckHost	String	7层健康检查请求的头部域会带指定的host字段，例如"localhost"，默认""。当健康检查协议为"HTTP"时生效
serverTimeout	int	后端服务器最大超时时间（单位：秒）
redirectPort	int	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定
certIds	List	加载的SSI证书，目前HTTPS监听器只能绑定一个SSL证书
dualAuth	bool	是否开启双向认证，默认为关闭
clientCertIds	List	当dualAuth为true时，加载的客户端证书链
encryptionType	String	加密类型
encryptionProtocols	List	加密协议

appliedCiphers	String	加密套件

AppListenerModel

参数名称	类型	描述
listenerPort	int	监听器的监听端口
listenerType	String	监听器的监听类型
scheduler	String	负载均衡算法，值为"RoundRobin"/"LeastConnection"/"WeightLeastConn"/"Hash"，"RoundRobin"代表加权轮询、"LeastConnection"代表最小连接数、"WeightLeastConn"代表加权最小连接数,"Hash"代表根据源ip哈希
tcpSessionTimeout	int	tcp设置连接超时时间（单位：秒）
udpSessionTimeout	int	udp设置连接超时时间（单位：秒）
keepSession	bool	是否开启会话保持功能，即同一个Client发出的请求都会到达同一个后端服务器
keepSessionType	String	会话保持的cookie处理方式，当且仅当开启会话保持时有效
keepSessionTimeout	int	会话保持的cookie有效时间（单位：秒），当且仅当开启会话保持时有效
keepSessionCookieName	String	会话保持需要覆盖的cookie名称，当且仅当开启会话保持且keepSessionType="rewrite"时有效
xForwardFor	bool	是否开启获取Client真实IP，开启后后端服务器上可以通过X-Forwarded-For这个HTTP Header来获得Client端的真实地址
xForwardedProto	bool	是否开启获取监听协议，开启后后端服务器上可以通过X-Forwarded-Proto这个HTTP Header来获得监听协议
serverTimeout	int	后端服务器最大超时时间（单位：秒）
redirectPort	int	将此监听器收到的请求转发到HTTPS监听器，HTTPS监听器通过这个端口指定
certIds	List	加载的SSL证书，目前HTTPS监听器只能绑定一个SSL证书
dualAuth	bool	是否开启双向认证，默认为关闭
clientCertIds	List	当dualAuth为true时，加载的客户端证书链
encryptionType	String	加密类型
encryptionProtocols	List	加密协议
appliedCiphers	String	加密套件

Auth

参数名称	类型	描述
uid	String	用户id，所有人使用""
auth	String	鉴权方式，取值：allow/deny，分别表示允许/拒绝

Service

参数名称	类型	描述
serviceId	String	服务发布点的id
name	String	服务发布点的名称
description	String	描述
serviceName	String	服务名称
bindType	String	绑定服务类型，目前仅支持绑定BLB实例
instanceId	String	绑定实例ID
status	String	发布点状态，取值范围inService/available/unavailable/dead/free，分别表示：服务中/可用/不可用/故障/未绑定
service	String	服务发布点唯一对应域名
createTime	String	创建时间
endpointCount	int	关联的服务网卡数量
endpointList	List< RelatedEndpoint >	关联的服务网卡列表
authList	List< Auth >	授权列表

RelatedEndpoint

参数名称	类型	描述
endpointId	String	服务网卡的id
uid	String	服务网卡对应用户id
attachTime	String	关联时间

文档更新记录

2018-01-23

- 支持用户创建 BLB 实例时指定所属 VPC。

2019-03-21

- 新增应用型BLB相关API接口，及其对应的数据结构
- 普通型BLB接口新增创建、查询、更新SSL监听器接口
- 普通型BLB增加查询BLB示例详情接口
- BLB支持的服务域名更新

SDK

Java-SDK

概述

本文档主要介绍BLB Java SDK的安装和使用。在使用本文档前，您需要先了解BLB的一些基本知识，并已经开通了BLB服务。若您还不了解BLB，可以参考[产品描述](#)和[快速入门](#)。

安装SDK工具包

运行环境

1. 运行环境 Java SDK工具包可在 jdk1.7、jdk1.8环境下运行。
2. 鉴权和认证 要使用百度云产品，您需要拥有一个百度云账号和一个有效的 AK(Access Key ID)、SK(Secret Access Key)用来进行签名认证。

可以通过如下步骤获得并了解您的AK/SK信息：

1. [注册百度云账号](#)
2. [创建AK/SK](#)

安装SDK

方式一：使用Maven安装

在 Maven 的 pom.xml 文件中添加 bce-java-sdk 的依赖：

```
<dependency>
  <groupId>com.baidubce</groupId>
  <artifactId>bce-java-sdk</artifactId>
  <version>{version}</version>
</dependency>
```

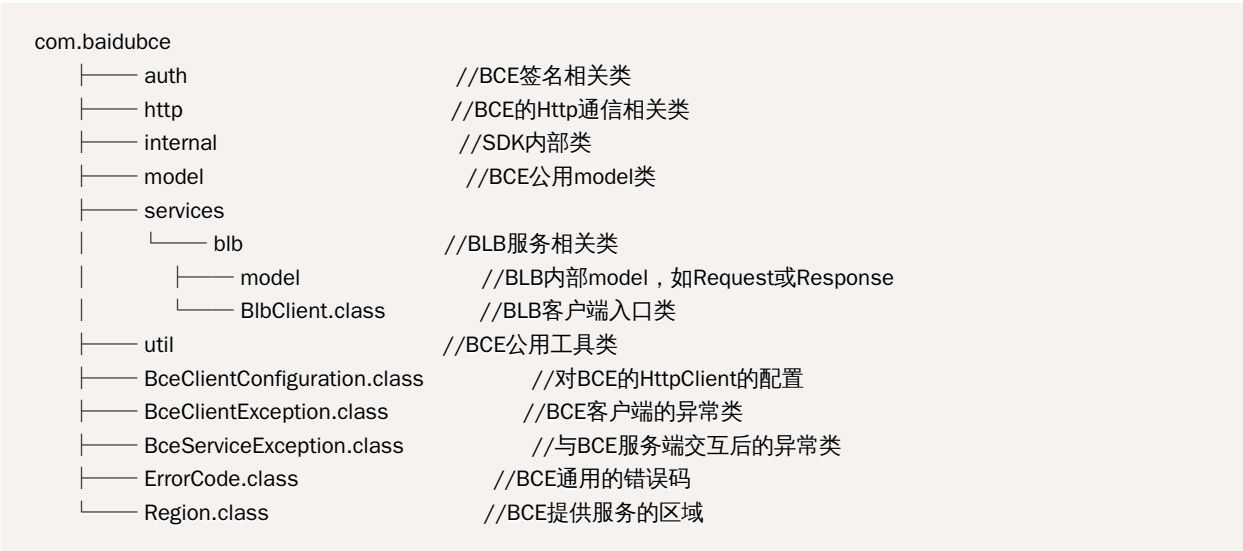
其中，{version}为版本号，可以在[开发者资源SDK中心](#)找到。

方式二：直接使用JAR包安装

1. 在[官方网站](#)下载Java SDK压缩工具包。
2. 将下载的bce-java-sdk-version.zip解压后，复制到工程文件夹中。
3. 在Eclipse右键“工程 -> Properties -> Java Build Path -> Add JARs”。
4. 添加SDK工具包lib/bce-java-sdk-version.jar和第三方依赖工具包third-party/*.jar。

其中，version为版本号。

SDK目录结构



🔗 卸载SDK

预期卸载 SDK 时，删除 pom 依赖或源码即可。

初始化

BlbClient是Blb服务的Java客户端，用于管理BLB资源。使用Java SDK发起BLB请求，您需要初始化一个BlbClient实例，并根据需要修改BlbClientConfiguration的默认配置项。

🔗 确认Endpoint

在确认您使用SDK时配置的Endpoint时，可先阅读开发人员指南中关于[BLB访问域名](#)的部分，理解Endpoint相关的概念。 百度云目前开放了多区域支持，请参考[区域选择说明](#)。

目前支持“北京”、“广州”、“苏州”、“香港”、“武汉”、“保定”、“上海”、“新加坡”等区域。

服务域名的对应信息为：

访问区域	对应Endpoint
bj	blb.bj.baidubce.com
gz	blb.gz.baidubce.com
su	blb.su.baidubce.com
hkg	blb.hkg.baidubce.com
fwh	blb.fwh.baidubce.com
bd	blb.bd.baidubce.com
fsh	blb.fsh.baidubce.com
sin	blb.sin.baidubce.com

🔗 新建BlbClient

使用AK/SK新建BlbClient

通过AK/SK方式访问BLB，用户可以参考如下代码新建一个BlbClient：

```
public class Sample {  
public static void main(String[] args) {  
    String ACCESS_KEY_ID = <your-access-key-id>;           // 用户的Access Key ID  
    String SECRET_ACCESS_KEY = <your-secret-access-key>;    // 用户的Secret Access Key  
  
    // 初始化一个BlibClient  
    BlibClientConfiguration config = new BlibClientConfiguration();  
    config.setCredentials(new DefaultBceCredentials(ACCESS_KEY_ID, SECRET_ACCESS_KEY));  
    BlibClient client = new BlibClient(config);  
    }  
}
```

在上面代码中，ACCESS_KEY_ID对应控制台中的“Access Key ID”，SECRET_ACCESS_KEY对应控制台中的“Access Key Secret”，获取方式请参考《操作指南 [管理ACCESSKEY](#)》。

上面的方式使用默认域名作为BLB的服务地址，如果用户需要自己指定域名，可以通过传入ENDPOINT参数来指定。

```
String ACCESS_KEY_ID = <your-access-key-id>;           // 用户的Access Key ID  
String SECRET_ACCESS_KEY = <your-secret-access-key>;    // 用户的Secret Access Key  
String ENDPOINT = <domain-name>;                      // 用户自己指定的域名  
  
BlibClientConfiguration config = new BlibClientConfiguration();  
config.setCredentials(new DefaultBceCredentials(ACCESS_KEY_ID, SECRET_ACCESS_KEY));  
config.setEndpoint(ENDPOINT);  
BlibClient client = new BlibClient(config);
```

注意：ENDPOINT参数只能用指定的包含区域的域名来进行定义，不指定时默认为北京区域http://blb.bj.baidubce.com。

设置网络参数

用户可以用BlibClientConfiguration对基本网络参数进行设置：

```
BlibClientConfiguration config = new BlibClientConfiguration();  
  
// 设置HTTP最大连接数为10  
config.setMaxConnections(10);  
  
// 设置TCP连接超时为5000毫秒  
config.setConnectionTimeout(5000);  
  
// 设置Socket传输数据超时的时间为2000毫秒  
config.setSocketTimeout(2000);
```

参数说明

通过BlibClientConfiguration能指定的所有参数如下表所示：

参数	说明
UserAgent	用户代理，指HTTP的User-Agent头
Protocol	连接协议类型
ProxyDomain	访问NTLM验证的代理服务器的Windows域名
ProxyHost	代理服务器主机地址
ProxyPort	代理服务器端口
ProxyUsername	代理服务器验证的用户名
ProxyPassword	代理服务器验证的密码
ProxyPreemptiveAuthenticationEnabled	是否设置用户代理认证
ProxyWorkstation	NTLM代理服务器的Windows工作站名称
LocalAddress	本地地址
ConnectionTimeoutInMillis	建立连接的超时时间（单位：毫秒）
SocketTimeoutInMillis	通过打开的连接传输数据的超时时间（单位：毫秒）
MaxConnections	允许打开的最大HTTP连接数
RetryPolicy	连接重试策略
SocketBufferSizeInBytes	Socket缓冲区大小
StreamBufferSize	流文件缓冲区大小

普通型BLB实例

🔗 获取Endpoint

在确认您使用SDK配置的Endpoint时，可先阅读开发人员指南中关于[BLB服务域名](#)的部分，理解Endpoint相关的概念。 百度智能云目前开放了多区域支持，请参考[区域选择说明](#)中网络产品BLB的部分。

注意： 负载均衡BLB API支持HTTP和HTTPS两种调用方式。为了提升数据的安全性，建议通过HTTPS调用

🔗 获取AK/SK

要使用百度智能云负载均衡BLB，您需要拥有一个有效的 AK（Access Key ID）和SK(Secret Access Key)用来进行签名认证。AK/SK是由系统分配给用户的，均为字符串，用于标识用户，为访问服务做签名验证。 可以通过如下步骤获得并了解您的AK/SK信息：

- 1. [注册百度智能云账号](#)
- 2. [创建AK/SK](#)

🔗 新建BlbClient

BlbClient是负载均衡BLB服务的客户端，为开发者与负载均衡BLB服务进行交互提供了一系列的方法。 新建BlbClient时，需要先使用Endpoint、AK、SK对BceClientConfigurationI类型的config实例进行配置，再使用config实例对BlbClient进行配置，具体配置方法如下：

```
static final String ENDPOINT = ""
static final String AK = "";
static final String SK = "";

BlbClientConfiguration config = new BlbClientConfiguration();
config.setCredentials(new DefaultBceCredentials(ak, sk));
config.setEndpoint(endpoint);
BlbClient blbClient = new BlbClient(config);
```

创建BLB实例

函数声明

```
public CreateBlbResponse createBlb(String name, String desc, String vpcId, String subnetId) {
}
public CreateBlbResponse createBlb(CreateBlbRequest createBlbRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [createloadbalancer创建实例请求参数](#)

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "blbId": "lb-BLuOPSLZ",
  "name": "blb-for-test",
  "address": "192.168.0.24",
  "desc": "myblb"
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateBlb.java](#)

查询已有的BLB实例

函数声明

```
public ListBlbResponse listBlbs(String address, String name, String blbId, String bcId) {
}
public ListBlbResponse listBlbs(ListBlbRequest listBlbRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeloadbalancers](#)查询已有的blb实例请求参数

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "blbList": [
    {
      "blbId": "lb-a7e5zPPk",
      "status": "available",
      "name": "test-blb",
      "desc": "desc",
      "address": "192.168.0.23"
    },
    {
      "blbId": "lb-gj5gVpeq",
      "status": "available",
      "name": "nametest",
      "desc": "desc",
      "address": "192.168.0.24",
      "publicIp": "123.2.3.4"
    }
  ],
  "marker": "blb-0A20F971",
  "nextMarker": "blb-0A20FB09",
  "isTruncated": true,
  "maxKeys": 2
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListBlbs.java](#)

查询BLB实例详情

函数声明

```
public BlbInstance blbDetail(BlbDetailRequest request) {
}
```

参数含义

请参考OpenAPI文档 [describeloadbalancerdetail](#)查询blb实例详情请求参数

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "blbId": "lb-gj5gVpeq",
  "status": "available",
  "name": "nametest",
  "desc": "",
  "address": "192.168.0.2",
  "publicIp": "123.2.3.4",
  "cidr": "192.168.0.0/16",
  "vpcName": "test",
  "createTime": "2019-03-07T02:35:31Z",
  "releaseTime": "2019-03-07T02:35:31Z",
  "listener": [
    {
      "port": 80,
      "backendPort": 90,
      "type": "TCP",
      "healthCheckType": "TCP",
      "healthCheckStatus": "Alive"
    }
  ]
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleBlbDetail.java](#)

更新BLB实例

函数声明

```
public void modifyBlbAttributes(String blbId, String name, String desc) {
}
public void modifyBlbAttributes(ModifyBlbAttributesRequest modifyBlbAttributesRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [updateLoadBalancer更新实例请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleModifyBlbAttributes.java](#)

🔗 释放BLB实例

函数声明

```
public void deleteBlb(String blbId) {  
}  
public void deleteBlb(DeleteBlbRequest deleteBlbRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [deleteloadbalancer释放blb实例请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteBlb.java](#)

🔗 更新BLB实例ACL功能

函数声明

```
public void updateLoadBalancerAcl(UpdateLoadBalancerAclRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [updateloadbalanceracl更新实例acl功能请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateLoadBalancerAcl.java](#)

🔗 创建TCP监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createtcplistener创建tcp监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateTcpListener.java](#)

🔗 创建UDP监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createudplistener创建udp监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateUdpListener.java](#)

🔗 创建HTTP监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createhttplistener创建http监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateHttpListener.java](#)

🔗 创建HTTPS监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createhttpslistener创建https监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateHttpsListener.java](#)

🔗 创建SSL监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createssllistener创建ssl监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateSslListener.java](#)

🔗 查询TCP监听器

函数声明

```
public ListListenerResponse<TcpListener> listTcpListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describetcplisteners](#)查询tcp监听器请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "listenerList": [  
    {  
      "listenerPort": 80,  
      "backendPort": 80,  
      "scheduler": "RoundRobin",  
      "healthCheckTimeoutInSecond": 3,  
      "healthCheckInterval": 3,  
      "unhealthyThreshold": 3,  
      "healthyThreshold": 3  
    },  
    {  
      "listenerPort": 88,  
      "backendPort": 88,  
      "scheduler": "RoundRobin",  
      "healthCheckTimeoutInSecond": 2,  
      "healthCheckInterval": 4,  
      "unhealthyThreshold": 3,  
      "healthyThreshold": 3  
    }  
  ],  
  "marker": "listener-0050",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 2  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListTcpListener.java](#)

🔗 查询UDP监听器

函数声明

```
public ListListenerResponse<UdpListener> listUdpListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeudplisteners查询udp监听器请求参数](#)

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "listenerList": [  
    {  
      "listenerPort": 80,  
      "backendPort": 80,  
      "scheduler": "RoundRobin",  
      "healthCheckType": "UDP",  
      "healthCheckTimeoutInSecond": 3,  
      "healthCheckInterval": 3,  
      "unhealthyThreshold": 3,  
      "healthyThreshold": 3,  
      "healthCheckString": "healthCheck",  
      "udpSessionTimeout": 90  
    },  
    {  
      "listenerPort": 88,  
      "backendPort": 88,  
      "scheduler": "RoundRobin",  
      "healthCheckType": "UDP",  
      "healthCheckTimeoutInSecond": 2,  
      "healthCheckInterval": 4,  
      "unhealthyThreshold": 3,  
      "healthyThreshold": 3,  
      "healthCheckString": "healthCheck",  
      "udpSessionTimeout": 90  
    }  
  ],  
  "marker": "listener-0050",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 2  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListUdpListener.java](#)

🔗 查询HTTP监听器

函数声明

```
public ListListenerResponse<HttpListener> listHttpListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describehttplisteners查询http监听器请求参数](#)

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 90,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_2xx|http_3xx",
      "serverTimeout": 30,
      "redirectPort": 80
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 95,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_1xx|http_2xx",
      "serverTimeout": 1800,
      "redirectPort": 80
    }
  ],
  "marker": "listener-005A",
  "nextMarker": "listener-0322",
  "isTruncated": true,
  "maxKeys": 2
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListHttpListener.java](#)

 查询HTTPS监听器

函数声明

```
public ListListenerResponse<HttpsListener> listHttpsListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describehttpslisteners](#)查询https监听器请求参数

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 90,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_2xx|http_3xx",
      "serverTimeout": 30,
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 95,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_1xx|http_2xx",
      "serverTimeout": 1800,
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-005A",
  "nextMarker": "listener-0322",
  "isTruncated": true,
  "maxKeys": 2
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateHttpsListener.java](#)

 查询SSL监听器

函数声明

```
public ListListenerResponse<SslListener> listSslListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describesssllisteners](#)查询ssl监听器请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "listenerList": [  
    {  
      "listenerPort": 80,  
      "backendPort": 80,  
      "scheduler": "RoundRobin",  
      "healthCheckTimeoutInSecond": 3,  
      "healthCheckInterval": 3,  
      "unhealthyThreshold": 3,  
      "healthyThreshold": 3  
    },  
    {  
      "listenerPort": 88,  
      "backendPort": 88,  
      "scheduler": "RoundRobin",  
      "healthCheckTimeoutInSecond": 2,  
      "healthCheckInterval": 4,  
      "unhealthyThreshold": 3,  
      "healthyThreshold": 3  
    }  
  ],  
  "marker": "listener-0050",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 2  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListSslListener.java](#)

 查询所有监听

函数声明

```
public ListListenerResponse<AllListener> listAllListener(String blbId) {  
}  
public ListListenerResponse<AllListener> listAllListener(ListAllListenerRequest request) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describealllisteners查询所有监听请求参数](#)

返回值

- 操作成功

```

{
  "metadata": "具体信息参考附录metadata部分",
  "marker" : "listener-0035",
  "isTruncated" : false,
  "maxKeys" : 1000,
  "listenerList" : [ {
    "listenerPort" : 53,
    "backendPort" : 53,
    "backendPortType" : "UDP",
    "scheduler" : "RoundRobin",
    "healthCheckTimeoutInSecond" : 3,
    "healthCheckInterval" : 3,
    "unhealthyThreshold" : 3,
    "healthyThreshold" : 3,
    "getBlbIp" : false,
    "healthCheckValid" : 1,
    "listenerType" : "UDP",
    "udpSessionTimeout" : 30,
    "healthCheckString" :
    "\\00\\01\\01\\00\\00\\01\\00\\00\\00\\00\\00\\00\\05baidu\\03com\\00\\00\\01\\00\\01",
    "healthCheckType" : "UDP"
  }, {
    "listenerPort" : 80,
    "backendPort" : 80,
    "backendPortType" : "TCP",
    "scheduler" : "RoundRobin",
    "healthCheckTimeoutInSecond" : 3,
    "healthCheckInterval" : 3,
    "unhealthyThreshold" : 3,
    "healthyThreshold" : 3,
    "getBlbIp" : false,
    "healthCheckValid" : 1,
    "listenerType" : "TCP",
    "tcpSessionTimeout" : 900
  }, {
    "listenerPort" : 81,
    "backendPort" : 80,
    "backendPortType" : "HTTP",
    "scheduler" : "RoundRobin",
    "healthCheckTimeoutInSecond" : 3,
    "healthCheckInterval" : 3,
    "unhealthyThreshold" : 3,
    "healthyThreshold" : 3,
    "getBlbIp" : false,
    "healthCheckValid" : 1,
    "listenerType" : "HTTP",
    "keepSession" : false,
    "xForwardFor" : true,
    "healthCheckType" : "HTTP",
    "healthCheckPort" : 80,
    "healthCheckURI" : "/",
    "healthCheckNormalStatus" : "http_2xx|http_3xx",
    "healthCheckHost" : "",
    "serverTimeout" : 30,
    "redirectPort" : 0
  } ]
}

```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListAllListener.java](#)

更新TCP监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updatetcplistener更新tcp监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateTcpListener.java](#)

更新UDP监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updateudplistener更新udp监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateUdpListener.java](#)

更新HTTP监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updatehttplistener更新http监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateHttpListener.java](#)

更新HTTPS监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updatehttpslistener更新https监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateHttpsListener.java](#)

更新SSL监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updatessllistener更新ssl监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateSslListener.java](#)

🔗 释放监听器

函数声明

```
public void deleteListener(String blbId, List<Integer> portList) {  
}  
public void deleteListener(DeleteListenerRequest deleteListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [deletelisteners](#)释放监听器请求参数

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteListeners.java](#)

🔗 添加后端服务器

函数声明

```
public void addBackendServers(String blbId, List<BackendServer> backendServerList) {  
}  
public void addBackendServers(AddBackendServersRequest addBackendServersRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [addbackendservers](#)添加后端服务器请求参数

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleAddBackendServers.java](#)

🔗 查询后端服务器健康状态

函数声明

```
public ListBackendServerStatusResponse listBackendServerStatus(String blbId, int listenerPort) {  
}  
public ListBackendServerStatusResponse listBackendServerStatus(ListBackendServerStatusRequest  
listBackendServerStatusRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describehealthstatus查询后端服务器健康状态请求参数](#)

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "backendServerList": [  
    {  
      "instanceId": "i-YfAibI4A",  
      "weight": 50,  
      "status": "Alive"  
    },  
    {  
      "instanceId": "i-jBXABQ8D",  
      "weight": 60,  
      "status": "Dead"  
    }  
  ],  
  "type": "TCP",  
  "listenerPort": 88,  
  "backendPort": 88,  
  "marker": "rs-OA6BE9BB",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 1000  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListBackendServerStatus.java](#)

🔗 查询后端服务器列表

函数声明

```
public ListBackendServerResponse listBackendServers(String blbId) {  
}  
public ListBackendServerResponse listBackendServers(ListBackendServerRequest listBackendServerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describebackendservers](#)查询后端服务器列表请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "backendServerList": [  
    {  
      "instanceId": "i-YfAib14A",  
      "weight": 50,  
    }  
  ],  
  "marker": "rs-0A6BE9BB",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 1000  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListBackendServers.java](#)

🔗 更新后端服务器

函数声明

```
public void modifyBackendServerAttributes(String blbId, List<BackendServer> backendServerList) {  
}  
public void modifyBackendServerAttributes(ModifyBSAttributesRequest modifyBSAttributesRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [updatebackendservers更新后端服务器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleModifyBackendServerAttributes.java](#)

🔗 释放后端服务器

函数声明

```
public void deleteBackendServers(String blbId, List<String> backendServerList) {  
}  
public void deleteBackendServers>DeleteBSRequest deleteBSRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [removebackendservers释放后端服务器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteBackendServers.java](#)

🔗 绑定普通安全组

函数声明

```
public void bindSg(SgOperateRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [bindsecuritygroups绑定普通安全组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleBindSg.java](#)

🔗 解绑普通安全组

函数声明

```
public void unBindSg(SgOperateRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [unbindsecuritygroups解绑普通安全组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUnBindSg.java](#)

🔗 查询BLB实例普通安全组列表

函数声明

```
public ListBlbSgResponse listBlbSg(ListBlbSgRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [describsecuritygroups查询blb实例普通安全组列表请求参数](#)

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "blbSecurityGroups": [
    {
      "securityGroupDesc": "",
      "securityGroupId": "g-4NxWoxeq",
      "securityGroupName": "securitygroup-name",
      "vpcName": "vpc-name",
      "securityGroupRules": [
        {
          "destGroupId": "",
          "destIp": "all",
          "direction": "egress",
          "ethertype": "IPv4",
          "portRange": "1-65535",
          "protocol": "all",
          "securityGroupRuleId": "r-gkv8yupumvx2"
        }
      ]
    }
  ]
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListBlbSg.java](#)

🔗 绑定企业安全组

函数声明

```
public void bindEsg(EsgOperateRequest request) {
}
```

参数含义

请参考OpenAPI文档 [bindenterprisesecuritygroups绑定企业安全组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleBindEsg.java](#)

🔗 解绑企业安全组

函数声明


```
public void unBindEsg(EsgOperateRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [unbindenterprisesecuritygroups解绑企业安全组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUnBindEsg.java](#)

查询BLB实例企业安全组列表

函数声明

```
public ListBlbEsgResponse listBlbEsg(ListBlbSgRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [describeenterprisesecuritygroups查询blb实例企业安全组列表请求参数](#)

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "enterpriseSecurityGroups": [{
    "enterpriseSecurityGroupId": "esg-g8z4qfj0m0bu",
    "enterpriseSecurityGroupName": "test0627",
    "enterpriseSecurityGroupDesc": "",
    "enterpriseSecurityGroupRules": [{
      "remark": "全部协议",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv6",
      "portRange": "",
      "sourcelp": "",
      "destlp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu",
      "protocol": "all"
    },
    {
      "remark": "",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "",
      "sourcelp": "",
      "destlp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9je09z6bd9wv",
      "protocol": "tcp"
    },
    {
      "remark": "",
      "direction": "ingress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "2",
      "sourcelp": "all",
      "destlp": "",
      "enterpriseSecurityGroupRuleId": "esgr-jcej4juqbh49",
      "protocol": "tcp"
    }
  ]
}]
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListBlbEsg.java](#)

附录

公共返回信息metadata格式

```
{
  "bceRequestId": "f5f0821d-45fe-439b-bbf4-fc48b639f84a",
  "contentLength": 28,
  "contentType": "application/json;charset=UTF-8",
  "date": "1701917695000",
  "server": "openresty/1.15.8.1"
}
```

应用型BLB实例

🔗 获取Endpoint

在确认您使用SDK配置的Endpoint时，可先阅读开发人员指南中关于[BLB服务域名](#)的部分，理解Endpoint相关的概念。百度智能云目前开放了多区域支持，请参考[区域选择说明](#)中网络产品BLB的部分。

注意： 负载均衡BLB API支持HTTP和HTTPS两种调用方式。为了提升数据的安全性，建议通过HTTPS调用

🔗 获取AK/SK

要使用百度智能云负载均衡BLB，您需要拥有一个有效的 AK（Access Key ID）和SK(Secret Access Key)用来进行签名认证。AK/SK是由系统分配给用户的，均为字符串，用于标识用户，为访问服务做签名验证。可以通过如下步骤获得并了解您的AK/SK信息：

1. [注册百度智能云账号](#)
2. [创建AK/SK](#)

🔗 新建AppBlbClient

AppBlbClient是负载均衡BLB（应用型）服务的客户端，为开发者与负载均衡BLB服务进行交互提供了一系列的方法。新建AppBlbClient时，需要先使用Endpoint、AK、SK对BceClientConfigurationI类型的config实例进行配置，再使用config实例对AppBlbClient进行配置，具体配置方法如下：

```
static final String ENDPOINT = "";
static final String AK = "";
static final String SK = "";

BlbClientConfiguration config = new BlbClientConfiguration();
config.setCredentials(new DefaultBceCredentials(ak, sk));
config.setEndpoint(endpoint);
AppBlbClient appBlbClient = new AppBlbClient(config);
```

🔗 创建BLB实例

函数声明

```
public CreateBlbResponse createBlb(String name, String desc, String vpId, String subnetId, List<Tag> tags) {
}
public CreateBlbResponse createBlb(CreateBlbRequest createBlbRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [createloadbalancer创建实例请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "blbId": "lb-BLuOPSLZ",
  "name": "blb-for-test",
  "address": "192.168.0.24",
  "desc": "myblb"
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateBlb.java](#)

更新BLB实例

函数声明

```
public void modifyBlbAttributes(String blbId, String name, String desc) {
}
public void modifyBlbAttributes(ModifyBlbAttributesRequest modifyBlbAttributesRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [updateloadbalancer更新实例请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleModifyAppBlbAttributes.java](#)

查询已有的BLB实例

函数声明

```
public ListBlbResponse listBlbs(String address, String name, String blbld, String bcld) {  
}  
public ListBlbResponse listBlbs(ListBlbRequest listBlbRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeloadbalancers](#)查询已有的blb实例请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "blbList": [  
    {  
      "blbld": "lb-a7e5zPPk",  
      "status": available,  
      "name": "test-blb",  
      "desc": desc,  
      "address": "192.168.0.23"  
    },  
    {  
      "blbld": "lb-gj5gVpeq",  
      "status": available,  
      "name": "nametest",  
      "desc": desc,  
      "address": "192.168.0.24",  
      "publicIp": "123.2.3.4"  
    }  
  ],  
  "marker": "blb-0A20F971",  
  "nextMarker": "blb-0A20FB09",  
  "isTruncated": true,  
  "maxKeys": 2  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListAppBlbs.java](#)

查询BLB实例详情

函数声明

```
public BlbInstance blbDetail(BlbDetailRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [describeloadbalancers](#)查询已有的blb实例请求参数

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "blbId": "lb-gj5gVpeq",
  "status": "available",
  "name": "nametest",
  "desc": "",
  "address": "192.168.0.2",
  "publicIp": "123.2.3.4",
  "cidr": "192.168.0.0/16",
  "vpcName": "test",
  "createTime": "2019-03-07T02:35:31Z",
  "releaseTime": "2019-03-07T02:35:31Z",
  "listener": [
    {
      "port": 80,
      "backendPort": 90,
      "type": "TCP",
      "healthCheckType": "TCP",
      "healthCheckStatus": "Alive"
    }
  ]
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleAppBlbDetail.java](#)

🔗 释放BLB实例

函数声明

```
public void deleteBlb(String blbId) {
}
public void deleteBlb(DeleteBlbRequest deleteBlbRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [deleteloadbalancer](#)释放blb实例请求参数

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteAppBlb.java](#)

🔗 创建TCP监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createapptcplistener创建tcp监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateAppTcpListener.java](#)

🔗 创建UDP监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createappudplistener创建udp监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateAppUdpListener.java](#)

🔗 创建HTTP监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createapphttplistener创建http监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateAppHttpListener.java](#)

🔗 创建HTTPS监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createapphttpslistener创建https监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateAppHttpsListener.java](#)

🔗 创建SSL监听器

函数声明

```
public void createListener(BlbListenerRequest blbListenerRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createappssllistener创建ssl监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateAppSslListener.java](#)

更新TCP监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updateapptcplistener更新tcp监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateAppTcpListener.java](#)

更新UDP监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updateappudplistener更新udp监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateAppUdpListener.java](#)

 更新HTTP监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updateapphttplistener更新http监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateAppHttpListener.java](#)

 更新HTTPS监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updateapphttpslistener更新https监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateAppHTTPSListener.java](#)

 更新SSL监听器

函数声明

```
public void modifyListenerAttributes(BlbListenerRequest modifyListenerAttributesRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updateappssllistener更新ssl监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateAppSSLListener.java](#)

🔗 查询TCP监听器

函数声明

```
public ListListenerResponse<TcpListener> listTcpListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeapptcplisteners](#)查询应用型tcp监听器请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "listenerList": [  
    {  
      "listenerPort": 800,  
      "scheduler": "RoundRobin",  
      "tcpSessionTimeout": 900  
    },  
    {  
      "listenerPort": 808,  
      "scheduler": "RoundRobin",  
      "tcpSessionTimeout": 900  
    }  
  ],  
  "marker": "listener-0050",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 1000  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeAppTCPListeners.java](#)

🔗 查询UDP监听器

函数声明

```
public ListListenerResponse<UdpListener> listUdpListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeappudplisteners查询udp监听器请求参数](#)

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "listenerList": [  
    {  
      "listenerPort": 53,  
      "scheduler": "RoundRobin",  
      "udpSessionTimeout": 30  
    },  
    {  
      "listenerPort": 54,  
      "scheduler": "RoundRobin",  
      "udpSessionTimeout": 30  
    }  
  ],  
  "marker": "listener-0050",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 1000  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeAppUDPListeners.java](#)

🔗 查询HTTP监听器

函数声明

```
public ListListenerResponse<HttpListener> listHttpListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeapphttplisteners](#)查询http监听器请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "listenerList": [  
    {  
      "listenerPort": 80,  
      "scheduler": "RoundRobin",  
      "keepSession": false,  
      "xForwardedFor": true,  
      "xForwardedProto": false,  
      "serverTimeout": 30,  
      "redirectPort": 0  
    },  
    {  
      "listenerPort": 801,  
      "scheduler": "RoundRobin",  
      "keepSession": false,  
      "xForwardedFor": true,  
      "xForwardedProto": false,  
      "serverTimeout": 30,  
      "redirectPort": 0  
    }  
  ],  
  "marker": "listener-005A",  
  "nextMarker": "listener-0322",  
  "isTruncated": true,  
  "maxKeys": 1000  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeAppHTTPListeners.java](#)

 查询HTTPS监听器

函数声明

```
public ListListenerResponse<HttpsListener> listHttpsListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeaphttpslisteners查询https监听器请求参数](#)

返回值

- 操作成功

```

{
  "metadata": "具体信息参考附录metadata部分",
  "listenerList": [
    {
      "listenerPort" : 443,
      "scheduler" : "RoundRobin",
      "keepSession" : false,
      "xForwardedFor" : true,
      "xForwardedProto" : false,
      "serverTimeout" : 30,
      "certIds" : [ "cert-gs8bktrm7drp" ],
      "encryptionType" : "tls_cipher_policy_default",
      "encryptionProtocols" : [ "tlsv10", "tlsv11", "tlsv12" ],
      "appliedCiphers" : "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-
AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-
RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-
SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA",
      "dualAuth" : false,
      "clientCertIds" : [ ]
    },
    {
      "listenerPort" : 448,
      "scheduler" : "RoundRobin",
      "keepSession" : false,
      "xForwardedFor" : true,
      "xForwardedProto" : false,
      "serverTimeout" : 30,
      "certIds" : [ "cert-gs8bktrm7drp" ],
      "encryptionType" : "tls_cipher_policy_default",
      "encryptionProtocols" : [ "tlsv10", "tlsv11", "tlsv12" ],
      "appliedCiphers" : "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-
AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-
RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-
SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA",
      "dualAuth" : false,
      "clientCertIds" : [ ]
    }
  ],
  "marker": "listener-005A",
  "nextMarker": "listener-0322",
  "isTruncated": true,
  "maxKeys": 1000
}

```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeAppHTTPListeners.java](#)

🔗 查询SSL监听器

函数声明

```
public ListListenerResponse<SslListener> listSslListener(String blbId) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeappssllisteners](#)查询应用型ssl监听器请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "listenerList": [  
    {  
      "listenerPort": 444,  
      "scheduler": "RoundRobin",  
      "certIds": [ "cert-gs8bktrm7drp" ],  
      "encryptionType": "tls_cipher_policy_default",  
      "encryptionProtocols": [ "tlsv10", "tlsv11", "tlsv12" ],  
      "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA",  
      "dualAuth": false,  
      "clientCertIds": [ ],  
      "serverTimeout": 900  
    },  
    {  
      "listenerPort": 445,  
      "scheduler": "RoundRobin",  
      "certIds": [ "cert-4cn1jhp747" ],  
      "encryptionType": "tls_cipher_policy_default",  
      "encryptionProtocols": [ "tlsv10", "tlsv11", "tlsv12" ],  
      "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA",  
      "dualAuth": false,  
      "clientCertIds": [ ],  
      "serverTimeout": 900  
    }  
  ],  
  "marker": "listener-0050",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 1000  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeAppSSLListeners.java](#)

🔗 查询所有监听器

函数声明

```
public ListListenerResponse<AllListener> listAllListener(String blbld) {  
}  
public ListListenerResponse listListener(ListListenerRequest listListenerRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeappalllisteners](#)查询所有监听请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "listenerList": [  
    {  
      "listenerPort": 53,  
      "scheduler": "RoundRobin",  
      "listenerType": "UDP",  
      "udpSessionTimeout": 900  
    },  
    {  
      "listenerPort": 80,  
      "scheduler": "RoundRobin",  
      "listenerType": "HTTP",  
      "keepSession": false,  
      "xForwardFor": true,  
      "xForwardedProto": false,  
      "serverTimeout": 30,  
      "redirectPort": 0  
    },  
    {  
      "listenerPort": 443,  
      "scheduler": "RoundRobin",  
      "listenerType": "HTTPS",  
      "keepSession": false,  
      "xForwardFor": true,  
      "xForwardedProto": false,  
      "serverTimeout": 30,  
      "certIds": [ "cert-gs8bktrm7drp" ],  
      "dualAuth": false,  
      "encryptionType": "tls_cipher_policy_default",  
      "encryptionProtocols": [ "tlsv10", "tlsv11", "tlsv12" ],  
      "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-
```

```
AES256-SHA:ECDSA-AES128-GCM-SHA256:ECDSA-AES128-SHA256:ECDSA-AES128-SHA:ECDSA-
RSA-AES256-GCM-SHA384:ECDSA-RSA-AES256-SHA384:ECDSA-RSA-AES256-SHA:ECDSA-RSA-AES128-GCM-
SHA256:ECDSA-RSA-AES128-SHA256:ECDSA-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-
SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA"
    },
    {
        "listenerPort" : 444,
        "scheduler" : "RoundRobin",
        "listenerType" : "SSL",
        "serverTimeout" : 900,
        "certIds" : [ "cert-gs8bktrm7drp" ],
        "dualAuth" : false,
        "encryptionType" : "tls_cipher_policy_default",
        "encryptionProtocols" : [ "tlsv10", "tlsv11", "tlsv12" ],
        "appliedCiphers" : "ECDHE-ECDSA-AES256-GCM-SHA384:ECDSA-AES256-SHA384:ECDSA-
AES256-SHA:ECDSA-AES128-GCM-SHA256:ECDSA-AES128-SHA256:ECDSA-AES128-SHA:ECDSA-
RSA-AES256-GCM-SHA384:ECDSA-RSA-AES256-SHA384:ECDSA-RSA-AES256-SHA:ECDSA-RSA-AES128-GCM-
SHA256:ECDSA-RSA-AES128-SHA256:ECDSA-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-
SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA"
    },
    {
        "listenerPort" : 800,
        "scheduler" : "RoundRobin",
        "listenerType" : "TCP",
        "tcpSessionTimeout" : 900
    }
],
"marker": "listener-0050",
"nextMarker": null,
"isTruncated": false,
"maxKeys": 1000
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeAppAllListeners.java](#)

🔗 释放监听器

函数声明

```
public void deleteListener(String blbId, List<Integer> portList) {
}
public void deleteListener(DeleteListenerRequest deleteListenerRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [deleteapplisteners释放监听器请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteAppListeners.java](#)

🔗 创建策略

函数声明

```
public CreateAppPolicyResponse createPolicys(String blbId, Integer listenerPort, String type, List<AppPolicy> appPolicyVos) {  
}  
public CreateAppPolicyResponse createPolicys(AppPolicyRequest appPolicyRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [createpolicys创建策略请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreatePolicys.java](#)

🔗 查询策略列表

函数声明

```
public ListAppPolicyResponse listPolicys(String blbId, Integer port) {  
}  
public ListAppPolicyResponse listPolicys(ListAppPolicyRequest listAppPolicyRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describepolicys](#)查询对应blb端口下策略信息请求参数

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "policyList": [
    {
      "id": "policy-e18d7448",
      "desc": "",
      "appServerGroupId": "sg-b8221879",
      "appServerGroupName": "sgp-6FF0traU",
      "groupType": "Server",
      "frontendPort": 443,
      "backendPort": 80,
      "portType": "HTTP",
      "priority": 3,
      "ruleList": [
        {
          "key": "uri",
          "value": "/aa"
        }
      ]
    },
    {
      "id": "policy-ed24ea7d",
      "desc": "",
      "appServerGroupId": "sg-b8221879",
      "appServerGroupName": "sgp-6FF0traU",
      "groupType": "Server",
      "frontendPort": 443,
      "backendPort": 80,
      "portType": "HTTP",
      "priority": 2,
      "ruleList": [
        {
          "key": "host",
          "value": "www.abc.com"
        }
      ]
    }
  ],
  "marker": "policy-e18d7448",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribePolicys.java](#)

函数声明

```
public void deletePolicys(String blbId, Integer port, List<String> policyIdList) {  
}  
public void deletePolicys(DeleteAppPolicyRequest deleteAppPolicyRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [deletepolicys批量删除策略请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeletePolicys.java](#)

🔗 创建应用型服务器组

函数声明

```
public AppSgResponse createAppServerGroup(String blbId, String name, String desc, List<AppBackendServer>  
backendServerList) {  
}  
public AppSgResponse createAppServerGroup(AppSgRequest appSgRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [createaappservergroup创建应用型服务器组请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "id" : "sg-0aae4f4c",
  "name" : "sgp-udT9IBTO",
  "desc" : "",
  "status" : "available"
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateAppServerGroup.java](#)

更新服务器组

函数声明

```
public void modifyAppServerGroupAttributes(String blbId, String sgId, String name, String desc) {
}
public void modifyAppServerGroupAttributes(AppSgRequest appSgRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [updateappservergroup更新服务器组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateAppServerGroup.java](#)

查询服务器组列表

函数声明

```
public ListAppSgResponse listAppServerGroup(String blbId, String name) {
}
public ListAppSgResponse listAppServerGroup(ListAppSgRequest listAppSgRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeappservergroup查询服务器组请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "appServerGroupList": [
    {
      "id": "sg-0aae4f4c",
      "name": "sgp-udT9IBTO",
      "desc": "",
      "status": "available",
      "portList": [ ]
    },
    {
      "id": "sg-b8221879",
      "name": "sgp-6FF0traU",
      "desc": "",
      "status": "available",
      "portList": [
        {
          "id": "port-c5a87acf",
          "status": "available",
          "port": 80,
          "type": "HTTP",
          "healthCheck": "HTTP",
          "healthCheckPort": 80,
          "healthCheckUriPath": "/",
          "healthCheckTimeoutInSecond": 3,
          "healthCheckIntervalInSecond": 3,
          "healthCheckDownRetry": 3,
          "healthCheckUpRetry": 3,
          "healthCheckNormalStatus": "http_2xx|http_3xx",
          "udpHealthCheckString": ""
        }
      ]
    }
  ],
  "marker": "62672",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeAppServerGroup.java](#)

函数声明

```
public void deleteAppServerGroup(String blbId, String sgId) {  
}  
public void deleteAppServerGroup(AppSgRequest appSgRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [deleteappservergroup删除服务器组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteAppServerGroup.java](#)

🔗 创建服务器组端口

函数声明

```
public AppSgPortResponse createAppServerGroupPort(AppSgPortRequest appSgPortRequest) {  
}
```

参数含义

请参考OpenAPI文档 [createappservergroupport创建应用型服务器组端口请求参数](#)

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "id" : "port-0b2c33c9",  
  "status" : "available"  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateAppServerGroupPort.java](#)

 更新服务器组端口

函数声明

```
public void modifyAppServerGroupPortAttributes(AppSgPortRequest appSgPortRequest) {  
}
```

参数含义

请参考OpenAPI文档 [updateappservergroupport更新服务器组端口请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateAppServerGroupPort.java](#)

 删除服务器组端口

函数声明

```
public void deleteAppServerGroupPort(String blbId, String sgId, List<String> portIdList) {  
}  
public void deleteAppServerGroupPort(AppSgPortRequest appSgPortRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [deleteappservergroupport删除服务器组端口请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteAppServerGroupPort.java](#)

 添加应用型BLB后端RS

函数声明

```
public void createBlbRs(String blbId, String sgId, List<AppBackendServer> backendServerList) {  
}  
public void createBlbRs(AppSgRequest appSgRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [createblbrs添加应用型blb后端rs请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateBlbRs.java](#)

更新服务器组下挂载的RS权重

函数声明

```
public void modifyBlbRs(String blbId, String sgId, List<AppBackendServer> backendServerList) {  
}  
public void modifyBlbRs(AppSgRequest appSgRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [updateblbrs更新服务器组下挂载的rs权重请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateBlbRs.java](#)

🔗 查询服务器组下的RS列表信息

函数声明

```
public ListAppRsResponse listBlbRs(String blbId, String sgId) {  
}  
public ListAppRsResponse listBlbRs(ListAppRsRequest listAppRsRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeblbrs查询服务器组下的rs列表信息请求参数](#)

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "backendServerList" : [  
    {  
      "instanceId" : "i-q9dIHUJm",  
      "weight" : 100,  
      "privateIp" : "192.168.16.11",  
      "portList" : [  
        {  
          "listenerPort" : 80,  
          "backendPort" : "80",  
          "portType" : "HTTP",  
          "healthCheckPortType" : "HTTP",  
          "status" : "unknown",  
          "policyId" : "policy-58b2c7bf"  
        }  
      ]  
    }  
  ],  
  "marker": "appr-6458619E",  
  "nextMarker": null,  
  "isTruncated": false,  
  "maxKeys": 1000  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeBlbRs.java](#)

🔗 删除服务器组下挂载的RS

函数声明

```
public void deleteBlbRs(String blbId, String sgId, List<String> backendServerIdList) {  
}  
public void deleteBlbRs(AppRsRequest appRsRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [deleteblbrs删除服务器组下挂载的rs请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteBlbRs.java](#)

🔗 [查询服务器组下绑定的RS](#)

函数声明

```
public ListAppRsMountResponse listBlbRsMount(String blbId, String sgId) {  
}  
public ListAppRsMountResponse listBlbRsMount(ListAppRsRequest listAppRsRequest) {  
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describersmount查询服务器组下绑定的server请求参数](#)

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "backendServerList" : [
    {
      "instanceId" : "i-q9dIHUJm",
      "weight" : 100,
      "privateIp" : "192.168.16.11",
      "portList" : [
        {
          "listenerPort" : 80,
          "backendPort" : "80",
          "portType" : "HTTP",
          "healthCheckPortType" : "HTTP",
          "status" : "unknown",
          "policyId" : "policy-58b2c7bf"
        }
      ]
    }
  ]
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeRsMount.java](#)

🔗 查询服务器组下未绑定的RS

函数声明

```
public ListAppRsUnMountResponse listBlbRsUnMount(String blbId, String sgId) {
}
public ListAppRsUnMountResponse listBlbRsUnMount(ListAppRsRequest listAppRsRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describersunmount查询服务器组下未绑定的rs请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "backendServerList": [
    {
      "instanceId": "i-q9dIHUJm",
      "privateIp": "192.168.16.11"
    }
  ]
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDescribeRsUnMount.java](#)

🔗 绑定普通安全组

函数声明

```
public void bindSg(SgOperateRequest request) {
}
```

参数含义

请参考OpenAPI文档 [bindsecuritygroups绑定普通安全组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleBindSg.java](#)

🔗 解绑普通安全组

函数声明

```
public void unBindSg(SgOperateRequest request) {
}
```

参数含义

请参考OpenAPI文档 [unbindsecuritygroups解绑普通安全组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUnBindSg.java](#)

🔗 查询BLB实例普通安全组列表

函数声明

```
public ListBlbSgResponse listBlbSg(ListBlbSgRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [describesecuritygroups](#)查询blb实例普通安全组列表请求参数

返回值

- 操作成功

```
{  
  "metadata":具体信息参考附录metadata部分,  
  "blbSecurityGroups": [  
    {  
      "securityGroupDesc": "",  
      "securityGroupId": "g-4NxWoxeq",  
      "securityGroupName": "securitygroup-name",  
      "vpcName": "vpc-name",  
      "securityGroupRules": [  
        {  
          "destGroupId": "",  
          "destIp": "all",  
          "direction": "egress",  
          "ethertype": "IPv4",  
          "portRange": "1-65535",  
          "protocol": "all",  
          "securityGroupRuleId": "r-gkv8yupumvx2"  
        }  
      ]  
    }  
  ]  
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListAppBlbSg.java](#)

🔗 绑定企业安全组

函数声明

```
public void bindEsg(EsgOperateRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [bindenterprisesecuritygroups绑定企业安全组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleBindEsg.java](#)

解绑企业安全组

函数声明

```
public void unBindEsg(EsgOperateRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [unbindenterprisesecuritygroups解绑企业安全组请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUnBindEsg.java](#)

查询BLB实例企业安全组列表

函数声明

```
public ListBlbEsgResponse listBlbEsg(ListBlbSgRequest request) {  
}
```

参数含义

请参考OpenAPI文档 [describeenterprisesecuritygroups查询blb实例企业安全组列表请求参数](#)

返回值

- 操作成功


```

{
  "metadata": "具体信息参考附录metadata部分",
  "enterpriseSecurityGroups": [
    {
      "enterpriseSecurityGroupId": "esg-g8z4qfj0m0bu",
      "enterpriseSecurityGroupName": "test0627",
      "enterpriseSecurityGroupDesc": "",
      "enterpriseSecurityGroupRules": [
        {
          "remark": "全部协议",
          "direction": "egress",
          "action": "allow",
          "priority": 1000,
          "ethertype": "IPv6",
          "portRange": "",
          "sourceIp": "",
          "destIp": "all",
          "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu",
          "protocol": "all"
        },
        {
          "remark": "",
          "direction": "egress",
          "action": "allow",
          "priority": 1000,
          "ethertype": "IPv4",
          "portRange": "",
          "sourceIp": "",
          "destIp": "all",
          "enterpriseSecurityGroupRuleId": "esgr-9je09z6bd9wv",
          "protocol": "tcp"
        },
        {
          "remark": "",
          "direction": "ingress",
          "action": "allow",
          "priority": 1000,
          "ethertype": "IPv4",
          "portRange": "2",
          "sourceIp": "all",
          "destIp": "",
          "enterpriseSecurityGroupRuleId": "esgr-jcej4juqbh49",
          "protocol": "tcp"
        }
      ]
    }
  ]
}

```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListBlbEsg.java](#)

附录

公共返回信息metadata格式

```
{
  "bceRequestId": "f5f0821d-45fe-439b-bbf4-fc48b639f84a",
  "contentLength": 28,
  "contentType": "application/json;charset=UTF-8",
  "date": "1701917695000",
  "server": "openresty/1.15.8.1"
}
```

IPv6型BLB实例

🔗 获取Endpoint

在确认您使用SDK配置的Endpoint时，可先阅读开发人员指南中关于[BLB服务域名](#)的部分，理解Endpoint相关的概念。百度智能云目前开放了多区域支持，请参考[区域选择说明](#)中网络产品BLB的部分。

注意： 负载均衡BLB API支持HTTP和HTTPS两种调用方式。为了提升数据的安全性，建议通过HTTPS调用

🔗 获取AK/SK

要使用百度智能云IPv6型负载均衡BLB，您需要拥有一个有效的 AK（Access Key ID）和SK(Secret Access Key)用来进行签名认证。AK/SK是由系统分配给用户的，均为字符串，用于标识用户，为访问服务做签名验证。可以通过如下步骤获得并了解您的AK/SK信息：

1. [注册百度智能云账号](#)
2. [创建AK/SK](#)

🔗 新建BlbClient

BlbClient是IPv6型负载均衡BLB服务的客户端，为开发者与负载均衡BLB服务进行交互提供了一系列的方法。新建BlbClient时，需要先使用Endpoint、AK、SK对BceClientConfiguration类型的config实例进行配置，再使用config实例对BlbClient进行配置，具体配置方法如下：

```
static final String ENDPOINT = "";
static final String AK = "";
static final String SK = "";

BlbClientConfiguration config = new BlbClientConfiguration();
config.setCredentials(new DefaultBceCredentials(ak, sk));
config.setEndpoint(endpoint);
BlbClient blbClient = new BlbClient(config);
```

🔗 创建IPv6BLB实例

函数声明

```
public CreateBlbResponse createIpv6Blb(String name, String desc, String vpcId, String subnetId) {
}
public CreateBlbResponse createBlb(CreateBlbRequest createBlbRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [createloadbalancer创建实例请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "blbId": "lb-BLuOPSLZ",
  "name": "blb-for-test",
  "address": "2400:da00:e003:0:46b:2400:0:3",
  "desc": "myblb"
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateIpv6Blb.java](#)

 查询已有的IPv6BLB实例

函数声明

```
public ListBlbResponse listIpv6Blbs(String address, String name, String blbId, String bcId) {
}
public ListBlbResponse listBlbs(ListBlbRequest listBlbRequest) {
}
```

注意：

第一个接口只支持少量常用参数

第二个接口则可以支持较多参数，但需要创建request实例

参数含义

请参考OpenAPI文档 [describeloadbalancers查询已有的blb实例请求参数](#)

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "blbList": [
    {
      "blbId": "lb-a7e5zPPk",
      "status": "available",
      "name": "test-blb",
      "desc": "desc",
      "address": "2400:da00:e003:0:46b:2400:0:3"
    },
    {
      "blbId": "lb-gj5gVpeq",
      "status": "available",
      "name": "nametest",
      "desc": "desc",
      "address": "2400:da00:e003:0:46b:2400:0:4"
    }
  ],
  "marker": "blb-0A20F971",
  "nextMarker": "blb-0A20FB09",
  "isTruncated": true,
  "maxKeys": 2
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListIpv6Blb.java](#)

🔗 其他接口

其他接口与普通型BLB接口都一致，详见[普通型BLB接口](#)

🔗 附录

公共返回信息metadata格式

```
{
  "bceRequestId": "f5f0821d-45fe-439b-bbf4-fc48b639f84a",
  "contentLength": 28,
  "contentType": "application/json; charset=UTF-8",
  "date": "1701917695000",
  "server": "openresty/1.15.8.1"
}
```

负载均衡专属集群LBDC

🔗 获取Endpoint

在确认您使用SDK配置的Endpoint时，可先阅读开发人员指南中关于[LBDC服务域名](#)的部分，理解Endpoint相关的概念。百度智能云目前开放了多区域支持，请参考[区域选择说明](#)中网络产品LBDC的部分。

注意： 负载均衡专属集群LBDC API支持HTTP和HTTPS两种调用方式。为了提升数据的安全性，建议通过HTTPS调用

🔗 获取AK/SK

要使用百度智能云负载均衡BLB，您需要拥有一个有效的 AK（Access Key ID）和SK(Secret Access Key)用来进行签名认证。AK/SK是由系统分配给用户的，均为字符串，用于标识用户，为访问服务做签名验证。可以通过如下步骤获得并了解您的AK/SK信息：

1. [注册百度智能云账号](#)
2. [创建AK/SK](#)

新建LbdcClient

LbdcClient是负载均衡专属集群LBDC服务的客户端，为开发者与负载均衡专属集群LBDC服务进行交互提供了一系列的方法。新建LbdcClient时，需要先使用Endpoint、AK、SK对BceClientConfigurationI类型的config实例进行配置，再使用config实例对LbdcClient进行配置，具体配置方法如下：

```
String ak = "Your Ak";
String sk = "Your Sk";
String endpoint = "blb.bj.baidubce.com";

BceClientConfiguration config = new BceClientConfiguration();
config.setCredentials(new DefaultBceCredentials(ak, sk));
config.setEndpoint(endpoint);
LbdcClient lbdcClient = new LbdcClient(config);
```

创建LBDC

函数声明

```
public CreateLbdcResponse createLbdc(CreateLbdcRequest body, String clientToken) {
}
```

参数含义

请参考OpenAPI文档 [创建lbdc请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "id": "bgw_group-a3t786p1",
  "name": "abc",
  "type": "4Layer",
  "desc":""
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateLbdc.java](#)

升级LBDC

函数声明

```
public void upgradeLbdc(String id, UpgradeLbdcRequest body, String clientToken) {  
}
```

参数含义

请参考OpenAPI文档 [升级lbdc请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpgradeLbdc.java](#)

续费LBDC

函数声明

```
public void renewLbdc(String id, RenewLbdcRequest body, String clientToken) {  
}
```

参数含义

请参考OpenAPI文档 [续费lbdc请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleRenewLbdc.java](#)

LBDC列表

函数声明

```
public ListLbdcResponse listLbdc(String id, String name) {  
}
```

参数含义

请参考OpenAPI文档 [lbdc列表请求参数](#)

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "clusterList": [
    {
      "id" : "bgw_group-f632f4bb",
      "name" : "sss",
      "type" : "4Layer",
      "status" : "available",
      "ccuCount" : 2,
      "createTime" : "2023-08-28T06:00:00Z",
      "expireTime" : "2024-12-07T08:06:45Z",
      "desc" : "desc"
    },
    {
      "id" : "nginx_group-4f63ff24",
      "name" : "7layerName",
      "type" : "7Layer",
      "status" : "available",
      "ccuCount" : 1,
      "createTime" : "2023-08-28T06:00:00Z",
      "expireTime" : "2023-09-28T06:00:00Z",
      "desc" : "desc"
    }
  ],
  "marker": "bgw_group-f632f4bb",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListLbdc.java](#)

LBDC详情

函数声明

```
public GetLbdcResponse getLbdc(String id) {
}
```

参数含义

请参考OpenAPI文档 [lbdc详情请参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "id" : "bgw_group-1f1b6e17",
  "name" : "ccqLbdc1117",
  "type" : "4Layer",
  "status" : "increasing",
  "ccuCount" : "8",
  "createTime" : "2023-11-17T09:57:33Z",
  "expireTime" : "2023-12-17T09:42:43Z",
  "totalConnectCount" : "640000000",
  "newConnectCps" : "96000000",
  "networkInBps" : "960000000000",
  "networkOutBps" : "960000000000"
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleGetLbdc.java](#)

更新LBDC

函数声明

```
public void updateLbdc(String id, UpdateLbdcRequest body, String clientToken) {
}
```

参数含义

请参考OpenAPI文档 [更新lbdc请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateLbdc.java](#)

LBDC关联的BLB列表

函数声明

```
public GetBoundBIBListOfLbdcResponse getBoundBIBListOfLbdc(String id) {
}
```

参数含义

请参考OpenAPI文档 [lbdc关联的bib列表请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "blbList": [
    {
      "blbId" : "lb-38eff7a8",
      "name" : "ipv6blb-test",
      "blbType" : "ipv6",
      "bandwidth" : 0,
      "address" : "0.0.0.0",
      "ipv6" : "2400:da00:e003:0:46b:2400:0:3",
      "vpclId" : "7d54f4be-9fae-47e1-9842-359e3104e3dd",
      "subnetId" : "fc315d1a-0940-4531-b9c5-474e23335f4c"
    },
    {
      "blbId" : "lb-20e783dd",
      "name" : "blb_1204_1",
      "blbType" : "application",
      "bandwidth" : 0,
      "address" : "192.168.0.2",
      "ipv6" : "",
      "vpclId" : "309228b5-4aaf-45b8-a149-37c8c6eaaef2",
      "subnetId" : "e3efb8c1-0e8f-4bcc-9295-58a532b5fe6e"
    }
  ]
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleGetBoundBIBListOfLbdc.java](#)

附录

公共返回信息metadata格式

```
{
  "bceRequestId":"f5f0821d-45fe-439b-bbf4-fc48b639f84a",
  "contentLength":28,
  "contentType":"application/json;charset=UTF-8",
  "date":1701917695000,
  "server":"openresty/1.15.8.1"
}
```

服务发布点

获取Endpoint

在确认您使用SDK配置的Endpoint时，可先阅读开发人员指南中关于[服务发布点服务域名](#)的部分，理解Endpoint相关的概念。
百度智能云目前开放了多区域支持，请参考[区域选择说明](#)中网络产品服务发布点的部分。

注意： 服务发布点 API支持HTTP和HTTPS两种调用方式。为了提升数据的安全性，建议通过HTTPS调用

获取AK/SK

要使用百度智能云服务发布点，您需要拥有一个有效的 AK（Access Key ID）和SK(Secret Access Key)用来进行签名认证。AK/SK是由系统分配给用户的，均为字符串，用于标识用户，为访问服务做签名验证。可以通过如下步骤获得并了解您的AK/SK信息：

1. [注册百度智能云账号](#)
2. [创建AK/SK](#)

新建UserserviceClient

UserserviceClient是服务发布点服务的客户端，为开发者与服务发布点服务进行交互提供了一系列的方法。新建UserserviceClient时，需要先使用Endpoint、AK、SK对BceClientConfigurationI类型的config实例进行配置，再使用config实例对LbdcClient进行配置，具体配置方法如下：

```
String ak = "Your Ak";
String sk = "Your Sk";
String endpoint = "blb.bj.baidubce.com";

BceClientConfiguration config = new BceClientConfiguration();
config.setCredentials(new DefaultBceCredentials(ak, sk));
config.setEndpoint(endpoint);
UserserviceClient userserviceClient = new UserserviceClient(config);
```

创建服务发布点

函数声明

```
public CreateUserServiceResponse createUserService(CreateUserServiceRequest body, String clientToken) {
}
```

参数含义

请参考OpenAPI文档 [创建服务发布点请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "service" : "testService.userService-3a76ed46.beijing.baidubce.com"
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleCreateUserService.java](#)

更新服务发布点

函数声明

```
public void updateUserService(String service, UpdateUserServiceRequest body, String clientToken) {
}
```

参数含义

请参考OpenAPI文档 [更新服务发布点请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUpdateUserService.java](#)

🔗 服务发布点绑定实例

函数声明

```
public void bindInstance(String service, BindInstanceRequest body, String clientToken) {  
}
```

参数含义

请参考OpenAPI文档 [服务发布点绑定实例请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleBindInstance.java](#)

🔗 服务发布点解绑实例

函数声明

```
public void unbindInstance(String service, String clientToken) {  
}
```

参数含义

请参考OpenAPI文档 [服务发布点解绑实例请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleUnbindInstance.java](#)

🔗 新增服务发布点鉴权信息

函数声明

```
public void addAuth(String service, AddAuthRequest body, String clientToken) {  
}
```

参数含义

请参考OpenAPI文档 [新增服务发布点鉴权信息请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleAddAuth.java](#)

🔗 修改服务发布点鉴权信息

函数声明

```
public void editAuth(String service, EditAuthRequest body, String clientToken) {  
}
```

参数含义

请参考OpenAPI文档 [修改服务发布点鉴权信息请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleEditAuth.java](#)

🔗 删除服务发布点鉴权信息

函数声明

```
public void removeAuth(String service, RemoveAuthRequest body, String clientToken) {  
}
```

参数含义

请参考OpenAPI文档 [删除服务发布点鉴权信息请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleRemoveAuth.java](#)

查询服务发布点列表

函数声明

```
public ListUserServiceResponse listUserService(String marker, Integer maxKeys) {  
}
```

参数含义

请参考OpenAPI文档 [查询服务发布点列表请求参数](#)

返回值

- 操作成功

```
{
  "metadata": "具体信息参考附录metadata部分",
  "services": [
    {
      "serviceId": "user-service-a223vc",
      "name": "name",
      "description": "",
      "serviceName": "serviceName",
      "bindType": "blb",
      "instanceId": "lb-d3e2cf75",
      "status": "inService",
      "service": "xxx.baidubce.com",
      "createTime": "2019-03-07T02:35:31Z",
      "endpointCount": 1,
      "endpointList": [
        {
          "endpointId": "endpoint-a223vc",
          "uid": "e8deba4d81d3435bb4591ce97288f296",
          "attachTime": "2019-03-07T02:35:31Z",
        }
      ],
      "authList": [
        {
          "uid": "*",
          "auth": "allow"
        }
      ]
    }
  ],
  "marker": "user-service-a032va",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 100
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleListUserService.java](#)

 [查询服务发布点详情](#)

函数声明

```
public GetUserServiceResponse getUserService(String service) {
}
```

参数含义

请参考OpenAPI文档 [查询服务发布点详情请求参数](#)

返回值

- 操作成功

```
{
  "metadata":具体信息参考附录metadata部分,
  "serviceId": "user-service-a223vc",
  "name": "name",
  "description": "",
  "serviceName": "serviceName",
  "bindType": "blb",
  "instanceId": "lb-d3e2cf75",
  "status": "inService",
  "service": "xxx.baidubce.com",
  "createTime": "2019-03-07T02:35:31Z",
  "endpointCount": 1,
  "endpointList": [
    {
      "endpointId": "endpoint-a223vc",
      "uid": "e8deba4d81d3435bb4591ce97288f296",
      "attachTime": "2019-03-07T02:35:31Z",
    }
  ],
  "authList": [
    {
      "uid": "*",
      "auth": "allow"
    }
  ]
}
```

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleGetUserService.java](#)

🔗 删除服务发布点

函数声明

```
public void deleteUserService(String service, String clientToken) {
}
```

参数含义

请参考OpenAPI文档 [删除服务发布点请求参数](#)

返回值

- 操作成功

无返回值

- 操作失败

操作失败返回异常列表参考[负载均衡BLB异常列表](#)

代码示例

具体代码示例参考：[ExampleDeleteUserService.java](#)

🔗 附录

公共返回信息metadata格式

```
{
  "bceRequestId": "f5f0821d-45fe-439b-bbf4-fc48b639f84a",
  "contentLength": 28,
  "contentType": "application/json;charset=UTF-8",
  "date": "1701917695000",
  "server": "openresty/1.15.8.1"
}
```

版本更新说明

- Java SDK开发包[2019-09-05] 版本号 0.10.75

Python-SDK

概述

本文档主要介绍BLB Python SDK的安装和使用。在使用本文档前，您需要先了解BLB的一些基本知识，并已经开通了BLB服务。若您还不了解BLB，可以参考[产品描述](#)和[快速入门](#)。

安装

运行环境

1. 运行环境 Python SDK工具包可在Python 2.7、python3.x环境下运行。
2. 鉴权和认证 要使用百度云产品，您需要拥有一个百度云账号和一个有效的 AK(Access Key ID)、SK(Secret Access Key)用来进行签名认证。

可以通过如下步骤获得并了解您的AK/SK信息：

1. [注册百度云账号](#)
2. [创建AK/SK](#)

🔗 安装SDK

🔗 下载和安装

方式一：通过pip安装

您可以通过pip安装的方式将百度智能云Python SDK安装到您的环境中。联网状态下，在命令行中执行如下命令：

```
pip install bce-python-sdk
```

即可将Python SDK安装到本地。

方式二：将源码包下载到本地后进行安装

1. 在[开发者资源中心](#)下载Python SDK压缩工具包。
2. 命令行移动到压缩包所在路径，执行如下命令（version替换为包名称中的版本号）：

```
pip install bce-python-sdk-version.zip
```

即可将Python SDK安装到本地。

您也可以解压压缩包后执行如下命令（version替换为包名称中的版本号）


```
cd bce-python-sdk-version

python setup.py install
```

来完成SDK的安装。

SDK目录结构

```
com.baidubce
├── auth                //BCE签名相关类
├── http                //BCE的Http通信相关类
├── model               //BCE公用model类
├── services
│   └── blb             //BLB服务相关类
│       ├── app_blb_client.py    //APPBLB客户端入口类
│       └── blb_client.py        //BLB客户端入口类
├── bce_base_client.py    //BCE客户端入口类的基类
├── bce_client_configuration.py //对BCE的HttpClient的配置
├── bce_response.py       //BCE客户端的请求类
├── exception.py          //BCE客户端的异常类
├── protocol.py           //网络协议定义
├── region.py             //区域处理模块
├── retry_policy.py        //BCE服务公共配置类
└── utils.py              //BCE公用工具类
```

🔗 卸载SDK

预期卸载 SDK 时，使用pip卸载“bce-python-sdk”即可。

初始化

确认Endpoint

在确认您使用SDK时配置的Endpoint时，可先阅读开发人员指南中关于[BLB访问域名](#)的部分，理解Endpoint相关的概念。 百度云目前开放了多区域支持，请参考[区域选择说明](#)。

目前支持“北京”、“广州”、“苏州”、“香港”、“武汉”、“保定”、“上海”、“新加坡”等区域。

服务域名的对应信息为：

访问区域	对应Endpoint
bj	blb.bj.baidubce.com
gz	blb.gz.baidubce.com
su	blb.su.baidubce.com
hkg	blb.hkg.baidubce.com
fwh	blb.fwh.baidubce.com
bd	blb.bd.baidubce.com
fsh	blb.fsh.baidubce.com
sin	blb.sin.baidubce.com

🔗 新建BlbClient

使用AK/SK新建BlbClient

通过AK/SK方式访问Blb，用户可以参考如下代码新建一个BlbClient：

```
from baidubce import bce_client_configuration
from baidubce.services.blb import blb_client
from baidubce.auth import bce_credentials

class Sample():
    def setUp(self):
        AK = 'your-access-key-id'          # 用户的Access Key ID
        SK = 'your-secret-access-key'       # 用户的Secret Access Key

        # 初始化一个BlbClient
        config = BceClientConfiguration(credentials=BceCredentials(AK, SK))
        client = blb_client.BlbClient(config)
```

在上面代码中，AK对应控制台中的“Access Key ID”，SK对应控制台中的“Access Key Secret”，获取方式请参考《[操作指南 管理ACCESSKEY](#)》。

上面的方式使用默认域名作为BLB的服务地址，如果用户需要自己指定域名，可以通过传入ENDPOINT参数来指定。

```
AK = 'your-access-key-id'          # 用户的Access Key ID
SK = 'your-secret-access-key'       # 用户的Secret Access Key
HOST = 'domain-name'               # 用户自己指定的域名，域名选择可参考下方"Region列表"

# 初始化一个BlbClient
config = BceClientConfiguration(credentials=BceCredentials(AK, SK), endpoint=HOST)
client = blb_client.BlbClient(config)
```

注意：ENDPOINT参数只能用指定的包含区域的域名来进行定义，不指定时默认为北京区域http://blb.bj.baidubce.com。

普通型blb实例

普通型blb实例提供了负载均衡常规功能组件，能够满足用户对负载均衡的通用需求，具备轻量化、配置简单、价格便宜等特点。

self即blb_client；具体各个参数含义请参考blb官方文档，api参考下的[普通型blb接口](#)

创建blb实例

创建一个LoadBalancer，返回分配的服务地址及实例ID。付费方式为后付费。需要实名认证，若未通过实名认证可以前往百度智能云官网控制台中的安全认证下的实名认证中进行认证。

使用以下代码可以创建blb实例：

函数声明

```
def create_loadbalancer(self, vpc_id, subnet_id, name=None,
                        desc=None, ip_version=None, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateLoadBalancer创建例](#)

返回值

操作成功：

```
{
  blb_id:'lb-7587472b',
  name:'BLB-beijing-c5335c6f',
  desc:'',
  address:'192.168.0.7'
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_loadbalancer.py](#)

🔗 修改blb实例

更新指定的单个LoadBalancer配置信息，包括LoadBalancer实例的名称和描述。

使用以下代码可以修改blb实例：

函数声明

```
def update_loadbalancer(self, blb_id, name=None, desc=None,
                        client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[UpdateLoadBalancer更新实例](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_loadbalancer.py](#)

🔗 修改blb实例ACL

更新指定的单个LoadBalancer 支持ACL特性。

使用以下代码可以修改blb实例支持ACL：

函数声明

```
def update_loadbalancer_acl(self, blb_id, client_token=None, support_acl=None, config=None)
```

参数含义

请参考OpenAPI文档：[UpdateLoadBalancerAcl更新实例ACL功能](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_loadbalancer_acl.py](#)

🔗 查询blb实例列表

查询用户账户下所有LoadBalancer的信息。支持按LoadBalancer的id、name、address进行查询，匹配规则支持部分包含（不支持正则）。支持按bccld进行查询，目前只支持单个bccld查询，并且仅支持精确查询。查询结果为包含该后端服务器的负载均衡实例。返回结果是多重查询条件的交集。如果BLB实例开启了公网访问，返回结果将会带上公网IP。结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询blb列表：

函数声明

```
def describe_loadbalancers(self, address=None, name=None, blb_id=None,
                             bcc_id=None, marker=None, max_keys=None, ip_version=None,
                             config=None)
```

参数含义

请参考OpenAPI文档：[DescribeLoadBalancers查询已有的BLB实例](#)

返回值

操作成功：

```
blb_list:[{
  tags:None,
  blb_id:'lb-70c35170',
  status:'available',
  name:'testblb',
  desc:'justForTest',
  address:'192.168.0.3',
  vpc_id:'vpc-sr84haq14tcb',
  subnet_id:'sbn-kg7mj5wt4yw9',
  allow_delete:True,
  payment_timing:'Postpaid'
},
marker:u'blb-05FAOCCE',
next_marker:None,
is_truncated:False,
max_keys:1000
]
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_loadbalancers.py](#)

🔗 查询blb实例详情

按id查询用户账户下某个LoadBalancer的详细信息。包含LoadBalancer所有的监听器端口信息

使用以下代码可以查询blb详情：

函数声明

```
def describe_loadbalancer_detail(self, blb_id, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeLoadBalancerDetail](#)查询BLB实例详情

返回值

操作成功：

```
{
  tags:None,
  blb_id:'lb-70c35170',
  status:'available',
  name:'testblb',
  desc:'justForTest',
  address:'192.168.0.3',
  cidr:'192.168.0.0/16',
  vpc_name:'test',
  create_time:'2023-12-11T12:32:09Z',
  ipv6:'',
  allow_delete:True,
  listener:[],
  support_acl:False,
  payment_timing:'Postpaid',
  vpc_id:'vpc-sr84haq14tcb'
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_loadbalancer_detail.py](#)

🔗 删除blb实例

释放指定LoadBalancer，被释放的LoadBalancer无法找回

使用以下代码可以删除blb实例：

函数声明

```
def delete_loadbalancer(self, blb_id, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[DeleteLoadBalancer](#)释放BLB实例

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_loadbalancer.py](#)

🔗 创建TCP监听器

在指定LoadBalancer下，创建一个基于TCP协议的监听器，监听一个前端端口，将发往该端口的所有TCP流量，转发到后端服务器监听的后端端口上。TCP监听器只支持TCP协议的健康检查，检查端口和后端服务器监听端口一致。

使用以下代码可以创建TCP监听器：

函数声明

```
def create_tcp_listener(self, blb_id, listener_port,
                        backend_port, scheduler,
                        health_check_timeout_in_second=None,
                        health_check_interval=None,
                        unhealthy_threshold=None,
                        healthy_threshold=None,
                        client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateTCPLListener创建TCP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_tcp_listener.py](#)

🔗 创建UDP监听器

在指定LoadBalancer下，创建一个基于UDP协议的监听器，监听一个前端端口，将发往该端口的所有UDP流量，转发到后端服务器监听的后端端口上。UDP监听器支持UDP、ICMP协议的健康检查，检查端口和后端服务器监听端口一致。

使用以下代码可以创建UDP监听器：

函数声明

```
def create_udp_listener(self, blb_id, listener_port, backend_port,
                        scheduler, health_check_string,
                        health_check_timeout_in_second=None,
                        health_check_interval=None,
                        unhealthy_threshold=None,
                        healthy_threshold=None,
                        client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateUDPListener创建UDP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_udp_listener.py](#)

🔗 创建HTTP监听器

在指定LoadBalancer下，创建一个基于HTTP协议的监听器，监听一个前端端口，将发往该端口的所有HTTP请求，转发到后端服务器监听的后端端口上。

使用以下代码可以创建HTTP监听器：

函数声明

```
def create_http_listener(self, blb_id, listener_port,
                        backend_port, scheduler,
                        keep_session=None, keep_session_type=None,
                        keep_session_duration=None,
                        keep_session_cookie_name=None,
                        x_forward_for=None,
                        health_check_type=None, health_check_port=None,
                        health_check_uri=None,
                        health_check_timeout_in_second=None,
                        health_check_interval=None,
                        unhealthy_threshold=None,
                        healthy_threshold=None,
                        health_check_normal_status=None,
                        server_timeout=None, redirect_port=None,
                        client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateHTTPListener创建HTTP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_http_listener.py](#)

🔗 创建HTTPS监听器

在指定LoadBalancer下，创建一个基于HTTPS协议的监听器，监听一个前端端口，将发往该端口的所有HTTPS请求，先通过SSL卸载转换为HTTP请求后，再转发到后端服务器监听的后端端口上。

使用以下代码可以创建HTTPS监听器：

函数声明

```
def create_https_listener(self, blb_id, listener_port, backend_port,
                          scheduler, cert_ids, keep_session=None,
                          keep_session_type=None,
                          keep_session_duration=None,
                          keep_session_cookie_name=None,
                          x_forward_for=None, health_check_type=None,
                          health_check_port=None, health_check_uri=None,
                          health_check_timeout_in_second=None,
                          health_check_interval=None,
                          unhealth_threshold=None, health_threshold=None,
                          health_check_normal_status=None,
                          server_timeout=None, ie6_compatible=None,
                          encryption_type=None, encryption_protocols=None,
                          dual_auth=None, client_certIds=None,
                          client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateHTTPSListener创建HTTPS监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_https_listener.py](#)

🔗 创建SSL监听器

在指定LoadBalancer下，创建一个基于SSL协议的监听器，监听一个前端端口，将发往该端口的所有SSL流量，通过SSL卸载转换为TCP请求后转发到后端服务器监听的后端端口上。SSL监听器只支持TCP协议的健康检查，检查端口和后端服务器监听端口一致。

使用以下代码可以创建SSL监听器：

函数声明

```
def create_ssl_listener(self, blb_id, listener_port, backend_port,
                        scheduler, cert_ids,
                        health_check_timeout_in_second=None,
                        health_check_interval=None,
                        unhealth_threshold=None, health_threshold=None,
                        ie6_compatible=None, encryption_type=None,
                        encryption_protocols=None,
                        dual_auth=None, client_certIds=None,
                        client_token=None, config=None)
```


参数含义

请参考OpenAPI文档：[CreateSSLListener创建SSL监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_ssl_listener.py](#)

更新TCP监听器

在指定LoadBalancer下，创建一个基于SSL协议的监听器，监听一个前端端口，将发往该端口的所有SSL流量，通过SSL卸载转换为TCP请求后转发到后端服务器监听的后端端口上。SSL监听器只支持TCP协议的健康检查，检查端口和后端服务器监听端口一致。

使用以下代码可以更新TCP监听器：

函数声明

```
def update_tcp_listener(self, blb_id, listener_port,
                        backend_port=None, scheduler=None,
                        health_check_timeout_in_second=None,
                        health_check_interval=None,
                        unhealth_threshold=None,
                        health_threshold=None,
                        config=None)
```

参数含义

请参考OpenAPI文档：[UpdateTCPListener更新TCP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_tcp_listener.py](#)

更新UDP监听器

更新指定LoadBalancer下的UDP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新UDP监听器：

函数声明

```
def update_udp_listener(self, blb_id, listener_port, backend_port=None,
                        scheduler=None, health_check_string=None,
                        health_check_timeout_in_second=None,
                        health_check_interval=None,
                        unhealth_threshold=None,
                        health_threshold=None,
                        config=None)
```

参数含义

请参考OpenAPI文档：[UpdateUDPListener更新UDP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_udp_listener.py](#)

更新HTTP监听器

更新指定LoadBalancer下的HTTP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新HTTP监听器：

函数声明

```
def update_http_listener(self, blb_id, listener_port, backend_port=None,
                        scheduler=None, keep_session=None,
                        keep_session_type=None,
                        keep_session_duration=None,
                        keep_session_cookie_name=None,
                        x_forward_for=None,
                        health_check_type=None, health_check_port=None,
                        health_check_uri=None,
                        health_check_timeout_in_second=None,
                        health_check_interval=None,
                        unhealth_threshold=None, health_threshold=None,
                        health_check_normal_status=None,
                        server_timeout=None,
                        redirect_port=None, config=None)
```

参数含义

请参考OpenAPI文档：[UpdateHTTPListener更新HTTP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_http_listener.py](#)

更新HTTPS监听器

更新指定LoadBalancer下的HTTPS监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新HTTPS监听器：

函数声明

```
def update_https_listener(self, blb_id, listener_port,
                           backend_port=None,
                           scheduler=None, keep_session=None,
                           keep_session_type=None,
                           keep_session_duration=None,
                           keep_session_cookie_name=None,
                           x_forward_for=None, health_check_type=None,
                           health_check_port=None, health_check_uri=None,
                           health_check_timeout_in_second=None,
                           health_check_interval=None,
                           unhealth_threshold=None, health_threshold=None,
                           health_check_normal_status=None,
                           server_timeout=None,
                           cert_ids=None, ie6_compatible=None,
                           config=None)
```

参数含义

请参考OpenAPI文档：[UpdateHTTPSListener更新HTTPS监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_https_listener.py](#)

更新SSL监听器

更新指定LoadBalancer下的SSL监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新SSL监听器：

函数声明

```
def update_ssl_listener(self, blb_id, listener_port,
                        backend_port=None, scheduler=None,
                        health_check_timeout_in_second=None,
                        health_check_interval=None,
                        unhealth_threshold=None,
                        health_threshold=None, cert_ids=None,
                        ie6_compatible=None,
                        encryption_type=None,
                        encryption_protocols=None,
                        dual_auth=None, client_certids=None,
                        config=None)
```

参数含义

请参考OpenAPI文档：[UpdateSSLListener更新SSL监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_ssl_listener.py](#)

查询TCP监听器

- 查询指定LoadBalancer下所有TCP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询TCP监听器：

函数声明

```
def describe_tcp_listener(self, blb_id, listener_port=None,
                           marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeTCPListeners查询TCP监听器](#)

返回值

操作成功：

```
{
  "listener_list": [
    {
      "listener_port":80,
      "backend_port":80,
      "backend_port_type":u'TCP',
      "scheduler":u'RoundRobin',
      "health_check_timeout_in_second":3,
      "health_check_interval":3,
      "unhealthy_threshold":3,
      "healthy_threshold":3,
      "get_blb_ip":False,
      "health_check_valid":1,
      "tcp_session_timeout":900
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例 具体代码示例参考：[example_describe_tcp_listener.py](#)

查询UDP监听器

- 查询指定LoadBalancer下所有UDP监听器的信息
- 支持按监听器端口进行匹配查询
- 结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询UDP监听器：

函数声明

```
def describe_udp_listener(self, blb_id, listener_port=None, marker=None,
                           max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeUDPListeners查询UDP监听器](#)

返回值

操作成功：

```
{
  "listener_list": [
    {
      "listener_port":53,
      "backend_port":53,
      "backend_port_type":u'UDP',
      "scheduler":u'RoundRobin',
      "health_check_timeout_in_second":3,
      "health_check_interval":3,
      "unhealthy_threshold":3,
      "healthy_threshold":3,
      "get_blb_ip":False,
      "health_check_valid":1,
      "health_check_string":u'test',
      "udp_session_timeout":30,
      "health_check_type":u'UDP'
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_udp_listener.py](#)

查询HTTP监听器

查询指定LoadBalancer下所有HTTP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询HTTP监听器

函数声明

```
def describe_http_listener(self, blb_id, listener_port=None,
                           marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeHTTPListeners查询HTTP监听器](#)

返回值

操作成功：

```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 90,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_2xx|http_3xx",
      "serverTimeout": 30,
      "redirectPort": 80
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 95,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_1xx|http_2xx",
      "serverTimeout": 1800,
      "redirectPort": 80
    }
  ],
  "marker": "listener-005A",
  "nextMarker": "listener-0322",
  "isTruncated": true,
  "maxKeys": 2
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_http_listener.py](#)

查询HTTPS监听器

查询指定LoadBalancer下所有HTTPS监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询HTTPS监听器

函数声明

```
def describe_https_listener(self, blb_id, listener_port=None,
                             marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeHTTPSListeners](#)查询HTTPS监听器

返回值

操作成功：


```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 90,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_2xx|http_3xx",
      "serverTimeout": 30,
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "healthCheckTimeoutInSecond": 4,
      "healthCheckInterval": 5,
      "unhealthyThreshold": 2,
      "healthyThreshold": 3,
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionDuration": null,
      "keepSessionCookieName": null,
      "xForwardFor": false,
      "healthCheckType": "HTTP",
      "healthCheckPort": 95,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_1xx|http_2xx",
      "serverTimeout": 1800,
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-005A",
  "nextMarker": "listener-0322",
  "isTruncated": true,
  "maxKeys": 2
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_https_listener.py](#)

查询SSL监听器

查询指定LoadBalancer下所有SSL监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询SSL监听器

函数声明

```
def describe_ssl_listener(self, blb_id, listener_port=None,
                           marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeSSLListeners查询SSL监听器](#)

返回值

操作成功：

```
{
  "listenerList": [
    {
      "listenerPort": 80,
      "backendPort": 80,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3
    },
    {
      "listenerPort": 88,
      "backendPort": 88,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 2,
      "healthCheckInterval": 4,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_ssl_listener.py](#)

🔗 查询所有监听

查询指定LoadBalancer下所有监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询所有监听

函数声明

```
def describe_all_listeners(self, blb_id, listener_port=None,
                           marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeAllListeners查询所有监听](#)

返回值

操作成功：

```
{
  "nextMarker": "",
  "marker": "",
  "maxKeys": 1,
  "isTruncated": true,
  "listenerList": [
    {
      "listenerPort": 53,
      "backendPort": 53,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3,
      "getBlbIp": false,
      "listenerType": "UDP",
      "udpSessionTimeout": 90,
      "healthCheckString":
"\00\01\01\00\00\01\00\00\00\00\00\00\00\05baidu\03com\00\00\01\00\01",
      "healthCheckType": "UDP"
    },
    {
      "listenerPort": 82,
      "backendPort": 80,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3,
      "getBlbIp": false,
      "listenerType": "TCP",
      "tcpSessionTimeout": 900
    },
    {
      "listenerPort": 87,
      "backendPort": 80,
      "scheduler": "RoundRobin",
      "healthCheckTimeoutInSecond": 3,
      "healthCheckInterval": 3,
      "unhealthyThreshold": 3,
      "healthyThreshold": 3,
      "getBlbIp": false,
      "listenerType": "HTTP",
      "keepSession": false,
      "xForwardFor": true,
      "healthCheckType": "HTTP",
      "healthCheckPort": 80,
      "healthCheckURI": "/",
      "healthCheckNormalStatus": "http_2xx|http_3xx",
      "healthCheckHost": "",

```

```

"serverTimeout": 30,
"redirectPort": 0
},
{
  "listenerPort": 443,
  "backendPort": 80,
  "scheduler": "RoundRobin",
  "healthCheckTimeoutInSecond": 3,
  "healthCheckInterval": 3,
  "unhealthyThreshold": 3,
  "healthyThreshold": 3,
  "getBlbIp": false,
  "listenerType": "HTTPS",
  "keepSession": false,
  "xForwardFor": true,
  "healthCheckType": "HTTP",
  "healthCheckPort": 80,
  "healthCheckURI": "/",
  "healthCheckNormalStatus": "http_2xx|http_3xx",
  "healthCheckHost": "",
  "serverTimeout": 30,
  "certIds": [
    "cert-gs8bktrm7drp"
  ],
  "dualAuth": false,
  "encryptionType": "tls_cipher_policy_1_2_secure",
  "encryptionProtocols": [
    "tlsv12"
  ],
  "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA"
},
{
  "listenerPort": 446,
  "backendPort": 80,
  "scheduler": "RoundRobin",
  "healthCheckTimeoutInSecond": 3,
  "healthCheckInterval": 3,
  "unhealthyThreshold": 3,
  "healthyThreshold": 3,
  "getBlbIp": false,
  "listenerType": "SSL",
  "certIds": [
    "cert-gs8bktrm7drp"
  ],
  "dualAuth": true,
  "clientCertIds": [
    "cert-0x037gwe4fmg"
  ],
  "encryptionType": "tls_cipher_policy_default",
  "encryptionProtocols": [
    "tlsv10",
    "tlsv11",
    "tlsv12"
  ],
  "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA"
},

```

```
}  
]  
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_all_listeners.py](#)

🔗 释放监听器

释放指定LoadBalancer下的监听器，监听器通过监听端口来指定，支持批量释放。

使用以下代码可以释放监听器

函数声明

```
def delete_listeners(self, blb_id, portList, client_token=None, config=None)
```

参数含义 请参考OpenAPI文档：[DeleteListeners释放监听器](#)

返回值 操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_listeners.py](#)

🔗 新增后端服务器实例

为指定LoadBalancer添加后端服务器，支持批量添加。

使用以下代码可以新增后端服务器实例

函数声明

```
def add_backend_servers(self, blb_id, backend_server_list,  
                        client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[添加后端服务器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_add_backend_servers.py](#)

🔗 修改后端服务器实例

更新指定LoadBalancer下的后端服务器权重，支持批量修改。

使用以下代码可以修改后端服务器实例

函数声明

```
def update_backend_servers(self, blb_id, backend_server_list,
                           client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[更新后端服务器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_backend_servers.py](#)

🔗 查询后端服务器实例健康状态

查询指定LoadBalancer下的指定端口的后端服务器列表的健康状态。

使用以下代码可以查询后端服务器实例健康状态

函数声明

```
def describe_health_status(self, blb_id, listener_port,
                            marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[查询后端服务器健康状态](#)

返回值

操作成功：

```
{
  "backendServerList": [
    {
      "instanceId": "i-YfAib14A",
      "weight": 50,
      "status": "Alive"
    }
  ],
  "type": "TCP",
  "listenerPort": 88,
  "backendPort": 88,
  "marker": "rs-0A6BE9BB",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_health_status.py](#)

查询后端服务器实例

查询指定LoadBalancer下的后端服务器列表。

使用以下代码可以查询后端服务器实例

函数声明

```
def describe_backend_servers(self, blb_id, marker=None,
                             max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[查询后端服务器列表](#)

返回值

操作成功：

```
{
  "backendServerList": [
    {
      "instanceId": "i-YfAib14A",
      "weight": 50,
    }
  ],
  "marker": "rs-0A6BE9BB",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_backend_servers.py](#)

🔗 移除后端服务器实例

释放指定LoadBalancer下的后端服务器，后端服务器通过其标识符指定

使用以下代码可以移除后端服务器实例

函数声明

```
def remove_backend_servers(self, blb_id, backend_server_list,
                           client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[释放后端服务器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_remove_backend_servers.py](#)

🔗 绑定普通安全组

为指定LoadBalancer绑定普通安全组

使用以下代码可以绑定普通安全组

函数声明

```
def bind_security_groups(self, blb_id, enterprisesecuritygroupids,
                          client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[BindSecurityGroups绑定普通安全组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_bind_security_groups.py](#)

🔗 解绑普通安全组

为指定LoadBalancer解绑普通安全组

使用以下代码可以解绑普通安全组

函数声明

```
def unbind_security_groups(self, blb_id, securitygroupids,
                           client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[UnbindSecurityGroups解绑普通安全组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_unbind_security_groups.py](#)

🔗 查询BLB实例普通安全组列表

为指定LoadBalancer查询普通安全组

使用以下代码可以查询普通安全组

函数声明

```
def describe_security_groups(self, blb_id,
                             client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeSecurityGroups查询BLB实例普通安全组列表](#)

返回值

操作成功：

```
{
  "blbSecurityGroups": [
    {
      "securityGroupDesc": "",
      "securityGroupId": "g-4NxWoxeq",
      "securityGroupName": "securitygroup-name",
      "vpcName": "vpc-name",
      "securityGroupRules": [
        {
          "destGroupId": "",
          "destIp": "all",
          "direction": "egress",
          "ethertype": "IPv4",
          "portRange": "1-65535",
          "protocol": "all",
          "securityGroupRuleId": "r-gkv8yupumvx2"
        }
      ]
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_security_groups.py](#)

🔗 绑定企业安全组

为指定LoadBalancer绑定企业安全组

使用以下代码可以绑定企业安全组

函数声明

```
def bind_enterprise_security_groups(self, blb_id, enterprisesecuritygroupids,
                                   client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[BindEnterpriseSecurityGroups绑定企业安全组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_bind_enterprise_security_groups.py.py](#)

🔗 解绑企业安全组

为指定LoadBalancer解绑企业安全组

使用以下代码可以解绑企业安全组

函数声明

```
def unbind_enterprise_security_groups(self, blb_id, enterprisesecuritygroupids,
                                     client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[UnbindEnterpriseSecurityGroups解绑企业安全组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_unbind_enterprise_security_groups.py](#)

查询BLB实例企业安全组列表

为指定LoadBalancer查询企业安全组

使用以下代码可以查询企业安全组

函数声明

```
def describe_enterprise_security_groups(self, blb_id,
                                       client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeEnterpriseSecurityGroups查询BLB实例企业安全组列表](#)

返回值

操作成功：

```
{
  "enterpriseSecurityGroups": [{
    "enterpriseSecurityGroupId": "esg-g8z4qfj0m0bu",
    "enterpriseSecurityGroupName": "test0627",
    "enterpriseSecurityGroupDesc": "",
    "enterpriseSecurityGroupRules": [{
      "remark": "全部协议",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv6",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu",
      "protocol": "all"
    },
    {
      "remark": "",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9je09z6bd9wv",
      "protocol": "tcp"
    },
    {
      "remark": "",
      "direction": "ingress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "2",
      "sourceIp": "all",
      "destIp": "",
      "enterpriseSecurityGroupRuleId": "esgr-jcej4juqbh49",
      "protocol": "tcp"
    }
  ]
}]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_enterprise_security_groups.py](#)

应用型blb实例

应用型blb实例针对用户复杂应用部署架构，特别是大型网站架构。使用基于策略的网络管理框架构建，实现业务驱动流量负载均衡。

self即app_blb_client；具体各个参数含义请参考blb官方文档，api参考下的[应用型blb接口](#)

🔗 创建应用型blb实例

创建一个LoadBalancer，返回分配的服务地址及实例ID。付费方式为后付费。需要实名认证，若未通过实名认证可以前往百度智能云官网控制台中的安全认证下的实名认证中进行认证。

使用以下代码可以创建应用型blb实例：

函数声明

```
def create_app_loadbalancer(self, vpc_id, subnet_id, name=None, desc=None, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateLoadBalancer创建实例](#)

返回值

操作成功：

```
{
  blb_id:'lb-1a22b118',
  name:'BLB-beijing-a60ce0c4',
  desc:'',
  address:'192.168.0.7'
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_loadbalancer.py](#)

🔗 更新应用型blb实例

更新一个LoadBalancer的配置信息。

使用以下代码可以更新应用型blb实例：

函数声明

```
def update_app_loadbalancer(self, blb_id, name=None, desc=None, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[UpdateLoadBalancer更新实例](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_loadbalancer.py](#)

🔗 查询应用型blb实例列表

查询用户账户下所有LoadBalancer的信息。支持按LoadBalancer的id、name、address进行查询，匹配规则支持部分包含（不支持正则）。支持查找绑定指定BCC的LoadBalancer，通过bccId参数指定。返回结果是多重查询条件的交集。结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询应用型blb实例列表：

函数声明

```
def describe_app_loadbalancers(self, address=None, name=None, blb_id=None, bcc_id=None, marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeLoadBalancers查询已有的BLB实例](#)

返回值

操作成功：

```
{
  blb_list:[{
    tags:None,
    blb_id:'lb-cf2dc4e7',
    status:'available',
    name:'testappblb',
    desc:'appjustForTest',
    address:'192.168.0.4',
    vpc_id:'vpc-sr84haq14tcb',
    subnet_id:'sbn-kg7mj5wt4yw9',
    allow_delete:True,
    payment_timing:'Postpaid'},
  ]
  marker:'blb-05FAOCED',
  next_marker:None,
  is_truncated:False,
  max_keys:1000
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_loadbalancers.py](#)

查询应用型blb实例详情

按id查询用户账户下所有LoadBalancer的详细信息。包含LoadBalancer所有的监听器端口信息

使用以下代码可以查询应用型blb实例详情：

函数声明

```
def describe_app_loadbalancer_detail(self, blb_id, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeLoadBalancerDetail查询BLB实例详情](#)

返回值

操作成功：

```
{
  tags:None,
  blb_id:'lb-cf2dc4e7',
  status:'available',
  name:'testapblb',
  desc:'appjustForTest',
  address:'192.168.0.4',
  cidr:'192.168.0.0/16',
  vpc_name:'test',
  subnet_name:'test',
  subnet_cider:'192.168.0.0/24',
  create_time:'2023-12-11T14:14:36Z',
  ipv6:'',
  allow_delete:True,
  listener:[],
  support_acl:True,
  payment_timing:'Postpaid',
  vpc_id:'vpc-sr84haq14tcb'
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_loadbalancer_detail.py](#)

🔗 删除应用型blb实例

释放指定LoadBalancer，被释放的LoadBalancer无法找回

使用以下代码可以删除应用型blb实例：

函数声明

```
def delete_app_loadbalancer(self, blb_id, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[DeleteLoadBalancer释放BLB实例](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_app_loadbalancer.py](#)

🔗 创建TCP监听器

在指定LoadBalancer下，创建一个基于TCP协议的应用型blb监听器，监听一个前端端口，将发往该端口的所有TCP流量，根据策略进行转发。

使用以下代码可以创建TCP监听器

函数声明

```
def create_app_tcp_listener(self, blb_id, listener_port,
                           scheduler, client_token=None,
                           config=None)
```

参数含义

请参考OpenAPI文档：[CreateAppTCPListener创建TCP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_tcp_listener.py](#)

🔗 创建UDP监听器

在指定LoadBalancer下，创建一个基于UDP协议的应用型监听器，监听一个前端端口，将发往该端口的所有UDP流量，根据策略进行转发。

使用以下代码可以创建UDP监听器

函数声明

```
def create_app_udp_listener(self, blb_id, listener_port,
                           scheduler, client_token=None,
                           config=None)
```

参数含义

请参考OpenAPI文档：[CreateAppUDPLListener创建UDP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_udp_listener.py](#)

🔗 创建HTTP监听器

在指定LoadBalancer下，创建一个基于HTTP协议的应用型监听器，监听一个前端端口，将发往该端口的所有HTTP请求，根据策略转发到后端服务器监听的后端端口上。

使用以下代码可以创建HTTP监听器

函数声明

```
def create_app_http_listener(self, blb_id, listener_port,
                             scheduler, keep_session=None,
                             keep_session_type=None,
                             keep_session_timeout=None,
                             keep_session_cookie_name=None,
                             x_forwarded_for=None,
                             server_timeout=None,
                             redirect_port=None,
                             client_token=None,
                             config=None)
```

参数含义

请参考OpenAPI文档：[CreateAppHTTPListener创建HTTP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_http_listener.py](#)

🔗 创建HTTPS监听器

更新指定LoadBalancer下的HTTPS监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以创建HTTPS监听器

函数声明

```
def create_app_https_listener(self, blb_id, listener_port,
                              scheduler, cert_ids,
                              keep_session=None,
                              keep_session_type=None,
                              keep_session_timeout=None,
                              keep_session_cookie_name=None,
                              x_forwarded_for=None, server_timeout=None,
                              ie6_compatible=None, encryption_type=None,
                              encryption_protocols=None,
                              dual_auth=None, client_certIds=None,
                              client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateAppHTTPSListener创建HTTPS监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_https_listener.py](#)

🔗 创建SSL监听器

在指定LoadBalancer下，创建一个基于SSL协议的应用型监听器，监听一个前端端口，将发往该端口的所有SSL流量，根据策略进行转发。

使用以下代码可以创建SSL监听器

函数声明

```
def create_app_ssl_listener(self, blb_id, listener_port,
                           scheduler, cert_ids,
                           ie6_compatible=None,
                           encryption_type=None,
                           encryption_protocols=None,
                           dual_auth=None, client_certIds=None,
                           client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateAppSSLListener创建SSL监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_ssl_listener.py](#)

🔗 更新TCP监听器

更新指定LoadBalancer下的TCP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新TCP监听器

函数声明

```
def update_app_tcp_listener(self, blb_id, listener_port,
                           scheduler=None,
                           client_token=None,
                           config=None)
```

参数含义

请参考OpenAPI文档：[UpdateAppTCPListener更新TCP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_tcp_listener.py](#)

更新UDP监听器

更新指定LoadBalancer下的UDP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新UDP监听器

函数声明

```
def update_app_udp_listener(self, blb_id, listener_port,
                           scheduler=None, client_token=None,
                           config=None)
```

参数含义

请参考OpenAPI文档：[UpdateAppUDPLListener更新UDP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_udp_listener.py](#)

更新HTTP监听器

更新指定LoadBalancer下的HTTP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新HTTP监听器

函数声明

```
def update_app_http_listener(self, blb_id, listener_port,
                             scheduler=None, keep_session=None,
                             keep_session_type=None,
                             keep_session_timeout=None,
                             keep_session_cookie_name=None,
                             x_forwarded_for=None,
                             server_timeout=None,
                             redirect_port=None,
                             client_token=None,
                             config=None)
```

参数含义

请参考OpenAPI文档：[UpdateAppHTTPListener更新HTTP监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_http_listener.py](#)

🔗 更新HTTPS监听器

更新指定LoadBalancer下的HTTPS监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新HTTPS监听器

函数声明

```
def update_app_https_listener(self, blb_id, listener_port,
                              scheduler=None,
                              keep_session=None,
                              keep_session_type=None,
                              keep_session_timeout=None,
                              keep_session_cookie_name=None,
                              x_forwarded_for=None,
                              server_timeout=None,
                              cert_ids=None,
                              ie6_compatible=None,
                              encryption_type=None,
                              encryption_protocols=None,
                              dual_auth=None,
                              client_certIds=None,
                              client_token=None,
                              config=None)
```

参数含义

请参考OpenAPI文档：[UpdateAppHTTPSListener更新HTTPS监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_https_listener.py](#)

更新应用型SSL监听器

更新指定LoadBalancer下的SSL监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新应用型SSL监听器

函数声明

```
def update_app_ssl_listener(self, blb_id, listener_port,
                             scheduler=None,
                             cert_ids=None,
                             ie6_compatible=None,
                             encryption_type=None,
                             encryption_protocols=None,
                             dual_auth=None,
                             client_certIds=None,
                             client_token=None,
                             config=None)
```

参数含义

请参考OpenAPI文档：[UpdateAppSSLListener更新应用型SSL监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_ssl_listener.py](#)

查询应用型TCP监听器

查询指定LoadBalancer下所有TCP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过max_keys参数指定。

使用以下代码可以查询应用型TCP监听器

函数声明

```
def describe_app_tcp_listener(self, blb_id, listener_port=None,
                             marker=None, max_keys=None,
                             config=None)
```

参数含义

请参考OpenAPI文档：[DescribeAppTCPListeners查询应用型TCP监听器](#)

返回值

操作成功：

```
{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin"
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin"
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_tcp_listener.py](#)

查询UDP监听器

查询指定LoadBalancer下所有UDP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过max_keys参数指定。

使用以下代码可以查询UDP监听器

函数声明

```
def describe_app_udp_listener(self, blb_id, listener_port=None,
                              marker=None, max_keys=None,
                              config=None)
```

参数含义

请参考OpenAPI文档：[DescribeAppUDPLListeners查询UDP监听器](#)

返回值

操作成功：

```
{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin",
      "udpSessionTimeout": 90
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin",
      "udpSessionTimeout": 90
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_udp_listener.py](#)

查询HTTP监听器

查询指定LoadBalancer下所有HTTP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过max_keys参数指定。

使用以下代码可以查询HTTP监听器

函数声明

```
def describe_app_http_listener(self, blb_id, listener_port=None,
                                marker=None, max_keys=None,
                                config=None)
```

参数含义

请参考OpenAPI文档：[DescribeAppHTTPListeners查询HTTP监听器](#)

返回值

操作成功：

```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 30,
      "redirectPort": 80
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 1800,
      "redirectPort": 80
    }
  ],
  "marker": "listener-005A",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_https_listener.py](#)

查询HTTPS监听器

查询指定LoadBalancer下所有HTTPS监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过max_keys参数指定。

使用以下代码可以查询HTTPS监听器

函数声明

```
def describe_app_https_listener(self, blb_id, listener_port=None,
                                marker=None, max_keys=None,
                                config=None)
```

参数含义

请参考OpenAPI文档：[DescribeAppHTTPSListeners查询HTTPS监听器](#)

返回值

操作成功：


```
{
  "listenerList": [
    {
      "listenerPort": 90,
      "backendPort": 90,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 30,
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 95,
      "backendPort": 95,
      "scheduler": "LeastConnection",
      "keepSession": false,
      "keepSessionType": null,
      "keepSessionTimeout": null,
      "keepSessionCookieName": null,
      "xForwardedFor": false,
      "serverTimeout": 1800,
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-005A",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_https_listener.py](#)

查询应用型SSL监听器

查询指定LoadBalancer下所有SSL监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过max_keys参数指定。

使用以下代码可以查询应用型SSL监听器

函数声明

```
def describe_app_ssl_listener(self, blb_id, listener_port=None,
                              marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeAppSSLListeners查询应用型SSL监听器](#)

返回值

操作成功：

```
{
  "listenerList": [
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin",
      "certIds": ["cert-zfj2ey2z4kmm"]
    },
    {
      "listenerPort": 88,
      "scheduler": "RoundRobin",
      "certIds": ["cert-zfj2ey2z4kmm"]
    }
  ],
  "marker": "listener-0050",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 2
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_ssl_listener.py](#)

查询所有监听

查询指定LoadBalancer下所有监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过max_keys参数指定。

使用以下代码可以查询所有监听

函数声明

```
def describe_app_all_listener(self, blb_id, listener_port=None,
                              marker=None, max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeAppAllListeners查询所有监听](#)

返回值

操作成功：

```
{
  "nextMarker": "",
  "marker": "",
  "maxKeys": 1,
  "isTruncated": true,
  "listenerList": [
    {
      "listenerPort": 53,
      "scheduler": "RoundRobin",
      "listenerType": "UDP"
    },
    {
      "listenerPort": 80,
      "scheduler": "RoundRobin"
```

```

        "scheduler": "RoundRobin",
        "listenerType": "HTTP",
        "keepSession": false,
        "xForwardFor": true,
        "xForwardedProto": false,
        "serverTimeout": 30,
        "redirectPort": 0
    },
    {
        "listenerPort": 81,
        "scheduler": "RoundRobin",
        "listenerType": "TCP",
        "tcpSessionTimeout": 900
    },
    {
        "listenerPort": 443,
        "scheduler": "RoundRobin",
        "listenerType": "SSL",
        "certIds": [
            "cert-gs8bktrm7drp"
        ],
        "dualAuth": false,
        "encryptionType": "tls_cipher_policy_ie",
        "encryptionProtocols": [
            "ssl3",
            "tlsv10",
            "tlsv11",
            "tlsv12"
        ],
        "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-ECDSA-RC4-SHA:ECDHE-ECDSA-DES-CBC3-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-RSA-RC4-SHA:ECDHE-RSA-DES-CBC3-SHA:RC4-SHA:AES128-GCM-SHA256:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES256-SHA:AES128-SHA256:DES-CBC3-SHA"
    },
    {
        "listenerPort": 444,
        "scheduler": "RoundRobin",
        "listenerType": "HTTPS",
        "keepSession": false,
        "xForwardFor": false,
        "xForwardedProto": false,
        "serverTimeout": 30,
        "certIds": [
            "cert-gs8bktrm7drp"
        ],
        "dualAuth": false,
        "encryptionType": "tls_cipher_policy_1_1",
        "encryptionProtocols": [
            "tlsv11",
            "tlsv12"
        ],
        "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA"
    }
    ]
}

```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_all_listener.py](#)

🔗 释放监听器

释放指定LoadBalancer下的监听器，监听器通过监听端口来指定，支持批量释放。

使用以下代码可以释放监听器

函数声明

```
def delete_app_listeners(self, blb_id, portList,
                          client_token=None,
                          config=None)
```

参数含义

请参考OpenAPI文档：[DeleteAppListeners释放监听器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_app_listener.py](#)

🔗 创建策略

在指定应用型BLB监听器端口下创建策略。

使用以下代码可以创建策略

函数声明

```
def create_policys(self, blb_id, listener_port,
                   app_policy_vos, client_token=None,
                   config=None)
```

参数含义

请参考OpenAPI文档：[CreatePolicys创建策略](#)

返回值

操作成功：

```
{
  "listenerPort": 98,
  "type": "HTTP",
  "appPolicyVos": [
    {
      "appServerGroupId": "sg-db3975dc",
      "backendPort": 83,
      "priority": 100,
      "desc": "aa",
      "ruleList": [
        {
          "key": "*",
          "value": "*"
        }
      ]
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_policys.py](#)

查询对应BLB端口下策略信息

查询指定LoadBalancer下所有服务器组的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过max_keys参数指定。

使用以下代码可以查询对应BLB端口下策略信息

函数声明

```
def describe_policys(self, blb_id, listener_port,
                      marker=None, max_keys=None,
                      config=None)
```

参数含义

请参考OpenAPI文档：[DescribePolicys查询对应BLB端口下策略信息](#)

返回值

操作成功：

```
{
  "marker": "rs-012310",
  "isTruncated": true,
  "maxKeys": 1000,
  "nextMarker": "rs-013214",
  "policyList": [
    {
      "id": "policy_id",
      "appServerGroupId": "sg_id",
      "appServerGroupName": "sg_name",
      "frontendPort": 80,
      "type": "TCP",
      "backendPort": 98,
      "portType": "TCP",
      "priority": 100,
      "desc": "aa",
      "groupType": "Server",
      "ruleList": [
        {
          "key": "host",
          "value": "*.baidu.com"
        }
      ]
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_policys.py](#)

批量删除策略

批量删除对应BLB端口下的策略。

使用以下代码可以批量删除策略

函数声明

```
def delete_policys(self, blb_id, listener_port,
                    policys_list,
                    client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[DeletePolicys批量删除策略](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_policys.py](#)

创建应用型blb服务器组

在指定应用型BLB下，创建一个服务器组，用来绑定后端服务器，以及为监听器开放相应的端口。

使用以下代码可以创建应用型blb服务器组

函数声明

```
def create_app_server_group(self, blb_id,
                             name=None,
                             desc=None,
                             backend_server_list=None,
                             client_token=None,
                             config=None)
```

参数含义

请参考OpenAPI文档：[CreateAppServerGroup创建应用型服务器组](#)

返回值

操作成功：

```
{
  "status": "available",
  "name": "exmaple",
  "id": "sg-b71fc32a",
  "desc": "example"
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_server_group.py](#)

更新应用型blb服务器组

更新指定LoadBalancer下监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新应用型blb服务器组

函数声明

```
def update_app_server_group(self, blb_id, sg_id,
                             name=None,
                             desc=None,
                             client_token=None,
                             config=None)
```

参数含义

请参考OpenAPI文档：[UpdateAppServerGroup更新服务器组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_server_group.py](#)

🔗 查询应用型blb服务器组

查询指定LoadBalancer下所有服务器组的信息，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询应用型blb服务器组

函数声明

```
def describe_app_server_group(self, blb_id,
                                name=None,
                                exactly_match=None,
                                marker=None,
                                max_keys=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeAppServerGroup查询服务器组](#)

返回值

操作成功：

```
{
  "max_keys":1000,
  "next_marker":None,
  "app_server_group_list":[
    {
      "status":"available",
      "desc":"example2",
      "id":"sg-0a950a17",
      "name":"exmaple2",
      "port_list":[

    ]
  }
],
  "marker":"60742",
  "is_truncated":false
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_server_group.py](#)

🔗 删除应用型blb服务器组

删除服务器组，通过服务器组id指定。

使用以下代码可以删除应用型blb服务器组

函数声明

```
def delete_app_server_group(self, blb_id, sg_id,
                             client_token=None,
                             config=None)
```

参数含义

请参考OpenAPI文档：[DeleteAppServerGroup删除服务器组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_app_server_group.py](#)

🔗 创建应用型blb服务器组端口

在指定应用型BLB下，创建一个服务器组后端端口，将发往该端口的所有流量按权重轮询分发到其绑定的对应服务器列表中的服务器。

使用以下代码可以创建应用型blb服务器组端口

函数声明

```
def create_app_server_group_port(self, blb_id, sg_id,
                                   port, protocol_type,
                                   health_check=None,
                                   health_check_port=None,
                                   health_check_urlpath=None,
                                   health_check_timeout_insecond=None,
                                   health_check_interval_insecond=None,
                                   health_check_down_retry=None,
                                   health_check_up_retry=None,
                                   health_check_normal_status=None,
                                   udp_health_check_string=None,
                                   client_token=None,
                                   config=None)
```

参数含义

请参考OpenAPI文档：[CreateAppServerGroupPort创建应用型服务器组端口](#)

返回值

操作成功：

```
{
  "status": "available",
  "id": "port-ba811b4f"
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_server_group_port.py](#)

更新应用型blb服务器组端口

根据id更新服务器组端口。

使用以下代码可以更新应用型blb服务器组端口

函数声明

```
def update_app_server_group_port(self, blb_id, sg_id, port_id,
    health_check=None,
    health_check_port=None,
    health_check_urlpath=None,
    health_check_timeout_insecond=None,
    health_check_interval_insecond=None,
    health_check_down_retry=None,
    health_check_up_retry=None,
    health_check_normal_status=None,
    udp_health_check_string=None,
    client_token=None,
    config=None)
```

参数含义

请参考OpenAPI文档：[UpdateAppServerGroupPort更新服务器组端口](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_server_group_port.py](#)

删除应用型blb服务器组端口

删除服务器组端口，通过服务器组id指定。

使用以下代码可以删除应用型blb服务器组端口

函数声明

```
def delete_app_server_group_port(self, blb_id, sg_id,
                                port_list,
                                client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[DeleteAppServerGroupPort删除服务器组端口](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_app_server_group_port.py](#)

添加应用型blb后端服务器

在指定应用型BLB和服务器组下绑定后端服务器RS。

使用以下代码可以添加应用型blb后端服务器

函数声明

```
def create_app_blb_rs(self, blb_id, sg_id,
                       backend_server_list,
                       client_token=None,
                       config=None)
```

参数含义

请参考OpenAPI文档：[添加应用型BLB后端RS](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_app_blb_rs.py](#)

更新应用型blb后端服务器

更新指定LoadBalancer下的TCP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新应用型blb后端服务器

函数声明

```
def update_app_blb_rs(self, blb_id, sg_id,
                      backend_server_list,
                      client_token=None,
                      config=None)
```

参数含义

请参考OpenAPI文档：[更新服务器组下挂载的RS权重](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_app_blb_rs.py](#)

🔗 删除应用型blb后端服务器

删除服务器组，通过服务器组id指定。

使用以下代码可以删除应用型blb后端服务器

函数声明

```
def delete_app_blb_rs(self, blb_id, sg_id,
                      backend_server_list,
                      client_token=None,
                      config=None)
```

参数含义

请参考OpenAPI文档：[删除应用型blb的后端服务器](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_app_blb_rs.py](#)

🔗 查询服务器组下的RS信息

查询指定LoadBalancer下的指定服务器组下的RS信息

使用以下代码可以查询服务器组下的RS信息

函数声明

```
def describe_app_blb_rs(self, blb_id, sg_id,
                        marker=None, max_keys=None,
                        config=None)
```

参数含义

请参考OpenAPI文档：[查询服务器组下的RS列表信息](#)

返回值

操作成功：

```
{
  "marker": "rs-012310",
  "isTruncated": false,
  "maxKeys": 1000,
  "nextMarker": "rs-013214",
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "privateIp": "192.168.0.1",
      "weight": "WEIGHT",
      "portList": [
        {
          "listenerPort": "LISTENER_PORT1",
          "backendPort": "BACKEND_PORT1",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "portId": "port_xxx",
          "policyId": "policy_xxx",
          "status": "Alive"
        }
      ]
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_blb_rs.py](#)

🔗 查询应用型blb挂载的后端服务器

查询服务器组下绑定的server。

使用以下代码可以查询应用型blb挂载的后端服务器

函数声明

```
def describe_rs_mount(self, blb_id, sg_id, config=None)
```

参数含义

请参考OpenAPI文档：[查询服务器组下绑定的server](#)

返回值

操作成功：

```
{
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "privateIp": "192.168.0.1",
      "weight": "WEIGHT",
      "portList": [
        {
          "listenerPort": "LISTENER_PORT1",
          "backendPort": "BACKEND_PORT1",
          "portType": "HTTP",
          "healthCheckPortType": "HTTP",
          "portId": "port_xxx",
          "policyId": "policy_xxx",
          "status": "Alive"
        }
      ]
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_rs_mount.py](#)

🔗 查询应用型blb未挂载的后端服务器

查询服务器组下未绑定的RS。

使用以下代码可以查询应用型blb未挂载的后端服务器

函数声明

```
def describe_rs_unmount(self, blb_id, sg_id, config=None)
```

参数含义

请参考OpenAPI文档：[查询服务器组下未绑定的RS](#)

返回值

操作成功：

```
{
  "backendServerList": [
    {
      "instanceId": "BCC_UUID",
      "privateIp": null,
      "weight": null,
      "portList": []
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_rs_unmount.py](#)

🔗 绑定普通安全组

为指定LoadBalancer绑定普通安全组

使用以下代码可以绑定普通安全组

函数声明

```
def bind_app_security_groups(self, blb_id, securitygroupids,
                             client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[BindSecurityGroups绑定普通安全组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_bind_app_security_groups.py](#)

🔗 解绑普通安全组

为指定LoadBalancer解绑普通安全组

使用以下代码可以解绑普通安全组

函数声明

```
def unbind_app_security_groups(self, blb_id, securitygroupids,
                               client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[UnbindSecurityGroups解绑普通安全组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_unbind_app_security_groups.py](#)

🔗 查询BLB实例普通安全组列表

为指定LoadBalancer查询普通安全组

使用以下代码可以查询普通安全组

函数声明

```
def describe_app_security_groups(self, blb_id,
                                client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeSecurityGroups查询BLB实例普通安全组列表](#)

返回值

操作成功：

```
{
  "blbSecurityGroups": [
    {
      "securityGroupDesc": "",
      "securityGroupId": "g-4NxWoxeq",
      "securityGroupName": "securitygroup-name",
      "vpcName": "vpc-name",
      "securityGroupRules": [
        {
          "destGroupId": "",
          "destIp": "all",
          "direction": "egress",
          "ethertype": "IPv4",
          "portRange": "1-65535",
          "protocol": "all",
          "securityGroupRuleId": "r-gkv8yupumvx2"
        }
      ]
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_security_groups.py](#)

🔗 绑定企业安全组

为指定LoadBalancer绑定企业安全组

使用以下代码可以绑定企业安全组

函数声明


```
def bind_app_enterprise_security_groups(self, blb_id, enterprisesecuritygroupids,
                                       client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[BindEnterpriseSecurityGroups绑定企业安全组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_bind_app_enterprise_security_groups.py](#)

🔗 解绑企业安全组

为指定LoadBalancer解绑企业安全组

使用以下代码可以解绑企业安全组 **函数声明**

```
def unbind_app_enterprise_security_groups(self, blb_id, enterprisesecuritygroupids,
                                       client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[UnbindEnterpriseSecurityGroups解绑企业安全组](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_unbind_app_enterprise_security_groups.py](#)

🔗 查询BLB实例企业安全组列表

为指定LoadBalancer查询企业安全组

使用以下代码可以查询企业安全组

函数声明

```
def describe_app_enterprise_security_groups(self, blb_id,
                                       client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeEnterpriseSecurityGroups](#)查询BLB实例企业安全组列表

返回值

操作成功：

```
{
  "enterpriseSecurityGroups": [{
    "enterpriseSecurityGroupId": "esg-g8z4qfj0m0bu",
    "enterpriseSecurityGroupName": "test0627",
    "enterpriseSecurityGroupDesc": "",
    "enterpriseSecurityGroupRules": [{
      "remark": "全部协议",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv6",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu",
      "protocol": "all"
    },
    {
      "remark": "",
      "direction": "egress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "",
      "sourceIp": "",
      "destIp": "all",
      "enterpriseSecurityGroupRuleId": "esgr-9je09z6bd9wv",
      "protocol": "tcp"
    },
    {
      "remark": "",
      "direction": "ingress",
      "action": "allow",
      "priority": 1000,
      "ethertype": "IPv4",
      "portRange": "2",
      "sourceIp": "all",
      "destIp": "",
      "enterpriseSecurityGroupRuleId": "esgr-jcej4juqbh49",
      "protocol": "tcp"
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_app_enterprise_security_groups.py.py](#)

IPv6型BLB实例

创建一个LoadBalancer，返回分配的服务地址及实例ID。付费方式为后付费。需要实名认证，若未通过实名认证可以前往百度智能云官网控制台中的安全认证下的实名认证中进行认证。

使用以下代码可以创建blb实例：

函数声明

```
@required(vpc_id=(bytes, str),
          subnet_id=(bytes, str))
def create_ipv6_loadbalancer(self, vpc_id, subnet_id, name=None,
                             desc=None, client_token=None, config=None):
```

参数含义

请参考OpenAPI文档：[创建blb实例](#)

返回值

操作成功：

```
{
  blb_id:'lb-56443af4',
  name:'BLB-beijing-4f624697',
  desc:'',
  address:'2400:da00:e003:0:2cc:300:0:3'
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_ipv6_loadbalancer.py](#)

🔗 查询blb实例列表

查询用户账户下所有LoadBalancer的信息。支持按LoadBalancer的id、name、address进行查询，匹配规则支持部分包含（不支持正则）。支持按bccld进行查询，目前只支持单个bccld查询，并且仅支持精确查询。查询结果为包含该后端服务器的负载均衡实例。返回结果是多重查询条件的交集。如果BLB实例开启了公网访问，返回结果将会带上公网IP。结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询blb列表：

函数声明

```
def describe_ipv6_loadbalancers(self, address=None, name=None, blb_id=None,
                                 bcc_id=None, marker=None, max_keys=None,
                                 config=None):
```

参数含义

请参考OpenAPI文档：[查询已有的BLB实例](#)

返回值

操作成功：

```
{
  blb_list:[{
    tags:None,
    blb_id:'lb-a93a0d3d',
    status:'available',
    name:'BLB-beijing-4f7ecf2b',
    desc:'',
    address:'2400:da00:e003:0:2cc:300:0:2',
    vpc_id:'vpc-5wseb8brmpxt',
    subnet_id:'sbn-b2iq2g6f00u',
    allow_delete:True,
    payment_timing:'Postpaid'},
  ],
  marker:'blb-05FA1C7F',
  next_marker:None,
  is_truncated:False,
  max_keys:1000
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_describe_ipv6_loadbalancers.py](#)

负载均衡专属集群LBDC

创建LBDC

创建lbdc实例。

使用以下代码可以创建lbdc实例

函数声明

```
def create_lbdc(self, name, type, ccu_count, billing, desc=None, renew=None, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[CreateLBDC创建LBDC](#)

返回值

操作成功：

```
{
  "id": "bgw_group-a3t786p1",
  "type": "4Layer",
  "desc":""
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_lbdc.py](#)

🔗 升级LBDC

升级LBDC。

使用以下代码可以升级lbdc实例

函数声明

```
def upgrade_lbdc(self, lbdc_id, ccu_count, action=b'resize', client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[UpgradeLBDC升级LBDC](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_upgrade_lbdc.py](#)

🔗 续费LBDC

续费lbdc实例。

使用以下代码可以为lbdc实例续费

函数声明

```
def renew_lbdc(self, lbdc_id, billing, action=b'purchaseReserved', client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[RenewLBDC续费LBDC](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_renew_lbdc.py](#)

🔗 获取LBDC列表

获取LBDC列表。

使用以下代码可以获取lbdc列表

函数声明

```
def list_lbdc(self, lbdc_id=None, name=None, config=None)
```

参数含义

请参考OpenAPI文档：[ListLBDC获取LBDC列表](#)

返回值

操作成功：

```
{
  "clusterList": [{
    "id": "bgw_group-a3t786p1",
    "name": "abc",
    "type": "4Layer",
    "status": "available",
    "ccuCount": 1,
    "createTime": "2021-08-25T06:13:09Z",
    "expireTime": "2021-09-25T06:13:09Z",
    "desc": ""
  }
],
  "marker": "bgw_group-a3t786p1",
  "nextMarker": null,
  "isTruncated": false,
  "maxKeys": 1000
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_list_lbdc.py](#)

🔗 获取LBDC详情

获取lbdc详情。

使用以下代码可以获取lbdc详情

函数声明

```
def get_lbdc(self, lbdc_id, config=None)
```

参数含义

请参考OpenAPI文档：[DescribeLBDC获取LBDC详情](#)

返回值

操作成功：

```
{
  "id": "bgw_group-a3t786p1",
  "name": "abc",
  "type": "4Layer",
  "status": "available",
  "ccuCount": 1,
  "createTime": "2021-08-25T06:13:09Z",
  "expireTime": "2021-09-25T06:13:09Z",
  "desc": "",
  "totalConnectCount":100,
  "newConnectCps":100,
  "networkInBps":100,
  "networkOutBps":100
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_get_lbdc.py](#)

🔗 更新LBDC

更新lbdc实例。

使用以下代码可以更新lbdc

函数声明

```
def update_lbdc(self, lbdc_id, name=None, desc=None, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[UpdateLBDC更新LBDC](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_lbdc.py](#)

🔗 获取LBDC关联的BLB列表

获取lbdc关联的blb列表。

使用以下代码可以获取lbdc关联的blb列表

函数声明

```
def list_lbdc_blb(self, lbdc_id, config=None)
```

参数含义

请参考OpenAPI文档：[GetBlibListOfLBDC获取LBDC关联的BLB列表](#)

返回值

操作成功：

```
{
  "blbList": [
    {
      "blbId": "lb-1c791997",
      "name": "abc",
      "status": "available",
      "blbType": "normal",
      "publicIp": "10.10.10.10",
      "eipRouteType": "bgp",
      "bandwidth": 20,
      "address": "192.168.0.7",
      "ipv6": "2400:da00:e003:1820::4",
      "vpcId": "vpc-0n1hhh8759b0",
      "subnetId": "sbn-p09p0ihp7498"
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_list_lbdc_blb.py](#)

服务发布点

🔗 创建服务发布点

创建服务发布点

使用以下代码可以创建服务发布点

函数声明

```
def create_user_service(self, name, desc, serviceName,
                        instanceId, client_token=None,
                        authList=None, config=None)
```

参数含义

请参考OpenAPI文档：[创建服务发布点](#)

返回值

操作成功：

```
{
  "service": "xxx.baidubce.com"
}
```


操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_create_user_service.py](#)

🔗 更新服务发布点

更新服务发布点，name、description不能同时为空

使用以下代码可以更新服务发布点

函数声明

```
def update_user_service(self, name=None, desc=None, service=None,
                        client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[更新服务发布点](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_update_user_service.py](#)

🔗 服务发布点绑定实例

服务发布点绑定服务

使用以下代码可以服务发布点绑定实例

函数声明

```
def user_service_bind_instance(self, instanceId, service,
                              client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[服务发布点绑定实例](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_user_service_bind_instance.py](#)

🔗 服务发布点解绑实例

服务发布点解绑服务

使用以下代码可以服务发布点解绑实例

函数声明

```
def user_service_unbind_instance(self, service,
                                client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[服务发布点解绑服务](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_user_service_unbind_instance.py](#)

🔗 新增服务发布点鉴权信息

新增服务发布点鉴权信息，如果用户授权已存在则覆盖之前的授权信息

使用以下代码可以新增服务发布点鉴权信息

函数声明

```
def user_service_add_auth(self, service,
                           authList, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[新增服务发布点鉴权信息](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_user_service_add_auth.py](#)

🔗 修改服务发布点鉴权信息

修改服务发布点鉴权信息

使用以下代码可以修改服务发布点鉴权信息

函数声明

```
def user_service_edit_auth(self, service,
                           authList, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[修改服务发布点鉴权信息](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_user_service_edit_auth.py](#)

🔗 删除服务发布点鉴权信息

删除服务发布点鉴权信息

使用以下代码可以删除服务发布点鉴权信息

函数声明

```
def user_service_remove_auth(self, service,
                             uidList, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[删除服务发布点鉴权信息](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_user_service_remove_auth.py](#)

🔗 查询服务发布点列表

查询服务发布点列表。

使用以下代码可以查询服务发布点列表

函数声明

```
def get_user_service_list(self, marker=None,
                           maxKeys=None, client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[查询服务发布点列表](#)

返回值

操作成功：

```
{
  "nextMarker": "user-service-a223vc",
  "marker": "user-service-a032va",
  "maxKeys": 1,
  "isTruncated": true,
  "services": [
    {
      "serviceId": "user-service-a223vc",
      "name": "name",
      "description": "",
      "serviceName": "serviceName",
      "bindType": "blb",
      "instanceId": "lb-d3e2cf75",
      "status": "inService",
      "service": "xxx.baidubce.com",
      "createTime": "2019-03-07T02:35:31Z",
      "endpointCount": 1,
      "endpointList": [
        {
          "endpointId": "endpoint-a223vc",
          "uid": "e8deba4d81d3435bb4591ce97288f296",
          "attachTime": "2019-03-07T02:35:31Z",
        }
      ],
      "authList": [
        {
          "uid": "*",
          "auth": "allow"
        }
      ]
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_get_user_service_list.py](#)

查询服务发布点详情

查询服务发布点详情

使用以下代码可以查询服务发布点详情

函数声明

```
def get_user_service_detail(self, service,
                             client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[查询服务发布点详情](#)

返回值

操作成功：

```
{
  "serviceId": "uservice-a223vc",
  "name": "name",
  "description": "",
  "serviceName": "serviceName",
  "bindType": "blb",
  "instanceId": "lb-d3e2cf75",
  "status": "inService",
  "service": "xxx.baidubce.com",
  "createTime": "2019-03-07T02:35:31Z",
  "endpointCount": 1,
  "endpointList": [
    {
      "endpointId": "endpoint-a223vc",
      "uid": "e8deba4d81d3435bb4591ce97288f296",
      "attachTime": "2019-03-07T02:35:31Z",
    },
  ],
  "authList": [
    {
      "uid": "*",
      "auth": "allow"
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_get_user_service_detail.py](#)

🔗 删除服务发布点

删除服务发布点

使用以下代码可以删除服务发布点

函数声明

```
def delete_user_service(self, service,
                          client_token=None, config=None)
```

参数含义

请参考OpenAPI文档：[删除服务发布点](#)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](#)

代码示例

具体代码示例参考：[example_delete_user_service.py](#)

版本更新说明

- Python SDK开发包[2019-08-14] 版本号 0.8.28

PHP-SDK

概述

本文档主要介绍BLB PHP SDK的安装和使用。在使用本文档前，您需要先了解BLB的一些基本知识，并已经开通了BLB服务。若您还不了解BLB，可以参考[产品描述](#)和[快速入门](#)。

安装

运行环境

1. 运行环境 PHP SDK包要求运行环境至少为PHP 5.3.2 版本。
2. 鉴权和认证 要使用百度智能云产品，您需要拥有一个百度智能云账号和一个有效的 AK(Access Key ID)、SK(Secret Access Key)用来进行签名认证。

可以通过如下步骤获得并了解您的AK/SK信息：

1. [注册百度智能云账号](#)
2. [创建AK/SK](#)

安装SDK

1. 在[官方网站](#)下载Java SDK压缩工具包。
2. 解压ZIP包之后有三个文件，如下：

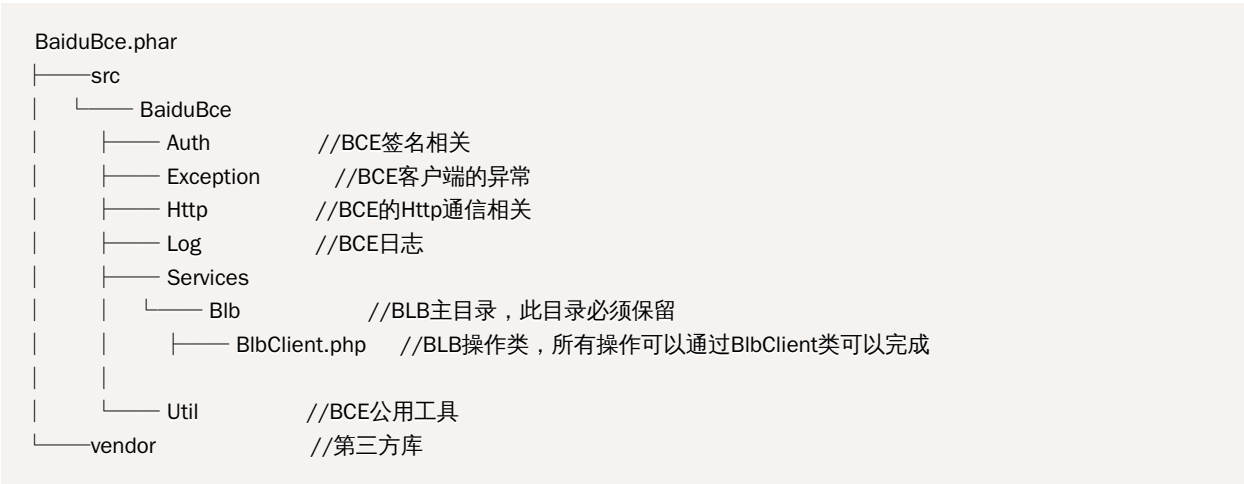
```
BaiduBce.phar //PHP SDK
BlbClient.php //示例
SampleConf.php //参考配置文件，具体内容见下文
```

3. 在脚本文件中添加以下代码，即可以使用SDK包：

```
include 'BaiduBce.phar';
require 'YourConf.php';
```

关于配置文件的引用请参考下文中的配置BlbClient。

SDK目录结构



🔗 卸载SDK

预期卸载 SDK 时，删除脚本中include的phar包即可。

初始化

🔗 确认Endpoint

在确认您使用SDK时配置的Endpoint时，可先阅读开发人员指南中关于[BLB访问域名](#)的部分，理解Endpoint相关的概念。 百度云目前开放了多区域支持，请参考[区域选择说明](#)。

目前支持“北京”、“广州”、“苏州”、“香港”、“武汉”、“保定”、“上海”、“新加坡”等区域。

服务域名的对应信息为：

访问区域	对应Endpoint
bj	blb.bj.baidubce.com
gz	blb.gz.baidubce.com
su	blb.su.baidubce.com
hkg	blb.hkg.baidubce.com
fwh	blb.fwh.baidubce.com
bd	blb.bd.baidubce.com
fsh	blb.fsh.baidubce.com
sin	blb.sin.baidubce.com

🔗 新建BlbClient

BlbClient是BLB服务的客户端，为开发者与BLB服务进行交互提供了一系列的方法。

在新建BlbClient之前，需要先创建配置文件对BlbClient进行配置，以下将此配置文件命名为YourConf.php，具体配置信息如下所示：

```
// 报告所有 PHP 错误
error_reporting(-1);

define('__BLB_CLIENT_ROOT', dirname(__DIR__));

// 设置BibClient的Access Key ID、Secret Access Key和ENDPOINT
$BLB_TEST_CONFIG =
    array(
        'credentials' => array(
            'accessKeyId' => 'your ak',
            'secretAccessKey' => 'your sk',
            'sessionToken' => 'your session token'
        ),
        'endpoint' => 'http://blb.bj.baidubce.com',
        'stsEndpoint' => 'http://sts.bj.baidubce.com',
    );

// 设置log的格式和级别
$__handler = new \Monolog\Handler\StreamHandler(STDERR, \Monolog\Logger::DEBUG);
$__handler->setFormatter(
    new \Monolog\Formatter\LineFormatter(null, null, false, true)
);
\BaiduBce\Log\LogFactory::setInstance(
    new \BaiduBce\Log\MonoLogFactory(array($__handler))
);
\BaiduBce\Log\LogFactory::setLogLevel(\Psr\Log\LogLevel::DEBUG);
```

注意：1.在上面代码中，ACCESS_KEY_ID对应控制台中的“Access Key ID”，SECRET_ACCESS_KEY对应控制台中的“Access Key Secret”，获取方式请参考《操作指南 [获取ACCESSKEY](#)》。

2.如果用户需要自己指定域名，可以通过传入ENDPOINT参数来指定，ENDPOINT参数需要用指定区域的域名来进行定义，如服务所在区域为北京，则为<http://blb.bj.baidubce.com>。

3.如果不使用STS验证方式，则stsEndpoint和sessionToken可以为空或者删除。

使用AK/SK新建BibClient

通过AK/SK方式访问BLB，用户可以参考如下代码新建一个BibClient：

```
//使用PHP SDK，并且使用自定义配置文件
include 'BaiduBce.phar';
require 'YourConf.php';

use BaiduBce\BceClientConfigOptions;
use BaiduBce\Util\Time;
use BaiduBce\Util\MimeTypes;
use BaiduBce\Http\HttpHeaders;
use BaiduBce\Services\Bib\BibClient;

//调用配置文件中的参数
global $BLB_TEST_CONFIG;
//新建BibClient
$client = new BibClient($BLB_TEST_CONFIG);
```

使用STS创建BibClient 申请STS token BLB可以通过STS机制实现第三方的临时授权访问。STS（Security Token Service）是百度智能云提供的临时授权服务。通过STS，您可以为第三方用户颁发一个自定义时效和权限的访问凭证。第三方用户可以使用该访问凭证直接调用百度智能云的API或SDK访问百度智能云资源。

通过STS方式访问BLB，用户需要先通过STS的client申请一个认证字符串，申请方式可参见[百度智能云STS使用介绍](#)。

用STS token新建BLBClient 申请好STS后，可将STSToken配置到BlbClient中，用户可以参考如下代码新建一个BlbClient： 1.首先进行STS的endpoint配置。STS的配置示例如下：

```
$BLB_TEST_CONFIG =
    array(
        'credentials' => array(
            'accessKeyId' => 'your ak',
            'secretAccessKey' => 'your sk',
        ),
        'stsEndpoint' => 'http://sts.bj.baidubce.com',
    );
```

2.StsClient的示例代码如下：

```
//新建StsClient
$client = new StsClient($BLB_TEST_CONFIG);
$request =
    array(
        'acl' => $aclArray, //用户定义的acl
        'durationSeconds' => 43200, //STS凭证有效时间
    );
$response = $client->getSessionToken($request);
$accessKeyId= $response->accessKeyId;
$secretAccessKey= $response->secretAccessKey;
$sessionToken = $response->sessionToken;
```

3.将获取到的accessKeyId/secretAccessKey/sessionToken写入到配置文件YourConf.php中并新建BlbClient。

```
$BLB_TEST_CONFIG =
    array(
        'credentials' => array(
            'accessKeyId' => 'your ak',
            'secretAccessKey' => 'your sk',
            'sessionToken' => 'your session token'
        ),
        'endpoint' => 'http://blb.bj.baidubce.com',
    );

//新建BlbClient
$client = new BlbClient($BLB_TEST_CONFIG);
```

注意：目前使用STS配置client时，无论对应BLB服务的endpoint在哪里，endpoint都需配置为<http://sts.bj.baidubce.com>。

配置BlbClient

PHP SDK在\BaiduBce\Bce.php中默认设置了一些基本参数，若用户想要对参数的值进行修改，可以参考此文件来创建自身的参数配置函数，并在构造BlbClient的时候传入，传入代码参考如下：

```
public function CustomizedConfig() {
    $customizedConfig = array(
        BceClientConfigOptions::PROTOCOL => 'http',
        BceClientConfigOptions::REGION => 'bj',
        BceClientConfigOptions::CONNECTION_TIMEOUT_IN_MILLIS => 120 * 1000,
        BceClientConfigOptions::SOCKET_TIMEOUT_IN_MILLIS => 300 * 1000,
        BceClientConfigOptions::SEND_BUF_SIZE => 5 * 1024 * 1024,
        BceClientConfigOptions::RECV_BUF_SIZE => 5 * 1024 * 1024,
        BceClientConfigOptions::CREDENTIALS => array(
            'ak' => 'your-access-key-id',
            'sk' => 'your-secret-access-key',
        ),
        'endpoint' => 'your-endpoint',
    );

    //利用自定义配置创建BlbClient
    $customizedClient = new BlbClient($customizedConfig);

    //通过自定义配置调用方法
    $options = array(BlbOptions::CONFIG=>$customizedConfig);
    $this->client->listBlbs($options);
}
```

参数说明

参数	说明	默认值
PROTOCOL	协议	http
REGION	区域	bj
CONNECTION_TIMEOUT_IN_MILLIS	请求超时时间（单位：毫秒）	50 * 1000
SOCKET_TIMEOUT_IN_MILLIS	通过打开的连接传输数据的超时时间（单位：毫秒）	0（指的是无限等待，若设置非0数值需要对文件大小和网速进行评估，否则上传大文件时会产生超时）
SEND_BUF_SIZE	发送缓冲区大小	1024 * 1024
RECV_BUF_SIZE	协议	http
PROTOCOL	接收缓冲区大小	10 1024 1024

普通型blb实例

普通型blb实例提供了负载均衡常规功能组件，能够满足用户对负载均衡的通用需求，具备轻量化、配置简单、价格便宜等特点。

具体各个参数含义请参考blb官方文档，api参考下的[普通型blb接口](#)

创建blb实例

使用以下代码可以创建blb实例：

```

/**
 * Create an nat with the specified options.
 *
 * @param string $name
 *     name of nat
 *
 * @param string $vpclId
 *     id of vpc which nat created in
 *
 * @param string $subnetId
 *     id of subnet which blb created in
 *
 * @param string $desc
 *     description of blb
 *
 * @param string $clientToken
 *     if the clientToken is not specified by the user, a random String
 *     generated by default algorithm will be used.
 *
 * @param array $options
 *     The optional bce configuration, which will overwrite the
 *     default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function createBlb($vpclId = null, $subnetId = null, $name = null, $desc = null, $clientToken = null, $options = array())
{
}

```

使用示例如下：

```

public function testCreateBlb()
{
    $resp = $this->client->createBlb("vpc-545hyqfg6m5s","sbn-e54j0dxwshj6");
    print_r($resp);
    $this->assertNotNull($resp);
}

```

🔗 修改blb实例

使用以下代码可以修改blb实例：

```

/**
 * update single blb
 *
 * @param string $blbld
 *      id of blb instance
 *
 * @param string $name
 *      name of blb instance to be updated
 *
 * @param string $desc
 *      description of blb instance to be updated
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function updateBlb($blbld, $name = null, $desc = null, $clientToken = null, $options = array())
{

}

```

使用示例如下：

```

public function testUpdateBlb()
{
    $resp = $this->client->updateBlb('lb-48872493', 'testName', 'testDesc');
    print_r($resp);
    $this->assertNotNull($resp);
}

```

🔗 查询blb实例列表

使用以下代码可以查询blb列表：

```

/**
 * get a list of blb owned by the authenticated user and specified conditions.
 *
 * @param string $address
 *     address of blb in vpc. The optional parameter
 *
 * @param string $blbId
 *     id of blb instance. The optional parameter
 *
 * @param string $name
 *     name of nat instance. The optional parameter
 *
 * @param string $bccId
 *     id of bcc instance which is bound to blb instance. The optional parameter
 *
 * @param string $marker
 *     The optional parameter marker specified in the original request to specify
 *     where in the results to begin listing.
 *
 * @param int $maxKeys
 *     The optional parameter to specifies the max number of list result to return.
 *     The default value is 1000.
 *
 * @param array $options
 *     The optional bce configuration, which will overwrite the
 *     default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function listBlbs($address = null, $name = null, $blbId = null, $bccId = null, $marker = null, $maxKeys = 1000,
$options = array())
{

}

```

使用示例如下：

```

public function testListBlb()
{
    $resp = $this->client->listBlbs();
    print_r($resp);
    $this->assertNotNull($resp);
}

```

🔗 删除blb实例

使用以下代码可以删除blb实例：

```
/**
 * delete the blb
 * Only the Postpaid instance or Prepaid which is expired can be released.
 *
 * @param string $blbId
 *      id of blb instance to be released
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function deleteBlb($blbId, $clientToken = null, $options = array())
{

}
```

使用示例如下：

```
public function testDeleteBlb()
{
    $resp = $this->client->deleteBlb('lb-90d60843');
    print_r($resp);
    $this->assertNotNull($resp);
}
```

创建tcpListener实例

使用以下代码可以创建tcpListener实例：

```

/**
 * Create a tcp listener with the specified options.
 *
 * @param string $blbId
 *      id of blb
 *
 * @param int $listenerPort
 *      front end port of listener
 *
 * @param int $backendPort
 *      back end port of listener
 *
 * @param string $scheduler
 *      scheduler of load balance
 *
 * @param int $healthCheckTimeoutInSecond
 *      timeout of health check, default 3 seconds
 *
 * @param int $healthCheckInterval
 *      interval of health check, default 3 seconds
 *
 * @param int $unhealthyThreshold
 *      times threshold of unhealthy check, default 3 times
 *
 * @param int $healthyThreshold
 *      times threshold of healthy check, default 3 times
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function createTcpListener($blbId, $listenerPort, $backendPort, $scheduler, $healthCheckTimeoutInSecond = 3,
$healthCheckInterval = 3, $unhealthyThreshold = 3, $healthyThreshold = 3, $clientToken = null, $options = array())
{
}

```

使用示例如下：

```

public function testCreateTcpListener()
{
    $resp = $this->client->createTcpListener('lb-48872493', 8000, 8008, 'RoundRobin');
    print_r($resp);
    $this->assertNotNull($resp);
}

```

创建udpListener实例

使用以下代码可以创建udpListener实例：

```

/**
 * Create a udp listener with the specified options.
 *
 * @param string $blbld
 *      id of blb
 *
 * @param int $listenerPort
 *      front end port of listener
 *
 * @param int $backendPort
 *      back end port of listener
 *
 * @param string $scheduler
 *      scheduler of load balance
 *
 * @param string $healthCheckString
 *      string to send for health check
 *
 * @param int $healthCheckTimeoutInSecond
 *      timeout of health check, default 3 seconds
 *
 * @param int $healthCheckInterval
 *      interval of health check, default 3 seconds
 *
 * @param int $unhealthyThreshold
 *      times threshold of unhealthy check, default 3 times
 *
 * @param int $healthyThreshold
 *      times threshold of healthy check, default 3 times
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function createUdpListener($blbld, $listenerPort, $backendPort, $scheduler, $healthCheckString,
$healthCheckTimeoutInSecond = 3, $healthCheckInterval = 3, $unhealthyThreshold = 3, $healthyThreshold = 3,
$clientToken = null, $options = array())
{

}

```

使用示例如下：

```

public function testCreateUdpListener()
{
    $resp = $this->client->createUdpListener('lb-48872493', 8001, 8008, 'RoundRobin', 'aaa');
    print_r($resp);
    $this->assertNotNull($resp);
}

```

创建httpListener实例

使用以下代码可以创建httpListener实例：


```
/**
 * Create a http listener with the specified options.
 *
 * @param string $blbId
 *     id of blb
 *
 * @param int $listenerPort
 *     front end port of listener
 *
 * @param int $backendPort
 *     back end port of listener
 *
 * @param string $scheduler
 *     scheduler of load balance
 *
 * @param boolean $keepSession
 *     switch for keep session
 *
 * @param string $keepSessionType
 *     type to keep session
 *
 * @param int $keepSessionDuration
 *     duration to keep session
 *
 * @param string $keepSessionCookieName
 *     cookie name for keep session
 *
 * @param boolean $xForwardFor
 *     switch for acquiring real ip
 *
 * @param string $healthCheckType
 *     type for health check
 *
 * @param int $healthCheckPort
 *     port for health check
 *
 * @param string $healthCheckURI
 *     uri for health check
 *
 * @param int $healthCheckTimeoutInSecond
 *     timeout of health check, default 3 seconds
 *
 * @param int $healthCheckInterval
 *     interval of health check, default 3 seconds
 *
 * @param int $unhealthyThreshold
 *     times threshold of unhealthy check, default 3 times
 *
 * @param int $healthyThreshold
 *     times threshold of healthy check, default 3 times
 *
 * @param string $healthCheckNormalStatus
 *     normal status for http health check
 *
 * @param int $serverTimeout
 *     timeout for backend server
 *
 * @param int $redirectPort
 *     port for redirect
 *
 * @param string $clientToken
 *     if the clientToken is not specified by the user, a random String
```

```

*      generated by default algorithm will be used.
*
* @param array $options
*      The optional bce configuration, which will overwrite the
*      default configuration that was passed while creating EipClient instance.
*
* @return mixed
*/
public function createHttpListener($blbId, $listenerPort, $backendPort, $scheduler, $keepSession = false,
                                   $keepSessionType = null, $keepSessionDuration = 3600, $keepSessionCookieName = null,
                                   $xForwardFor = false, $healthCheckType = null, $healthCheckPort = null, $healthCheckURI =
null,
                                   $healthCheckTimeoutInSecond = 3, $healthCheckInterval = 3, $unhealthyThreshold = 3,
$healthyThreshold = 3,
                                   $healthCheckNormalStatus = null, $serverTimeout = 30, $redirectPort = null, $clientToken =
null, $options = array())
{

}

```

使用示例如下：

```

public function testCreateHttpListener()
{
    $resp = $this->client->createHttpListener('lb-48872493', 8002, 8008, 'RoundRobin');
    print_r($resp);
    $this->assertNotNull($resp);
}

```

创建httpsListener实例

使用以下代码可以创建httpsListener实例：

```

/**
 * Create a https listener with the specified options.
 *
 * @param string $blbId
 *      id of blb
 *
 * @param int $listenerPort
 *      front end port of listener
 *
 * @param int $backendPort
 *      back end port of listener
 *
 * @param string $scheduler
 *      scheduler of load balance
 *
 * @param array $certIds
 *      SSL certs for listener
 *
 * @param boolean $keepSession
 *      switch for keep session
 *
 * @param string $keepSessionType
 *      type to keep session
 *
 * @param int $keepSessionDuration
 *      duration to keep session
 *
 * @param string $keepSessionCookieName

```

```

*      cookie name for keep session
*
* @param boolean $xForwardFor
*      switch for acquiring real ip
*
* @param string $healthCheckType
*      type for health check
*
* @param int $healthCheckPort
*      port for health check
*
* @param string $healthCheckURI
*      uri for health check
*
* @param int $healthCheckTimeoutInSecond
*      timeout of health check, default 3 seconds
*
* @param int $healthCheckInterval
*      interval of health check, default 3 seconds
*
* @param int $unhealthyThreshold
*      times threshold of unhealthy check, default 3 times
*
* @param int $healthyThreshold
*      times threshold of healthy check, default 3 times
*
* @param string $healthCheckNormalStatus
*      normal status for http health check
*
* @param int $serverTimeout
*      timeout for backend server
*
* @param boolean $ie6Compatible
*      switch for compatibility with IE6
*
* @param string $clientToken
*      if the clientToken is not specified by the user, a random String
*      generated by default algorithm will be used.
*
* @param array $options
*      The optional bce configuration, which will overwrite the
*      default configuration that was passed while creating EipClient instance.
*
* @return mixed
*/
public function createHttpsListener($blbId, $listenerPort, $backendPort, $scheduler, $certIds, $keepSession = false,
    $keepSessionType = null, $keepSessionDuration = 3600, $keepSessionCookieName = null,
    $xForwardFor = false, $healthCheckType = null, $healthCheckPort = null, $healthCheckURI =
null,
    $healthCheckTimeoutInSecond = 3, $healthCheckInterval = 3, $unhealthyThreshold = 3,
    $healthyThreshold = 3,
    $healthCheckNormalStatus = null, $serverTimeout = 30, $ie6Compatible = true, $clientToken
= null, $options = array())
{

}

```

使用示例如下：

```
public function testCreateHttpsListener()
{
    $resp = $this->client->createHttpsListener('lb-5538a312', 8000, 8008, 'RoundRobin', array('cert-ahc9dhfh7icf'));
    print_r($resp);
    $this->assertNotNull($resp);
}
```

🔗 查询tcpListener实例

使用以下代码可以查询tcpListener实例：

```
/**
 * list tcp listener.
 *
 * @param string $blbld
 *     id of blb instance
 *
 * @param int listenerPort
 *     port of listener
 *
 * @param string $marker
 *     The optional parameter marker specified in the original request to specify
 *     where in the results to begin listing.
 *
 * @param int $maxKeys
 *     The optional parameter to specifies the max number of list result to return.
 *     The default value is 1000.
 *
 * @param array $options
 *     The optional bce configuration, which will overwrite the
 *     default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function listTcpListeners($blbld, $listenerPort = null, $marker = null, $maxKeys = 1000, $options = array())
{
}
}
```

使用示例如下：

```
public function testListTcpListeners()
{
    $resp = $this->client->listTcpListeners('lb-48872493');
    print_r($resp);
    $this->assertNotNull($resp);
}
```

🔗 查询udpListener实例

使用以下代码可以查询udpListener实例：

```
/**
 * list udp listener.
 *
 * @param string $blbId
 *      id of blb instance
 *
 * @param int listenerPort
 *      port of listener
 *
 * @param string $marker
 *      The optional parameter marker specified in the original request to specify
 *      where in the results to begin listing.
 *
 * @param int $maxKeys
 *      The optional parameter to specifies the max number of list result to return.
 *      The default value is 1000.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function listUdpListeners($blbId, $listenerPort = null, $marker = null, $maxKeys = 1000, $options = array())
{

}
```

使用示例如下：

```
public function testListUdpListeners()
{
    $resp = $this->client->listUdpListeners('lb-48872493');
    print_r($resp);
    $this->assertNotNull($resp);
}
```

查询httpListener实例

使用以下代码可以查询httpListener实例：

```

/**
 * list http listener.
 *
 * @param string $blbld
 *      id of blb instance
 *
 * @param int listenerPort
 *      port of listener
 *
 * @param string $marker
 *      The optional parameter marker specified in the original request to specify
 *      where in the results to begin listing.
 *
 * @param int $maxKeys
 *      The optional parameter to specifies the max number of list result to return.
 *      The default value is 1000.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function listHttpListeners($blbld, $listenerPort = null, $marker = null, $maxKeys = 1000, $options = array())
{
}

```

使用示例如下：

```

public function testListHttpListeners()
{
    $resp = $this->client->listHttpListeners('lb-48872493');
    print_r($resp);
    $this->assertNotNull($resp);
}

```

🔗 查询httpsListener实例

使用以下代码可以查询httpsListener实例：

```

/**
 * list https listener.
 *
 * @param string $blbld
 *      id of blb instance
 *
 * @param int listenerPort
 *      port of listener
 *
 * @param string $marker
 *      The optional parameter marker specified in the original request to specify
 *      where in the results to begin listing.
 *
 * @param int $maxKeys
 *      The optional parameter to specifies the max number of list result to return.
 *      The default value is 1000.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function listHttpsListeners($blbld, $listenerPort = null, $marker = null, $maxKeys = 1000, $options = array())
{
}

```

使用示例如下：

```

public function testListHttpsListeners()
{
    $resp = $this->client->listHttpsListeners('lb-5538a312');
    print_r($resp);
    $this->assertNotNull($resp);
}

```

更新tcpListener实例

使用以下代码可以更新tcpListener实例：

```

/**
 * update a tcp listener with the specified options.
 *
 * @param string $blbld
 *      id of blb
 *
 * @param int $listenerPort
 *      front end port of listener
 *
 * @param int $backendPort
 *      back end port of listener
 *
 * @param string $scheduler
 *      scheduler of load balance
 *
 * @param int $healthCheckTimeoutInSecond
 *      timeout of health check, default 3 seconds
 *
 * @param int $healthCheckInterval
 *      interval of health check, default 3 seconds
 *
 * @param int $unhealthyThreshold
 *      times threshold of unhealthy check, default 3 times
 *
 * @param int $healthyThreshold
 *      times threshold of healthy check, default 3 times
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function updateTcpListener($blbld, $listenerPort, $backendPort = null, $scheduler = null,
$healthCheckTimeoutInSecond = null, $healthCheckInterval = null, $unhealthyThreshold = null, $healthyThreshold = null,
$clientToken = null, $options = array())
{
}

```

使用示例如下：

```

public function testUpdateTcpListener()
{
    $resp = $this->client->updateTcpListener('lb-48872493', 8000, 8009);
    print_r($resp);
    $this->assertNotNull($resp);
}

```

更新udpListener实例

使用以下代码可以更新udpListener实例：


```

/**
 * update a udp listener with the specified options.
 *
 * @param string $blbId
 *     id of blb
 *
 * @param int $listenerPort
 *     front end port of listener
 *
 * @param int $backendPort
 *     back end port of listener
 *
 * @param string $scheduler
 *     scheduler of load balance
 *
 * @param string $healthCheckString
 *     string to send for health check
 *
 * @param int $healthCheckTimeoutInSecond
 *     timeout of health check, default 3 seconds
 *
 * @param int $healthCheckInterval
 *     interval of health check, default 3 seconds
 *
 * @param int $unhealthyThreshold
 *     times threshold of unhealthy check, default 3 times
 *
 * @param int $healthyThreshold
 *     times threshold of healthy check, default 3 times
 *
 * @param string $clientToken
 *     if the clientToken is not specified by the user, a random String
 *     generated by default algorithm will be used.
 *
 * @param array $options
 *     The optional bce configuration, which will overwrite the
 *     default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function updateUdpListener($blbId, $listenerPort, $backendPort, $scheduler, $healthCheckString,
$healthCheckTimeoutInSecond = null, $healthCheckInterval = null, $unhealthyThreshold = null, $healthyThreshold = null,
$clientToken = null, $options = array())
{

}

```

使用示例如下：

```

public function testUpdateUdpListener()
{
    $resp = $this->client->updateUdpListener('lb-48872493', 8001, 8009, 'RoundRobin', 'bbb');
    print_r($resp);
    $this->assertNotNull($resp);
}

```

更新httpListener实例

使用以下代码可以更新httpListener实例：

```
/**
 * update a http listener with the specified options.
 *
 * @param string $blbId
 *     id of blb
 *
 * @param int $listenerPort
 *     front end port of listener
 *
 * @param int $backendPort
 *     back end port of listener
 *
 * @param string $scheduler
 *     scheduler of load balance
 *
 * @param boolean $keepSession
 *     switch for keep session
 *
 * @param string $keepSessionType
 *     type to keep session
 *
 * @param int $keepSessionDuration
 *     duration to keep session
 *
 * @param string $keepSessionCookieName
 *     cookie name for keep session
 *
 * @param boolean $xForwardFor
 *     switch for acquiring real ip
 *
 * @param string $healthCheckType
 *     type for health check
 *
 * @param int $healthCheckPort
 *     port for health check
 *
 * @param string $healthCheckURI
 *     uri for health check
 *
 * @param int $healthCheckTimeoutInSecond
 *     timeout of health check, default 3 seconds
 *
 * @param int $healthCheckInterval
 *     interval of health check, default 3 seconds
 *
 * @param int $unhealthyThreshold
 *     times threshold of unhealthy check, default 3 times
 *
 * @param int $healthyThreshold
 *     times threshold of healthy check, default 3 times
 *
 * @param string $healthCheckNormalStatus
 *     normal status for http health check
 *
 * @param int $serverTimeout
 *     timeout for backend server
 *
 * @param int $redirectPort
 *     port for redirect
 *
 * @param string $clientToken
 *     if the clientToken is not specified by the user, a random String
```

```

*      generated by default algorithm will be used.
*
* @param array $options
*      The optional bce configuration, which will overwrite the
*      default configuration that was passed while creating EipClient instance.
*
* @return mixed
*/
public function updateHttpListener($blbId, $listenerPort, $backendPort = null, $scheduler = null, $keepSession = null,
    $keepSessionType = null, $keepSessionDuration = null, $keepSessionCookieName = null,
    $xForwardFor = null, $healthCheckType = null, $healthCheckPort = null, $healthCheckURI = null,
    $healthCheckTimeoutInSecond = null, $healthCheckInterval = null, $unhealthyThreshold = null,
    $healthyThreshold = null,
    $healthCheckNormalStatus = null, $serverTimeout = null, $redirectPort = null, $clientToken =
    null, $options = array())
{
}

```

使用示例如下：

```

public function testUpdateHttpListener()
{
    $resp = $this->client->updateHttpListener('lb-48872493', 8002, 8009);
    print_r($resp);
    $this->assertNotNull($resp);
}

```

更新httpsListener实例

使用以下代码可以更新httpsListener实例：

```

/**
 * update a https listener with the specified options.
 *
 * @param string $blbId
 *      id of blb
 *
 * @param int $listenerPort
 *      front end port of listener
 *
 * @param int $backendPort
 *      back end port of listener
 *
 * @param string $scheduler
 *      scheduler of load balance
 *
 * @param array $certIds
 *      SSL certs for listener
 *
 * @param boolean $keepSession
 *      switch for keep session
 *
 * @param string $keepSessionType
 *      type to keep session
 *
 * @param int $keepSessionDuration
 *      duration to keep session
 *
 * @param string $keepSessionCookieName

```

```

*      cookie name for keep session
*
* @param boolean $xForwardFor
*      switch for acquiring real ip
*
* @param string $healthCheckType
*      type for health check
*
* @param int $healthCheckPort
*      port for health check
*
* @param string $healthCheckURI
*      uri for health check
*
* @param int $healthCheckTimeoutInSecond
*      timeout of health check, default 3 seconds
*
* @param int $healthCheckInterval
*      interval of health check, default 3 seconds
*
* @param int $unhealthyThreshold
*      times threshold of unhealthy check, default 3 times
*
* @param int $healthyThreshold
*      times threshold of healthy check, default 3 times
*
* @param string $healthCheckNormalStatus
*      normal status for http health check
*
* @param int $serverTimeout
*      timeout for backend server
*
* @param boolean $ie6Compatible
*      switch for compatibility with IE6
*
* @param string $clientToken
*      if the clientToken is not specified by the user, a random String
*      generated by default algorithm will be used.
*
* @param array $options
*      The optional bce configuration, which will overwrite the
*      default configuration that was passed while creating EipClient instance.
*
* @return mixed
*/
public function updateHttpsListener($blbId, $listenerPort, $backendPort = null, $scheduler = null, $certIds = null,
$keepSession = null,
                                $keepSessionType = null, $keepSessionDuration = null, $keepSessionCookieName = null,
                                $xForwardFor = null, $healthCheckType = null, $healthCheckPort = null, $healthCheckURI =
null,
                                $healthCheckTimeoutInSecond = null, $healthCheckInterval = null, $unhealthyThreshold = null,
$healthyThreshold = null,
                                $healthCheckNormalStatus = null, $serverTimeout = null, $ie6Compatible = null, $clientToken
= null, $options = array())
{
}

```

使用示例如下：

```
public function testUpdateHttpsListener()
{
    $resp = $this->client->updateHttpsListener('lb-5538a312', 8000, 8010);
    print_r($resp);
    $this->assertNotNull($resp);
}
```

🔗 删除Listener实例

使用以下代码可以删除Listener实例：

```
/**
 * delete a listener.
 *
 * @param string $blbld
 *     id of blb
 *
 * @param array $portList
 *     ports of listener to be deleted
 *
 * @param string $clientToken
 *     if the clientToken is not specified by the user, a random String
 *     generated by default algorithm will be used.
 *
 * @param array $options
 *     The optional bce configuration, which will overwrite the
 *     default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function deleteListeners($blbld, $portList, $clientToken = null, $options = array())
{
}
```

使用示例如下：

```
public function testDeleteListener()
{
    $resp = $this->client->deleteListeners('lb-48872493', array(8000, 8001));
    print_r($resp);
    $this->assertNotNull($resp);
}
```

🔗 增加后端服务器实例

使用以下代码可以增加后端服务器实例：

```

/**
 * add backend server
 *
 * @param string $blbId
 *      blb instance id
 *
 * @param array $backendServerList
 *      back end servers to be added
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function addBackendServer($blbId, $backendServerList, $clientToken = null, $options = array())
{

}

```

使用示例如下：

```

public function testAddServers()
{
    $b1 = new BackendServer('i-xtsPZTGY', 2);
    $b2 = new BackendServer('i-Xf3hfqWt', 3);
    $backendServerList = array($b1, $b2);
    $resp = $this->client->addBackendServer('lb-21e928b8', $backendServerList);
    print_r($resp);
    $this->assertNotNull($resp);
}

```

🔗 用端口查询后端服务器实例

使用以下代码可以查询后端服务器实例：

```

/**
 * list backend servers mounted with blb instance. Health Status will be presented while listener port is necessary
 *
 * @param string $blbId
 *      blb instance id
 *
 * @param int $listenerPort
 *      port of listener
 *
 * @param string $marker
 *      The optional parameter marker specified in the original request to specify
 *      where in the results to begin listing.
 *
 * @param int $maxKeys
 *      The optional parameter to specifies the max number of list result to return.
 *      The default value is 1000.
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function listBackendServersWithPort($blbId, $listenerPort, $marker = null, $maxKeys = 1000, $clientToken =
null, $options = array())
{

}

```

使用示例如下：

```

public function testListBackendServersWithPort()
{
    $resp = $this->client->listBackendServersWithPort('lb-21e928b8', 80);
    print_r($resp);
    $this->assertNotNull($resp);
}

```

🔗 查询后端服务器实例

使用以下代码可以查询后端服务器实例：

```

/**
 * list backend servers mounted with blb instance
 *
 * @param string $blbld
 *      blb instance id
 *
 * @param string $marker
 *      The optional parameter marker specified in the original request to specify
 *      where in the results to begin listing.
 *
 * @param int $maxKeys
 *      The optional parameter to specifies the max number of list result to return.
 *      The default value is 1000.
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function listBackendServers($blbld, $marker = null, $maxKeys = 1000, $clientToken = null, $options = array())
{
}

```

使用示例如下：

```

public function testListBackendServers()
{
    $resp = $this->client->listBackendServers('lb-21e928b8');
    print_r($resp);
    $this->assertNotNull($resp);
}

```

更新后端服务器实例

使用以下代码可以更新后端服务器实例：


```

/**
 * update weight of backend servers
 *
 * @param string $blbId
 *      blb instance id
 *
 * @param array $backendServerList
 *      back end servers to be updated
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function updateBackendServer($blbId, $backendServerList, $clientToken = null, $options = array())
{
}

```

使用示例如下：

```

public function testUpdateServers()
{
    $b1 = new BackendServer('i-xtsPZTGY', 20);
    $b2 = new BackendServer('i-Xf3hfqWt', 31);
    $backendServerList = array($b1, $b2);
    $resp = $this->client->updateBackendServer('lb-21e928b8', $backendServerList);
    print_r($resp);
    $this->assertNotNull($resp);
}

```

🔗 移除后端服务器实例

使用以下代码可以移除后端服务器实例：

```
/**
 * remove backend servers
 *
 * @param string $blbld
 *      blb instance id
 *
 * @param array $backendServerList
 *      back end servers to be removed
 *
 * @param string $clientToken
 *      if the clientToken is not specified by the user, a random String
 *      generated by default algorithm will be used.
 *
 * @param array $options
 *      The optional bce configuration, which will overwrite the
 *      default configuration that was passed while creating EipClient instance.
 *
 * @return mixed
 */
public function removeBackendServers($blbld, $backendServerList, $clientToken = null, $options = array())
{

}
```

使用示例如下：

```
public function testRemoveServers()
{
    $resp = $this->client->removeBackendServers('lb-21e928b8', array('i-xtsPZTGY', 'i-Xf3hfqWt'));
    print_r($resp);
    $this->assertNotNull($resp);
}
```

GO-SDK

概述

本文档主要介绍BLB GO-SDK的安装和使用。在使用本文档前，您需要先了解BLB的一些基本知识，并已经开通了BLB服务。若您还不了解BLB，可以参考[产品描述](#)和[快速入门](#)。

安装

运行环境

1. 运行环境 GO SDK可以在go1.3及以上环境下运行。

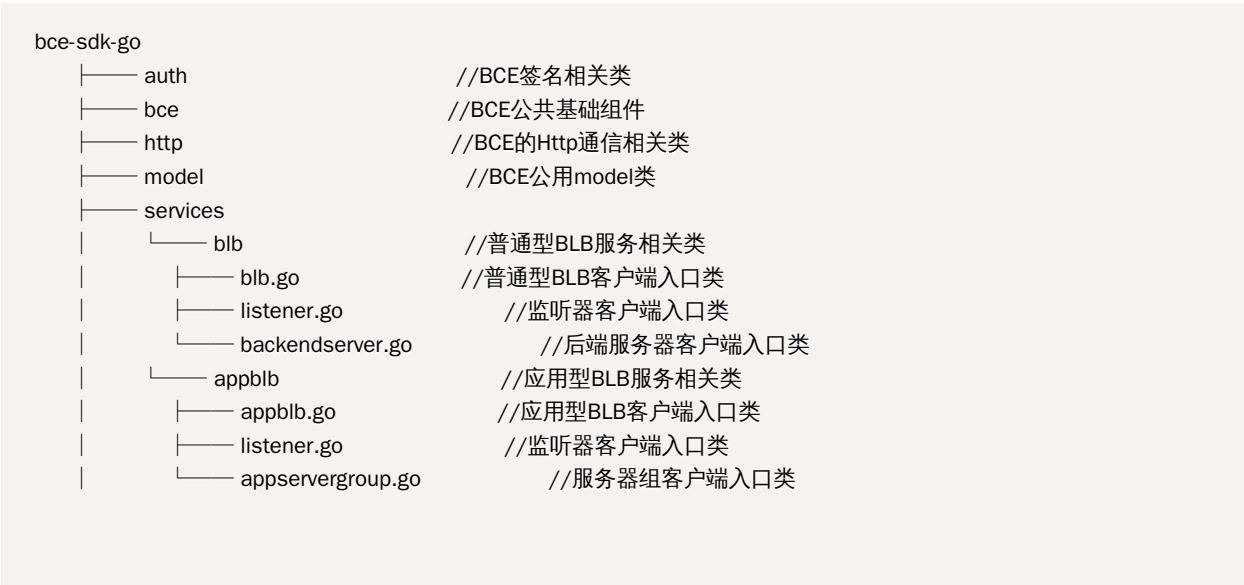
🔗 安装SDK

直接从github下载

使用go get工具从github进行下载：

```
go get github.com/baidubce/bce-sdk-go
```

SDK目录结构



🔗 卸载SDK

预期卸载 SDK 时，删除下载的源码即可。

初始化

🔗 初始化

🔗 确认Endpoint

在确认您使用SDK时配置的Endpoint时，可先阅读开发人员指南中关于[BLB访问域名](#)的部分，理解Endpoint相关的概念。 百度云目前开放了多区域支持，请参考[区域选择说明](#)。

目前支持“北京”、“广州”、“苏州”、“香港”、“武汉”、“保定”、“上海”、“新加坡”等区域。

服务域名的对应信息为：

访问区域	对应Endpoint
bj	blb.bj.baidubce.com
gz	blb.gz.baidubce.com
su	blb.su.baidubce.com
hkg	blb.hkg.baidubce.com
fwh	blb.fwh.baidubce.com
bd	blb.bd.baidubce.com
fsh	blb.fsh.baidubce.com
sin	blb.sin.baidubce.com

🔗 新建BlbClient

使用AK/SK新建BlbClient

通过AK/SK方式访问BOS，用户可以参考如下代码新建一个BosClient：

```
import (
    "github.com/baidubce/bce-sdk-go/services/blb"
    "github.com/baidubce/bce-sdk-go/services/appblb"
)
func main() {
    // 用户的Access Key ID和Secret Access Key
    ACCESS_KEY_ID, SECRET_ACCESS_KEY := <ak>, <sk>

    // 用户指定的Endpoint
    ENDPOINT := <domain-name>

    // 初始化一个BlbClient
    blbClient, err := blb.NewClient(AK, SK, ENDPOINT)
    // 初始化一个AppblbClient
    appblbClient, err := appblb.NewClient(AK, SK, ENDPOINT)
}
```

在上面代码中，ACCESS_KEY_ID对应控制台中的“Access Key ID”，SECRET_ACCESS_KEY对应控制台中的“Access Key Secret”，获取方式请参考《操作指南 [管理ACCESSKEY](#)》。

>**注意：**`ENDPOINT`参数只能用指定的包含区域的域名来进行定义，不指定时默认为北京区域`blb.bj.baidubce.com`。

普通型blb实例

创建BLB

<!--1、*必选，一句话描述函数功能。-->

创建一个BLB实例，返回分配的服务地址及实例ID。付费方式为后付费。需要实名认证，若未通过实名认证可以前往百度智能云官网控制台中的安全认证下的实名认证中进行认证

使用以下代码可以创建BLB实例

****函数声明****

```
func (c Client) CreateLoadBalancer(args CreateLoadBalancerArgs) (*CreateLoadBalancerResult, error)
```

****参数含义****

请参考OpenAPI文档：[CreateLoadBalancer创建实例]

(<https://cloud.baidu.com/doc/BLB/s/njwxnv79#createloadbalancer%E5%88%9B%E5%BB%BA%E5%AE%9E%E4%BE%8B>)

****返回值****

操作成功：

```
{ "blbId": "lb-BLuOPSLZ", "name": "blb-for-test", "address": "192.168.0.24", "desc": "myblb" }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxzw6)

****代码示例****

具体代码示例参考：[example_create_loadbalancer.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_loadbalancer.go)

查询已有BLB实例

<!--1、*必选，一句话描述函数功能。-->

查询用户账户下所有LoadBalancer的信息，支持按LoadBalancer的Name、Address进行模糊查询，支持按LoadBalancer的ID进行精确查询，支持按bccld进行查询，目前只支持单个bccld查询，并且仅支持精确查询。查询结果为包含该后端服务器的负载均衡实例。返回结果是多重查询条件的交集。如果BLB实例开启了公网访问，返回结果将会带上公网IP。结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询已有BLB实例

****函数声明****

```
func (c Client) DescribeLoadBalancers(args DescribeLoadBalancersArgs) (*DescribeLoadBalancersResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeLoadBalancers查询已有的BLB实例]

(https://cloud.baidu.com/doc/BLB/s/njwvxv79#describeloadbalancers%E6%9F%A5%E8%AF%A2%E5%B7%B2%E6%9C%89)

****返回值****

操作成功：

```
{ "blbList": [ { "blbld": "lb-a7e5zPPk", "status": available, "name": "test-blb", "desc": desc, "address": "192.168.0.23" }, {
"blbld": "lb-gj5gVpeq", "status": available, "name": "nametest", "desc": desc, "address": "192.168.0.24", "publicIp":
"123.2.3.4" } ], "marker": "blb-0A20F971", "nextMarker": "blb-0A20FB09", "isTruncated": true, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxzw6)

****代码示例****

具体代码示例参考：[example_describe_loadbalancers.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_loadbalancers.go)

查询BLB实例详情

<!--1、*必选，一句话描述函数功能。-->

按ID查询用户账户下所有LoadBalancer的详细信息，包含LoadBalancer所有的监听器端口信息

使用以下代码可以查询BLB实例详情

****函数声明****

```
func (c Client) DescribeLoadBalancerDetail(blbld string) (DescribeLoadBalancerDetailResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeLoadBalancerDetail查询BLB实例详情]

(<https://cloud.baidu.com/doc/BLB/s/njwvxnv79#describeloadbalancerdetail%E6%9F%A5%E8%AF%A2blb%E5%AE%9E%E4%E>)

****返回值****

操作成功：

```
{ "blbId": "lb-gj5gVpeq", "status": "available", "name": "nametest", "desc": "",
  "address": "192.168.0.2", "publicIp": "123.2.3.4", "cidr": "192.168.0.0/16", "vpcName": "test",
  "createTime": "2019-03-07T02:35:31Z",
  "releaseTime": "2019-03-07T02:35:31Z",
  "listener": [
    { "port": 80,
      "backendPort": 90,
      "type": "TCP",
      "healthCheckType": "TCP",
      "healthCheckStatus": "Alive"
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_describe_loadbalancer_detail.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_loadbalancer_detail.go)

更新BLB

<!--1、*必选，一句话描述函数功能。-->

更新指定的单个LoadBalancer配置信息，包括LoadBalancer实例的名称和描述

使用以下代码可以更新BLB实例

****函数声明****

```
func (c Client) UpdateLoadBalancer(blbId string, args UpdateLoadBalancerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateLoadBalancer更新实例]

(<https://cloud.baidu.com/doc/BLB/s/njwxnv79#updateloadbalancer%E6%9B%B4%E6%96%B0%E5%AE%9E%E4%BE%8B>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_update_loadbalancer.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_update_loadbalancer.go)

删除BLB

<!--1、*必选，一句话描述函数功能。-->

释放指定LoadBalancer，被释放的LoadBalancer无法找回

使用以下代码可以删除BLB实例

****函数声明****

```
func (c *Client) DeleteLoadBalancer(blbld string) error
```

****参数含义****

请参考OpenAPI文档：[DeleteLoadBalancer释放BLB实例]

(<https://cloud.baidu.com/doc/BLB/s/njwxnv79#deleteloadbalancer%E9%87%8A%E6%94%BEblb%E5%AE%9E%E4%BE%8B>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_delete_loadbalancer.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_delete_loadbalancer.go)

更新BLB ACL开关

<!--1、*必选，一句话描述函数功能。-->

更新指定的LoadBalancer是否支持ACL功能，支持更新类型（普通型、应用型、IPv6型）

使用以下代码可以更新BLB ACL开关

****函数声明****

```
func (c Client) UpdateLoadBalancerAcl(blbld string, args UpdateLoadBalancerAclArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateLoadBalancerAcl更新实例ACL功能]

(<https://cloud.baidu.com/doc/BLB/s/njwvxnv79#updateloadbalanceracl%E6%9B%B4%E6%96%B0%E5%AE%9E%E4%BE%8B>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_update_loadbalancer_acl.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_update_loadbalancer_acl.go)

创建TCP监听器

在指定LoadBalancer下，创建一个基于TCP协议的监听器，监听一个前端端口，将发往该端口的所有TCP流量，转发到后端服务器监听的后端端口上。TCP监听器只支持TCP协议的健康检查，检查端口和后端服务器监听端口一致。

使用以下代码创建TCP监听器

****函数声明****

```
func (c Client) CreateTCPListener(blbId string, args CreateTCPListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateTCPListener创建TCP监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnv16#createtcplistener%E5%88%9B%E5%BB%BAtcp%E7%9B%91%E5%90%AC%E5>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_create_tcp_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_tcp_listener.go)

创建UDP监听器

在指定LoadBalancer下，创建一个基于UDP协议的监听器，监听一个前端端口，将发往该端口的所有UDP流量，转发到后端服务器监听的后端端口上。UDP监听器支持UDP、ICMP协议的健康检查，检查端口和后端服务器监听端口一致。

使用以下代码创建UDP监听器

****函数声明****


```
func (c Client) CreateUDPLListener(bld string, args CreateUDPLListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateUDPLListener创建UDP监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnl6#createudplistener%E5%88%9B%E5%BB%BAudp%E7%9B%91%E5%90%AC%E>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_create_udp_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_udp_listener.go)

创建HTTP监听器

在指定LoadBalancer下，创建一个基于HTTP协议的监听器，监听一个前端端口，将发往该端口的所有HTTP请求，转发到后端服务器监听的后端端口上。

使用以下代码创建HTTP监听器

****函数声明****

```
func (c Client) CreateHTTPListener(bld string, args CreateHTTPListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateHTTPListener创建HTTP监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#createhttplistener%E5%88%9B%E5%BB%BAhttp%E7%9B%91%E5%90%AC%I>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_create_http_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_http_listener.go)

创建HTTPS监听器

在指定LoadBalancer下，创建一个基于HTTPS协议的监听器，监听一个前端端口，将发往该端口的所有HTTPS请求，先通过SSL卸载转换为HTTP请求后，再转发到后端服务器监听的后端端口上。

使用以下代码创建HTTPS监听器

****函数声明****

```
func (c Client) CreateHTTPSListener(blbld string, args CreateHTTPSListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateHTTPSListener创建HTTPS监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#createhttpslistener%E5%88%9B%E5%BB%BAhttps%E7%9B%91%E5%90%AC%I>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_create_https_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_https_listener.go)

创建SSL监听器

在指定LoadBalancer下，创建一个基于SSL协议的监听器，监听一个前端端口，将发往该端口的所有SSL流量，通过SSL卸载转换为TCP请求后转发到后端服务器监听的后端端口上。SSL监听器只支持TCP协议的健康检查，检查端口和后端服务器监听端口一致。

使用以下代码创建SSL监听器

****函数声明****

```
func (c Client) CreateSSLListener(blbid string, args CreateSSLListenerArgs) error
```

参数含义

请参考OpenAPI文档：[CreateSSLListener创建SSL监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#createssllistener%E5%88%9B%E5%BB%BAss%E7%9B%91%E5%90%AC%E5>)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

代码示例

具体代码示例参考：[example_create_ssl_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_ssl_listener.go)

查询TCP监听器

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有TCP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询tcp监听器

函数声明

```
func (c Client) DescribeTCPListeners(blbid string, args DescribeListenerArgs) (*DescribeTCPListenersResult, error)
```

参数含义

请参考OpenAPI文档：[DescribeTCPListeners查询TCP监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#describetcplisteners%E6%9F%A5%E8%AF%A2tcp%E7%9B%91%E5%90%AC>)

返回值

操作成功：

```
{ "listenerList": [ { "listenerPort": 80, "backendPort": 80, "scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 3,
"healthCheckInterval": 3, "unhealthyThreshold": 3, "healthyThreshold": 3 }, { "listenerPort": 88, "backendPort": 88,
"scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 2, "healthCheckInterval": 4, "unhealthyThreshold": 3,
"healthyThreshold": 3 } ], "marker": "listener-0050", "nextMarker": null, "isTruncated": false, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxnz6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxnz6)

****代码示例****

具体代码示例参考：[\[example_describe_tcp_listeners.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_tcp_listeners.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_tcp_listeners.go)

查询UDP监听器

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有UDP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询UDP监听器

****函数声明****

```
func (c Client) DescribeUDPListeners(blbld string, args DescribeListenerArgs) (*DescribeUDPListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[DescribeUDPListeners查询UDP监听器\]](https://cloud.baidu.com/doc/BLB/s/yjwvxnl6#describeudpllisteners%E6%9F%A5%E8%AF%A2udp%E7%9B%91%E5%90%AC)

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnl6#describeudpllisteners%E6%9F%A5%E8%AF%A2udp%E7%9B%91%E5%90%AC>)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 80, "backendPort": 80, "scheduler": "RoundRobin", "healthCheckType": "UDP",
"healthCheckTimeoutInSecond": 3, "healthCheckInterval": 3, "unhealthyThreshold": 3, "healthyThreshold": 3,
"healthCheckString":"healthCheck", "udpSessionTimeout":90 }, { "listenerPort": 88, "backendPort": 88, "scheduler":
"RoundRobin", "healthCheckType": "UDP", "healthCheckTimeoutInSecond": 2, "healthCheckInterval": 4,
"unhealthyThreshold": 3, "healthyThreshold": 3, "healthCheckString":"healthCheck", "udpSessionTimeout": 90 } ], "marker":
"listener-0050", "nextMarker": null, "isTruncated": false, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxnz6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxnz6)

****代码示例****

具体代码示例参考：[\[example_describe_udp_listeners.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_udp_listeners.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_udp_listeners.go)

查询HTTP监听器

查询指定LoadBalancer下所有HTTP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询应用型HTTP监听器信息

****函数声明****

```
func (c Client) DescribeHTTPListeners(blbld string, args DescribeListenerArgs) (*DescribeHTTPListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeHTTPListeners查询HTTP监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#describehttplisteners%E6%9F%A5%E8%AF%A2http%E7%9B%91%E5%90%AC>)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 90, "backendPort": 90, "scheduler": "LeastConnection", "healthCheckTimeoutInSecond": 4,
"healthCheckInterval": 5, "unhealthyThreshold": 2, "healthyThreshold": 3, "keepSession": false, "keepSessionType": null,
"keepSessionDuration": null, "keepSessionCookieName": null, "xForwardFor": false, "healthCheckType": "HTTP",
"healthCheckPort": 90, "healthCheckURI": "/", "healthCheckNormalStatus": "http_2xx|http_3xx", "serverTimeout": 30,
"redirectPort": 80 }, { "listenerPort": 95, "backendPort": 95, "scheduler": "LeastConnection", "healthCheckTimeoutInSecond":
4, "healthCheckInterval": 5, "unhealthyThreshold": 2, "healthyThreshold": 3, "keepSession": false, "keepSessionType": null,
"keepSessionDuration": null, "keepSessionCookieName": null, "xForwardFor": false, "healthCheckType": "HTTP",
"healthCheckPort": 95, "healthCheckURI": "/", "healthCheckNormalStatus": "http_1xx|http_2xx", "serverTimeout": 1800,
"redirectPort": 80 } ], "marker": "listener-005A", "nextMarker": "listener-0322", "isTruncated": true, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_describe_http_listeners.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_http_listeners.go)

****查询HTTPS监听器****

查询指定LoadBalancer下所有HTTPS监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询应用型HTTPS监听器信息

****函数声明****

```
func (c Client) DescribeHTTPSListeners(blbId string, args DescribeListenerArgs) (*DescribeHTTPSListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeHTTPSListeners查询HTTPS监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#describehttpslisteners%E6%9F%A5%E8%AF%A2https%E7%9B%91%E5%90%AC>)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 90, "backendPort": 90, "scheduler": "LeastConnection", "healthCheckTimeoutInSecond": 4,
"healthCheckInterval": 5, "unhealthyThreshold": 2, "healthyThreshold": 3, "keepSession": false, "keepSessionType": null,
"keepSessionDuration": null, "keepSessionCookieName": null, "xForwardFor": false, "healthCheckType": "HTTP",
"healthCheckPort": 90, "healthCheckURI": "/", "healthCheckNormalStatus": "http_2xx|http_3xx", "serverTimeout": 30,
"certIds": ["cert-zfj2ey2z4kmm"] }, { "listenerPort": 95, "backendPort": 95, "scheduler": "LeastConnection",
```

```
"healthCheckTimeoutInSecond": 4, "healthCheckInterval": 5, "unhealthyThreshold": 2, "healthyThreshold": 3, "keepSession":
false, "keepSessionType": null, "keepSessionDuration": null, "keepSessionCookieName": null, "xForwardFor": false,
"healthCheckType": "HTTP", "healthCheckPort": 95, "healthCheckURI": "/", "healthCheckNormalStatus": "http_1xx|http_2xx",
"serverTimeout": 1800, "certIds": [{"cert-zfj2ey2z4kmm"}] }, "marker": "listener-005A", "nextMarker": "listener-0322",
"isTruncated": true, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxnz6)

****代码示例****

具体代码示例参考：[example_describe_https_listeners.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_https_listeners.go)

查询SSL监听器

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有SSL监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询SSL监听器

****函数声明****

```
func (c Client) DescribeSSLListeners(blbd string, args DescribeListenerArgs) (*DescribeSSLListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeSSLListeners查询SSL监听器]

(https://cloud.baidu.com/doc/BLB/s/yjwvxnv16#describessllisteners%E6%9F%A5%E8%AF%A2ssl%E7%9B%91%E5%90%AC%)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 80, "backendPort": 80, "scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 3,
"healthCheckInterval": 3, "unhealthyThreshold": 3, "healthyThreshold": 3 }, { "listenerPort": 88, "backendPort": 88,
"scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 2, "healthCheckInterval": 4, "unhealthyThreshold": 3,
"healthyThreshold": 3 } ], "marker": "listener-0050", "nextMarker": null, "isTruncated": false, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxznw6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxznw6)

****代码示例****

具体代码示例参考：[\[example_describe_ssl_listeners.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_ssl_listeners.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_ssl_listeners.go)

查询所有监听器

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询所有监听器

****函数声明****

```
func (c Client) DescribeAllListeners(blbl string, args DescribeListenerArgs) (*DescribeAllListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[DescribeAllListeners查询所有监听\]](https://cloud.baidu.com/doc/BLB/s/yjwvxnl6#describealllisteners%E6%9F%A5%E8%AF%A2%E6%89%80%E6%9C%89%E7%9)

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnl6#describealllisteners%E6%9F%A5%E8%AF%A2%E6%89%80%E6%9C%89%E7%9>)

****返回值****

操作成功：

```
{ "nextMarker": "", "marker": "", "maxKeys": 1, "isTruncated": true, "listenerList": [ { "listenerPort": 53, "backendPort": 53,
"scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 3, "healthCheckInterval": 3, "unhealthyThreshold": 3,
"healthyThreshold": 3, "getBlblp": false, "listenerType": "UDP", "udpSessionTimeout": 90, "healthCheckString":
"\00\01\01\00\00\01\00\00\00\00\00\00\05baidu\03com\00\00\01\00\01", "healthCheckType": "UDP" }, {
"listenerPort": 82, "backendPort": 80, "scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 3, "healthCheckInterval":
3, "unhealthyThreshold": 3, "healthyThreshold": 3, "getBlblp": false, "listenerType": "TCP", "tcpSessionTimeout": 900 }, {
"listenerPort": 87, "backendPort": 80, "scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 3, "healthCheckInterval":
3, "unhealthyThreshold": 3, "healthyThreshold": 3, "getBlblp": false, "listenerType": "HTTP", "keepSession": false,
"xForwardFor": true, "healthCheckType": "HTTP", "healthCheckPort": 80, "healthCheckURI": "/", "healthCheckNormalStatus":
"http_2xx|http_3xx", "healthCheckHost": "", "serverTimeout": 30, "redirectPort": 0 }, { "listenerPort": 443, "backendPort": 80,
"scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 3, "healthCheckInterval": 3, "unhealthyThreshold": 3,
"healthyThreshold": 3, "getBlblp": false, "listenerType": "HTTPS", "keepSession": false, "xForwardFor": true,
"healthCheckType": "HTTP", "healthCheckPort": 80, "healthCheckURI": "/", "healthCheckNormalStatus": "http_2xx|http_3xx",
"healthCheckHost": "", "serverTimeout": 30, "certIds": [ "cert-gs8bktrm7drp" ], "dualAuth": false, "encryptionType":
"tls_cipher_policy_1_2_secure", "encryptionProtocols": [ "tlsv12" ], "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-
SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-
AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-
AES256-SHA:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES128-SHA" }, { "listenerPort":
446, "backendPort": 80, "scheduler": "RoundRobin", "healthCheckTimeoutInSecond": 3, "healthCheckInterval": 3,
"unhealthyThreshold": 3, "healthyThreshold": 3, "getBlblp": false, "listenerType": "SSL", "certIds": [ "cert-gs8bktrm7drp" ],
"dualAuth": true, "clientCertIds": [ "cert-Ox037gwe4fmg" ], "encryptionType": "tls_cipher_policy_default",
"encryptionProtocols": [ "tlsv10", "tlsv11", "tlsv12" ], "appliedCiphers": "ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-
ECDSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-
```

```
SHA256:ECDSA-AES128-SHA:ECDSA-AES256-GCM-SHA384:ECDSA-AES256-SHA384:ECDSA-AES256-SHA:ECDSA-AES128-GCM-SHA256:ECDSA-AES128-SHA256:ECDSA-AES128-SHA:AES128-GCM-SHA256:AES256-SHA:AES128-SHA:AES256-GCM-SHA384:AES256-SHA256:AES128-SHA256:DES-CBC3-SHA" ] }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxzw6)

****代码示例****

具体代码示例参考：[\[example_describe_all_listeners.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_all_listeners.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_all_listeners.go)

更新TCP监听器

更新指定LoadBalancer下的TCP监听器参数，请求参数中所有指定的字段都会被更新，未指定的字段保持不变，监听器通过端口指定。

使用以下代码更新TCP监听器

****函数声明****

```
func (c Client) UpdateTCPListener(blbd string, args UpdateTCPListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[\[UpdateTCPListener更新TCP监听器\]](https://cloud.baidu.com/doc/BLB/s/yjwvxvnl6#updatetcp listener%E6%9B%B4%E6%96%B0tcp%E7%9B%91%E5%90%AC%E)

(<https://cloud.baidu.com/doc/BLB/s/yjwvxvnl6#updatetcp listener%E6%9B%B4%E6%96%B0tcp%E7%9B%91%E5%90%AC%E>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxzw6)

****代码示例****

具体代码示例参考：[\[example_update_tcp_listener.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_update_tcp_listener.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_update_tcp_listener.go)

更新UDP监听器

更新指定LoadBalancer下的UDP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新UDP监听器

****函数声明****

```
func (c Client) UpdateUDPListener(blbd string, args UpdateUDPListenerArgs) error
```


****参数含义****

请参考OpenAPI文档：[UpdateUDPListener更新UDP监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#updateudplistener%E6%9B%B4%E6%96%B0udp%E7%9B%91%E5%90%AC%>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_update_udp_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blk/example_update_udp_listener.go)

更新HTTP监听器

更新指定LoadBalancer下的HTTP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新HTTP监听器

****函数声明****

```
func (c Client) UpdateHTTPListener(blId string, args UpdateHTTPListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateHTTPListener更新HTTP监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#updatehttplistener%E6%9B%B4%E6%96%B0http%E7%9B%91%E5%90%AC%>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_update_http_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blk/example_update_http_listener.go)

更新HTTPS监听器

更新指定LoadBalancer下的HTTPS监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新HTTPS监听器

****函数声明****

```
func (c Client) UpdateHTTPSListener(blbl string, args UpdateHTTPSListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateHTTPSListener更新HTTPS监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#updatehttpslistener%E6%9B%B4%E6%96%B0https%E7%9B%91%E5%90%A>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_update_https_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_update_https_listener.go)

更新SSL监听器

更新指定LoadBalancer下的SSL监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新SSL监听器

****函数声明****

```
func (c Client) UpdateSSLListener(blbl string, args UpdateSSLListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateSSLListener更新SSL监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnvI6#updatessllistener%E6%9B%B4%E6%96%B0ssl%E7%9B%91%E5%90%AC%E>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_update_ssl_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_update_ssl_listener.go)

释放监听器

释放指定LoadBalancer下的监听器，监听器通过监听端口来指定；支持批量释放

使用以下代码可以释放监听器

****函数声明****

```
func (c Client) DeleteListeners(blbl string, args DeleteListenersArgs) error
```

****参数含义****

请参考OpenAPI文档：[DeleteListeners释放监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnl6#deletelisteners%E9%87%8A%E6%94%BE%E7%9B%91%E5%90%AC%E5%99%>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_delete_listeners.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_delete_listeners.go)

添加后端服务器

为指定LoadBalancer添加后端服务器，支持批量添加

使用以下代码可以添加后端服务器

****函数声明****

```
func (c Client) AddBackendServers(blbl string, args AddBackendServersArgs) error
```

****参数含义****

请参考OpenAPI文档：[AddBackendServers添加后端服务器]

(<https://cloud.baidu.com/doc/BLB/s/Ujwvxnx6#addbackendservers%E6%B7%BB%E5%8A%A0%E5%90%8E%E7%AB%AF%E6%>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_add_backend_servers.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_add_backend_servers.go)

查询后端服务器健康状态

查询指定LoadBalancer下的后端服务器信息，需要指定监听端口，结果会返回后端服务器的健康状态；结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询后端服务器健康状态

****函数声明****

```
func (c Client) DescribeHealthStatus(blbId string, args DescribeHealthStatusArgs) (*DescribeHealthStatusResult, error)
```

参数含义

请参考OpenAPI文档：[DescribeHealthStatus查询后端服务器健康状态]

(<https://cloud.baidu.com/doc/BLB/s/Ujwvxnvxe#describehealthstatus%E6%9F%A5%E8%AF%A2%E5%90%8E%E7%AB%AF%E>)

返回值

操作成功：

```
{ "backendServerList": [ { "instanceId": "i-YfAixxx", "weight": 50, "status": "Alive" } ], "type": "TCP", "listenerPort": 80, "backendPort": 80, "marker": "rs-0A6BE9BB", "isTruncated": false, "maxKeys": 1000 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

代码示例

具体代码示例参考：[example_describe_health_status.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_health_status.go)

查询后端服务器列表

查询指定LoadBalancer下的后端服务器列表；结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询后端服务器列表

函数声明

```
func (c Client) DescribeBackendServers(blbId string, args DescribeBackendServersArgs) (*DescribeBackendServersResult, error)
```

参数含义

请参考OpenAPI文档：[DescribeBackendServers查询后端服务器列表]

(<https://cloud.baidu.com/doc/BLB/s/Ujwvxnvxe#describebackendservers%E6%9F%A5%E8%AF%A2%E5%90%8E%E7%AB%AF%E>)

返回值

操作成功：

```
{ "backendServerList": [ { "instanceId": "i-YfAixxx", "weight": 50, } ], "marker": "rs-0A6BE9BB", "isTruncated": false, "maxKeys": 1000 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_describe_backend_servers.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blk/example_describe_backend_servers.go)

更新后端服务器

更新指定LoadBalancer下的后端服务器权重，支持批量修改

使用以下代码可以更新后端服务器

****函数声明****

```
func (c Client) UpdateBackendServers(blbId string, args UpdateBackendServersArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateBackendServers更新后端服务器]

(https://cloud.baidu.com/doc/BLB/s/Ujwvxnvxe#updatebackendservers%E6%9B%B4%E6%96%B0%E5%90%8E%E7%AB%AF)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_update_backend_servers.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blk/example_update_backend_servers.go)

释放后端服务器

释放指定LoadBalancer下的后端服务器，后端服务器通过其标识符指定；支持批量释放

使用以下代码可以释放后端服务器

****函数声明****

```
func (c Client) RemoveBackendServers(blbId string, args RemoveBackendServersArgs) error
```

****参数含义****

请参考OpenAPI文档：[RemoveBackendServers释放后端服务器]

(<https://cloud.baidu.com/doc/BLB/s/Ujwvxnvxe#removebackendservers%E9%87%8A%E6%94%BE%E5%90%8E%E7%AB%AF>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_remove_backend_servers.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_remove_backend_servers.go)

绑定普通安全组

<!--1、*必选，一句话描述函数功能。-->

为指定LoadBalancer（普通型、应用型、IPv6型）绑定普通安全组，支持批量绑定

使用以下代码可以为指定LoadBalancer绑定普通安全组

****函数声明****

```
func (c Client) BindSecurityGroups(blbl string, args UpdateSecurityGroupsArgs) error
```

****参数含义****

请参考OpenAPI文档：[BindSecurityGroups绑定普通安全组]

(<https://cloud.baidu.com/doc/BLB/s/VI7ygu1rk#bindsecuritygroups%E7%BB%91%E5%AE%9A%E6%99%AE%E9%80%9A%E5>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_bind_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_bind_security_groups.go)

解绑普通安全组

<!--1、*必选，一句话描述函数功能。-->

为指定LoadBalancer（普通型、应用型、IPv6型）解绑普通安全组，支持批量解绑

使用以下代码可以为指定LoadBalancer解绑普通安全组

****函数声明****

```
func (c Client) UnbindSecurityGroups(blbld string, args UpdateSecurityGroupsArgs) error
```

****参数含义****

请参考OpenAPI文档：[UnbindSecurityGroups解绑普通安全组]
(<https://cloud.baidu.com/doc/BLB/s/VI7ygu1rk#unbindsecuritygroups%E8%A7%A3%E7%BB%91%E6%99%AE%E9%80%9A%E>
)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_unbind_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_unbind_security_groups.go)

查询BLB实例普通安全组列表

<!--1、*必选，一句话描述函数功能。-->

按ID查询用户账户下指定LoadBalancer的普通安全组信息

使用以下代码可以按ID查询用户账户下指定LoadBalancer的普通安全组信息

****函数声明****

```
func (c Client) DescribeSecurityGroups(blbld string) (DescribeSecurityGroupsResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeSecurityGroups查询BLB实例普通安全组列表]
(<https://cloud.baidu.com/doc/BLB/s/VI7ygu1rk#describesecuritygroups%E6%9F%A5%E8%AF%A2blb%E5%AE%9E%E4%BE%E>
)

****返回值****

操作成功：

```
{ "blbSecurityGroups": [ { "securityGroupDesc": "", "securityGroupId": "g-4NxWoxeq", "securityGroupName": "securitygroup-name", "vpcName": "vpc-name", "securityGroupRules": [ { "destGroupId": "", "destIp": "all", "direction": "egress", "ethertype": "IPv4", "portRange": "1-65535", "protocol": "all", "securityGroupRuleId": "r-gkv8yupumvx2" } ] } ] }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_describe_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blk/example_describe_security_groups.go)

绑定企业安全组

<!--1、*必选，一句话描述函数功能。-->

为指定LoadBalancer（普通型、应用型、IPv6型）绑定企业安全组，支持批量绑定

使用以下代码可以为指定LoadBalancer绑定企业安全组

****函数声明****

```
func (c Client) BindEnterpriseSecurityGroups(blbid string, args UpdateEnterpriseSecurityGroupsArgs) error
```

****参数含义****

请参考OpenAPI文档：[BindEnterpriseSecurityGroups绑定企业安全组]

(https://cloud.baidu.com/doc/BLB/s/Vl7ygu1rk#bindenterprisesecuritygroups%E7%BB%91%E5%AE%9A%E4%BC%81%E4%B)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_bind_enterprise_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blk/example_bind_enterprise_security_groups.go)

解绑企业安全组

<!--1、*必选，一句话描述函数功能。-->

为指定LoadBalancer（普通型、应用型、IPv6型）解绑企业安全组，支持批量解绑

使用以下代码可以为指定LoadBalancer解绑企业安全组

****函数声明****

```
func (c Client) UnbindEnterpriseSecurityGroups(blbid string, args UpdateEnterpriseSecurityGroupsArgs) error
```


****参数含义****

请参考OpenAPI文档：[\[UnbindEnterpriseSecurityGroups解绑企业安全组\]](#)

(<https://cloud.baidu.com/doc/BLB/s/Vl7ygu1rk#unbindenterpriseSecurityGroups%E8%A7%A3%E7%BB%91%E4%BC%81%E4%B6%9C>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_unbind_enterprise_security_groups.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_unbind_enterprise_security_groups.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_unbind_enterprise_security_groups.go)

查询bib实例企业安全组列表

<!--1、*必选，一句话描述函数功能。-->

按ID查询用户账户下指定LoadBalancer的企业安全组信息

使用以下代码可以按ID查询用户账户下指定LoadBalancer的企业安全组信息。

****函数声明****

```
func (c Client) DescribeEnterpriseSecurityGroups(bld string) (DescribeEnterpriseSecurityGroupsResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[DescribeEnterpriseSecurityGroups查询BLB实例企业安全组列表\]](#)

(<https://cloud.baidu.com/doc/BLB/s/Vl7ygu1rk#describeenterprisesecuritygroups%E6%9F%A5%E8%AF%A2b1b%E5%AE%9E>)

****返回值****

操作成功：

```
{ "enterpriseSecurityGroups": [{ "enterpriseSecurityGroupId": "esg-g8z4qf0m0bu", "enterpriseSecurityGroupName": "test0627", "enterpriseSecurityGroupDesc": "", "enterpriseSecurityGroupRules": [{ "remark": "全部协议", "direction": "egress", "action": "allow", "priority": 1000, "ethertype": "IPv6", "portRange": "", "sourceIp": "", "destIp": "all", "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu", "protocol": "all" }, { "remark": "", "direction": "egress", "action": "allow", "priority": 1000, "ethertype": "IPv4", "portRange": "", "sourceIp": "", "destIp": "all", "enterpriseSecurityGroupRuleId": "esgr-9je09z6bd9wv", "protocol": "tcp" }, { "remark": "", "direction": "ingress", "action": "allow", "priority": 1000, "ethertype": "IPv4", "portRange": "2", "sourceIp": "all", "destIp": "", "enterpriseSecurityGroupRuleId": "esgr-jcej4juqbh49", "protocol": "tcp" } ] }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxznw6)

****代码示例****

具体代码示例参考：[example_describe_enterprise_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_enterprise_security_groups.go)

应用型blb实例

创建BLB

<!--1、*必选，一句话描述函数功能。-->

创建一个BLB实例，返回分配的服务地址及实例ID。付费方式为后付费。需要实名认证，若未通过实名认证可以前往百度智能云官网控制台中的安全认证下的实名认证中进行认证

使用以下代码可以创建BLB实例

****函数声明****

```
func (c Client) CreateLoadBalancer(args CreateLoadBalancerArgs) (*CreateLoadBalanceResult, error)
```

****参数含义****

请参考OpenAPI文档：[CreateLoadBalancer创建实例]

(https://cloud.baidu.com/doc/BLB/s/6jwvxnyhi#createloadbalancer%E5%88%9B%E5%BB%BA%E5%AE%9E%E4%BE%8B)

****返回值****

操作成功：

```
{ "blbId": "lb-BLuOPSLZ", "name": "blb-for-test", "desc": "", "address": "192.168.0.24" }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxznw6)

****代码示例****

具体代码示例参考：[example_create_loadbalancer.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_create_loadbalancer.go)

更新BLB

<!--1、*必选，一句话描述函数功能。-->

更新指定的单个LoadBalancer配置信息，包括LoadBalancer实例的名称和描述

使用以下代码可以更新BLB实例

****函数声明****

```
func (c Client) UpdateLoadBalancer(blbId string, args UpdateLoadBalancerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateLoadBalancer更新实例]

(<https://cloud.baidu.com/doc/BLB/s/6jwvxnyhi#updateloadbalancer%E6%9B%B4%E6%96%B0%E5%AE%9E%E4%BE%8B>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_update_loadbalancer.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_update_loadbalancer.go)

查询已有BLB实例

<!--1、*必选，一句话描述函数功能。-->

查询用户账户下所有LoadBalancer的信息，支持按LoadBalancer的Name、Address进行模糊查询，支持按LoadBalancer的ID进行精确查询，支持按bccld进行查询，目前只支持单个bccld查询，并且仅支持精确查询。查询结果为包含该后端服务器的负载均衡实例。返回结果是多重查询条件的交集。如果BLB实例开启了公网访问，返回结果将会带上公网IP。结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询已有BLB实例

****函数声明****

```
func (c Client) DescribeLoadBalancers(args DescribeLoadBalancersArgs) (*DescribeLoadBalancersResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeLoadBalancers查询已有的BLB实例]

(<https://cloud.baidu.com/doc/BLB/s/6jwvxnyhi#describeloadbalancers%E6%9F%A5%E8%AF%A2%E5%B7%B2%E6%9C%89%>)

****返回值****

操作成功：

```
{ "blbList": [ { "blbId": "lb-a7e5zPPk", "status": available, "name": "test-blb", "desc": "用于生产环境", "vpclId": "vpc-fy6vdykpwkqb", "address": "192.168.0.24" }, { "blbId": "lb-gj5gVpeq", "status": available, "name": "nametest", "desc": "用于测试环境", "vpclId": "vpc-a8n5p6kybbx4", "address": "192.168.0.25" } ], "marker": "blb-0A20F971", "nextMarker": "blb-0A20FB09", "isTruncated": true, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_describe_loadbalancers.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_loadbalancers.go)

查询BLB实例详情

<!--1、*必选，一句话描述函数功能。-->

按ID查询用户账户下所有LoadBalancer的详细信息，包含LoadBalancer所有的监听器端口信息

使用以下代码可以查询BLB实例详情

****函数声明****

```
func (c Client) DescribeLoadBalancerDetail(blbId string) (DescribeLoadBalancerDetailResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeLoadBalancerDetail查询BLB实例详情]

(https://cloud.baidu.com/doc/BLB/s/6jwvxnyhi#describeloadbalancerdetail%E6%9F%A5%E8%AF%A2bIb%E5%AE%9E%E4%BI)

****返回值****

操作成功：

```
{ "blbId": "lb-gj5gVpeq",
  "status": "available",
  "name": "nametest",
  "desc": "",
  "address": "192.168.0.2",
  "publicIp": "123.2.3.4",
  "cidr": "192.168.0.0/16",
  "vpcName": "test",
  "subnetName": "系统预定义子网",
  "subnetCidr": "192.168.0.0/20",
  "createTime": "2019-03-07T02:35:31Z",
  "releaseTime": "2019-03-07T02:35:31Z",
  "listener": [
    { "port": 80,
      "type": "TCP"
    }
  ]
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_describe_loadbalancer_detail.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_loadbalancer_detail.go)

删除BLB

<!--1、*必选，一句话描述函数功能。-->

释放指定LoadBalancer，被释放的LoadBalancer无法找回

使用以下代码可以删除BLB实例

****函数声明****

```
func (c *Client) DeleteLoadBalancer(blbId string) error
```

****参数含义****

请参考OpenAPI文档：[DeleteLoadBalancer释放BLB实例]

(https://cloud.baidu.com/doc/BLB/s/6jwvxnyhi#deleteloadbalancer%E9%87%8A%E6%94%BEblb%E5%AE%9E%E4%BE%8B)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_delete_loadbalancer.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_delete_loadbalancer.go)

创建TCP监听器

在指定LoadBalancer下，创建一个基于TCP协议的应用型blb监听器，监听一个前端端口，将发往该端口的所有TCP流量，根据策略进行转发。

使用以下代码创建TCP监听器

****函数声明****

```
func (c Client) CreateAppTCPListener(blbId string, args CreateAppTCPListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateAppTCPListener创建TCP监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwxnyux#createapptcplistener%E5%88%9B%E5%BB%BAtcp%E7%9B%91%E5%90%AC>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_create_app_tcp_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_create_app_tcp_listener.go)

创建UDP监听器

在指定LoadBalancer下，创建一个基于UDP协议的应用型监听器，监听一个前端端口，将发往该端口的所有UDP流量，根据策略进行转发。

使用以下代码创建UDP监听器

****函数声明****

```
func (c Client) CreateAppUDPLListener(blblid string, args CreateAppUDPLListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateAppUDPLListener创建UDP监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwxnyux#createappudplistener%E5%88%9B%E5%BB%BAudp%E7%9B%91%E5%90%A>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_create_app_udp_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_create_app_udp_listener.go)

创建HTTP监听器

在指定LoadBalancer下，创建一个基于HTTP协议的应用型监听器，监听一个前端端口，将发往该端口的所有HTTP请求，根据策略转发到后端服务器监听的后端端口上。

使用以下代码创建HTTP监听器

****函数声明****

```
func (c Client) CreateAppHTTPListener(blbld string, args CreateAppHTTPListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateAppHTTPListener创建HTTP监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwxnyux#createapphttplistener%E5%88%9B%E5%BB%BAhttp%E7%9B%91%E5%90%/>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_create_app_http_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_create_app_http_listener.go)

创建HTTPS监听器

在指定LoadBalancer下，创建一个基于HTTPS协议的应用型监听器，监听一个前端端口，将发往该端口的所有HTTPS请求，先通过SSL卸载转换为HTTP请求后，再根据策略转发到后端服务器监听的后端端口上。

使用以下代码创建HTTPS监听器

****函数声明****

```
func (c Client) CreateAppHTTPSListener(blbld string, args CreateAppHTTPSListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateAppHTTPSListener创建HTTPS监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#createapphttpslistener%E5%88%9B%E5%BB%BAhttps%E7%9B%91%E5%9C>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_create_app_https_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_create_app_https_listener.go)

创建SSL监听器

在指定LoadBalancer下，创建一个基于SSL协议的应用型监听器，监听一个前端端口，将发往该端口的所有SSL流量，根据策略进行转发。

使用以下代码创建SSL监听器

****函数声明****

```
func (c Client) CreateAppSSLListener(blbld string, args CreateAppSSLListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateAppSSLListener创建SSL监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#createappssllistener%E5%88%9B%E5%BB%BAssl%E7%9B%91%E5%90%AC>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_create_app_ssl_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_create_app_ssl_listener.go)

更新TCP监听器

更新指定LoadBalancer下的TCP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新TCP监听器

****函数声明****


```
func (c Client) UpdateAppTCPListener(blbld string, args UpdateAppTCPListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateAppTCPListener更新TCP监听器]
(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#updateapptcp listener%E6%9B%B4%E6%96%B0tcp%E7%9B%91%E5%90%A>
)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_update_app_tcp_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_update_app_tcp_listener.go)

更新UDP监听器

更新指定LoadBalancer下的UDP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新UDP监听器

****函数声明****

```
func (c Client) UpdateAppUDPLListener(blbld string, args UpdateAppUDPLListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateAppUDPListener更新UDP监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwxnyux#updateappudplistener%E6%9B%B4%E6%96%B0udp%E7%9B%91%E5%90%>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_update_app_udp_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_update_app_udp_listener.go)

更新HTTP监听器

更新指定LoadBalancer下的HTTP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新HTTP监听器

****函数声明****

```
func (c Client) UpdateAppHTTPListener(blbld string, args UpdateAppHTTPListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateAppHTTPListener更新HTTP监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwxnyux#updateapphttplistener%E6%9B%B4%E6%96%B0http%E7%9B%91%E5%90%>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_update_app_http_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_update_app_http_listener.go)

更新HTTPS监听器

更新指定LoadBalancer下的HTTPS监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新HTTPS监听器

****函数声明****

```
func (c Client) UpdateAppHTTPSListener(blbl string, args UpdateAppHTTPSListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateAppHTTPSListener更新HTTPS监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnnyux#updateapphttpslistener%E6%9B%B4%E6%96%B0https%E7%9B%91%E5%90>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxznw6>)

****代码示例****

具体代码示例参考：[example_update_app_https_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_update_app_https_listener.go)

更新SSL监听器

更新指定LoadBalancer下的SSL监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码更新SSL监听器

****函数声明****

```
func (c Client) UpdateAppSSLListener(blbl string, args UpdateAppSSLListenerArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateAppSSLListener更新SSL监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#updateappssllistener%E6%9B%B4%E6%96%B0ssl%E7%9B%91%E5%90%A>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_update_app_ssl_listener.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_update_app_ssl_listener.go)

查询TCP监听器

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有TCP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询TCP监听器

****函数声明****

```
func (c Client) DescribeAppTCPListeners(blbld string, args DescribeAppListenerArgs) (*DescribeAppTCPListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeAppTCPListeners查询应用型TCP监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#describeapptcplisteners%E6%9F%A5%E8%AF%A2%E5%BA%94%E7%94%A8>)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 80, "scheduler": "RoundRobin" }, { "listenerPort": 88, "scheduler": "RoundRobin" } ],  
"marker": "listener-0050", "nextMarker": null, "isTruncated": false, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_describe_app_tcp_listeners.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_app_tcp_listeners.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_app_tcp_listeners.go)

查询UDP监听器

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有UDP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询UDP监听器

****函数声明****

```
func (c Client) DescribeAppUDPListeners(blbl string, args DescribeAppListenerArgs) (*DescribeAppUDPListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[DescribeAppUDPListeners查询UDP监听器\]](https://cloud.baidu.com/doc/BLB/s/ujwxnyux#describeappudplisteners%E6%9F%A5%E8%AF%A2udp%E7%9B%91%E5%90)

(<https://cloud.baidu.com/doc/BLB/s/ujwxnyux#describeappudplisteners%E6%9F%A5%E8%AF%A2udp%E7%9B%91%E5%90>)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 80, "scheduler": "RoundRobin", "udpSessionTimeout":90 }, { "listenerPort": 88, "scheduler": "RoundRobin", "udpSessionTimeout":90 } ], "marker": "listener-0050", "nextMarker": null, "isTruncated": false, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_describe_app_udp_listeners.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_app_udp_listeners.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_app_udp_listeners.go)

查询HTTP监听器

查询指定LoadBalancer下所有HTTP监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询应用型HTTP监听器信息

****函数声明****

```
func (c Client) DescribeAppHTTPListeners(blbl string, args DescribeAppListenerArgs) (*DescribeAppHTTPListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeHTTPListeners查询HTTP监听器]

(<https://cloud.baidu.com/doc/BLB/s/yjwvxnl6#describehttplisteners%E6%9F%A5%E8%AF%A2http%E7%9B%91%E5%90%AC>)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 90, "backendPort": 90, "scheduler": "LeastConnection", "keepSession": false,
"keepSessionType": null, "keepSessionTimeout": null, "keepSessionCookieName": null, "xForwardedFor": false,
"serverTimeout": 30, "redirectPort": 80 }, { "listenerPort": 95, "backendPort": 95, "scheduler": "LeastConnection",
"keepSession": false, "keepSessionType": null, "keepSessionTimeout": null, "keepSessionCookieName": null,
"xForwardedFor": false, "serverTimeout": 1800, "redirectPort": 80 } ], "marker": "listener-005A", "nextMarker": null,
"isTruncated": false, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_describe_app_http_listeners.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_app_http_listeners.go)

查询HTTPS监听器

查询指定LoadBalancer下所有HTTPS监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定。

使用以下代码可以查询应用型HTTPS监听器信息

****函数声明****

```
func (c Client) DescribeAppHTTPSListeners(blbId string, args DescribeAppListenerArgs) (*DescribeAppHTTPSListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeAppHTTPSListeners查询HTTPS监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#describeapphttpslisteners%E6%9F%A5%E8%AF%A2https%E7%9B%91%E5%>)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 90, "backendPort": 90, "scheduler": "LeastConnection", "keepSession": false,
"keepSessionType": null, "keepSessionTimeout": null, "keepSessionCookieName": null, "xForwardedFor": false,
"serverTimeout": 30, "certIds": ["cert-zfj2ey2z4kmm"] }, { "listenerPort": 95, "backendPort": 95, "scheduler":
"LeastConnection", "keepSession": false, "keepSessionType": null, "keepSessionTimeout": null, "keepSessionCookieName":
null, "xForwardedFor": false, "serverTimeout": 1800, "certIds": ["cert-zfj2ey2z4kmm"] } ], "marker": "listener-005A",
"nextMarker": null, "isTruncated": false, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxznw6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxznw6)

****代码示例****

具体代码示例参考：[\[example_describe_app_https_listeners.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_app_https_listeners.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_describe_app_https_listeners.go)

查询应用型SSL监听器

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有SSL监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询应用型SSL监听器

****函数声明****

```
func (c Client) DescribeAppSSLListeners(blbld string, args DescribeAppListenerArgs) (*DescribeAppSSLListenersResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[DescribeAppSSLListeners查询应用型SSL监听器\]](https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#describeappssllisteners%E6%9F%A5%E8%AF%A2%E5%BA%94%E7%94%A8)

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#describeappssllisteners%E6%9F%A5%E8%AF%A2%E5%BA%94%E7%94%A8>)

****返回值****

操作成功：

```
{ "listenerList": [ { "listenerPort": 80, "scheduler": "RoundRobin", "certIds": ["cert-zfj2ey2z4kmm"] }, { "listenerPort": 88, "scheduler": "RoundRobin", "certIds": ["cert-zfj2ey2z4kmm"] } ], "marker": "listener-0050", "nextMarker": null, "isTruncated": false, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxznw6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxznw6)

****代码示例****

具体代码示例参考：[\[example_describe_app_ssl_listeners.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_app_ssl_listeners.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_app_ssl_listeners.go)

查询所有监听器

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有监听器的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询所有监听器

****函数声明****

```
func (c Client) DescribeAppAllListeners(blbld string, args DescribeAppListenerArgs) (*DescribeAppAllListenersResult, error)
```


****参数含义****

请参考OpenAPI文档：[DeleteAppListeners释放监听器]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#deleteapplisteners%E9%87%8A%E6%94%BE%E7%9B%91%E5%90%AC%E5%90%A5>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_delete_app_listeners.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_delete_app_listeners.go)

创建策略

<!--1、*必选，一句话描述函数功能。-->

在指定应用型BLB监听器端口下创建策略

使用以下代码可以在指定应用型BLB监听器端口下创建策略

****函数声明****

```
func (c Client) CreatePolicys(blbl string, args CreatePolicysArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreatePolicys创建策略]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#createpolicys%E5%88%9B%E5%BB%BA%E7%AD%96%E7%95%A5>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_create_policys_server_group.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_create_policys_server_group.go)

查询策略列表

<!--1、*必选，一句话描述函数功能。-->

查询指定LoadBalancer下所有服务器组的信息，支持按监听器端口进行匹配查询，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询指定LoadBalancer下所有服务器组的信息

****函数声明****

```
func (c Client) DescribePolicys(blbl string, args DescribePolicysArgs) (*DescribePolicysResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribePolicys查询对应BLB端口下策略信息]

(<https://cloud.baidu.com/doc/BLB/s/ujwxnyux#describepolicys%E6%9F%A5%E8%AF%A2%E5%AF%B9%E5%BA%94blb%E7%>)

****返回值****

操作成功：

```
{ "marker": "rs-012310",
  "isTruncated": true, "maxKeys": 1000, "nextMarker": "rs-013214",
  "policyList": [ { "id": "policy_id", "appServerGroupId": "sg_id", "appServerGroupName": "sg_name", "frontendPort": 80,
    "type": "TCP", "backendPort": 98, "portType": "TCP", "priority": 100, "desc": "aa", "groupType": "Server", "ruleList": [ { "key":
      "host", "value": "*.baidu.com" } ] } ] }
```

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_describe_policys_server_group.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_policys_server_group.go)

删除策略

<!-- 1、*必选，一句话描述函数功能。 -->

批量删除对应BLB端口下的策略

使用以下代码可以批量删除对应BLB端口下的策略

****函数声明****

```
func (c Client) DeletePolicys(blbl string, args DeletePolicysArgs) error
```

****参数含义****

请参考OpenAPI文档：[DeletePolicys批量删除策略]

(<https://cloud.baidu.com/doc/BLB/s/ujwvxnyux#deletepolicys%E6%89%B9%E9%87%8F%E5%88%A0%E9%99%A4%E7%AD%9A>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_delete_policys.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_delete_policys.go)

添加应用型BLB 服务器组

在指定应用型BLB下，创建一个服务器组，用来绑定后端服务器，以及为监听器开放相应的端口

使用以下代码可以创建应用型BLB服务器组

****函数声明****

```
func (c Client) CreateAppServerGroup(blbld string, args CreateAppServerGroupArgs) (*CreateAppServerGroupResult, error)
```

****参数含义****

请参考OpenAPI文档：[CreateAppServerGroup创建应用型服务器组]

(<https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#createaappservergroup%E5%88%9B%E5%BB%BA%E5%BA%94%E7%94%A8>)

****返回值****

操作成功：

```
{ "id": "sg_short_id", "name": "name", "desc": "aaa", "status": "available"
}
```

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_create_app_server_group.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_app_server_group.go)

更新应用型BLB 服务器组

更新指定LoadBalancer下的TCP监听器参数，所有请求参数中指定的域都会被更新，未指定的域保持不变，监听器通过端口指定。

使用以下代码可以更新应用型BLB服务器组

****函数声明****

```
func (c Client) UpdateAppServerGroup(blbld string, args UpdateAppServerGroupArgs) error
```

参数含义

请参考OpenAPI文档：[UpdateAppServerGroup更新应用型服务器组]

(<https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#updateappservergroup%E6%9B%B4%E6%96%B0%E6%9C%8D%E5%8A%A1>)

返回值

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

代码示例

具体代码示例参考：[example_update_app_server_group.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blr/example_update_app_server_group.go)

查询应用型BLB 服务器组

查询指定LoadBalancer下所有服务器组的信息，结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询应用型BLB服务器组

函数声明

```
func (c Client) DescribeAppServerGroup(blbld string, args DescribeAppServerGroupArgs) (*DescribeAppServerGroupResult, error)
```

参数含义

请参考OpenAPI文档：[DescribeAppServerGroup查询服务器组]

(<https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#describeappservergroup%E6%9F%A5%E8%AF%A2%E6%9C%8D%E5%8A%A1>)

返回值

操作成功：

```
{ "appServerGroupList": [ { "id": "sg_short_id", "name": "name", "desc": "bbb", "status": "available", "portList": [ { "id": "port_id", "port": "port", "type": "TCP", "healthCheck": "TCP", "healthCheckNormalStatus": "^[1-5][0-9]*$", "healthCheckPort": 90, "healthCheckTimeoutInSecond": 2, "healthCheckDownRetry": 3, "healthCheckUpRetry": 3, "healthCheckIntervalInSecond": 1, "healthCheckUriPath": "HEALTH_CHECK_URL_PATH", "healthCheckHost": "www.xxx", "status": "available" } ] }, { "marker": "sg-012310", "isTruncated": false, "maxKeys": 1000, "nextMarker": "sg-013214" } ] }
```

操作失败：

抛出异常，异常列表参考[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[example_describe_app_server_group.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blk/example_describe_app_server_group.go)

删除应用型BLB 服务器组

删除服务器组，通过服务器组id指定

使用以下代码可以删除应用型BLB服务器组

****函数声明****

```
func (c Client) DeleteAppServerGroup(blbld string, args DeleteAppServerGroupArgs) error
```

****参数含义****

请参考OpenAPI文档：[DeleteAppServerGroup删除应用型服务器组]

(https://cloud.baidu.com/doc/BLB/s/Bjwxny4u#deleteappservergroup%E5%88%A0%E9%99%A4%E6%9C%8D%E5%8A%A1%)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[example_delete_app_server_group.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blk/example_delete_app_server_group.go)

添加应用型BLB 服务器组端口

在指定应用型BLB下，创建一个服务器组后端端口，将发往该端口的所有流量按权重轮询分发到其绑定的对应服务器列表中的服务器

使用以下代码可以创建应用型BLB服务器组端口

****函数声明****

```
func (c Client) CreateAppServerGroupPort(blbld string, args CreateAppServerGroupPortArgs) (*CreateAppServerGroupPortResult, error)
```

****参数含义****

请参考OpenAPI文档：[CreateAppServerGroupPort创建应用型服务器组端口]

(https://cloud.baidu.com/doc/BLB/s/Bjwxny4u#createappservergroupport%E5%88%9B%E5%BB%BA%E5%BA%94%E7%94%)

****返回值****

操作成功：

```
{ "sgId":"sgid", "type":"TCP", "port": 80, "healthCheck":"TCP",
  "healthCheckTimeoutInSecond": 2,
  "healthCheckDownRetry": 3,
  "healthCheckUpRetry": 3,
  "healthCheckIntervalInSecond":3
}
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_create_app_server_group_port.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_app_server_group_port.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_create_app_server_group_port.go)

更新应用型BLB 服务器组端口

根据id更新服务器组端口。

使用以下代码可以更新应用型BLB服务器组端口

****函数声明****

```
func (c Client) UpdateAppServerGroupPort(blbId string, args UpdateAppServerGroupPortArgs) error
```

****参数含义****

请参考OpenAPI文档：[\[UpdateAppServerGroupPort更新应用型服务器组端口\]\(https://cloud.baidu.com/doc/BLB/s/Bjwxny4u#updateappservergroupport\)](https://cloud.baidu.com/doc/BLB/s/Bjwxny4u#updateappservergroupport)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_update_app_server_group_port.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_update_app_server_group_port.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_update_app_server_group_port.go)

删除应用型BLB 服务器组端口

删除服务器组端口，通过服务器组id指定

使用以下代码可以删除应用型BLB服务器组端口

****函数声明****

```
func (c Client) DeleteAppServerGroupPort(blbId string, args DeleteAppServerGroupPortArgs) error
```

****参数含义****

请参考OpenAPI文档：[DeleteAppServerGroupPort删除应用型服务器组端口](https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#deleteappservergroupport%E5%88%A0%E9%99%A4%E6%9C%8D%E5%8A%)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxnz6)

****代码示例****

具体代码示例参考：[example_delete_app_server_group_port.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/blb/example_delete_app_server_group_port.go)

添加应用型BLB后端RS

在指定应用型BLB和服务器组下绑定后端服务器RS

使用以下代码可以添加应用型BLB后端RS

****函数声明****

```
func (c Client) CreateBlbRs(blbl string, args CreateBlbRsArgs) error
```

****参数含义****

请参考OpenAPI文档：[CreateBlbRs添加应用型BLB后端RS](https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#createblbrs%E6%B7%BB%E5%8A%A0%E5%BA%94%E7%94%A8%E5%9E%8)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxnz6)

****代码示例****

具体代码示例参考：[example_create_blb_rs.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_create_blb_rs.go)

更新服务器组下挂载的RS权重

更新指定服务器组下的RS信息

使用以下代码可以更新服务器组下挂载的RS权重

****函数声明****

```
func (c Client) UpdateBlbRs(blbl string, args UpdateBlbRsArgs) error
```

****参数含义****

请参考OpenAPI文档：[UpdateBlbRs更新服务器组下挂载的RS权重]

(<https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#updateblbrs%E6%9B%B4%E6%96%B0%E6%9C%8D%E5%8A%A1%E5%99%>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_update_blb_rs.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_update_blb_rs.go)

查询服务器组下的RS列表信息

查询指定LoadBalancer下所有服务器组的信息；支持按监听器端口进行匹配查询；结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询服务器组下的RS列表信息

****函数声明****

```
func (c Client) DescribeBlbRs(blbld string, args DescribeBlbRsArgs) (*DescribeBlbRsResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeBlbRs查询服务器组下的RS列表信息]

(<https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#describeblbrs%E6%9F%A5%E8%AF%A2%E6%9C%8D%E5%8A%A1%E5%99%>)

****返回值****

操作成功：

```
{ "marker": "rs-012310",  
  "isTruncated": false, "maxKeys": 1000, "nextMarker": "rs-013214",  
  "backendServerList": [ { "instanceId": "i-fvdixxxx", "privateIp": "192.168.0.1",  
    "weight": 100, "portList": [  
      { "listenerPort": 80, "backendPort": "80", "portType": "HTTP", "healthCheckPortType": "HTTP", "portId": "port-xxx",  
        "policyId": "policy-xxx",  
        "status": "Alive" } ] } ] }
```


操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxnz6)

****代码示例****

具体代码示例参考：[example_describe_blb_rs.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_blb_rs.go)

删除服务器组下挂载的RS

删除服务器组下的后端RS，通过服务器组ID和后端RS ID指定；支持批量释放

使用以下代码可以删除服务器组下挂载的RS

****函数声明****

```
func (c Client) DeleteBlbRs(blbId string, args DeleteBlbRsArgs) error
```

****参数含义****

请参考OpenAPI文档：[DeleteBlbRs删除服务器组下挂载的rs]

(https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#deleteblbrs%E5%88%A0%E9%99%A4%E6%9C%8D%E5%8A%A1%E5%99%A)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxnz6)

****代码示例****

具体代码示例参考：[example_delete_blb_rs.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_delete_blb_rs.go)

查询服务器组下绑定的RS

查询服务器组下绑定的RS

使用以下代码可以查询服务器组下绑定的RS

****函数声明****

```
func (c Client) DescribeRsMount(blbId, sgId string) (DescribeRsMountResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeRsMount查询服务器组下绑定的server]

(https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#describersmount%E6%9F%A5%E8%AF%A2%E6%9C%8D%E5%8A%A1%E5%99%A)

****返回值****

操作成功：

```
{
```

```
"backendServerList": [ { "instanceId": "i-fvdixxxx", "privateIp": "192.168.0.1",
"weight": 100, "portList": [
{ "listenerPort": 80, "backendPort": "80", "portType": "HTTP", "healthCheckPortType": "HTTP", "portId": "port-xxx",
"policyId": "policy-xxx",
"status": "Alive" } ] } ] }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxznw6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxznw6)

****代码示例****

具体代码示例参考：[\[example_describe_rs_mount.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_rs_mount.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_rs_mount.go)

查询服务器组下未绑定的RS

查询服务器组下未绑定的RS

使用以下代码可以查询服务器组下未绑定的RS

****函数声明****

```
func (c Client) DescribeRsUnMount(blbId, sgId string) (DescribeRsMountResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[DescribeRsUnMount查询服务器组下未绑定的RS\]](https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#describersunmount%E6%9F%A5%E8%AF%A2%E6%9C%8D%E5%8A%A1%E5)

(<https://cloud.baidu.com/doc/BLB/s/Bjwvxny4u#describersunmount%E6%9F%A5%E8%AF%A2%E6%9C%8D%E5%8A%A1%E5>)

****返回值****

操作成功：



```
{
"backendServerList": [ { "instanceId": "i-fvdixxxx", "privateIp": "192.168.0.1",
"weight": 100, "portList": [
{ "listenerPort": 80, "backendPort": "80", "portType": "HTTP", "healthCheckPortType": "HTTP", "portId": "port-xxx",
"policyId": "policy-xxx",
"status": "Alive" } ] } ] }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwvxznw6\)](https://cloud.baidu.com/doc/BLB/s/Djwvxznw6)

****代码示例****

具体代码示例参考：[\[example_describe_rs_unmount.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_rs_unmount.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_rs_unmount.go)

绑定普通安全组

<!--1、*必选，一句话描述函数功能。-->

为指定LoadBalancer（普通型、应用型、IPv6型）绑定普通安全组，支持批量绑定

使用以下代码可以为指定LoadBalancer绑定普通安全组

****函数声明****

```
func (c Client) BindSecurityGroups(blbl string, args UpdateSecurityGroupsArgs) error
```

****参数含义****

请参考OpenAPI文档：[\[BindSecurityGroups绑定普通安全组\]](#)

(<https://cloud.baidu.com/doc/BLB/s/Vl7ygu1rk#bindsecuritygroups%E7%BB%91%E5%AE%A%E6%99%AE%E9%80%9A%E5%B8%B6%E7%BD%A%E5%AD%A>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_bind_security_groups.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_bind_security_groups.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_bind_security_groups.go)

解绑普通安全组

<!--1、*必选，一句话描述函数功能。-->

为指定LoadBalancer（普通型、应用型、IPv6型）解绑普通安全组，支持批量解绑

使用以下代码可以为指定LoadBalancer解绑普通安全组

****函数声明****

```
func (c Client) UnbindSecurityGroups(blbl string, args UpdateSecurityGroupsArgs) error
```

****参数含义****

请参考OpenAPI文档：[UnbindSecurityGroups解绑普通安全组]

(<https://cloud.baidu.com/doc/BLB/s/VI7ygu1rk#unbindsecuritygroups%E8%A7%A3%E7%BB%91%E6%99%AE%E9%80%9A%E>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxzw6>)

****代码示例****

具体代码示例参考：[example_unbind_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_unbind_security_groups.go)

查询BLB实例普通安全组列表

<!--1、*必选，一句话描述函数功能。-->

按ID查询用户账户下指定LoadBalancer的普通安全组信息

使用以下代码可以按ID查询用户账户下指定LoadBalancer的普通安全组信息

****函数声明****

```
func (c Client) DescribeSecurityGroups(blbId string) (DescribeSecurityGroupsResult, error)
```

****参数含义****

请参考OpenAPI文档：[DescribeSecurityGroups查询BLB实例普通安全组列表]

(<https://cloud.baidu.com/doc/BLB/s/VI7ygu1rk#describesecuritygroups%E6%9F%A5%E8%AF%A2blb%E5%AE%9E%E4%BE%E>)

****返回值****

操作成功：

```
{ "blbSecurityGroups": [ { "securityGroupDesc": "", "securityGroupId": "g-4NxWoxeq", "securityGroupName": "securitygroup-name", "vpcName": "vpc-name", "securityGroupRules": [ { "destGroupId": "", "destIp": "all", "direction": "egress", "ethertype": "IPv4", "portRange": "1-65535", "protocol": "all", "securityGroupRuleId": "r-gkv8yupumvx2" } ] } ] }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_describe_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_security_groups.go)

绑定企业安全组

<!--1、*必选，一句话描述函数功能。-->

为指定LoadBalancer（普通型、应用型、IPv6型）绑定企业安全组，支持批量绑定

使用以下代码可以为指定LoadBalancer绑定企业安全组

****函数声明****

```
func (c Client) BindEnterpriseSecurityGroups(blId string, args UpdateEnterpriseSecurityGroupsArgs) error
```

****参数含义****

请参考OpenAPI文档：[BindEnterpriseSecurityGroups绑定企业安全组]

(https://cloud.baidu.com/doc/BLB/s/Vl7ygu1rk#bindenterprisesecuritygroups%E7%BB%91%E5%AE%9A%E4%BC%81%E4%B)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_bind_enterprise_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_bind_enterprise_security_groups.go)

解绑企业安全组

<!--1、*必选，一句话描述函数功能。-->

为指定LoadBalancer（普通型、应用型、IPv6型）解绑企业安全组，支持批量解绑

使用以下代码可以为指定LoadBalancer解绑企业安全组

****函数声明****

```
func (c Client) UnbindEnterpriseSecurityGroups(blId string, args UpdateEnterpriseSecurityGroupsArgs) error
```

****参数含义****

请参考OpenAPI文档：[\[UnbindEnterpriseSecurityGroups解绑企业安全组\]](#)

[illegible]

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_unbind_enterprise_security_groups.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/appb1b/example_unbind_enterprise_security_groups.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appb1b/example_unbind_enterprise_security_groups.go)

查询bib实例企业安全组列表

<!--1、*必选，一句话描述函数功能。-->

按ID查询用户账户下指定LoadBalancer的企业安全组信息

使用以下代码可以按ID查询用户账户下指定LoadBalancer的企业安全组信息。

****函数声明****

```
func (c Client) DescribeEnterpriseSecurityGroups(bld string) (DescribeEnterpriseSecurityGroupsResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[DescribeEnterpriseSecurityGroups查询BLB实例企业安全组列表\]](#)

(<https://cloud.baidu.com/doc/BLB/s/Vl7ygu1rk#describeenterprisesecuritygroups%E6%9F%A5%E8%AF%A2b1b%E5%AE%9E>)

****返回值****

操作成功：

```
{ "enterpriseSecurityGroups": [{ "enterpriseSecurityGroupId": "esg-g8z4qf0m0bu", "enterpriseSecurityGroupName": "test0627", "enterpriseSecurityGroupDesc": "", "enterpriseSecurityGroupRules": [{ "remark": "全部协议", "direction": "egress", "action": "allow", "priority": 1000, "ethertype": "IPv6", "portRange": "", "sourceIp": "", "destIp": "all", "enterpriseSecurityGroupRuleId": "esgr-9tzd4yijcwqu", "protocol": "all" }, { "remark": "", "direction": "egress", "action": "allow", "priority": 1000, "ethertype": "IPv4", "portRange": "", "sourceIp": "", "destIp": "all", "enterpriseSecurityGroupRuleId": "esgr-9je09z6bd9wv", "protocol": "tcp" }, { "remark": "", "direction": "ingress", "action": "allow", "priority": 1000, "ethertype": "IPv4", "portRange": "2", "sourceIp": "all", "destIp": "", "enterpriseSecurityGroupRuleId": "esgr-jcej4juqbh49", "protocol": "tcp" } ] }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxzw6)

****代码示例****

具体代码示例参考：[example_describe_enterprise_security_groups.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/appblb/example_describe_enterprise_security_groups.go)

IPv6型BLB实例

创建BLB

<!--1、*必选，一句话描述函数功能。-->

创建一个BLB实例，返回分配的服务地址及实例ID。付费方式为后付费。需要实名认证，若未通过实名认证可以前往百度智能云官网控制台中的安全认证下的实名认证中进行认证

使用以下代码可以创建BLB实例

****函数声明****

```
func (c Client) CreateLoadBalancer(args CreateLoadBalancerArgs) (*CreateLoadBalancerResult, error)
```

****参数含义****

请参考OpenAPI文档：[IPv6BLB相关接口]
(https://cloud.baidu.com/doc/BLB/s/Ukx0417d1)

****返回值****

操作成功：

```
{ "blbId": "lb-BLuOPSLZ", "name": "blb-for-test", "desc": "", "address": "192.168.0.24" }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwvxzw6)

****代码示例****

具体代码示例参考：[example_create_loadbalancer.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/ipv6blb/example_create_loadbalancer.go)

查询已有BLB实例

<!--1、*必选，一句话描述函数功能。-->

查询用户账户下所有LoadBalancer的信息，支持按LoadBalancer的Name、Address进行模糊查询，支持按LoadBalancer的ID进行精确查询，支持按bccId进行查询，目前只支持单个bccId查询，并且仅支持精确查询。查询结果为包含该后端服务器的负载均衡实例。返回结果是多重查询条件的交集。如果BLB实例开启了公网访问，返回结果将会带上公网IP。结果支持marker分页，分页大小默认为1000，可通过maxKeys参数指定

使用以下代码可以查询已有BLB实例

****函数声明****

```
func (c Client) DescribeLoadBalancers(args DescribeLoadBalancersArgs) (*DescribeLoadBalancersResult, error)
```

****参数含义****

请参考OpenAPI文档：[IPv6BLB相关接口]
(<https://cloud.baidu.com/doc/BLB/s/Ukx0417d1>)

****返回值****

操作成功：

```
{ "blbList": [ { "blbId": "lb-a7e5zPPk", "status": available, "name": "test-blb", "desc": "用于生产环境", "vpcId": "vpc-fy6vdykpwkqb", "address": "192.168.0.24" }, { "blbId": "lb-gj5gVpeq", "status": available, "name": "nametest", "desc": "用于测试环境", "vpcId": "vpc-a8n5p6kybbx4", "address": "192.168.0.25" } ], "marker": "blb-0A20F971", "nextMarker": "blb-0A20FB09", "isTruncated": true, "maxKeys": 2 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_describe_loadbalancers.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/ipv6blb/example_describe_loadbalancers.go)

负载均衡专属集群LBDC**##### 创建LBDC**

<!--1、*必选，一句话描述函数功能。-->
创建lbdc实例。

使用以下代码可以创建lbdc

****函数声明****

```
func (c Client) CreateLbdc(args CreateLbdcArgs) (*CreateLbdcResult, error)
```

****参数含义****

请参考OpenAPI文档：[创建LBDC](<https://cloud.baidu.com/doc/BLB/s/6kszygx4#%E5%88%9B%E5%BB%BAlbdc>)

****返回值****

操作成功：

```
{ "id": "bgw_group-a3t786p1", "type": "4Layer", "desc":"" }
```


操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[example_create_lbdc.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/lbdc/example_create_lbdc.go)

升级LBDC

<!--1、*必选，一句话描述函数功能。-->

升级LBDC

使用以下代码可以升级lbdc实例

****函数声明****

```
func (c Client) UpgradeLbdc(args UpgradeLbdcArgs) error
```

****参数含义****

请参考OpenAPI文档：[升级LBDC](https://cloud.baidu.com/doc/BLB/s/6kszygx4#%E5%8D%87%E7%BA%A7lbdc)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[example_upgrade_lbdc.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/lbdc/example_upgrade_lbdc.go)

续费LBDC

<!--1、*必选，一句话描述函数功能。-->

续费lbdc

使用以下代码可以为lbdc实例续费

****函数声明****

```
func (c Client) RenewLbdc(args RenewLbdcArgs) error
```

****参数含义****

请参考OpenAPI文档：[续费LBDC](https://cloud.baidu.com/doc/BLB/s/6kszygx4#%E7%BB%AD%E8%B4%B9lbdc)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[example_renew_lbdc.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/lbdc/example_renew_lbdc.go)

LBDC列表

<!--1、*必选，一句话描述函数功能。-->

获取LBDC列表

使用以下代码可以获取lbdc列表

****函数声明****

```
func (c Client) ListLbdc(args ListLbdcArgs) (*ListLbdcResult, error)
```

****参数含义****

请参考OpenAPI文档：[LBDC列表](https://cloud.baidu.com/doc/BLB/s/6kszygx4#lbdc%E5%88%97%E8%A1%A8)

****返回值****

操作成功：

```
{ "clusterList": [{ "id": "bgw_group-a3t786p1", "name": "abc", "type": "4Layer", "status": "available", "ccuCount": 1,
"createTime": "2021-08-25T06:13:09Z", "expireTime": "2021-09-25T06:13:09Z", "desc": "" } ], "marker": "bgw_group-a3t786p1", "nextMarker": null, "isTruncated": false, "maxKeys": 1000 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[example_list_lbdc.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/lbdc/example_list_lbdc.go)

LBDC详情

<!--1、*必选，一句话描述函数功能。-->

获取lbdc详情

使用以下代码可以获取lbdc详情

****函数声明****

```
func (c Client) GetLbdcDetail(lbdcId string) (GetLbdcDetailResult, error)
```

****参数含义****

请参考OpenAPI文档：[LBDC详情](https://cloud.baidu.com/doc/BLB/s/6kszygx4#lbd%E8%AF%A6%E6%83%85)

****返回值****

操作成功：

```
{ "id": "bgw_group-a3t786p1", "name": "abc", "type": "4Layer", "status": "available", "ccuCount": 1, "createTime": "2021-08-25T06:13:09Z", "expireTime": "2021-09-25T06:13:09Z", "desc": "", "totalConnectCount":100, "newConnectCps":100, "networkInBps":100, "networkOutBps":100 }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_get_lbd_detail.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/lbdc/example_get_lbd_detail.go)

更新LBDC

<!-- 1、*必选，一句话描述函数功能。-->

更新lbdc

使用以下代码可以更新lbdc

****函数声明****

```
func (c Client) UpdateLbdc(args UpdateLbdcArgs) error
```

****参数含义****

请参考OpenAPI文档：[更新LBDC](https://cloud.baidu.com/doc/BLB/s/6kszygx4#%E6%9B%B4%E6%96%B0lbd)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_update_lbdc.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/lbdc/example_update_lbdc.go)

LBDC关联的BLB列表

<!-- 1、*必选，一句话描述函数功能。-->

获取lbdc关联的blb列表

使用以下代码可以获取lbdc关联的blb列表

****函数声明****

```
func (c Client) GetBoundBIBListOfLbdc(lbdcId string) (GetBoundBIBListOfLbdcResult, error)
```

****参数含义****

请参考OpenAPI文档 [LBDC关联的BLB列表]

(<https://cloud.baidu.com/doc/BLB/s/6kszygx4#lbdc%E5%85%B3%E8%81%94%E7%9A%84blb%E5%88%97%E8%A1%A8>)

****返回值****

操作成功：

```
{ "blbList": [ { "blbId": "lb-1c791997", "name": "abc", "status": "available", "blbType": "normal", "publicIp": "10.10.10.10",
"eipRouteType": "bgp", "bandwidth": 20, "address": "192.168.0.7", "ipv6": "2400:da00:e003:1820::4", "vpclId": "vpc-
On1hhh8759b0", "subnetId": "sbn-p09p0ihp7498" } ] }
```

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6>)

****代码示例****

具体代码示例参考：[example_get_bound_blb_list_of_lbdc.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/lbdc/example_get_bound_blb_list_of_lbdc.go)

服务发布点**##### 创建服务发布点**

<!--1、*必选，一句话描述函数功能。-->

创建服务发布点

使用以下代码可以创建服务发布点

****函数声明****

```
func (c Client) CreateUserService(args CreateUserServiceArgs) (*CreateUserServiceResult, error)
```

****参数含义****

请参考OpenAPI文档：[创建服务发布点]

(<https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E5%88%9B%E5%BB%BA%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8>)

****返回值****

操作成功：

```
{ "service": "xxx.baidubce.com" }
```

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_create_userservice.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_create_userservice.go)

更新服务发布点

<!--1、*必选，一句话描述函数功能。-->

更新服务发布点

使用以下代码可以更新服务发布点

****函数声明****

```
func (c Client) UpdateUserService(service string, args UpdateServiceArgs) error
```

****参数含义****

请参考OpenAPI文档：[更新服务发布点]

(https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E6%9B%B4%E6%96%B0%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](https://cloud.baidu.com/doc/BLB/s/DjwvxnzW6)

****代码示例****

具体代码示例参考：[example_update_userservice.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_update_userservice.go)

服务发布点绑定实例

<!--1、*必选，一句话描述函数功能。-->

服务发布点绑定实例

使用以下代码可以服务发布点绑定实例

****函数声明****

```
func (c Client) UserServiceBindInstance(service string, args UserServiceBindArgs) error
```

****参数含义****

请参考OpenAPI文档：[服务发布点绑定实例]

(<https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8%83%E7%82%B9%E7%BE>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_userservice_bind_instance.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_userservice_bind_instance.go)

服务发布点解绑服务

<!--1、*必选，一句话描述函数功能。-->

服务发布点解绑服务

使用以下代码可以服务发布点解绑服务

****函数声明****

```
func (c Client) UserServiceUnBindInstance(service string, args UserServiceUnBindArgs) error
```

****参数含义****

请参考OpenAPI文档：[服务发布点解绑服务]

(<https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8%83%E7%82%B9%E8%A7>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_userservice_unbind_instance.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_userservice_unbind_instance.go)

新增服务发布点鉴权信息

<!--1、*必选，一句话描述函数功能。-->

新增服务发布点鉴权信息

使用以下代码可以新增服务发布点鉴权信息

****函数声明****

```
func (c Client) UserServiceAddAuth(service string, args UserServiceAuthArgs) error
```

****参数含义****

请参考OpenAPI文档：[新增服务发布点鉴权信息]

(<https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E6%96%B0%E5%A2%9E%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwvxnz6>)

****代码示例****

具体代码示例参考：[example_userservice_add_auth.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_userservice_add_auth.go)

修改服务发布点鉴权信息

<!--1、*必选，一句话描述函数功能。-->

修改服务发布点鉴权信息

使用以下代码可以修改服务发布点鉴权信息

****函数声明****

```
func (c Client) UserServiceEditAuth(service string, args UserServiceAuthArgs) error
```

****参数含义****

请参考OpenAPI文档：[修改服务发布点鉴权信息]

(<https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E4%BF%AE%E6%94%B9%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_userservice_edit_auth.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_userservice_edit_auth.go)

删除服务发布点鉴权信息

<!--1、*必选，一句话描述函数功能。-->

删除服务发布点鉴权信息

使用以下代码可以删除服务发布点鉴权信息

****函数声明****

```
func (c Client) UserServiceRemoveAuth(service string, args UserServiceRemoveAuthArgs) error
```

****参数含义****

请参考OpenAPI文档：[删除服务发布点鉴权信息]

(<https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E5%88%A0%E9%99%A4%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[异常列表](<https://cloud.baidu.com/doc/BLB/s/Djwxnzw6>)

****代码示例****

具体代码示例参考：[example_userservice_remove_auth.go](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_userservice_remove_auth.go)

查询服务发布列表

<!--1、*必选，一句话描述函数功能。-->

查询服务发布列表

使用以下代码可以查询服务发布列表

****函数声明****


```
func (c Client) DescribeUserServices(args DescribeUserServicesArgs) (*DescribeUserServicesResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[查询服务发布列表\]](#)

(<https://cloud.baidu.com/doc/BLB/s/Nkwzwy9m6#%E6%9F%A5%E8%AF%A2%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8>)

****返回值****

操作成功：

```
{ "nextMarker": "user-service-a223vc", "marker": "user-service-a032va", "maxKeys": 1, "isTruncated": true, "services": [ {
  "serviceId": "user-service-a223vc", "name": "name", "description": "", "serviceName": "serviceName", "bindType": "blb",
  "instanceId": "lb-d3e2cf75", "status": "inService", "service": "xxx.baidubce.com", "createTime": "2019-03-07T02:35:31Z",
  "endpointCount": 1, "endpointList": [ { "endpointId": "endpoint-a223vc", "uid": "e8deba4d81d3435bb4591ce97288f296",
  "attachTime": "2019-03-07T02:35:31Z", }, }, "authList": [ { "uid": "*", "auth": "allow" } ]
```

$$\left. \begin{array}{l} 1 \\ \end{array} \right\}$$

}

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_describe_userservices.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_describe_userservices.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_describe_userservices.go)

查询服务发布详情

<!--1、*必选，一句话描述函数功能。-->

查询服务发布详情

使用以下代码可以查询服务发布详情

****函数声明****

```
func (c Client) DescribeUserServiceDetail(service string) (DescribeUserServiceDetailResult, error)
```

****参数含义****

请参考OpenAPI文档：[\[查询服务发布详情\]](#)

(<https://cloud.baidu.com/doc/BLB/s/Nkwzwy9m6#%E6%9F%A5%E8%AF%A2%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8>)

****返回值****

操作成功：

```
{ "serviceId": "user-service-a223vc", "name": "name", "description": "", "serviceName": "serviceName", "bindType": "blb",  
  "instanceId": "lb-d3e2cf75", "status": "inService", "service": "xxx.baidubce.com", "createTime": "2019-03-07T02:35:31Z",  
  "endpointCount": 1, "endpointList": [ { "endpointId": "endpoint-a223vc", "uid": "e8deba4d81d3435bb4591ce97288f296".
```

```
"attachTime": "2019-03-07T02:35:31Z", }, }, "authList": [ { "uid": "*", "auth": "allow" } ] }
```

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_describe_userservice_detail.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_describe_userservice_detail.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_describe_userservice_detail.go)

删除服务发布点

<!--1、*必选，一句话描述函数功能。-->

删除服务发布点

使用以下代码可以删除服务发布点

****函数声明****

```
func (c *Client) DeleteUserService(service string) error
```

****参数含义****

请参考OpenAPI文档：[\[删除服务发布点\]](https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E5%88%A0%E9%99%A4%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8)

(<https://cloud.baidu.com/doc/BLB/s/Nkwzzv9m6#%E5%88%A0%E9%99%A4%E6%9C%8D%E5%8A%A1%E5%8F%91%E5%B8>)

****返回值****

操作成功：

返回200

操作失败：

抛出异常，异常列表参考：[\[异常列表\]\(https://cloud.baidu.com/doc/BLB/s/Djwxnzw6\)](https://cloud.baidu.com/doc/BLB/s/Djwxnzw6)

****代码示例****

具体代码示例参考：[\[example_delete_userservice.go\]\(https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_delete_userservice.go\)](https://github.com/baidubce/bce-sdk-go/blob/master/examples/userservice/example_delete_userservice.go)

版本变更记录

- GO SDK开发包[2020-05-20] 版本号 0.9.11

常见问题

常见问题总览

一般类问题

* [BLB可以支持非百度智能云服务器（BCC）的机器吗？](BLB/常见问题/一般类问题.md#BLB可以支持非百度智能云服务器（BCC）的机器吗？)

* [BLB实例启用/不启用公网访问有什么区别？](BLB/常见问题/一般类问题.md#BLB实例启用/不启用公网访问有什么区别？)

* [是否可以在百度智能云上自己搭建负载分担服务？](BLB/常见问题/一般类问题.md#是否可以在百度智能云上自己搭建负载分担服务？)

- * [BLB实例总数超过上限，目前业务需要能否增加BLB配额？](BLB/常见问题/一般类问题.md#BLB实例总数超过上限，目前业务需要能否增加BLB配额？)
- * [后端服务器没有公网EIP，能否通过BLB访问公网？](BLB/常见问题/一般类问题.md#后端服务器没有公网EIP，能否通过BLB访问公网？)
- * [BLB支持IPv6吗？](BLB/常见问题/一般类问题.md#BLB支持IPv6吗？)
- * [windows操作系统可以使用BLB吗？](BLB/常见问题/一般类问题.md#windows操作系统可以使用BLB吗？)
- * [关于ping 负载均衡实例内网IP 的说明](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#关于ping-负载均衡实例内网ip-的说明)
- * [关于Telnet负载均衡监听端口的说明](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#关于telnet负载均衡监听端口的说明)
- * [什么情况下需要 HTTPS 双向认证？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#什么情况下需要-https-双向认证？)
- * [BLB是否支持同一监听下挂载同一后端服务器的不同端口？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb是否支持同一监听下挂载同一后端服务器的不同端口？)
- * [BLB支持同一端口创建多个不同协议的监听吗？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb支持同一端口创建多个不同协议的监听吗？)
- * [BLB支持同region跨vpc访问吗？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb支持同region跨vpc访问吗？)
- * [BLB 和后端云服务器之间是使用公网还是内网通信？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb-和后端云服务器之间是使用公网还是内网通信？)
- * [BLB UDP 协议与 TCP 协议有什么区别？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb-udp-协议与-tcp-协议有什么区别？)
- * [BLB可以为哪些 TCP 端口执行负载均衡？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb可以为哪些-tcp-端口执行负载均衡？)
- * [BLB转发请求的长度限制是多少？是否支持调整？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb转发请求的长度限制是多少？是否支持调整？)
- * [BLB级联场景使用？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb级联场景使用？)
- * [负载均衡 Cookies 会话保持方式的原理是什么？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#负载均衡-cookies-会话保持方式的原理是什么？)
- * [HTTPS 监听使用什么端口？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#https-监听使用什么端口？)
- * [tcp监听修改为https/http监听后流量统计为什么会下跌？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#tcp监听修改为https/http监听后流量统计为什么会下跌？)
- * [什么是后端服务器权重？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#什么是后端服务器权重？)
- * [请求响应为500、502、503、504状态码的解释及可能原因是什么？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#请求响应为500、502、503、504状态码的解释及可能原因是什么？)
- * [如何检查请求不均衡？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#如何检查请求不均衡？)
- * [如何检查负载均衡业务访问延时大？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#如何检查负载均衡业务访问延时大？)
- * [BLB实例是否支持 Client IP 透传？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb实例是否支持-client-ip-透传？)
- * [BLB实例后端服务器是分批次发回响应请求的，但是在客户端看到的是只收到一个请求，BLB 是否支持不缓存后端服务器的响应请求？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb实例后端服务器是分批次发回响应请求的，但是在客户端看到的是只收到一个请求，blb-是否支持不缓存后端服务器的响应请求？)
- * [BLB产品支持哪些调度算法？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#blb产品支持哪些调度算法？)
- * [普通型BLB和应用型BLB有哪些差别？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#普通型blb和应用型blb有哪些差别？)
- * [在何种情况下购买BLB要搭配EIP？](https://cloud.baidu.com/doc/BLB/s/ijwvx0Ou7#在何种情况下购买blb要搭配eip？)

性能类问题

- * [一个用户最多可以创建多少个BLB实例？](BLB/常见问题/性能类问题.md#一个用户最多可以创建多少个BLB实例？)
- * [BLB是否限制带宽？](BLB/常见问题/性能类问题.md#BLB是否限制带宽？)
- * [BLB支持哪些转发规则？](BLB/常见问题/性能类问题.md#BLB支持哪些转发规则？)
- * [目前BLB提供哪些监控信息？](BLB/常见问题/性能类问题.md#目前BLB提供哪些监控信息？)
- * [健康检查是如何实现的？](BLB/常见问题/性能类问题.md#健康检查是如何实现的？)

- * [如何获得来访者的真实IP？](BLB/常见问题/性能类问题.md#如何获得来访者的真实IP？)
 - * [BLB是否支持会话保持功能？](BLB/常见问题/性能类问题.md#BLB是否支持会话保持功能？)
 - * [后端BCC需要外网带宽吗？其与BLB的服务能力有关吗？](BLB/常见问题/性能类问题.md#后端BCC需要外网带宽吗？其与BLB的服务能力有关吗？)
 - * [BLB是否具有DDOS防护功能？](BLB/常见问题/性能类问题.md#BLB是否具有DDOS防护功能？)
 - * [负载均衡是否支持巨型帧（Jumbo frames）转发？](https://cloud.baidu.com/doc/BLB/s/Ojwvx015r#负载均衡是否支持巨型帧（jumbo-frames）转发？)
- ##### 故障类问题
- * [BCC实例异常会影响BLB服务吗？](BLB/常见问题/故障类问题.md#BCC实例异常会影响BLB服务吗？)
 - * [健康检查的状态总是异常怎么办？](BLB/常见问题/故障类问题.md#健康检查的状态总是异常怎么办？)
 - * [转发规则是源IP时，支持回环避免吗？](BLB/常见问题/故障类问题.md#转发规则是源IP时，支持回环避免吗？)
 - * [通过IP 分片访问负载均衡TCP/UDP/ICMP协议，网络不通怎么办？](https://cloud.baidu.com/doc/BLB/s/gjwvx02a8#通过ip-分片访问负载均衡tcpudpicmp协议，网络不通怎么办？)
 - * [客户端为何无法访问负载均衡？](https://cloud.baidu.com/doc/BLB/s/gjwvx02a8#客户端为何无法访问负载均衡？)
- ##### 配置类问题
- * [TCP超时时间配置？](BLB/常见问题/配置类问题.md#TCP超时时间配置？)
 - * [后端服务器能够向所在负载均衡实例发送请求吗？](BLB/常见问题/配置类问题.md#后端服务器可以向所在负载均衡实例发送请求吗？)
 - * [健康检查的相关配置，是否有相对合理的推荐值？](BLB/常见问题/配置类问题.md#健康检查的相关配置，是否有相对合理的推荐值？)
 - * [一组云服务器可以搭建多个网站并同时进行负载均衡吗？](BLB/常见问题/配置类问题.md#一组云服务器可以搭建多个网站并同时进行负载均衡吗？)
 - * [BLB支持的协议有哪些？](BLB/常见问题/配置类问题.md#BLB支持的协议有哪些？)
 - * [同一个BLB是否可以配置两个https证书，分别监听443和442端口呢？](BLB/常见问题/配置类问题.md#同一个BLB是否可以配置两个https证书，分别监听443和442端口呢？)
 - * [一台服务器能否配置在2个不同的BLB上？](BLB/常见问题/配置类问题.md#一台服务器能否配置在2个不同的BLB上？)
 - * [BLB监听设置中，TCP监听和HTTP监听有什么区别吗？](BLB/常见问题/配置类问题.md#BLB监听设置中，TCP监听和HTTP监听有什么区别吗？)
 - * [负载均衡会得的多个IP，哪一个为终端用户的真实IP呢？](BLB/常见问题/配置类问题.md#负载均衡会得的多个IP，哪一个为终端用户的真实IP呢？)
 - * [如何对BLB资源状况进行定期的全面检查？](BLB/常见问题/配置类问题.md#如何对BLB资源状况进行定期的全面检查？)
 - * [应用型BLB通过IP组挂载IDC侧的服务器健康检查失败？](BLB/常见问题/配置类问题.md#应用型BLB通过IP组挂载IDC侧的服务器健康检查失败？)
 - * [关于内网回环问题的说明](https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#关于内网回环问题的说明)
 - * [四层负载均衡后端RS权重重置为0与删除RS的区别](https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#四层负载均衡后端rs权重重置为0与删除rs的区别)
 - * [七层负载均衡后端RS权重重置为0与删除RS的区别](https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#七层负载均衡后端rs权重重置为0与删除rs的区别)
 - * [监听上的连接数较少时，权重相同的wrr或rr调度策略的各个rs为何连接数不相同？](https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#监听上的连接数较少时，权重相同的wrr或rr调度策略的各个rs为何连接数不相同？)

常见问题FAQ：

- * [四层负载均衡源地址哈希调度算法后端RS权重置为0仍能接收处理流量?]
(<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#四层负载均衡源地址哈希调度算法后端rs权重置为0仍能接收处理流量>)
- * [四层负载均衡删除后端服务器RS后，删除的RS上存量连接如何处理？]
(<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#四层负载均衡删除后端服务器rs后，删除的rs上存量连接如何处理？>)
- * [如何启用WebSocket支持？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#如何启用websocket支持？>)
- * [是否支持在业务不中断的前提下，将共享型负载均衡升级为规格型负载均衡？]
(<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#是否支持在业务不中断的前提下，将共享型负载均衡升级为规格型负载均衡？>)
- * [LB支持上传证书吗？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#lb支持上传证书吗？>)
- * [一个监听器可以绑定多少个 HTTPS 证书？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#一个监听器可以绑定多少个-https-证书？>)
- * [证书区分地域吗？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#证书区分地域吗？>)
- * [证书需要上传到后端服务器吗？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#证书需要上传到后端服务器吗？>)
- * [HTTP和HTTPS使用同样的后端提供服务，访问HTTP监听正常，访问HTTPS监听出现排版错误，如何排查？]
(<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#http和https使用同样的后端提供服务，访问http监听正常，访问https监听出现排版错误，如何排查？>)
- * [BLB释放监听后，流量是否继续转发？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#blb释放监听后，流量是否继续转发？>)
- * [负载均衡是否可以获取-Client-端-IP地址？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#负载均衡是否可以获取-client-端-ip地址？>)
- * [开启了通过x-forwarded-for来获取客户端IP，但还是没拿到真实CIP，如何排查？]
(<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#开启了通过x-forwarded-for来获取客户端ip，但还是没拿到真实cip，如何排查？>)
- * [当负载均衡器正在运行过程是否可以调整后端rs的数量？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#当负载均衡器正在运行过程是否可以调整后端rs的数量？>)
- * [IPv6-BLB挂载BBC BCC失败？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#ipv6-blb挂载bbcbcc失败？>)
- * [百度BLB是否支持-ecc-p256的-服务器证书与CA证书？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#百度blb是否支持-ecc-p256的-服务器证书与ca证书？>)
- * [BLB如何关联安全组？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#blb如何关联安全组？>)
- * [如何修改监听的健康检查配置？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#如何修改监听的健康检查配置？>)
- * [负载均衡健康检查探测源-IP-网段是100640010？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#负载均衡健康检查探测源-ip-网段是100640010？>)
- * [健康检查提示异常该如何处理？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#健康检查提示异常该如何处理？>)
- * [健康检查提示正常，无法访问业务需要如何处理？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#健康检查提示正常，无法访问业务需要如何处理？>)
- * [BLB的健康检查频率是否过高？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#blb的健康检查频率是否过高？>)
- * [BLB的控制台上后端服务器健康检查状态的转变生效时间是多少](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#blb-的控制台上后端服务器健康检查状态的转变生效时间是多少>)
- * [为什么100网段的IP在访问后端服务器？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#为什么100网段的ip在访问后端服务器？>)
- * [为什么100网段的IP在访问后端服务器？](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#为什么100网段的ip在访问后端服务器？>)
- * [LB实例开启CIP透传，为什么server上还能收到100640010-网段内的地址建连？]
(<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#lb实例开启cip透传，为什么server上还能收到100640010-网段内的地址建连？>)
- * [为什么server上还能收到大量100640010-网段内，以3和4结尾的IP作为源IP向server建连？]
(<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#为什么server上还能收到大量100640010-网段内，以3和4结尾的ip作为源ip向server建连？>)
- * [负载均衡实例的后端服务器状态怎么查看](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#负载均衡实例的后端服务器状态怎么查看>)
- * [负载均衡实例添加后端服务器后，后端服务器过多久开始承载流量](<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#负载均衡实例添加后端服务器后，后端服务器过多久开始承载流量>)
- * [在后端服务器上看到了很多100段-ip-发过来的健康检查请求，这个后面会减少吗]
(<https://cloud.baidu.com/doc/BLB/s/Ujwvx01ul#在后端服务器上看到了很多100段-ip-发过来的健康检查请求，这个后面会减少吗>)

安全类问题

- * [发现很多IP地址在频繁地访问我的BCC，是什么原因？]([BLB/常见问题/安全类问题.md#发现很多IP地址在频繁地访问我的BCC，是什么原因？](#))

* [BLB支持HTTPS协议吗？](BLB/常见问题/安全类问题.md#BLB支持HTTPS协议吗？)

* [BLB可以防攻击吗？](BLB/常见问题/安全类问题.md#BLB可以防攻击吗？)

计费类问题

* [BCC加入BLB会导致其计费规则发生变化吗？](BLB/常见问题/计费类问题.md#BCC加入BLB会导致其计费规则发生变化吗？)

* [健康检查产生的流量会算入到计费流量中吗？](BLB/常见问题/计费类问题.md#健康检查产生的流量会算入到计费流量中吗？)

* [BLB提供哪些计量数据？](BLB/常见问题/计费类问题.md#BLB提供哪些计量数据？)

专属集群类问题

* [购买了专属集群，创建BLB实例时为什么看不到这个专属集群](https://cloud.baidu.com/doc/BLB/s/Nlowjg9i6#购买了专属集群，创建blb实例时为什么看不到这个专属集群)

* [专属集群上创建的BLB实例，主备模式主主模式如何选择？](https://cloud.baidu.com/doc/BLB/s/Nlowjg9i6#专属集群上创建的blb实例，主备模式主主模式如何选择？)

* [BLB实例在专属集群共享集群间或者不同专属集群间迁移，数据流是否有损，以及预期有损时长多少？]
(https://cloud.baidu.com/doc/BLB/s/Nlowjg9i6#blb实例在专属集群共享集群间或者不同专属集群间迁移，数据流是否有损，以及预期有损时长多少？)

* [专属集群上的BLB实例执行主备机房切换主备主主模式切换时，数据流是否有损，以及预期有损时长多少？]
(https://cloud.baidu.com/doc/BLB/s/Nlowjg9i6#专属集群上的blb实例执行主备机房切换主备主主模式切换时，数据流是否有损，以及预期有损时长多少？)

* [购买了专属集群，创建BLB实例时为什么看不到这个专属集群](https://cloud.baidu.com/doc/BLB/s/Nlowjg9i6#购买了专属集群，创建blb实例时为什么看不到这个专属集群-1)

* [专属集群上创建的BLB实例，主备模式主主模式如何选择？](https://cloud.baidu.com/doc/BLB/s/Nlowjg9i6#专属集群上创建的blb实例，主备模式主主模式如何选择？-1)

* [BLB实例在专属集群共享集群间或者不同专属集群间迁移，数据流是否有损，以及预期有损时长多少？]
(https://cloud.baidu.com/doc/BLB/s/Nlowjg9i6#blb实例在专属集群共享集群间或者不同专属集群间迁移，数据流是否有损，以及预期有损时长多少？-1)

* [专属集群上的BLB实例执行主备机房切换主备主主模式切换时，数据流是否有损，以及预期有损时长多少？]
(https://cloud.baidu.com/doc/BLB/s/Nlowjg9i6#专属集群上的blb实例执行主备机房切换主备主主模式切换时，数据流是否有损，以及预期有损时长多少？-1)

一般类问题

BLB可以支持非百度智能云的服务器吗？

可以，目前BLB的后端服务器除了支持百度智能云的[云服务器BCC](https://bce.baidu.com/product/bcc.html)、[专属服务器DCC](https://bce.baidu.com/product/dcc.html)以及[弹性裸金属服务器BBC](https://cloud.baidu.com/product/bbc.html)外，还能通过[应用型BLB的IP组](https://cloud.baidu.com/doc/BLB/s/Njwxnt8e#配置ip组)挂载非百度智能云的服务器。

BLB实例启用/不启用公网访问有什么区别？

如不启用公网访问，则默认提供内网IP来进行内网流量的均衡服务；如启用，则为您购买并绑定公网IP来实现对公网访问流量的均衡服务。

是否可以在百度智能云上自己搭建负载分担服务？

您可以通过购买BCC云主机，并自行部署HAproxy、Nginx实现负载均衡服务；但是从成本、维护性和可扩展性角度考虑，建议您直接使用BLB服务。

BLB实例总数超过上限，目前业务需要能否增加BLB配额？

目前一个用户默认最多可创建60个BLB实例，如果需要创建更多BLB实例，可通过[配额管理] (https://cloud.baidu.com/doc/QUOTA_CENTER/s/iks9z7yuw) 进行申请。其他配额可参见：[BLB使用限制] (<https://cloud.baidu.com/doc/BLB/s/Hjwvxnouu>)。

后端服务器没有公网EIP，能否通过BLB访问公网？

BLB只作为负载均衡进行使用，不支持后端服务器通过BLB访问外网。您可以通过BCC绑定EIP，或创建NAT网关实现。

BLB支持IPv6吗？

BLB现已支持IPv6，需要白名单通过才能使用，有需要的用户请提交工单进行申请。

windows操作系统可以使用BLB吗？

BLB负责对HTTP和HTTPS协议请求的解析和转发，应用防火墙WAF负责在中间提供安全防护功能。对于后端服务器的操作系统没有特殊要求。

关于ping 负载均衡实例内网IP 的说明

ping报文是由负载均衡设备代答的，无法测试到服务端的连通性。如果要测试客户端到服务端连通性，请采用telnet等方式验证。

对于单台客户端ping方式出现丢包时，请尝试不同客户端同时ping，排除单台客户端网络异常问题。

对于多台客户端ping方式均丢包或不通时，请提工单进一步排查。

关于Telnet负载均衡监听端口的说明

创建四层（TCP）监听器后，如果不绑定后端服务器，则无法Telnet通监听端口；绑定后端服务器后，健康检查正常后，可以Telnet通监听端口。

什么情况下需要 HTTPS 双向认证？

有些客户对数据访问安全要求较高，如涉及到金融服务的客户等。他们不仅需要对服务端进行 HTTPS 认证，也需要对客户端来源进行 HTTPS 认证。

为了满足这些客户的需求，我们推出 HTTPS 双向认证功能。

BLB是否支持同一监听下挂载同一后端服务器的不同端口？

当前应用型BLB可以通过挂载IP组的方式支持后端rs端口不相同功能。

BLB支持同一端口创建多个不同协议的监听吗？

支持，用户可以在控制台或者通过openapi 创建监听选择不同协议。【注意】tcp/http协议不能同时存在于同一端口。

BLB支持同region跨vpc访问吗？

支持。用户可以创建应用型BLB，通过挂载IP组的形式，在控制台勾选同跨vpc选项。用户可以通过配置对等连接打通vpc，提交工单由技术支持配合配置obip路由。【注意】当前只支持虚机为普四及其以上机型。当前新创建的虚机都是普四以上机型，所以购买新的虚机均支持此能力。

BLB 和后端云服务器之间是使用公网还是内网通信？

在流量到达BLB实例的指定监听端口后，BLB与云服务器之间都是使用内网通信。当通过弹性公网IP（EIP）访问BLB产品时在客户端云服务器与弹性公网EIP之间是公网通讯，后端BLB至云服务器依然采用内网通信。

BLB UDP 协议与 TCP 协议有什么区别？

TCP 是面向连接的协议，在正式收发数据前，必须和对方建立可靠的连接。UDP 是面向非连接的协议，它在数据发送前不需要与对方先进行三次握手建立链接，而是直接进行数据包发送。UDP 协议主要适用于关注实时性而相对不注重可靠性的场景，如视频聊天、金融实时行情推送、DNS、物联网等。

BLB可以为哪些 TCP 端口执行负载均衡？

BLB产品 TCP类型监听端口支持范围1 - 65535，建议使用业界常用的端口提供服务，有一些端口可能存在公网访问时被封禁的风险。

BLB转发请求的长度限制是多少？是否支持调整？

访问BLB请求的URI长度不能超过32 KB，请求header长度也不能超过32 KB，但是支持通过client_header_buffer_size和large_client_header_buffers参数进行自定义调整

large_client_header_buffer参数进行自定义调整。

- * 如果客户端的请求大小超限，可能会返回400或414状态码。
- * 如果数据量很大建议采用POST传输数据，POST请求的body最大支持 10 GB。

BLB级联场景使用？

对于LB级联使用场景，建议都是用非三角模式的数据流。不要使用DR，IMF三角模式的数据流，可能会导致数据流不通。不建议用户级联BLB使用，可能会导致数据流不通。如果用户在使用过程中出现不通的情况，可以联系技术支持。

负载均衡 Cookies 会话保持方式的原理是什么？

负载均衡cookies会话保持有cookie植入（insert）和cookie重写（rewrite）两种方式：植入状态下会返回cookie的超时时间给客户端。重写状态下没有超时时间。

在 Cookie 植入模式下，BLB 负责将cookie植入客户端，后端服务器无需作出任何修改。当客户端进行第一次请求时，客户端 HTTP 请求（不带 Cookie）进入 BLB，BLB 根据负载均衡算法策略选择后端一台服务器，并将请求发送至该服务器，后端服务器进行 HTTP 回复（不带 Cookie）被发回 BLB，然后 BLB 通过set-cookie头插入响应返回给客户端。

当客户请求再次发生时，客户 HTTP 请求（带有上次 BLB 插入的 Cookie）进入 BLB，然后 BLB 读出 Cookie 里的会话保持数值，cookie值与后端服务器进行对比，将HTTP请求（带有与上面同样的 Cookie）发到指定的服务器，然后后端服务器进行请求回复，由于服务器并不写入 Cookie，HTTP 回复将不带有 Cookie，此时BLB也不会再把cookie插入响应返回给客户端。

如果出现BLB解析客户端cookie失败或者对应cookie的rs健康检查DOWN的情况，那么将根据用户配置的负载均衡算法重新选择后端服务器，并重新将cookie植入响应返回给客户端。

在 Cookie 重写模式下，重写模式使用服务端的标识符来覆盖后端服务器中用于进行会话保持的cookie。不同于insert模式，如果后端服务器在响应头中没有设置对应名称的cookie，insert模式会直接插入cookie，而rewrite模式则会认为当前请求不需要进行session sticky，不进行任何操作。当客户端进行第一次请求时，客户端 HTTP 请求（不带 Cookie）进入 BLB，BLB 根据负载均衡算法策略选择后端一台服务器，并将请求发送至该服务器，后端服务器进行 HTTP 回复（带 Cookie）被发回 BLB，然后 BLB 解析后端服务器返回的响应，对比BLB设置好的cookie name，找到后，重写cookie name为设置的cookie name，然后插入客户端。

当客户请求再次发生时，客户 HTTP 请求（带有上次 BLB 插入的 Cookie）进入 BLB，然后 BLB 读出 Cookie 里的会话保持数值，cookie值与后端服务器进行对比，将HTTP请求（带有与上面同样的 Cookie）发到指定的服务器。后端服务器进行 HTTP 回复（带 Cookie）被发回 BLB，然后 BLB 解析后端服务器返回的响应，对比BLB设置好的cookie name，找到后，重写cookie name为设置的cookie name，然后插入客户端。

HTTPS 监听使用什么端口？

不强制，建议使用443端口。

tcp监听修改为https/http监听后流量统计为什么会下跌？

tcp监听是四层负载均衡，流量统计的数据为ip数据报文的长度；
将tcp监听修改为http/https监听，是七层负载均衡，流量统计的数据为http/https报文的body的长度；
因此tcp监听修改为http/https监听后从流量统计上看会有一些数值上的下降。

什么是后端服务器权重？

用户可以指定后端服务器池内各 RS 的转发权重，权重比越高的 RS 将被分配到更多的访问连接，用户可以根据后端 RS 本身负载和对外服务能力情况来设定。

请求响应为500、502、503、504状态码的解释及可能原因是什么？

500 (Internal Server Error)

后端服务器内部错误，无法执行请求。

可能原因：

- * 后端服务器直接返回500，BLB透传后端状态码给客户端，请排查后端服务器返回500的原因。
- * 后端服务器在发送完响应之前异常关闭连接，请在后端服务器抓包确定并排查请求异常关闭的原因。

502 (Bad Gateway)

HTTP或HTTPS监听接收到客户端连接请求后，BLB由于无法正常将请求转发至后端服务器或无法从后端服务器收到响应，则会向客户端发送HTTP 502 Bad Gateway状态码。

可能原因：

- * 后端服务器直接返回502，BLB透传后端状态码给客户端，请排查后端服务器返回502的原因。
- * BLB后端服务器返回其他错误的状态码，但BLB统一返回502。建议排查访问日志的upstream_status和status字段或抓包排查后端服务器是否有异常。
- * BLB与后端服务器之间TCP通信异常，请排查后端服务器的状态是否正常，服务端是否被正常监听，或抓包查看TCP握

手报文是否正常。

- * 后端服务器Backlog已满，导致报文丢弃。建议通过netstat查看后端服务器的网络统计结果是否有drop的计数，例如netstat -s | grep -i listen。
- * 同一个服务器组的所有后端服务器健康检查都异常，与之关联的BLB实例无法转发请求，直接返回502。建议通过日志排查后端服务器是否有异常，或者检查健康检查配置是否存在异常。
- * 客户端发送的报文长度超过了后端服务器的最大传输单元（Maximum Transmission Unit，MTU），表现为健康检查或报文较短的包正常，但报文较长的包异常。建议在后端服务器抓包分析报文长度是否符合要求。
- * 后端服务器返回的报文格式异常或者有非法的HTTP标头。建议在后端服务器抓包，确认HTTP格式是否正常。
- * BLB后端服务器没有及时完成请求处理，请排查后端服务器的日志，并查看对应后端服务器的CPU、内存等的占用率。

503 (Service Temporarily Unavailable)

暂时无法使用服务器。

可能原因：

- * 后端服务器直接返回503状态码，BLB透传后端状态码给客户端，请排查后端服务器返回503的原因。
- * 请求没有请求到BLB对应的监听上，监听配置在BLB未找到，可能是请求没有匹配到配置的监听策略。

504 (Gateway Time-out)

后端服务器响应超时。

可能原因：

- * 后端服务器直接返回504状态码，请排查后端服务器的负载是否过高。
- * BLB向后端服务器建立连接时，后端服务器建立连接超时，超时时间默认设置为30秒，您可以查看访问日志中的upstream_connect_time字段是否为30秒或者大于30秒。建议抓包排查后端服务器响应超时的原因。
- * 后端服务器负载增大，响应请求的时间大于所配置请求超时时间。比如配置请求超时时间是60秒，如果响应时间为60.001秒，BLB将返回504。访问日志中查看upstream_response_time。

如何检查请求不均衡？

一般情况下，影响负载均衡分配的因素包括分配策略、会话保持、长连接、权重等。换言之，最终是否均匀分配不仅与分配策略相关，还与使用的长短连接、后端的性能负载等相关。

1. 检查是否开启了会话保持。如果配置了会话保持，而客户端的个数又比较少时，很容易导致不均衡。
2. 检查后端云服务器的健康检查状态是否正常，特别要关注下是否有健康检查状态一会正常一会异常的情况。健康检查异常或者状态切换都会导致流量不均衡。
3. 检查负载均衡算法是否是源IP算法。此时同一个IP发过来的请求都会分发到同一个后端，导致流量不均衡。
4. 后端服务是否开启了TCP keepalive保持长连接。如果开启，则有可能因为长连接上的请求数不同导致流量不均衡。
5. 将云服务器添加到LB后端时是否设置了权重，权重不同，分发的流量也不同。
6. 后端服务器添加到LB时间不同也会影响请求分布不均衡。

后端RS上流量不均衡，如何排查？

BLB当前选择RS的调度单位是基于连接调度，当一条新建连接选中某一个RS后，流量将一直转发到该RS直到连接结束或超时。

因此当前端RS上出现流量不均衡的情况，需要检查连接请求是否均衡。

可以从以下几点来排查：

- * 检查调度算法，是否使用了基于权重的调度算法，比如加权轮询（wrr），RS上配置的权重是否一致，如果不一致，则有可能出现流量不均衡。
- * 检查后端所有RS上的每秒新建连接数（cps）是否均衡，如果不均衡，则有可能出现流量不均衡。
- * 检查后端所有RS是否有长短连接混合使用的情况。
- * 检查连接数是否太少。

如何检查负载均衡业务访问延时大？

一般从如下几方面检查：

1. 将EIP绑定到后端云服务器，不经过弹性负载均衡直接访问后端服务，查看访问延时。用来判断是弹性负载均衡的问题，还是前端网络问题或者后端服务问题。
2. 查看业务流量是否超过了EIP的带宽限制，超带宽会产生拥塞、丢包等异常情况。
3. 如果直接访问后端存在业务访问延时大，需要排查后端服务是否压力过大，是否配置了安全策略等。
4. 查看异常主机数的监控来判断后端云服务器的健康检查状态是否有跳变。在后端服务状况不稳定时，因为弹性负载均衡的重试机制，如果连接一台后端超时，请求会重新发往下一台后端，请求成功，这样业务就表现为访问成功，但是延时很大。
5. 如果问题依然存在，请通过工单联系我们。

BLB实例是否支持 Client IP 透传？

目前 BLB 的 TCP/UDP 类型，新增挂载云上 BCC 类型的后端服务器，默认开启 Client IP 透传（在后端服务器上看到访问报文的源地址为 Client IP）。

HTTP/HTTPS类型的监听，目前暂不支持 CIP 透传，用户可开启『获取真实IP X-Forwarded-For方式获取客户端真实IP』选项，在后端服务器上通过解析 HTTP header 头来获取 Client IP。

SSL 类型监听，目前支持 ProxyProtocol 协议，用户可启用该选项，同时在后端服务器上解析 ProxyProtocol 标准协议的方式，来获取 Client IP。

BLB实例后端服务器是分批次发回响应请求的，但是在客户端看到的是只收到一个请求，BLB 是否支持不缓存后端服务器的响应请求？

BLB 实例开启代理缓冲功能，BLB 会将后端服务器的响应请求缓存在本地内存或磁盘中，再以合适的速度发送给客户端。

BLB 提供定制配置功能 给客户，来关闭此项功能。

关闭代理缓冲功能方法：开启 定制配置 选项，在命令输入框中输入 『proxy_buffering:off』

其他的定制配置字段参见：[定制配置操作指南](https://cloud.baidu.com/doc/BLB/s/gkk3kb8ic)。

BLB产品支持哪些调度算法？

BLB目前支持加权轮询、最小连接数、源IP的调度算法。

1. 加权轮询

调度算法设置为加权轮询时，会根据服务器之间的权重设置依次查询服务器，使得各服务器的实际连接数符合权重设置。常用于短连接服务，如HTTP服务。

2. 最小连接数

调度算法设置为最小连接数时，权重设置失效，blb每次选择当前连接数最小的后台服务器机建立连接。常用于长连接服务，如数据库服务。

3. 源IP

调度算法设置为源IP时，权重设置失效，将请求的源IP进行hash运算后派发请求至某匹配的服务器，可以保证同一个客户端IP的请求始终被派发至某特定的服务器。

在何种情况下购买BLB要搭配EIP？

用户需要申请一个BLB实例时，如需要公网服务，则给其绑定一个EIP即可。

性能类问题

一个用户最多可以创建多少个BLB实例？BLB产品在使用时有哪些规格与配额限制？

目前一个用户最多支持创建60个BLB实例，如果需要创建更多实例，欢迎通过[配额管理](https://cloud.baidu.com/doc/QUOTA_CENTER/s/iks9z7yuw)进行申请。

BLB产品配额限制请参考[文档](https://cloud.baidu.com/doc/BLB/s/Hjwvxnouu)，如需申请更多配额，请通过工单申请。

BLB是否限制带宽？

BLB实例自身没有带宽限制；但若需公网访问，您需要绑定弹性公网IP，弹性公网IP的带宽范围:普通EIP最大带宽200Mbps，增强型BGP EIP最大带宽5000Mbps。了解[弹性公网IP](https://cloud.baidu.com/doc/EIP/s/Pjwvz2pno)。

BLB支持哪些转发规则？

- 4层（TCP协议、UDP协议）监听器支持加权轮询、最小连接数、源IP三种转发规则；
- 7层（HTTP协议、HTTPS协议）监听器支持加权轮询、最小连接数两种转发规则。

目前BLB提供哪些监控信息？

- 4层提供监听器级别的从外网进入的流量、流向外网的流量、实例数量、网络流量、网络数据包、活跃连接数。

- 7层还提供监听器级别的从外网进入的流量、流向外网的流量、实例数量、网络流量、网络数据包、活跃连接数、请求数。

健康检查是如何实现的？

BLB通过健康检查自动屏蔽故障节点，实现业务高可用。

BLB的健康检查分为TCP协议健康检查和HTTP协议监控检查两种。

- TCP协议健康检查通过发送tcp syn包实现；
- HTTP协议健康检查通过发送HTTP GET请求实现。

BLB是否支持会话保持功能？

4层（TCP协议、UDP协议）监听器可以通过源IP转发规则实现会话保持；7层（HTTP、HTTPS协议）通过植入/重写cookie实现会话保持。

后端BCC需要外网带宽吗？其与BLB的服务能力有关吗？

如果只通过 BLB 连接 BCC，此 BCC 可以没有公网带宽。

BLB是否具有DDOS防护功能？

ddos攻击在超过5G后，EIP会被网络运营商设置黑洞，无法解绑无法访问，服务器本身是可正常使用的，黑洞的只是BLB上的公网EIP，DDOS防御建议您使用高防IP来防护，[详细参考](<https://cloud.baidu.com/doc/DDOS/s/HkOf63jrg>)

负载均衡是否支持巨型帧（Jumbo frames）转发？

巨型帧是指有效负载超过IEEE 802.3标准所限制的1500字节的以太网帧，增大的有效载荷百分比有助于提高链路利用率，获得更好的网络性能。BLB负载均衡目前暂不支持巨型帧的转发和分片后转发。

故障类问题

BCC实例异常会影响BLB服务吗？

BLB会主动检查后端BCC实例的状态，如果某个BCC实例异常，BLB就会把新的请求发送到其它健康的BCC实例上，所以只要有一台正常的BCC实例，整个服务就可以正常工作。

健康检查的状态总是异常怎么办？

1.确保云主机上的健康检查端口有进程/服务在监听。

a)检查BCC实例是否正常；

b)检查BCC实例上部署的服务是否正常，并确认健康检查端口有进程、服务在监听器。注意TCP监听器的健康检查端口默认和后端服务端口一致。

2.确保云主机的安全组策略做了正确的配置，详情请参看[后端服务器安全组设置](BLB/操作指南/普通型BLB实例/创建BLB普通型实例.md#设置后端服务器安全组)。

3.您可以手动请求云主机的端口，来检查是否可以正常访问。

- 如果使用的是TCP健康检查，可以使用telnet/nmap命令尝试和端口建立TCP连接，eg. `nmap ipaddress -p port`；
 - 如果使用的是HTTP健康检查，可以使用curl命令确保访问返回的HTTP状态码是2XX或3XX，eg. `curl -I ipaddress:port`。
- 4.如后端成员是IP地址，该IP地址需要和应用型BLB实例在同一个VPC下。

转发规则是源IP时，支持回环避免吗？

负载均衡支持同一个BCC既作为客户端又作为服务器，此时BLB看到的 Client IP 和 Server IP 是一样的，需要注意的是源IP的转发规则不支持回环避免。您可以选择加权轮询、加权最小连接数的转发规则来实现回环避免。

的转发规则防止出现回环避免。您可以选择加权轮询、加权最小连接数的转发规则来避免回环避免。

通过IP 分片访问负载均衡TCP/UDP/ICMP协议，网络不通怎么办？

BLB暂不支持分片报文的转发，分片报文到达负载均衡后会被丢弃；因此使用TCP\UDP进行通信时，需要严格控制客户端和服务端发出的数据包的整包大小不超过默认的MTU(1500字节)。

udp服务报文被分片常见的现象有服务不通。

tcp服务报文被分片常见的现象tcp三次握手，4次挥手等小包文传输没问题，业务请求和响应大报文被丢包引起重传，引起连接不通。

解决方案：缩小客户端和服务端发送消息大小，以确保数据包不分片，且尺寸小于等于1500字节。

客户端为何无法访问负载均衡？

四层监听的后端服务器默认不支持同时作为客户端和服务端使用，检查client和server是否是同一台服务器

健康检查异常，检查RS健康检查是否正常

检查后端服务是否正常，监听是否正常，client直接访问RS服务是否正常工作。

后端服务上是否有防火墙策略，是否放开服务端和client ip的访问权限。

访问控制列表、安全组是否放开相应的client ip和port访问权限。

配置类问题

TCP超时时间配置？

指定TCP连接的超时时间，输入范围为10~4000间的整数，默认900秒。TCP超时时间过短会导致长期不用的链接中断，过长会导致后端服务器一直保持连接，建议您根据自己的后端服务程序的需求调整。

后端服务器可以向所在负载均衡实例发送请求吗？

负载均衡支持同一个BCC既作为客户端又作为服务器，此时BLB看到的 Client IP 和 Server IP 是一样的，需要注意的是源IP的转发规则不支持回环避免。您可以选择加权轮询、加权最小连接数的转发规则来实现回环避免。

健康检查的相关配置，是否有相对合理的推荐值？

关于健康检查的参数配置，提供如下参考建议：

- 健康检查端口：默认使用后端服务端口，如果有特殊需求，可以自定义。
- 响应超时时间：3秒
- 健康检查间隔：3秒
- 不健康阈值：3
- 健康阈值：3

一组云服务器可以搭建多个网站并同时进行负载均衡吗？

可以。

- 一个BLB实例最多支持20个服务监听配置，每个服务监听配置可以对应后端云服务器上的一个应用（BLB的前端端口对应后端BCC上应用服务端口）。

- 可通过对后端BCC上的多个应用，搭建多个网站并进行负载均衡。

负载均衡是否可以获取 Client 端 IP？

七层 IPv4 和 IPv6 负载均衡监听提供 X-Forwarded-For 的方式获取访问者真实 IP，负载均衡侧默认开启，需要后端服务做相应配置来获取 Client IP。

> **注意：**如果客户端发给BLB的请求头已经包含了HTTP Header: X-Forwarded-For，BLB会改写X-Forwarded-For，将客户端真实IP以“,”为分隔符拼接在原内容之后，比如：原内容, 客户端真实IP。

四层IPv4 和 IPv6 负载均衡（TCP/UDP监听）服务因后端支持挂载多种类型的后端服务器(BCC/BBC/物理机/容器自定义 Pod IP/专线对端各种类型服务器等)，每种后端服务器类型由于网络架构限制需要采用不同的数据流转发，拿到Client IP的

方式可能存在不一样的方式；常规BCC虚拟机与二代BBC以实例形态挂载在LB做后端服务器时，支持直接从 server 端获取真实的Client IP，无需额外配置。下述一些特殊场景不支持直接从 server 端获取真实的Client IP，需要特殊配置。

* IPv6 NAT64 负载均衡不支持直接获取Client IP，可以选择启用ProxyProtocol协议以获取客户端源IP、源Port，访问目的IP、目的Port，访问协议，开启前请务必先配置后端服务器支持ProxyProtocol协议

* 以一代BBC、物理机、容器自定义Pod IP、专线对端各种类型服务器等或直接以IP组挂在四层负载均衡场景时，使用的是经典的fullnat转发模式，负载均衡将报文中的源ip以SNAT转成给后端服务器的网段IP（当前是100.64.0.0/10网段），而把真实的Client IP地址放入TCP报文自定义Option 中或IP Option中，这样后端服务器无法获得真实源 IP，因此需要通过安装开源的 TTM 模块或选择启用ProxyProtocol协议进行获取。如需了解详情可[提交工单] (<https://console.bce.baidu.com/ticket/#/>)联系技术支持。

同一个BLB是否可以配置两个https证书，分别监听443和442端口呢？

可以配置多个https证书。在监听设置中添加监听设置后，每一个https监听设置都可以选择一个服务器证书的。

一台服务器能否配置在2个不同的BLB上？

一台服务器可以添加到两个BLB中作为后端服务器。

BLB监听设置中，TCP监听和HTTP监听有什么区别吗？

4层（TCP协议）监听器支持加权轮询、最小连接数、源IP三种转发规则；7层（HTTP协议、HTTPS协议）监听器支持加权轮询、最小连接数两种转发规则；TCP是传输层协议，HTTP是应用层的协议。

负载均衡会得到多个IP，哪一个是终端用户的真实IP呢？

对于7层（HTTP协议、HTTPS协议）监听器，预先在配置BLB服务监听时开启“获取真实访问IP”的功能，可以通过HTTP Header: X-Forwarded-For获取，对于4层（TCP协议）监听器，无需任何配置即可获得来自外网的来访者真实IP；如果来访者是内网BCC，暂时无法获得来访者真实IP，也请您注意下如果客户端发给BLB的请求头已经包含了HTTP Header: X-Forwarded-For，目前BLB的处理逻辑是会在请求头里再额外添加一个新的HTTP Header: X-Forwarded-For，即请求头里会有两个HTTP Header: X-Forwarded-For，后一个是BLB添加的。

BLB怎么设置会话保持？

关于BLB会话保持的配置操作可以参考以下[文档](BLB/操作指南/普通型BLB实例/创建BLB普通型实例.md#配置监听器)。

如何对BLB资源状况进行定期的全面检查？

您可开通“云顾问 Cloud Advisor”服务，定期获得云上资源在安全、可用性、性能、成本方面的检查报告。报告中包含若干BLB相关检查项目，如：BLB-监听安全、BLB-没有后端服务器、BLB-负载均衡等。了解或开通云顾问服务，请前往 [云顾问主页](<https://cloud.baidu.com/product/aca.html>)

应用型BLB通过IP组挂载IDC侧的服务器健康检查失败？

IDC侧通过专线与VPC打通后，应用型BLB通过IP组挂载IDC侧的服务器健康检查失败，通常有两种解决方式：（1）配置100网段的健康检查路由；（2）用户配置最低默认路由如0.0.0.0/0到专线网关。第二种用户这边如果没其它冲突会更友好，不用关注100网段路由。

关于内网回环问题的说明

为了避免把一个服务器既当client又当后端server，这样会导致client ip和server ip是一样的，导致访问不通，BLB调度里有回环避免的开关，默认现在是开启的。

所以当客户端要当后端服务器时，请至少绑定两个后端服务器，否则是会访问不通。当客户端访问负载均衡时，负载均衡会自动调度到非此客户端A的后端服务器上。

目前wrr（加权轮询调度）和wlc（最小连接数调度）可以在console界面上配置回环避免

四层负载均衡后端RS权重置为0与删除RS的区别

后端权重置为0：新建的连接不再调度转发到此后端服务器上，但存量的连接会继续转发，直到无流量超时断开连接或client/server结束连接为止；

删除 RS：新建连接都不会再调度转发到此后端服务器上，存量udp连接会不再转发，存量tcp连接虽然会继续转发，但存量tcp连接会设置加快老化时间；

另外注意wrr和wlc是带权重的调度策略，而rr是不带权重的调度策略，使用rr调度策略时调整后端权重为0没有作用

七层负载均衡后端RS权重置为0与删除RS的区别

HTTPS/HTTP/SSL_TCP 监听的后端权重置为0 和 删除 RS 当前逻辑几乎无区别：新建的连接不再调度转发到此后端服务器上，但存量的连接会继续转发，直到无流量超时断开连接或client/server结束连接为止。

监听上的连接数较少时，权重相同的wrr或rr调度策略的各个rs为何连接数不相同？

四层负载均衡上一个节点上有好几个worker线程，现在实现的wrr和rr调度就基于perworker的，也就是说每个worker的调度算法是单独调度的，

当监听上的连接数较少时，网卡RSS分配到各个worker的数据流的首包的数量并不完全一样，甚至有的worker有流量，有的worker没流量，这导致wrr和rr调度在各个worker线程上调度的结果也会有差异，所以会出现监听上的连接数较少时，权重相同的wrr或rr调度策略的各个后端rs连接数的差异；

当监听上的连接数很多时，网卡RSS分配到各个worker的数据流的首包的数量相对是均匀的，wrr和rr调度在各个worker之间也会相对比较均匀，所以后端rs连接数也会比较均匀；

四层负载均衡源地址哈希调度算法后端RS权重置为0仍能接收处理流量？

源地址哈希调度算法的实现中是不关注权重的调度策略，使用这个调度策略时调整后端RS权重为0不会起作用，仍然可以新建连接或存量连接的报文转发给后端RS。

四层负载均衡删除后端服务器RS后，删除的RS上存量连接如何处理？

1. 如果删除tcp或udp监听上的所有RS后，客户端发起的存量连接不再转发直到连接断开或超时，后端RS也接收不到报文。
2. 对于tcp监听存在多个RS时，删除某一个RS后，对应这个RS上的存量连接默认会继续接收到转发报文，直到连接结束或超时。这种情况建议在RS删除时先主动关闭服务端正常结束连接。
3. 对于udp监听存在多个RS时，删除某一个RS后，对应RS上的存量连接不再转发，后续报文触发新建并重新选择可用的RS。

如何启用WebSocket支持？

无需额外配置，当选用HTTP监听时，默认支持无加密版本WebSocket协议（WS协议）；当选择HTTPS监听时，默认支持加密版本的WebSocket协议（WSS协议）。

是否支持在业务不中断的前提下，将共享型负载均衡升级为规格型负载均衡？

支持，在列表页面点击“性能调整”按钮进行变更。

LB支持上传证书吗？

不支持。LB是直接调用IAM证书服务的证书，需要您将证书上传至证书管理控制台，因此不支持在LB进行证书上传操作。证书管理控制台在右上角安全认证中，或在负载均衡控制台左侧点击证书管理可以跳转到证书管理控制台

一个监听器可以绑定多少个 HTTPS 证书？

如果用户使用 HTTPS 单向认证，则一个监听只能绑定一个服务器证书；若用户使用 HTTPS 双向认证，则一个监听需要绑定一个服务器证书 + 一个 CA 证书。

证书区分地域吗？

不区分。证书购买和颁发后，安装部署不受地域的限制。

证书需要上传到后端服务器吗？

不需要，负载均衡 HTTPS 提供证书管理系统管理和存储用户证书，证书不需要上传到后端服务器，用户上传到证书管理系统的私钥都会加密存储。

HTTP和HTTPS使用同样的后端提供服务，访问HTTP监听正常，访问HTTPS监听出现排版错误，如何排查？

因BLB默认不会修改js文件的传输，所以建议您从以下方向进行排查

- * 浏览器与证书安全级别的兼容情况
- * 核证书发布的正规情况，确保发布正规第三方证书
- * 检查后端服务器代码，是否存在HTTPS页面加载HTTP资源情况。

BLB释放监听后，流量是否继续转发？

释放后，对应监听的流量都不再转发到后端RS。

负载均衡是否可以直接获取 Client 端 IP地址？

七层 IPv4 和 IPv6 负载均衡监听提供 X-Forwarded-For 的方式获取访问者真实 IP，负载均衡侧默认开启，需要后端服务做相应配置来获取 Client IP。

注意：如果客户端发给BLB的请求头已经包含了HTTP Header: X-Forwarded-For，BLB会改写X-Forwarded-For，将客户端真实IP以“,”为分隔符拼接在原内容之后，形如：原内容, 客户端真实IP。

四层IPV4 和 IPV6 负载均衡（TCP/UDP监听）服务后端支持挂载各种类型后端服务器（ECS/DBS/物理机/容器化应用）

四层IPv4 和 IPv6 负载均衡（TCP/UDP监听）服务因后端支持挂载多种类型的后端服务器（BCC/BBC/物理机/容器自定义 pod ip/专线对端各种类型服务器等），每种后端服务器类型由于网络架构限制需要采用不同的数据流转发，拿到client ip的方式可能存在不一样的方式；常规BCC虚拟机与二代BBC以实例形态挂载在LB做后端服务器时，支持直接从 server 端获取真实的 client IP，无需额外配置。下述一些特殊场景不支持直接从 server 端获取真实的 client IP，需要特殊配置。

* IPv6 NAT64 负载均衡不支持直接获取 Client IP, 可以选择启用ProxyProtocol协议以获取客户端源IP、源Port，访问目的IP、目的Port，访问协议，开启前请务必先配置后端服务器支持ProxyProtocol协议

* 以一代BBC、物理机、容器自定义pod ip、专线对端各种类型服务器等或直接以ip组挂在四层负载均衡场景时，使用的是经典的fullnat转发模式，负载均衡将报文中的源ip以SNAT转成给后端服务器的网段ip（当前是100.64.0.0/10网段），而把真实的client ip地址放入TCP报文自定义 option 中或ip option中，这样后端服务器无法获得真实源 IP，因此需要通过安装开源的TTM模块或选择启用ProxyProtocol协议进行获取。请联系技术支持或提交工单详细了解

开启了通过X-Forwarded-For来获取客户端IP，但还是没拿到真实cip，如何排查？

若已经开通了 X-Forwarded-For来获取客户端ip，但还是没有拿到真实cip，可能是以下原因，建议进行排查

- * 建议检查BLB上配置情况，确保X-Forwarded-For配置正确
- * 建议检查协议情况，某些协议存在不支持 X-Forwarded-For头字段的情况，如WebSocket。
- * 建议检查是否经过BLB后是否引入其他链路模块及其情况，如使用了代理服务器转发请求，有可能是代理服务器上未配置正确，影响后续的转发。
- * 建议检查客户端情况，检查客户端是否存在删除或篡改 X-Forwarded-For头字段的可能性。
- * 建议检查后段服务上解析情况，检查后段服务上是否正确解析了X-Forwarded-For头字段。

当负载均衡器正在运行过程是否可以调整后端RS的数量？

负载均衡支持在任意时刻增加或减少后端服务器的数量，且支持不同的后端服务器切换操作。但是为了保证您对外业务的稳定，请确保在执行上述操作的时候关注后端服务器的健康检查状态，保证负载均衡器后端至少有一台可以正常运行的服务器。

IPv6 BLB挂载BBC/BCC失败？

请先排查一下BCC/BBC是否已分配有IPv6地址。

百度BLB是否支持 ECC p256的 服务器证书与CA证书？
支持

BLB如何关联安全组？

方式一：1 登录VPC控制台，安全组访问控制下的安全组，创建对应的安全组。

2 登录负载均衡控制台，在左侧导航栏，选择负载均衡实例，找到具体实例，点击实例进入详细信息界面。

3 在实例详细信息界面选择安全组页面，点击关联安全组，即可绑定安全组。

方式二：1 登录VPC控制台，安全组访问控制下的安全组，创建对应的安全组。

2 在安全组详情页面选择关联负载均衡，即可为对应的BLB关联安全组。

如何修改监听的健康检查配置？

1. 登录负载均衡控制台，在左侧导航栏，选择应用型负载均衡实例，找到具体实例，点击实例进入详细信息界面。
2. 在实例详细信息界面选择目标组 > 服务器组/IP组 页面，找到目标服务器/IP组，点击端口协议列的编辑按钮。
3. 在健康检查设置对话框，根据需求修改健康检查配置。然后单击确定。

负载均衡健康检查探测源 IP 网段是100.64.0.0/10？

负载均衡使用了100.64.0.0/10 网段内的地址做为健康探测源 IP，无需在购买云主机后端服务器的安全组中额外针对该网段配置放行策略。

另外若在后端服务器中配置 iptables 等其他安全策略，请务必放行该网段，否则将会导致健康探测失败。

健康检查提示异常该如何处理？

当健康检查提示异常时，请按如下步骤进行排查：

检查后端服务器已开启了相应的服务端口。

检查后端服务器内部是否有防火墙之类的防护软件，拦截健康检查报文。

检查后端服务器上有没有做 iptables 规则限制。

检查客户端直接访问后端服务器应用服务是否可以成功。

检查负载均衡检查参数设置是否正确。

七层负载均衡建议使用静态页面来健康检查。

检查后端服务器是否有高负载导致服务器对外响应慢。

健康检查提示正常，无法访问业务需要如何处理？

查看负载均衡监听上是否访问控制列表、安全组等访问策略限制了客户端访问，确保客户端被放行；

若调度算法为wlc、wrr等带有权重属性的算法，查看RS上配置的权重是否是0，如果是0，则将权重修改为大于0的值；

查看当前负载均衡的带宽使用量是否超过购买的带宽；

UDP监听配置ICMP健康检查，检查后端RS服务端口是否已启动；

BLB的健康检查频率是否过高？

BLB通过探测后端服务器的健康检查状态，来判断后端服务的可用性。

服务器必须是通过连续n次的探测，且探测结果均为健康才会视后端服务业务正常。

其中n 即为健康检查阈值，为了避免受到网络抖动的影响，进行多次健康检查是有必要的。

八、负载均衡实例健康检查相关问题

若用户感知过于频繁，可通过自行修改健康检查配置进行调节。

BLB 的控制台上后端服务器健康检查状态的转变生效时间是多少？

这个状态更新的时间与所处地域实例数有关，实例数越多，更新时间会越长。

四层协议对应的 RS 健康检查状态更新时间 <= 150s

七层协议对应的 RS 健康检查状态更新时间 <= 240s

目前控制台RS状态展示逻辑为独立的逻辑，只影响控制台展示，实际业务不受此影响。

为什么100网段的IP在访问后端服务器？

100开头的IP所属网段为100.64.0.0/10。负载均衡实例以该网段IP作为源地址，向后端服务器做健康检查探测和特殊转发数据流模式时转发前端业务流量使用。

因此，为了保证您所配置的负载均衡实例可正常提供服务，请确保后端服务器的 iptables 安全策略已放通100.64.0.0/10网段。

lb实例开启cip透传，为什么Server上还能收到100.64.0.0/10 网段内的地址建连？

负载均衡使用了100.64.0.0/10 网段内的地址做为健康探测源 IP，一般使用.3和.4结尾的IP作为源ip向Server发健康探测报文，不会对业务有影响，属于正常行为。

为什么Server上还能收到大量100.64.0.0/10 网段内，以.3和.4结尾的IP作为源ip向Server建连？

一个blb集群有多台物理机构成，每台机器都会使用.3和.4结尾的IP对Server做健康检查探测，所以同一台Server上可以看到不同的在100.64.0.0/10 网段内，以.3和.4结尾的IP作为源ip向Server建连。

负载均衡实例的后端服务器状态怎么查看？

负载均衡查看后端服务器状态有两种方式：

1. 登录百度智能云控制台，在左侧导航栏，选择负载均衡 BLB->选择对应的实例->详情页选择后端服务器 即可查看实时后端服务器健康检查状态

2. 登录百度智能云控制台，在左侧导航栏，选择负载均衡 BLB->选择对应的实例->详情页选择监控 在端口监控中可以查看后端服务器正常/异常数量

负载均衡实例添加后端服务器后，后端服务器过多久开始承载流量？

负载均衡实例挂载新的后端服务器时，默认后端服务器为不可用状态，当后端服务器通过健康检查之后，后端服务器状态转变为可用状态。此时，后端服务器才会开始承载流量。

因此，新添加的后端服务器生效时间，与健康检查生效时间有关。

当后端服务为正常的情况下，

健康检查最长生效时间 = 健康检查超时时间 * 健康检查次数 + 健康检查间隔时间 * (健康检查次数 - 1)

默认健康检查配置下，健康检查最长生效时间 = 3s * 3 + 3s * 2 = 15s。

在用户在控制台操作添加后端服务器后，配置下发网关设备还需要一定的时间，实际场景中，也需要把这一部分考虑上。此部分和集群配置规模存在一定的关系，一般可以认为在 30s 之内。

所以，用户在控制台上操作挂载一个服务状态正常的后端服务器之后，默认健康检查配置下，最久 45s 后该服务器就实际承载流量了。

在后端服务器上看到了很多100段 IP 发过来的健康检查请求，这个后面会减少吗？

负载均衡实例对后端服务器进行健康检查，都是使用的100段 IP。在分集群之前，后端服务器会被整集群的机器进行健康检查，可能会存在上述现象。

当前负载均衡实例各个集群已经采用分集群架构，单个用户后端服务器只会被所在的分集群进行健康检查，该类健康检查请求已相对减少。

安全类问题

发现很多IP地址在频繁地访问我的BCC，这是什么原因？

访问BCC的请求可能来自用户（正常访问、攻击流量）或者BLB实例的健康检查。建议您通过检查日志、使用`netstat -an`等命令分析访问频率和访客IP地址来源。根据结果来分析到底是正常访问还是攻击流量。

BLB支持HTTPS协议吗？

BLB支持HTTPS协议，您需要先上传服务器证书，然后配置HTTPS监听器。

BLB可以防攻击吗？

负载均衡设备本身具有一定的防DDoS攻击能力，确保服务稳定。同时还配合[应用防火墙WAF](<https://cloud.baidu.com/doc/WAF/index.html>)为您提供更全面的安全保障。

计费类问题

BCC加入BLB会导致其计费规则发生变化吗？

不会。BLB和BCC是根据客户的使用情况分别计量计费并结算的。

健康检查产生的流量会算入到计费流量中吗？

不会。

BLB提供哪些计量数据？

BLB以实例为单位提供该实例上一个计费周期的计量数据，包括：实例配置费。BLB按分钟计费，每小时出账单。更多计费相关信息请参考[产品定价](https://cloud.baidu.com/doc/BLB/s/Wkwn5lral)。

负载均衡专属集群类问题

购买了专属集群，创建BLB实例时为什么看不到这个专属集群？

1. 首先请确认该专属集群处于可用状态，刚购买的专属集群需要等到集群搭建完成，才会自动跳转到可用状态，可从集群列表页展示，如果集群列表页未展示，说明还处于搭建状态，请等待
2. 其次如果集群处于可用状态，请确认该集群详情页的<新建BLB实例>开关已开启，如果该开关关闭，会导致该集群不允许新建实例，创建BLB实例时无法选择到该集群
3. 如果要选择该集群，请跳转到该集群的详情页开启<新建BLB实例>开关即可

专属集群上创建的BLB实例，主备模式/主主模式如何选择？

1. 对于覆盖两个不同机房的专属集群，集群上创建的实例支持选择主备模式/主主模式，默认为主备模式
2. 主备模式与主主模式都是相对于客户端访问BLB的数据流访问特点，主备场景下无论客户端从云上哪个机房发起访问，默认都会到对应实例在专属集群的主机房，主机房宕机场景才会将流量转发到备机房，主主模式遵循就近访问，会依据客户端发起访问的机房按照物理链路最短，选择专属集群对应的机房承接流量
3. 如果该BLB的后端服务器集中在一个云机房，且希望对该BLB的访问具有后端链路亲和性，建议选择主备模式，且将主机房设置为跟后端服务器一致或者物理链路更靠近的机房，如果访问该BLB实例的客户端机房不限制，且希望该BLB实例具有向前链路亲和性，可以选择主主模式。
4. 产品上建议选择主备模式，数据流更清晰，更容易评估专属集群不同机房的负载情况，且定位问题数据流链路更明确

BLB实例在专属集群/共享集群间或者不同专属集群间迁移，数据流是否有损，以及预期有损时长多少？

1. 对该BLB实例4层监听上的访问，长短链接数据流有损均受影响，需要客户端主动重连恢复，有损时长受该监听器的健康检查参数设置影响，具体时长为该监听器上的健康检查间隔*健康阈值
2. 为了尽可能减少迁移的有损影响，建议在业务低峰期或者关闭服务后再执行迁移动作，另外在迁移前将监听器上这两个健康检查参数设置为最小值，迁移结束后再设置回原始值

专属集群上的BLB实例执行主备机房切换/主备/主主模式切换时，数据流是否有损，以及预期有损时长多少？

对该BLB实例4层监听上的访问，新建链接无损，主备机房切换时，存量长、短链接均有损；主备/主主切换时，存量长、短链接概率有损；客户端发起重连即可恢复，具体有损时长和客户端配置的重试时间有关。

服务等级协议SLA

BLB服务等级协议SLA(V2.0)

****协议生效时间：2019年01月15日****

本服务等级协议（Service Level Agreement，以下简称"SLA"）规定了百度智能云向客户提供的负载均衡（Baidu Load Balance，简称"BLB"）的服务可用性等级指标及赔偿方案。

1. 定义

****服务周期：**** 一个服务周期为一个自然月。

****服务周期总分钟数：**** 按照服务周期内的总天数 * 24（小时）* 60（分钟）计算。

****单实例不可用分钟数：**** 当某一分钟内，客户所有试图与指定的BLB实例中某个监听所建立连接的连续尝试均失败，则视为该分钟内该监听的服务不可用。在一个服务周期内单BLB实例所包含的所有监听的服务不可用分钟数之和。

****实例服务周期总分钟数：**** 在一个服务周期内单BLB实例所包含的所有监听的服务周期总分钟数之和。

****月度服务费用：**** 客户在一个自然月中就单个BLB实例所支付的服务费用总额，若客户一次性支付了多个月份的服务费用，则将按照所购买的月数分摊计算月度服务费用。

2. 服务可用性

2.1 服务可用性计算方式

![[图片]](http://bjhw-bce-online-public0.bjhw.baidu.com:8109/proxy-external/?url=https://doc.bce.baidu.com/bce-documentation/BLB/SLA_BLB_01.png)

2.2 服务可用性承诺

百度负载均衡 BLB 承诺单实例99.95%的服务可用性。

如负载均衡服务未达到上述服务可用性承诺，客户可以根据本协议第三条约定获得赔偿。赔偿范围不包括以下原因所导致的不可用：

- (1) 百度智能云预先通知用户后进行系统维护所引起的，包括割接、维修、升级和模拟故障演练；
- (2) 任何百度智能云所属设备以外的网络、设备故障或配置调整引起的；
- (3) 用户的应用程序或数据信息受到黑客攻击而引起的；
- (4) 用户维护不当或保密不当致使数据、口令、密码等丢失或泄漏所引起的；
- (5) 用户的疏忽或由用户授权的操作所引起的；
- (6) 用户未配置后端服务器或后端服务器故障导致的负载均衡服务无法正常使用；
- (7) 不可抗力以及意外事件引起的；
- (8) 负载均衡服务只赔付负载均衡自身，不赔付负载均衡实例所关联的其他云服务。
- (9) 其他非百度智能云原因所造成的负载均衡服务无法正常使用。

3. 服务赔偿条款

3.1 赔偿标准

根据BLB单实例月度服务可用性，按照下表中的标准计算赔偿金额，赔偿方式仅限于用于购买BLB产品的代金券，且赔偿总额不超过未达到服务可用性承诺当月客户就该BLB支付的月度服务费用。

服务可用性	赔偿代金券金额
-----	-----
低于99.95%但等于或高于99.00%	月度服务费用的10%
低于99.00%但等于或高于95.00%	月度服务费用的25%
低于95.00%	月度服务费用的100%

3.2 赔偿申请时限

客户可以在每月第五（5）个工作日后对上个月没有达到可用性的实例提出赔偿申请。赔偿申请必须限于在BLB实例没有达到可用性的相关月份结束后两（2）个月内提出。超出申请时限的赔偿申请将不被受理。

4. 其他说明

- (1) 在法律法规允许的范围内，百度智能云负责对本协议进行解释说明。
- (2) 本协议一经公布立即生效，百度智能云有权对本SLA条款作出修改。如本SLA条款有任何修改，百度智能云将提前30天以网站公示或发送邮件的方式通知您。如您不同意百度智能云对SLA所做的修改，您有权停止使用BLB服务，如您继续使用BLB服务，则视为您接受修改后的SLA。
- (3) 协议项下百度智能云对于用户所有的通知均可通过网页公告、站内信、电子邮件、手机短信或其他形式等方式进行；该等通知于发送之日视为已送达收件人。因用户未及时获知百度智能云的服务变更或终止条款遭受损失的，百度智能云不承担任何责任。

本协议内其他。

(4) 本协议的订立、执行和解释及争议的解决均应适用中国法律并受中国法院管辖。如双方就本协议内容或其执行发生任何争议，双方应尽量友好协商解决；协商不成时，任何一方均可向北京市海淀区人民法院提起诉讼。

(5) 本协议构成双方对本协议之约定事项及其他有关事宜的完整协议，除本协议规定的之外，未赋予本协议各方其他权利。

(6) 如本协议中的任何协议无论因何种原因完全或部分无效或不具有执行力，本协议的其余协议仍有效并且有约束力。

(7) 关于用户约束条款，详见[《百度智能云用户服务协议》](Agreements/百度智能云用户服务协议/用户的权利与义务.md)中的"用户的权利与义务"相关条款内容。

(8) 关于服务商免责声明，详见[《百度智能云用户服务协议》](Agreements/百度智能云用户服务协议/免责声明.md)中的"免责声明"相关条款内容。

LBDC服务等级协议SLA

协议生效时间：2021年09月01日

本服务等级协议（Service Level Agreement，以下简称 "SLA"）规定了百度智能云向客户提供的负载均衡专属集群（Load Balance Dedicated Cluster，简称"LBDC"）服务可用性等级指标及赔偿方案。

1. 定义

**服务周期：*一个服务周期为一个自然月。

**服务周期总分钟数：*按照服务周期内的总天数 * 24（小时） * 60（分钟）计算。

**单LBDC服务不可用分钟数：*当某一分钟内，客户所有试图与指定LBDC服务中的某个BLB实例所建立连接的连续尝试均失败，则视为该分钟内该LBDC服务不可用。在一个服务周期内单LBDC服务所包含的所有BLB实例的服务不可用分钟数之和。

**月度服务费用：*客户在一个自然月中就单个LBDC服务所支付的服务费用总额，若客户一次性支付了多个月份的服务费用，则将按照所购买的月数分摊计算月度服务费用。

2. 服务可用性

2.1 服务可用性计算方式

2.2 服务可用性承诺

百度智能云LBDC承诺单集群99.975%的服务可用性。

如LBDC服务未达到上述服务可用性承诺，客户可以根据本协议第三条约定获得赔偿。赔偿范围不包括以下原因所导致的不可用：

(1) 百度智能云预先通知用户后进行系统维护所引起的，包括割接、维修、升级和模拟故障演练；

(2) 任何百度智能云所属设备以外的网络、设备故障或配置调整引起的；

(3) 用户的应用程序或数据信息受到黑客攻击而引起的；

(4) 用户维护不当或保密不当致使数据、口令、密码等丢失或泄漏所引起的；

(5) 用户的疏忽或由用户授权的操作所引起的；

(6) 不可抗力以及意外事件引起的；

(7) LBDC服务只赔付LBDC自身，不赔付LBDC服务所关联的其他云服务；

(8) 其他非百度智能云原因所造成的LBDC服务无法正常使用。

3. 服务赔偿条款

3.1 赔偿标准

根据LBDC单集群月度服务可用性，按照下表中的标准计算赔偿金额，赔偿方式仅限于用于购买LBDC产品的代金券，且赔偿总额不超过未达到服务可用性承诺当月客户就该LBDC支付的月度服务费用。

服务可用性	赔偿代金券金额
-----	-----
低于99.975%但等于或高于99.00%	月度服务费用的10%
低于99.00%但等于或高于95.00%	月度服务费用的25%
低于95.00%	月度服务费用的100%

3.2 赔偿申请时限

客户可以在每月第五（5）个工作日后对上个月没有达到可用性的实例提出赔偿申请。赔偿申请必须限于在LBDC服务没有达到可用性的相关月份结束后两（2）个月内提出。超出申请时限的赔偿申请将不被受理。

4. 其他说明

(1) 在法律法规允许的范围内，百度智能云负责对本协议进行解释说明。

(2) 本协议一经公布立即生效，百度智能云有权对本SLA条款作出修改。如本SLA条款有任何修改，百度智能云将以网站公示或发送邮件的方式通知您。如您不同意百度智能云对SLA所做的修改，您有权停止使用LBDC服务，如您继续使用LBDC服务，则视为您接受修改后的SLA。

(3) 协议项下百度智能云对于用户所有的通知均可通过网页公告、站内信、电子邮件、手机短信或其他形式等方式进行；该等通知于发送之日视为已送达收件人。因用户未及时获知百度智能云的服务变更或终止条款遭受损失的，百度智能云不承担任何责任。

(4) 本协议的订立、执行和解释及争议的解决均应适用中国法律并受中国法院管辖。如双方就本协议内容或其执行发生任何争议，双方应尽量友好协商解决；协商不成时，任何一方均可向北京市海淀区人民法院提起诉讼。

(5) 本协议构成双方对本协议之约定事项及其他有关事宜的完整协议，除本协议规定的之外，未赋予本协议各方其他权利。

(6) 如本协议中的任何协议无论因何种原因完全或部分无效或不具有执行力，本协议的其余协议仍有效并且有约束力。

(7) 关于用户约束条款，详见[《百度智能云用户服务协议》](Agreements/百度智能云用户服务协议/用户的权利与义务.md)中的"用户的权利与义务"相关条款内容。

(8) 关于服务商免责条款，详见[《百度智能云用户服务协议》](Agreements/百度智能云用户服务协议/免责声明.md)中的"免责声明"相关条款内容。

百度云负载均衡可信云认证服务可用性说明

百度云负载均衡可信云认证服务可用性说明

1. 服务范围

百度负载均衡BLB（Baidu Load Balance）通过将同一区域的多台百度智能云服务器虚拟成一个组，设置一个内网或外网的服务地址，将前端并发访问转发给后台多台云服务器，实现应用程序的流量均衡，性能上实现业务水平扩展。负载均衡还通过故障自动切换及时地消除服务的单点故障，提升服务的可用性。

2. 服务可用性

2.1 服务可用性计算方式

2.2 服务可用性承诺

百度负载均衡 BLB 承诺单实例99.95%的服务可用性。

如负载均衡服务未达到上述服务可用性承诺，客户可以根据本协议第三条约定获得赔偿。赔偿范围不包括以下原因所导致的不可用：

(1) 百度智能云预先通知用户后进行系统维护所引起的，包括割接、维修、升级和模拟故障演练；

(2) 任何百度智能云所属设备以外的网络、设备故障或配置调整引起的；

(3) 用户的应用程序或数据信息受到黑客攻击而引起的；

(4) 用户维护不当或保密不当致使数据、口令、密码等丢失或泄漏所引起的；

(5) 用户的疏忽或由用户授权的操作所引起的；

(6) 用户未配置后端服务器或后端服务器故障导致的负载均衡服务无法正常使用；

(7) 不可抗力以及意外事件引起的；

(8) 负载均衡服务只赔付负载均衡自身，不赔付负载均衡实例所关联的其他云服务。

(9) 其他非百度智能云原因所造成的负载均衡服务无法正常使用。

3.数据知情权

除应当地法律法规、或政府监管部门的监管、审计要求，用户的所有数据、应用及行为日志不会提供给第三方。用户的所有数据不会被用于境内外业务或数据分析。用户的数据存储在用户选择的区域（数据中心）内。百度云数据中心在华北&华东&华南及海外均有数据中心,其中在北京的数据中心为北京亦庄数据中心、北京实体店数据中心；华东数据中心为上海万国数据外高桥数据中心、广州为广州联通数据中心；香港为香港联通葵涌数据中心；百度智能云用户负载均衡相关配置数据均留存在百度智能云相关管理服务器上,相关数据均位于百度智能云可自主控制的服务器上

4.服务可审查性

百度智能云在依据现有法律法规体系下，为了配合政府监管部门的监管或安全取证调查等原因，在符合流程和手续完备的情况下，可以提供负载均衡相关信息，包括各关键组件的运行日志、运维人员的操作记录、用户操作记录等信息。

5.服务功能

百度负载均衡 BLB（Baidu Load Balance）通过将不同区域的多台百度智能云服务器虚拟成一个组，设置一个内网或外网的服务地址，将前端并发访问转发给后台多台云服务器，实现应用程序的流量均衡，性能上实现业务水平扩展。负载均衡还通过故障自动切换及时地消除服务的单点故障，提升服务的可用性。

6.服务调配能力

基于用户业务实际需求,可以将负载均衡在资源使用上分为共享型负载均衡和专属型负载均衡两种,以满足不同用户对于负载均衡性能需求。

共享型负载均衡即为所有用户共享一组负载均衡服务器,所有的流量转发全部混合经由这两个负载均衡服务器转发,用户之间的负载均衡共享带宽,共享资源。

专属集群型负载均衡为某个客户单独使用,充分保障客户的带宽单独使用性,充分保障单租户使用专属集群的独享资源承载。负载均衡专属集群提供性能可控、资源独享、物理资源隔离的专属负载均衡服务，满足超高性能和独占资源需求。

负载均衡专属集群和负载均衡共享集群的区别如下:

![image.png](http://bjhw-bce-online-public0.bjhw.baidu.com:8109/proxy-external/?url=http://bce-cdn.bj.bcebos.com/doc/bce-doc/BLB/image_6dd1c7a.png)

（注：说明每个用户可用负载均衡的带宽上下限，数量上下限等对外售卖的资源调配能力

解释：BLB实例在共享集群，多租户性能共享，当出现资源争抢时，性能可能会受到限制，目前没有单独限速。）

7.故障恢复能力

百度智能云通过故障监控、快速定位、自我修复、自动告知等一系列故障管控体系，保证负载均衡服务的故障恢复能力。同时,云平台部署负载均衡服务器时是由多个服务器进行Session同步承载,保障在单服务器产生故障时能快速恢复,同时对于客户问题,百度智能云提供7*24小时技术支持服务保障出现故障时及时恢复。

8.网络接入性能

租户开通负载均衡服务时,可自行选择配置或不配置公网访问,在公网访问中支持在BLB购买页面直接购买公网IP或先进行BLB购买,随后进行EIP绑定,EIP使用的运营商可以为电信、联通、移动，带宽选择可以从1M到500M(特殊白名单可以到1000M以上)

![image.png](http://bjhw-bce-online-public0.bjhw.baidu.com:8109/proxy-external/?url=http://bce-cdn.bj.bcebos.com/doc/bce-doc/BLB/image_533e8e0.png)

且在EIP的选择上支持标准型和增强型BGP购买,具备高网络接入性能

9.服务计量准确性

负载均衡目前支持后付费计费方式,可基于后台价格系统进行价格配置,以分钟级进行计费,充分保障服务计量准确性，同时对于计费日志,百度智能云平台至少保持3个月的计费日志,保障计费准确性。![image.png](http://bjhw-bce-online-public0.bjhw.baidu.com:8109/proxy-external/?url=http://bce-cdn.bj.bcebos.com/doc/bce-doc/BLB/image_a17760f.png)

10.服务变更、终止条款

10. 服务变更、终止条款

鉴于网络服务的特殊性，用户了解并同意，百度智能云有权变更、中断或终止部分或全部的网络服务。如变更、中断或终止的网络服务属于免费网络服务，百度智能云无需针对用户进行单独通知。

用户理解，百度智能云需要定期或不定期地对提供网络服务的平台或相关的设备进行检修或者维护，可能造成收费网络服务在合理时间内的中断。百度智能云无需承担责任，但百度智能云应尽可能进行事先通告。

百度智能云服务可能变更或删除部分功能。此时，百度智能云无需针对用户进行单独通知。用户理解并接受，一旦服务取消，用户使用本服务的权利将立即终止，并且用户在本服务中储存的任何信息可能无法恢复。

11. 服务赔偿条款

11.1 赔偿标准

根据BLB单实例月度服务可用性，按照下表中的标准计算赔偿金额，赔偿方式仅限于用于购买BLB产品的代金券，且赔偿总额不超过未达到服务可用性承诺当月客户就该BLB支付的月度服务费用。

!<img.png>(http://bjhw-bce-online-public0.bjhw.baidu.com:8109/proxy-external/?url=http://bce-cdn.bj.bcebos.com/doc/bce-doc/BLB/image_1548307.png)

11.2 赔偿申请时限

客户可以在每月第五（5）个工作日后对上个月没有达到可用性的实例提出赔偿申请。赔偿申请必须限于在BLB实例没有达到可用性的相关月份结束后两（2）个月内提出。超出申请时限的赔偿申请将不被受理。

12. 用户约束条款

用户可在法律允许的范围内，使用百度智能云提供的服务；用户确保在接受本协议时已明确了解百度智能云所提供服务的內容并已知悉百度智能云官网关于该技术服务的详细介绍、产品服务等级协议（SLA）、使用方法等全部产品服务规则。

用户可以在订购期限内享受百度智能云按照现有技术和条件标准提供的服务，百度智能云应按照本协议条款和服务标准尽商业上的合理努力为用户提供连贯而安全的服务，具体服务标准参见百度智能云官网公布的产品服务等级协议（SLA）及其他产品服务规则。

订购期限内，用户可以享受百度智能云提供的如下产品服务：

(1) 百度智能云为用户提供7×24小时的在线工单服务系统，解答用户在使用中的问题。

(2) 百度智能云将于用户联系百度智能云后24小时内消除用户非人为操作所出现的故障并恢复产品服务的正常使用，但因用户的原因和/或不可抗力以及非百度智能云控制范围之内的事项除外。

* 用户应当确保订单信息填写准确无误以便百度智能云进行计费统计。除用户有确切证据证明百度智能云数据错误外，计费统计以百度智能云为准。因用户订单信息填写错误而导致的计费统计错误，百度智能云不承担责任，且无需退还用户已支付的服务款项。

* 用户应当按照本协议约定支付服务款项。如未按照本协议支付，百度智能云有权单方面拒绝或终止向用户提供产品服务，同时保留追讨欠款的法律权利。

* 用户不应在使用产品服务时安装、使用盗版软件。

* 用户对自己使用产品服务时存放的数据以及进入和管理百度智能云各类产品服务的口令、密码等信息的完整性和保密性负责；用户账户下发生的行为将全部视为用户的行为，无论该等行为是否取得用户的授权，或由用户、用户的员工或第三方（包括用户的承包商或代理商）实施，用户应对用户账户下的所有行为或活动负责，百度智能云及其关联方不对用户的账户遭到未授权访问承担责任。

* 用户对自己使用产品服务时存放的数据、信息内容负责。如因用户帐户中上传、发布的公开信息违反法律法规、部门规章或国家政策，百度智能云有权直接更改或删除上述内容同时有权停止为用户提供产品服务。

* 出于行业技术水平的客观限制，百度智能云无法保证其所提供的产品服务毫无瑕疵。用户同意，即使百度智能云提供的产品服务存在瑕疵，但上述瑕疵是当时行业技术水平所无法避免的，其将不被视为百度智能云违约。用户同意和百度智能云一同合作解决上述瑕疵问题。

* 用户同意，数据备份系用户的义务和责任。百度智能云没有义务完全备份用户的数据或信息，亦不对用户数据备份工作或结果承担任何责任。

* 用户在使用百度智能云产品服务时应当遵守《网络安全法》、《电信条例》等法律法规及相关国家政策，维护互联网秩序和安全，不得从事违法违规行为也不得为上述行为提供便利。上述违法违规行为包括但不限于：

(1) 实施涉嫌博彩、赌博、“私服”、“外挂”等非法互联网活动或其他欺诈、误导行为；

(2) 发布违法、违规、虚假信息或未经接收方许可进行商业短信、邮件发送或其他骚扰式营销行为；

(3) 实施侵害百度智能云或第三方肖像权、名誉权、隐私权、知识产权等任何合法权益的行为；

(4) 以过度负荷或其他任何方式导致网络中断，或实施未经授权的截取、盗用、干扰或监测等违法行为；

(5) 实施任何破坏或试图破坏网络安全的行为，包括但不限于以病毒、木马、恶意代码、钓鱼等方式，对网站、服务器进行恶意扫描、非法侵入系统、非法获取数据等；

(6) 实施任何改变或试图改变产品服务提供的系统配置或破坏系统安全的行为；

(7) 利用技术或其他手段破坏、扰乱百度智能云的运营或影响他人对产品或服务的使用；

(8) 利用任何技术手段或以任何方式对产品服务和相关数据进行破译、破解等行为，刺探或试图获取百度智能云用户个人信息或相关用户信息；

(9) 利用百度智能云提供的互联网资源经营任何用户不具备资质的互联网业务；

(10) 其它可能给百度智能云造成不良影响或损失的行为。

为实现本协议目的，用户可能需提供用户或其他第三方的相关信息。用户应当保证收集和提供上述信息的行为已遵循合法、正当、必要的原则，已明示收集、使用信息的目的、方式和范围且已获得对应自然人的明确授权。

百度智能云将通过合理努力追求数据分析、数据验证等产品服务的结果准确性，但不对所提供分析结果的准确性或特定用途适用性作出保证。百度智能云所提供的分值、分级等分析结果仅可作为参考，无法完成替代用户进行商业决策的制定。如用户依据百度智能云分析结果进行商业决策而遭受任何损失，百度智能云完全免责。用户对所获得的所有数据分析和验证结果应当履行保密义务，不得向第三方提供或售卖数据分析或验证结果。

未经百度智能云书面许可，用户不得将百度智能云所提供的产品服务（包括但不限于程序、接口、软件、源代码、工具、文档）转售、转租或提供给其他任何第三方使用。

13. 免责条款

如用户严重违反《百度智能云线上订购协议》约定，百度智能云有权提前终止本协议，并不退还用户已支付的服务款项；上述提前终止自百度智能云向用户发出的通知送达时生效。

对于因履行本协议所导致，或与其相关或因其发生的任何损失、指控、处罚、主张、判决或费用，尤其是因用户违反本协议约定而造成第三方的损失(包括诉讼费、律师费用和第三方费用等)，用户应确保百度智能云及其雇员、代理、管理层、或其关联公司免于承担因以上损失造成的任何责任，并赔偿百度智能云因此向该第三方支付损害赔偿。

用户应遵守相关的规则对享受的折扣优惠内容予以保密（如有），否则，百度智能云有权随时采取以下一项或多项措施：

- (1) 单方暂时中止、终止向用户提供部分或全部优惠。

- (2) 要求用户补足已经享受的免费或优惠的产品服务相对应的费用。

如果确因百度智能云原因造成用户不能正常使用百度智能云服务的，百度智能云按照服务赔偿标准向用户赔偿相应损失，具体服务赔偿标准参见百度智能云官网公示。

责任免除用户同意，在法律允许的范围内，下述情况不属于百度智能云违约，百度智能云不承担任何责任：

(1) 用户由于下述原因无法正常使用产品服务的情况：

- ① 用户遭到黑客攻击，导致数据丢失，服务中断；

- ② 用户由于自身系统、网络配置变更导致数据丢失，服务中断；

- ③ 用户由于自身程序bug导致的客户服务不可正常使用；

- ④ 其他由于非百度智能云原因导致的服务异常情况。

(2) 百度智能云在进行服务器配置、维护时，需要短时间中断服务，且已提前通知用户；

(3) 非百度智能云原因导致的Internet上的通路阻塞，造成用户网站访问速度下降等服务异常或中断，包括但不限于：

- ① 地震、爆炸、洪水、火灾、暴风雨、台风等天灾；

- ② 战争、类似战争行为、敌对行为、武装冲突、恐怖活动、谋反、政变等情况；

- ③ 罢工、暴动、民众骚乱或恶意行为；

- ④ 电力或动力故障、公用设施或其他电信故障；

- ⑤ 政府行为、政策变化、封锁、禁运、电信部门技术调整、政府管制等；

(4) 产品服务规则中对相关产品服务使用约定的其他免责情况
