百度智能云

IAM 文档

目录

功能发布记录

发布时间	功能概述
2025-02	• IAM角色联合：支持多角色联合登录 • IAM角色联合：支持角色在用户中心修改绑定手机号
2025-01	• 安全认证：新增API Key认证鉴权机制，支持千帆ModelBuilder、千帆AppBuilder、AI开放平台认证鉴权新能力
2024-08	• 权限策略：API粒度权限支持资源选择 • 账号管理：一个手机号限制只能激活10个账号 • 体验优化：用户中心升级到新版视觉效果
2024-07	• 体验优化：SAML辅助域名支持"-"

	• 体验优化：子用户登录页提示信息优化
2024-06	• 权限策略：支持API粒度权限(操作级权限) • OpenAPI&SDK：提供了证书管理的Go SDK • 体验优化：子用户详情页面支持解绑虚拟MFA
2024-05	• 权限策略：支持API粒度权限(操作级权限) • 体验优化：子用户登录页整体样式升级 • 体验优化：子用户登录页自动检测SSO，默认切换到企业账号登录
2024-04	• 权限策略：系统策略支持查看引用记录 • 设置：登录session默认生效时间修改为24小时 • 体验优化：主账号登录页整体样式升级

	• 体验优化：账号注销、头像及导航升级到最新视觉效果
2024-03	• 企业组织：继承实名的子账户退出企业组织增加约束条件 • 证书管理：子用户ak支持调用Open API • OpenAPI：获取证书详情open api支持查询证书指纹 • 体验优化：切换角色页面升级新版视觉效果
2024-02	• 账号管理：支持使用第三方账号(微信/QQ/微博)登录百度账号 • 设置：优化IP白名单配置体验 • OpenAPI：新增IAM角色联合SSO OpenAPI • 体验优化：安全认证升级新版视觉效果 • 体验优化：移动网页端登录体验升级，突出百度APP一键登录

2024-01	• 角色管理：角色列表支持角色名称或描述搜索 • 角色管理：支持解绑BCC实例角色
2023-11	• 设置：登录session有效期默认修改为3小时
2023-09	• 权限策略：IAM服务自身支持自定义权限策略
2023-08	• 权限策略：新增CCR支持标签鉴权 • 单点登录：支持百度小程序授权后可SSO到百度智能云
2023-06	• 账号管理：体验升级，增加"上次登录"标记 • 企业组织：支持子账户退出财务圈

2023-04	• 企业组织：快速新建子账户支持直接设置密码，快速完成实名认证 • 账号管理：云账号注册激活流程合并 • 账号管理：移动网页端登录页样式升级 • 权限策略：Billing权限细化，新增订单创建权限
2023-03	• 安全认证：AK增加前缀：ALTAK。以提升安全性 • 账号管理：百度账单支持短信验证码登录
2022-09	• 新增接入主子用户服务：云智能网 CSN • 新增接入主子用户服务：解析器 RESOLVER
2022-07	• 新增接入主子用户服务：智能网络接入服务 SMART_WAN

2022-06	• 新增接入主子用户服务：云防火墙 CFW • 新增接入主子用户服务：轻量应用服务器 LS
2022-02	• 新增接入主子用户服务：云原生数据库 GaiaDB-S
2021-12	• 新增接入主子用户服务：史宾格安全及隐私合规平台 SPRINGER
2021-06	• 新增接入主子用户服务：API网关
2021-01	• 新增接入主子用户服务：数据库审计 DBAudit
2020-12	• 新增接入主子用户服务：边缘计算 BEC

2020-11	• 新增接入主子用户服务：数据传输服务 DTS
2020-07	• 新增 IAM java sdk
2019-12	• 多用户访问控制、企业组织、消息中心国际化。 • 证书管理新增通过证书名称查询证书详情、证书状态接口。 • 企业组织新增账户级联合认证(SSO)。你可以基于SAML 2.0协议配置你的企业目录到企业组织子账户间的联合认证。 • 自定义策略限制条件：新增对象存储 BOS支持secureTransport。你可以通过secureTransport字段要求子用户或服务必须使用https访问BOS资源。 • 安全设置新增AccessKey泄漏监测与告警。在开启了IP白名单访问的前提下，你可以开启AK泄漏监测与告警开关，对于非IP白名单内的访问，满足一定条件后可以通过云监控的事件监控服务进行告警。 • 新增接入标签授权服务：音视频直播 LSS。 • 新增接入主子用户服务：云智能顾问 ABC Advisor。

2019-11	• 企业组织服务控制策略SCP支持一条策略配置多个服务。在此之前，你只能为每个服务配置一条SCP。 • 百度智能云账号提供自助账号注销功能。你可以通过百度智能云App发起云账号的注销，在此之前，你只能通过直接注销百度账号来注销云服务。 • 外部账号接入支持IAM角色联合和IAM用户联合，分别以自定义的IAM角色和IAM子用户的身份单点登录(SSO)到百度智能云，在此之前，你只能以系统管理员的角色SSO到百度智能云。 • 自定义策略限制条件：新增对象存储 BOS支持sourceVpc。你可以通过vpc字段限制对BOS桶中存储文件的访问权限。 • 安全设置新增密码策略。你可以在设置中配置子用户密码服务度、密码有效期、密码过期策略、历史密码检查、错误密码尝试等详细密码策略，在此之前，你只能设置子用户的密码有效期。 • 新增接入标签授权服务：私有网络 VPC、对象存储 BOS。 • 新增接入主子用户服务：安全计算节点 BEC。
2019-10	• 云账号密保信息与登录账号(百度账号、推广账号)保持一致。对于之前存在不一致的账号，你可以在用户中心中选择立即同步，或是通过直接修改登录账号的密保信息进行同步。 • 证书管理支持托管客户端CA证书。

	• 新增接入标签授权服务：负载均衡 BLB。 • 新增接入主子用户服务：安全计算节点 BEC。
2019-09	• 安全设置中控制台默认会话时间从30分钟调整到1小时，并优化了会话时间设置体验。 • 自定义策略支持限制条件。当前你可以为每条定义的自定义策略加上访问时间限制条件。 • 企业组织发布第一期OpenAPI。你可以使用OpenAPI进行企业组织创建、管理子账户、管理组织单元等。 • 新增接入标签授权服务：物理服务器 BBC。 • 新增接入主子用户服务：安全检测SRD。
	• 新增签名排查工具提供不同阶段的中间结果帮助快速排查签名问题，同时提供鉴权认证的SDK包 • MFA新特性支持你为自身账户，或是(管理员)为子用户同时配置2种MFA验证方式，只要任一渠道的验证通过，即可通过2次身份认证，在保证安全访问的同时，提升用户的便利性。在此之前，你开启MFA时，只能选择短信或是虚拟MFA的方式 • 证书管理提供查询、更新、删除证书OpenAPI。证书管理新发布的OpenAPI，支持你在不变更证书ID的情况下，更新不再使用的证书内容，以确保证书使用的更新和使用的连续性。通过删除证书API，你可以批量删除过期且不在使用状态的证书

批量删除过期且不在使用状态的证书

2019-08

- 自定义策略特定资源选项优化。调整了自定义策略 > 按策略生成器生成 > 特定资源列表的显示字段，待选框最多显示3列，已选框仅显示一列，以确保列表字段展示足够多的信息
- 接入标签授权的服务支持根据地域展示已选实例清单和数量。在此之前，你只能通过服务实例的整体数量查看被授权的实例清单
- 标签授权服务新接入了弹性公网EIP、云数据库RDS、云数据库SCS等服务
- 优化获取子用户列表性能和获取子账户列表性能

- 自定义策略特性增强，策略生成器生成的自定义策略目前已支持如下新功能
  - 一条策略同时配置多个服务，且同时支持多个区域的实例授权
  - 开放拒绝权限效力，拒绝权限始终优先于允许权限
  - 区分所有资源和特定资源，使用所有资源以支持创建/添加等无资源实例权限操作
- IP白名单功能当前已支持限制部分服务的OpenAPI调用，以确保只有被允许的IP地址才能通过编程的方式访问你的云资源，防止企业数据的泄漏，满足企业的强安全需求
- 组织单元可应用于企业内的组织架构层级、代理商模式下的代理商和客户层级场景下，账户的分组和权限的同一限制管理等。本次改版主要提升了如下特性：

2019-07	• 拆分了组织单元与账户的逻辑关系，凸显组织单元的容器特性 • 可视化组织单元与账户内SCP策略的作用逻辑 • 升级了用户的交互体验 • 在为子用户、组、角色授权特定服务的系统策略时，支持为其关联的服务一并授权，当前已支持的原始服务有：BCC、DCC、RDS、SCS、BLB、NAT网关、VPN网关、对等连接、专线网关、BMR等，被关联的权限为VPC、子网、安全组等 • 被授权对象——子用户、用户组、角色——等已支持批量授权和移除自定义策略，以提升你的授权操作效率和便利性 • 标签授权服务新接入了云服务器BCC、物理服务器DCC、云磁盘CDS等服务
2019-06	用户可以通过提供的REST API接口，进行用户和组的管理、及其权限授予等操作
2019-04	新增角色管理功能。IAM角色(后文统一称角色)是一种虚拟身份，同用户身份一样，可以关联权限对资源进行操作，但它没有确定的身份认证密钥，且需要被一个受信的实体用户扮演才能正常使用。你可以使用角色作为桥梁，向需要访问你云账户资源的用户、应用程序或服务提供访问权限
2019-03	支持设置登录session过期时间。登录session过期是指用户登录后，如果在有效时间内不进行任何操作，为保障账户安全，需要用户进行重新登录
2018-11	Cloud Trail新增接入产品DCC/RDS/DTS/DNS/EIP/TAG/SCS/BES/BCH/EIPBP/SMS/BTS。
2018-09	可以在IAM的操作记录中查看所有用户（主账户／子用户）的访问记录，以及对您的资源实例所进行的运维管控类记录，用于进行安全分析、资源变更以及合规性审计

产品描述

产品概述

多用户访问控制(Identity and Access Management, IAM)，主要用于百度智能云的身份管理和访问控制，解决云账户的集中授权与管理、资源分享与多用户协同工作等问题。多用户访问控制适用于企业内的不同职能角色，你可以对不同员工赋予产品的不同权限，以共享你账户内的资源，完成他们的工作。当你的企业存在需要多用户协同工作、分享资源时，推荐你使用多用户访问控制。

以下是多用户访问控制适用的典型用户和场景：

- 中大型企业客户：对公司内多个部门的不同员工进行集中资源和权限管理；
- 独立软件服务商(ISV)或SaaS平台商：对代理客户进行集中的资源和权限管理；
- 中小开发者或小企业：添加项目成员或协作者，进行资源和权限管理。

概念

在使用IAM服务前，你需要了解和熟悉IAM服务的关键概念，这对于你能针对企业的实际情况灵活地运用IAM的能力至关重要。IAM主要解决云账户的身份管理和访问控制问题，其相关概念也将围绕这两个方面展开。

身份体系

账户

百度智能云上的最小资源隔离和计费主体，是客户云上资源的集合和拥有者。客户在智能云控制台上注册时自动生成，未来用于云资源管理、出账单等的独立空间。

主用户

客户云账户创建时，由系统自动创建的超级管理员用户。由于主用户拥有云账户的所有权限，为保证账户资源的安全，**强烈建议**不要直接使用主用户直接管理你的云账户，建议使用主用户创建管理员子用户进行后续的资源管理和操作。

子用户

IAM 身份体系下一种用户，用于共享或协作主用户的云上资源。用户名在账户下唯一。子用户可以是人、也可以是服务或应用程序，即子用户可以通过账号密码的方式登录控制台，也可以使用API通过编程访问的方式访问账户内云资源。

消息接收人

IAM 身份体系下一种特殊用户，仅用做消息的接收，不能通过任何方式访问云账户内资源，常用于主用户将特殊的智能云消息发送给其企业和团队内的成员。子用户默认具备消息接收人的属性。

用户组

拥有相同职能子用户或消息接收人的集合。为用户组授权，组内的用户将自动继承组拥有的所有权限。一个子用户同时加入多个用户组。你可以将新入职的员工加入特定用户组，以快速获得相应职能的所有权限，对于转岗的员工，也可以直接将其从用户组中移除，以移除其不需要的权限。

角色

IAM角色是一种虚拟身份，同用户身份一样，可以关联权限对资源进行操作，但它没有确定的身份认证密钥，且需要被一个受信的实体用户扮演才能正常使用。

联合身份

企业已有自己的身份系统，作为身份服务提供商(IdP),以百度智能云为服务提供商(SP)，使用企业的身份账号可单点登录进入百度智能云。通常与角色结合使用，用于访问账户内的云资源。

用户凭证

与用户关联的安全凭证，用于验证用户身份。IAM的用户凭证当前分为如下三类

- 用户密码：密码的设定取决于用户是否需要登录控制台；
- AccessKey：用户通过编程访问控制台时的凭证，用于访问API或使用SDK时做签名验证；

- Token：临时授权的场景下，Token由STS服务提供给扮演角色的用户。

访问控制

权限

允许或拒绝一个用户对某种资源执行某种操作。比如，资源管控：创建、删除一台BCC服务器；资源运维操作：停止、重启BCC服务器，不涉及资源生命周期的变更；只读：查看。

策略

用户权限的集合，定义了用户对云账户内的资源可以进行什么操作。

策略语法

策略的描述，每条策略关联一条 ACL，JSON格式。

资源

云服务呈现给用户与之交互的对象实体的一种抽象，如BCC实例，BOS存储桶。

策略与身份

策略可以被关联到用户、用户组、角色，通过策略实现对云资源的访问控制。

产品功能

集中授权和分权访问控制

IAM集中管理了所有百度智能云上的资源权限，可以把你的云账户内的不同资源按职能授予给你企业或团队内的成员，以实现资源共享以及分权控制。

细粒度的权限管理

IAM包含了系统策略和自定义策略。系统策略可控制的权限粒度为产品级别，即产品内的所有资源实例；而对于已支持自定义策略的产品，可以将权限粒度精细控制到实例级别，比如，你可以为某一用户授予某一台BCC服务器的只读权限，也可以为另一用户授予一个特定BOS存储桶的管理权限。

跨账户的资源访问

在一些业务场景下，如代运维场景，你需要跨账户对其他云账户的资源进行访问，IAM提供角色管理的功能支持用户在获得显示授权的前提下，临时获得其他云账户的访问权限，以支持完成对其他云账户的资源访问操作。

双因素安全认证

双因素安全认证(Multi Factor Authentication,MFA)用于保证云账户的身份访问安全，IAM支持使用手机短信和虚拟MFA App的方式，对需要访问智能云的用户进行二次身份校验，同时，对于在云账户内的关键操作进行保护，以保证你的云账户安全。

联合身份

中大型企业客户通常在企业内部IT具备身份管理系统，已经打通企业内部的其他服务，此时企业客户在使用百度智能云时，期望把智能云作为其服务提供商(SP)之一，实现身份的联合认证，企业自身此时扮演身份服务提供商(IdP)的角色。IAM提供了支持SAML2.0标准协议的联合身份管理功能，你可以通过简单的配置将企业的内部身份管理系统与百度智能云的账户系统实现联通，详细功能请参见外部账号接入。

审计支持

IAM的关键用户操作行为已经纳入百度智能云云审计 BCT中，你可以查询账户内用户90天内的操作记录，也可以创建跟踪，以实现长期的审计记录存储。

应用场景

企业子用户管理与分权

某企业的A账户购买了多种云资源（例如：云服务器BCC，对象存储BOS，内容分发网络CDN等）。该企业拥有许多员工，包括开发人员、测试人员、运维人员等，由于每个员工的工作职责不同，所以需要的权限也不同，且员工不需要对操作产生的费用单独支付。

场景说明：

- 某企业的A账户可以为不同的工作人员赋予使用产品的不同权限，例如，只读，运维，管理，同时可以将控制权限细化到资源层级，例如，BCC某个instance的操作权限。
- 企业员工使用子用户账号进行登录使用，且不需要对操作产生的费用单独支付。

不同企业之间的资源管理与授权参考

- 企业组织
- 企业组织vs多用户访问控制

企业组织vs多用户访问控制

企业组织

企业组织：适用于企业组织间，代理商与客户等，各个主体都是独立账户（具有独立账单），这些独立账户之间有组织层级关系，上层可以管理下层的财务和资源、控制下层账户的操作权限。

场景一：A公司拥有多个子公司，同一主体，A公司和子公司之间既是相互关联的一个整体，也相对独立运行。

场景说明：

- A公司和子公司都希望具有单独的使用账户，且可在百度智能云上单独使用。
- A公司和子公司相互独立管理资源，A公司拥有对子公司的监管权，可管理子公司的资源。
- A公司和子公司同一主体，A公司来统一支付A公司和子公司的账单。

场景二：代理商B拥有多个客户，代理商有管理这些客户的需求，同时代理商和每个客户之间有独立管理资源的需求；

场景说明：

- 出于安全考虑，代理商B希望客户有单独的使用账户。
- 代理商B拥有对客户的资源使用监管权限。
- 如果代理商B和客户合同终止，代理商B可以随时解除客户的授权。

多用户访问控制

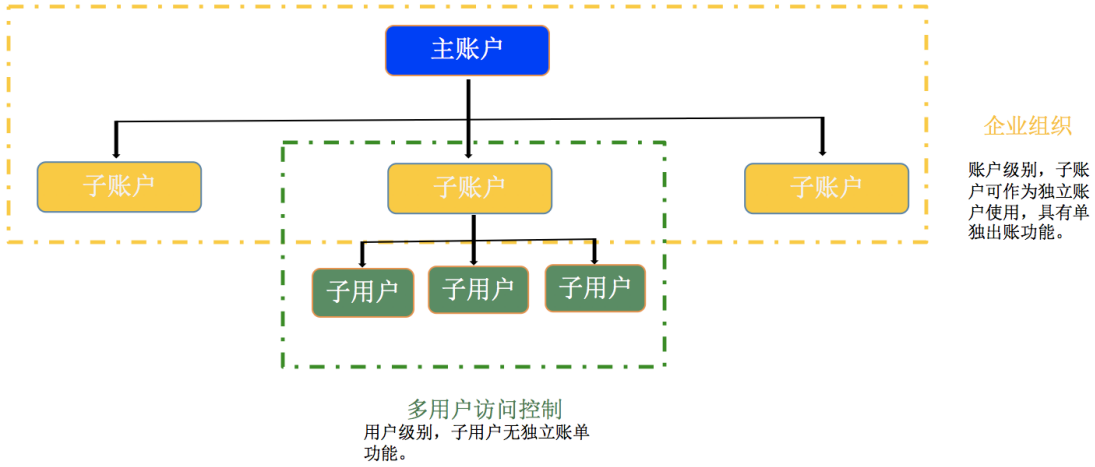
多用户访问控制：适用于企业组织内的不同角色，可以对不同的工作人员赋予使用产品的不同权限，例如，只读，运维，管理，也可细化到资源级别，且子用户不需要对操作产生的费用单独支付。当您的企业存在多用户协同操作资源时，推荐您使用多用户访问控制。

应用场景：某企业的A账户购买了多种云资源（例如：云服务器BCC,对象存储BOS,内容分发网络CDN等）。该企业拥有许多员工，包括开发人员、测试人员、运维人员等，由于每个员工的工作职责不同，所以需要的权限也不同，且员工不需要对操作产生的费用单独支付。

场景说明：

- 某企业的A账户可以为不同的工作人员赋予使用产品的不同权限，例如，只读，运维，管理，同时可以将控制权限细化到资源层级，例如，BCC某个instance的操作权限。
- 企业员工使用子用户账号进行登录使用，且不需要对操作产生的费用单独支付。

区别



区别	企业组织	多用户访问控制
资源归属	资源归属于各个账户，哪个账户开通/购买，就属于哪个账户。	资源归属于主账户，不属于子用户。
资金和账单归属	企业组织中的每个账户为资金的拥有者，可以单独出账单，同时主账户可以申请开通财务管理权限，通过资金划拨的方式统一支付组织中所有子账户的账单。	账户为资金和计费的载体，子用户不会单独出账单，账户下所有子用户产生的资源费用都记在主账户下。
使用场景	适用于企业组织间，各个主体都是独立账户（具有独立账单），这些独立账户之间有组织层级关系，上层可以管理下层的财务和资源、控制下层账户的操作权限。	适用于企业组织内的不同角色，可以对不同的工作人员赋予使用产品的不同权限，例如，只读，运维，管理，也可细化到资源级别，且子用户不需要对操作产生的费用单独支付。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。
区别	A公司拥有多个子公司，同一主体，A公司和子公司之间既是相互关联的一个整体，也相对独立运行。 场景说明： 1.A公司和子公司都希望具有单独的使用账户，且可在百度智能云上单独使用。 2.A公司和子公司相互独立管理资源，A公司拥有对子公司的监管权，可管理子公司的资源。 3.A公司和子公司同一主体，A公司拥有整体的财务结算权，A公司来统一支付A公司和子公司的账单。	某企业的A账户购买了多种云资源（例如：云服务器BCC,对象存储BOS,内容分发网络CDN等）。该企业拥有许多员工，包括开发人员、测试人员、运维人员等，由于每个员工的工作职责不同，所以需要的权限也不同，且员工不需要对操作产生的费用单独支付。 场景说明： 1.某企业的A账户可以为不同的工作人员赋予使用产品的不同权限，例如，只读，运维，管理，同时可以将控制权限细化到资源层级，例如，BCC某个instance的操作权限。 企业员工使用子用户账号进行登录使用，且不需要对操作产生的费用单独支付。

目前支持的产品线

IAM 作为百度智能云的身份和访问控制服务，提供了集中管理云平台产品服务权限的功能，相应云服务都需要接入IAM，从而实现产品内的权限管控。本文档将详细介绍已经接入IAM服务的云产品，其支持的权限颗粒度、相关使用文档等。IAM目前主要为云产品线提供2种主要类型的服务：

- 多用户访问控制（IAM），主要解决主子用户的身份、授权、鉴权等问题；
- 临时身份凭证（STS），IAM 为产品或服务提供的临时身份管理服务。

平台模块权限

平台策略主要描述的百度智能云平台通用服务模块的策略，包含但不限于系统层级的管理员、运维、只读，财务、工单、证书

管理等服务模块，平台策略属于IAM的系统策略。

权限名称	策略说明	相关文档
系统管理员权限	拥有管理所有百度智能云资源的权限	-
系统运维权限	包括接入鉴权运维类的所有产品线	-
系统只读权限	包括接入鉴权只读类的所有产品线	-
财务权限	拥有查看、支付、取消订单的权限	-
证书管理	支持证书只读、运维权限	证书管理
工单系统管理员(TicketFullControlPolicy)	管理全局工单的权限。支持创建工单，以及查看、回复、删除账户下全部工单	-
工单基础管理权限(TicketUserControlPolicy)	管理子用户自身工单的权限。支持创建工单，以及查看、回复、删除当前登录子用户工单	-
IAM系统管理员(IAMFullControlAccessPolicy)	拥有管理多用户访问控制的权限	-
IAM只读权限(IAMReadAccessPolicy)	拥有只读多用户访问控制的权限，包含访问报告的下载权限	-
AK管理权限(IAMManageAccessKeyPolicy)	增删管理子用户自身AccessKey的权限，创建子用户时，如勾选了使用编程方式访问，则默认授予该子用户当前权限	-
云审计管理权限	拥有管理云审计记录、下载审计日志、管理跟踪等所有权限	-
云审计只读权限	拥有管理云审计记录、下载审计日志、查看跟踪等只读权限	-

已接入IAM的产品服务说明

本节将描述已接入IAM和STS的产品服务，如下表格中字段的意义说明如下：

- 产品名称：百度智能云产品服务的中文+英文简称；
- 权限粒度：包含服务级和资源级，服务级表示将云产品作为一个整体授权，资源级表示可以精确授权到一个云产品下的实例，比如某台BCC服务器；
- 系统支持操作权限：服务级权限粒度下，云产品所支持的系统策略；
- 临时身份凭证(STS): “ ✓”表示支持，“-”表示不支持；
- 标签授权： 可以根据所选标签，筛选所需要授权的权限和资源，“ ✓”表示支持，“-”表示不支持；
- 相关文档：有链接表示当前产品已关联了相应权限说明文档，“-”表示暂无。

计算

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	标签授权	相关文档
云服务器 BCC	资源级	只读、运维、管理权限	✓	✓	BCC
专属服务器 DCC	资源级	只读、运维、管理权限	✓	✓	-
物理服务器 BBC	资源级	支持只读、运维权限	✓	-	-
容器引擎 CCE	资源级	开发、运维、管理权限	✓	-	CCE
应用引擎专业版 BAEPRO	资源级	-	✓	-	BAEPRO
函数计算 CFC	服务级	只读、管理权限	-	-	-
轻量应用服务器 LS	资源级	只读、运维、管理权限	-	-	LS

网络

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
弹性公网IP EIP	资源级	只读、运维、管理权限	✓	✓	EIP
共享带宽 EIPGROUP	资源级	只读、运维、管理权限	✓	-	EIPGROUP
EIP带宽包 EIP_BP	服务级	只读、运维、管理权限	-	-	EIP_BP
负载均衡 BLB	资源级	只读、运维、管理权限	✓	✓	BLB
私有网络 VPC	资源级	只读、运维、管理权限	✓	✓	NETWORK
子网 subnet	资源级	只读、运维、管理权限	✓	-	subnet
安全组securityGroup	资源级	只读、运维、管理权限	✓	✓	安全组
访问控制列表 ACL	资源级	只读、运维权限	✓	-	ACL
路由表Route	资源级	只读、运维权限	✓	-	路由表
专线网关	资源级	只读、运维、管理权限	✓	-	专线网关
VPN 网关	资源级	只读、运维、管理权限	✓	-	VPN网关
NAT网关	资源级	只读、运维、管理权限	✓	-	NAT网关
IPv6 公网网关	资源级	只读、运维、管理权限	-	-	IPv6
对等连接 PEERCONN	资源级	只读、运维、管理权限	✓	-	对等连接
专线ET	服务级	只读、运维、管理权限	-	-	专线ET
智能网络接入服务 SMART_WAN	服务级	运维、管理权限	-	-	-
云智能网络 CSN	服务级	管理权限	-	-	-
解析器 RESOLVER	服务级	只读、运维、管理权限	-	-	DNS

安全和管理

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
流量审计 IDS	资源级	只读、运维权限	-	-	-
应用防火墙 WAF	资源级	只读、运维权限	✓	-	-
主机安全客户端 HOSTEYE	资源级	只读、运维、管理权限	-	-	-
DDoS高防服务 ADAS	服务级	只读、运维、管理权限	✓	-	-
业务安全风控 AFD	资源级	管理权限	-	-	AFD
安全检测服务 SRD	资源级	只读、运维、管理权限	✓	-	SRD
云审计 BCT	服务级	只读、管理权限	-	-	-
史宾格安全及隐私合规助手 SPRINGER	服务级	管理权限	-	-	SPRINGER
云防火墙 CFW	服务级	只读、管理权限	-	-	-

存储和CDN

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
对象存储 BOS	资源级	只读、管理权限	✓	✓	BOS
云磁盘 CDS	资源级	只读、运维、管理权限	✓	✓	-
内容分发网络 CDN	资源级	只读、运维、管理权限	✓	✓	CDN
文件存储 CFS	资源级	只读、运维、管理权限	-	-	CFS
百度表格服务 BTS	资源级	只读、运维权限	-	-	BTS
边缘计算节点 BEC	服务级	只读、运维、管理权限	-	-	-

数据分析

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
百度MapReduce BMR	资源级	-	✓	-	BMR
百度Elasticsearch BES	资源级	只读、运维、管理权限	✓	-	BES
百度消息服务 BMS	资源级	只读、运维、管理权限	-	-	BMS
百度日志服务 BLS	资源级	只读、运维、管理权限	-	-	BLS

数据库

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
关系型数据库 RDS	资源级	只读、运维、管理权限	✓	✓	RDS
简单缓存服务 SCS	资源级	只读、运维、管理权限	✓	✓	SCS
文档数据库 MongoDB	资源级	只读、运维、管理权限	-	-	MongoDB
数据库审计 DBAudit	服务级	只读、运维、管理权限	-	-	DBAudit
云原生数据库 GaiaDB-S	资源级	只读、运维、管理权限	-	-	GaiaDB-S
数据传输服务 DTS	服务级	支持只读、运维、管理权限	-	-	DTS

智能多媒体服务

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
音视频直播 LSS	服务级	只读、管理权限	✓	-	-
音视频点播 VOD	服务级	管理权限	✓	-	-
音视频转码 MCT	服务级	管理、只读权限	✓	-	-
视频内容审核 VCR	服务级	只读、管理权限	✓	-	-

物联网服务

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
云秘智能客服 CVCA	资源级	-	-	-	-
时序数据库 TSDB	资源级	-	✓	-	TSDB
百度智能边缘 BIE	资源级	只读、管理权限	-	-	BIE

网站服务

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
云主机管家服务 HME	服务级	支持只读、管理权限	-	-	-
智能流量管理 ITM	服务级	支持只读、管理权限	-	-	ITM
域名服务 BCD	服务级	支持只读、运维、管理权限	✓	-	-

应用服务

产品名称	权限粒度	系统支持操作权限	临时身份凭证(STS)	基于标签授权	相关文档
简单消息服务 SMS	服务级	支持只读、管理权限	✓	-	-

系统限制

限制项	限制值
一个主账号下的子用户总数	500
一个主账号下的用户组总数	50
一个用户组关联的子用户数	20
关联到用户组的策略数	5
每个用户可创建的AccessKey个数	20
每个用户最少AccessKey个数	1
用户名称	255字符
用户密码	8-64字符
用户组名称	64字符
用户组备注	128字符
自定义权限策略名称	64字符

产品定价

产品定价

说明

百度智能云IAM（Identity and Access Management）是百度智能云为用户提供的免费服务，开通后即可使用，IAM不收取服务费用。

快速入门

创建子用户管理员

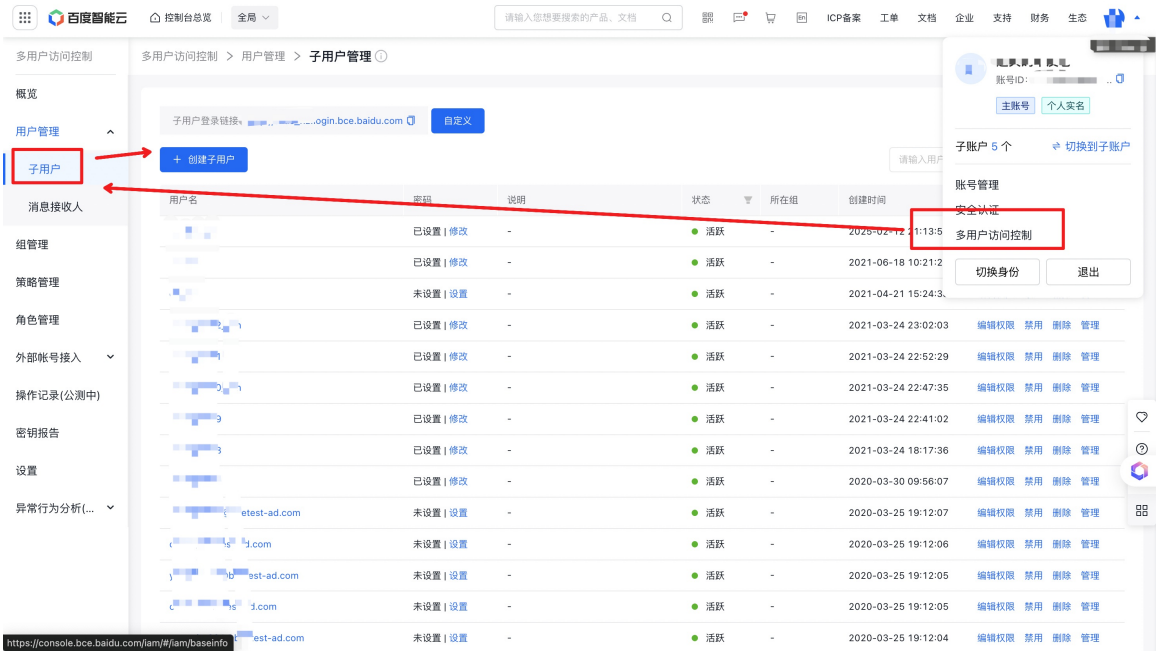
你可以快速创建子用户管理员，替代主账号进行云上资源的操作和管理，子用户系统管理员具备对云上所有资源的管理权限。

为什么需要创建子用户管理员

百度智能云主账号拥有对云账户内所有资源的全部操作权限，一旦泄露会对云上资源管理和业务运行带来巨大的风险，因此我们强烈建议你使用子用户管理员进行云账户和资源的管理工作。系统管理员策略拥有与主账号相同的权限范围，足以满足你日常的管理需求，同时可以保障云账户的安全性。为每个独立的管理员分配子用户，以避免主账号被共享使用，还能提升操作事件的可追溯性。

2步快速创建子用户管理员

1. 登录百度智能云管理控制台，鼠标移动到右上角头像处，在下拉列表中选择 **多用户访问控制**。选择**用户管理 > 子用户 > 创建子用户**。

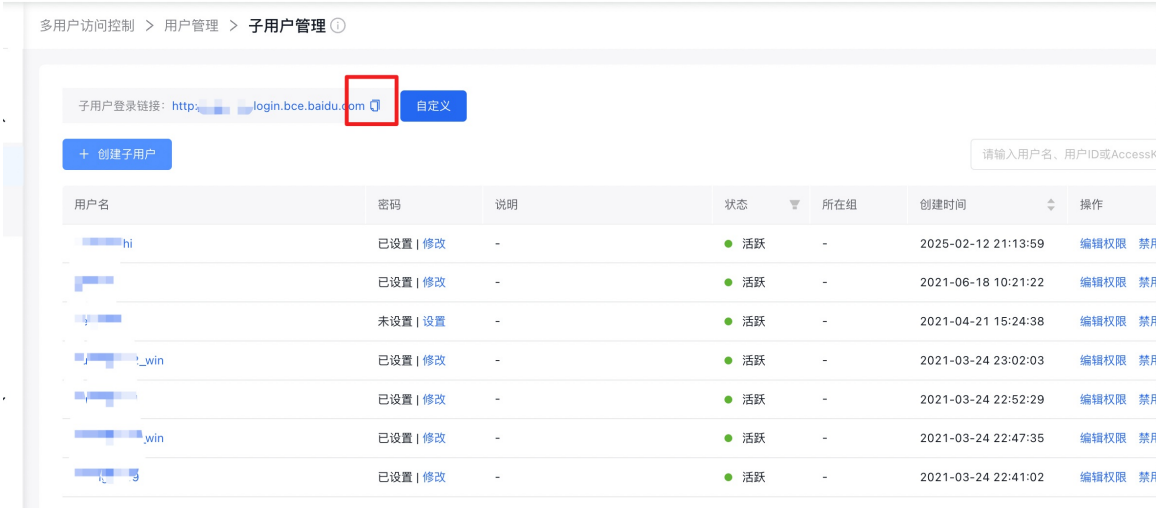


2. 在弹窗中输入用户名，点击勾选**系统管理员**，访问方式勾选**控制台密码访问**，按照密码规则要求填入密码，快速授权处勾选**系统管理员** ，点击确定，即可完成子用户管理员的创建

> 提示：你可以在这一步中勾选 **要求用户下次登录时必须重置密码** ，以避免因密码分发带来用户共享的问题。

使用管理员子用户登录云平台

1. 在**多用户访问控制 > 用户管理 > 子用户** 页面，用户列表上方，找到 **子用户登录链接**，点击复制。



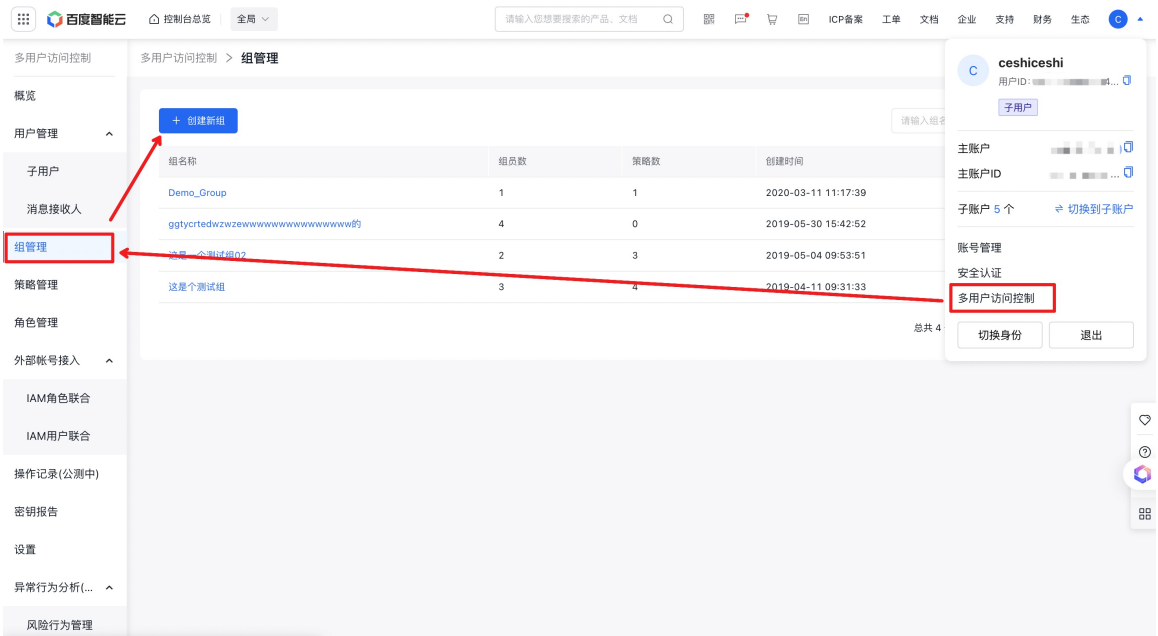
2.在浏览器链接粘贴打开(推荐使用谷歌浏览器)，输入上面步骤创建的用户名和密码，点击登录即可。

创建用户组并授权

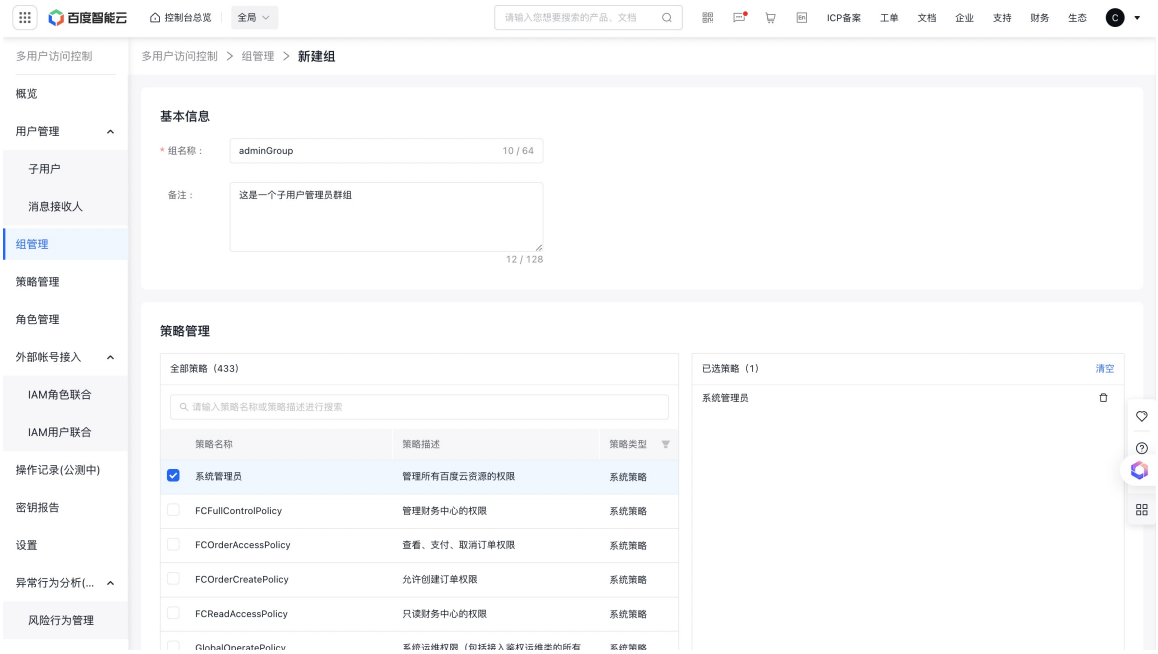
用户组是子用户的集合，通常将同一业务组或是需要同一类职责权限的用户放到一个，便于提升管理效率。后续如果有人员的变动，无需为每个人员单独授权或取消权限，只需要将人员所对应的子用户移入移出用户组即可。

创建用户组并授权

1. 登录百度智能云控制台，鼠标移动到右上角头像处，在下拉列表中选择 **多用户访问控制** > **组管理**，点击 **创建新组**。

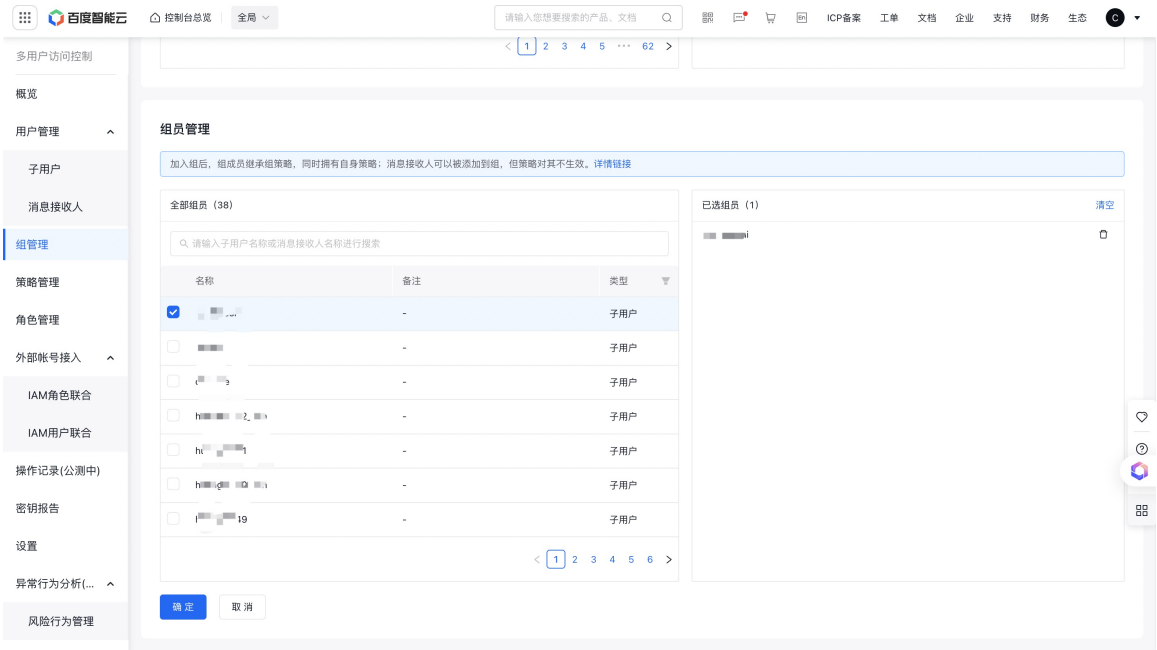


2. 在表单页面中填写组名称、备注信息。



3. 在 **策略管理** 模块选择需要为该用户组授予的权限。

4. 在 **组员管理** 模块选择需要添加到组的子用户，点击 **确认** 即可完成组创建。



你也可以暂时不对用户组添加权限策略或是子用户，创建完用户组并保存后，返回用户组列表页，点击 **编辑** 按钮，为用户组再添加权限策略和子用户。

操作指南

用户

用户管理

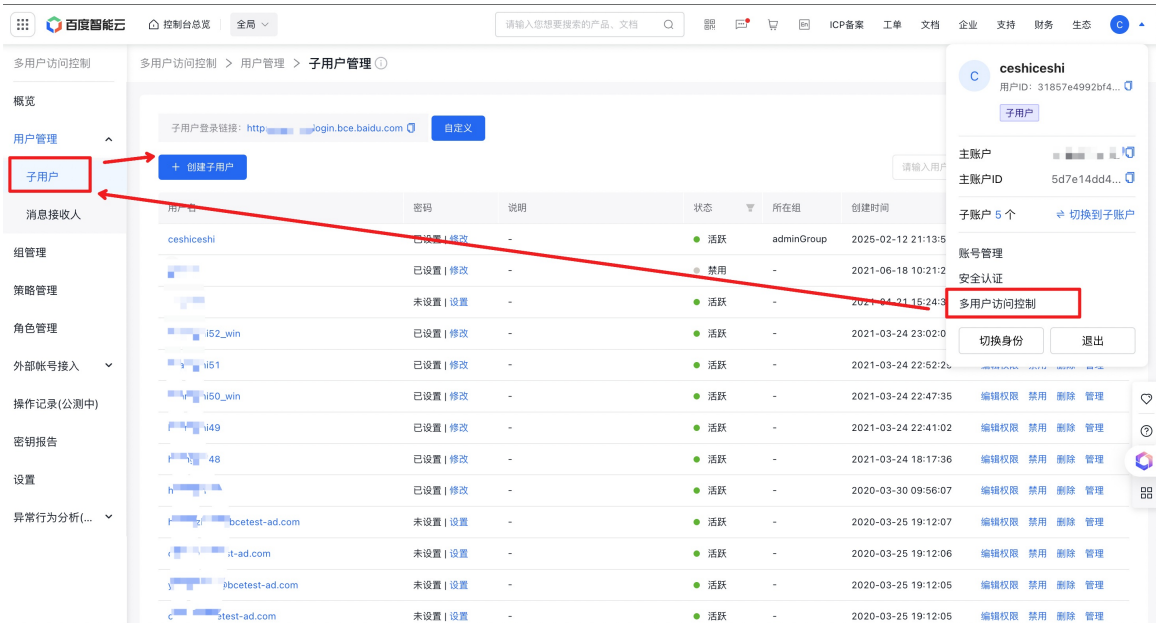
IAM用户管理分为两部分，即子用户管理和消息接收人管理：

- 通过使用子用户管理，你可以创建并管理账户下的子用户，规定子用户访问云账户内资源的形式，为子用户授予必要的资源访问权限，以及决定是否通过MFA的手段强制要求子用户使用多因素认证等，子用户默认具备消息接收人功能；
- 通过使用消息接收人管理，你可以创建并管理账户下的消息接收人，消息接收人在接收来自账户的消息通知前，需要通过验证，以防消息的滥发；

子用户管理

创建子用户

1. 登录百度智能云管理控制台，鼠标移动到右上角头像，下拉列表选择**多用户访问控制 > 用户管理 > 子用户**, 点击**创建子用户**。



2. 如果用户名，备注(可选)字段，你可以选择该子用户访问云账户资源的方式：

- 编程访问：授权子用户可通过编程方式访问云资源，系统默认为该子用户生成一对有效的AccessKey和SecretKey，用于调用智能云的API或通过SDK管理或使用云资源；
- 控制台密码访问：授权子用户可通过控制台登录云账户，可以选择创建全新的密码、系统自动生成密码以及绑定百度内网账号的方式设置密码。你还可以选择要求首次登录用户必须修改密码。

> 通常情况下，你只需要为子用户指定一种云资源的访问方式，你也可以为同一子用户同时指定2种访问方式。在后续的操作中，你也可以随时取消子用户对账户的访问。

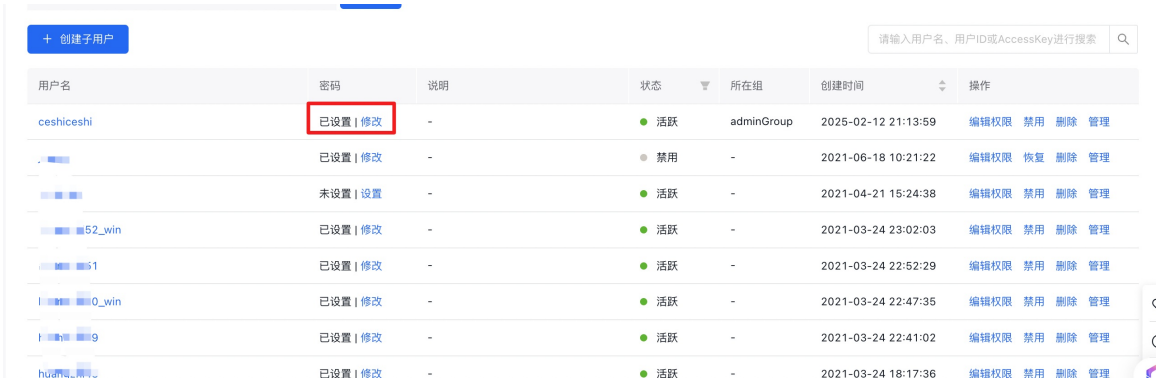
3. 在**快速授权**选项，你可以选择是否为子用户快速授权系统管理员权限，勾选后，该子用户将拥有该账户下所有云资源的管理权限，通常用于快速为团队创建云管理员。

设置子用户密码

为新建的子用户设置登录密码，子用户登录的两种选择：

- 方式一：推荐方式，使用设置的用户名+设置的密码，进行登录操作；
- 方式二：绑定百度账号，使用百度账号+对应的账号密码，进行登录操作；

1. 在**多用户访问控制** > **用户管理** > **子用户**菜单子用户列表中，选择子用户点击**设置**或**修改**进行密码的设置和修改，出现“设置密码”弹框。设置成功后，用户可以通过该【用户名】和【密码】进行登录。

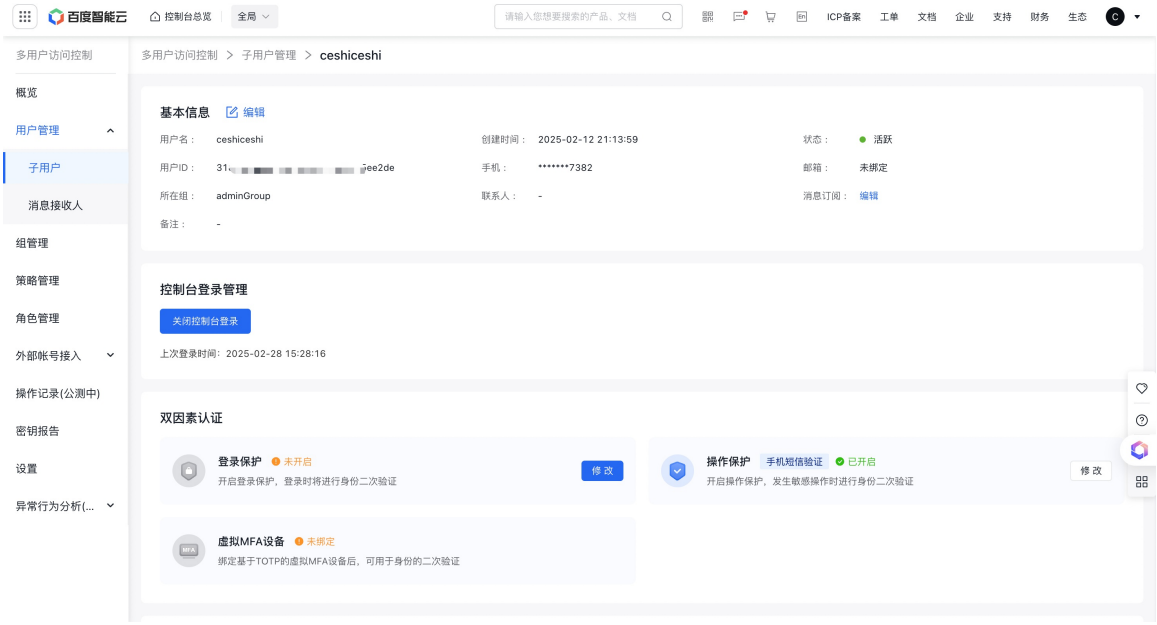


2. 你可以选择勾选“要求用户下次登录时必须重置密码”，则子用户在下次登录时，系统会要求子用户重设密码。

3. 你也可以选择“绑定内网账号”，选择“百度账号”，子用户登录时，使用百度账号和该账号的密码进行登录。

管理子用户信息

1. 在**多用户访问控制 > 用户管理 > 子用户**菜单子用户列表中，直接点击**用户名**或者**管理**，进入子用户详情页。
2. 修改子用户基本信息。



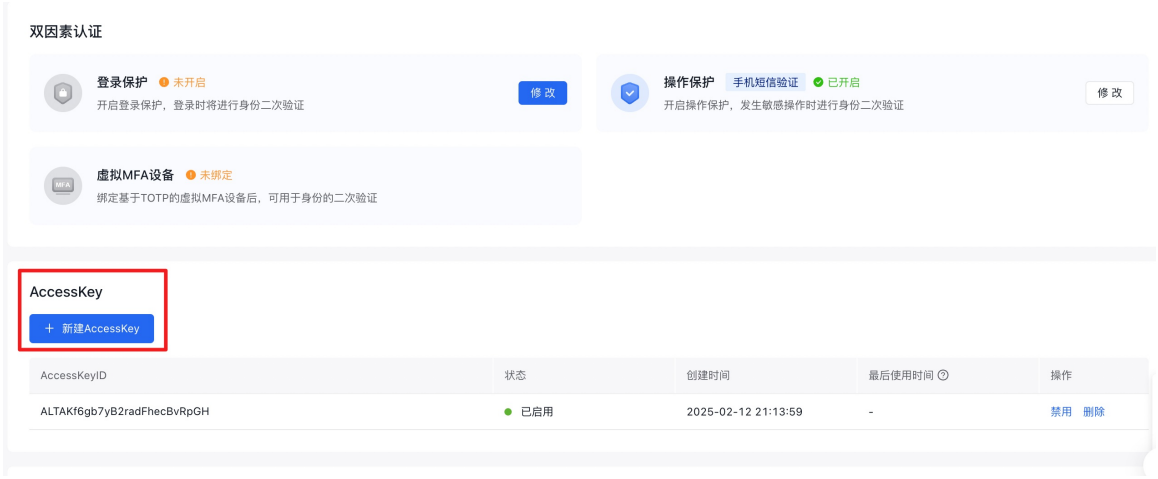
子用户双因素认证

双因素验证是一种简单有效的安全认证方法。它能够在用户名和密码之外，再增加一次额外的验证方式，通常使用短信验证码或是基于TOTP协议的一次性软件动态码的形式进行验证。

目前双因素认证可以提供登录保护和操作保护，分别对子用户的登录行为，和其对云资源的操作行为进行二次验证。详情请参考双因素验证。

管理子用户AccessKey

管理AccessKey：您可以根据实际情况删除或者新建AccessKey。子用户最多可以同时拥有20对AccessKey和SecretKey，但建议你只保存和使用2对，用于轮换。你可以删除最后一对AccessKey，以取消子用户对云资源的编程访问。



> 建议定期刷新AccessKey，以保证账户安全。

子用户授权

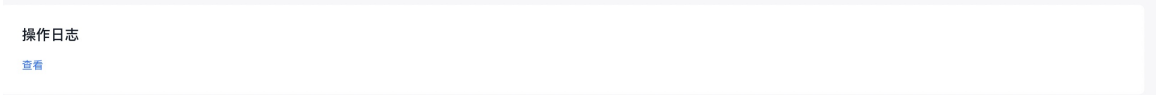
管理权限信息：在“权限信息”栏，您可以根据实际情况添加或删除子用户的权限。你可以为同一子用户授予多种服务、不同的资源访问权限。

管理子用户API Key 在AccessKey和SecretKey认证机制之外，IAM为开发者用户提供了API Key认证机制，子用户可以使用API Key直接使用千帆ModelBuilder、AppBuilder、AI开放平台内的服务能力，详细可见API Key简介。

作为管理员用户，你可以直接管理子用户的API Key，包含其生命周期的管理，以及Key权限的管理。当前1个子用户最多可以拥有200个API Key。

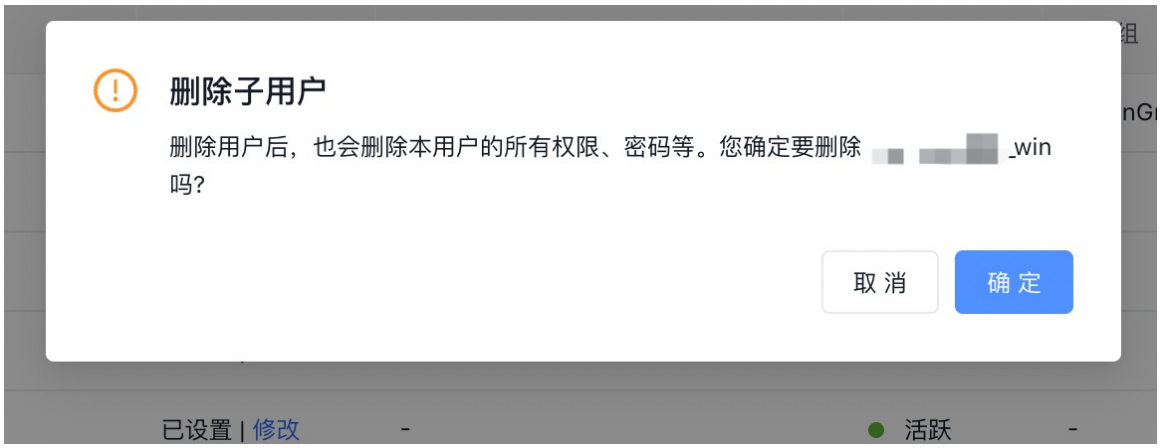
子用户操作日志

操作日志：目前已记录到云审计产品，点击**查看**会跳转到云审计页面。



删除子用户

在**多用户访问控制 > 用户管理 > 子用户**菜单子用户列表中，选择对应的子用户，点击删除，弹窗确认后可进行删除

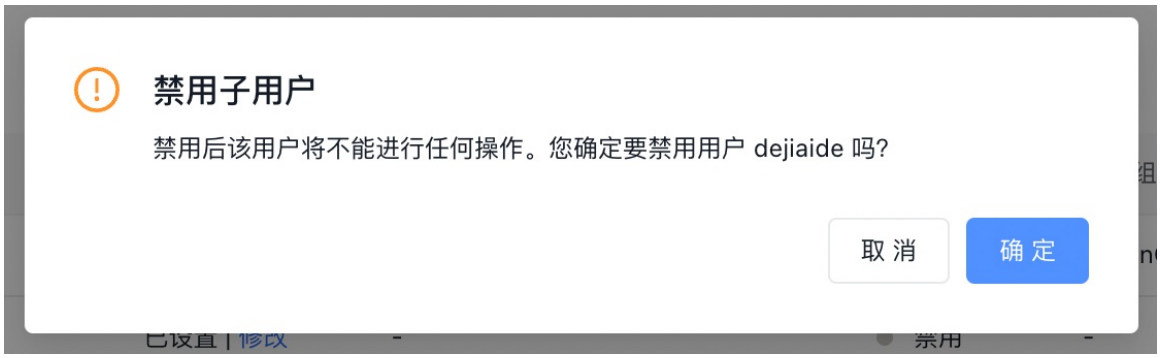


> 注意：删除子用户后，子用户对应的权限、密码也会随之删除，将无法恢复，所以在未能明确不需要子用户时，建议使用子用户的禁用功能。

禁用子用户

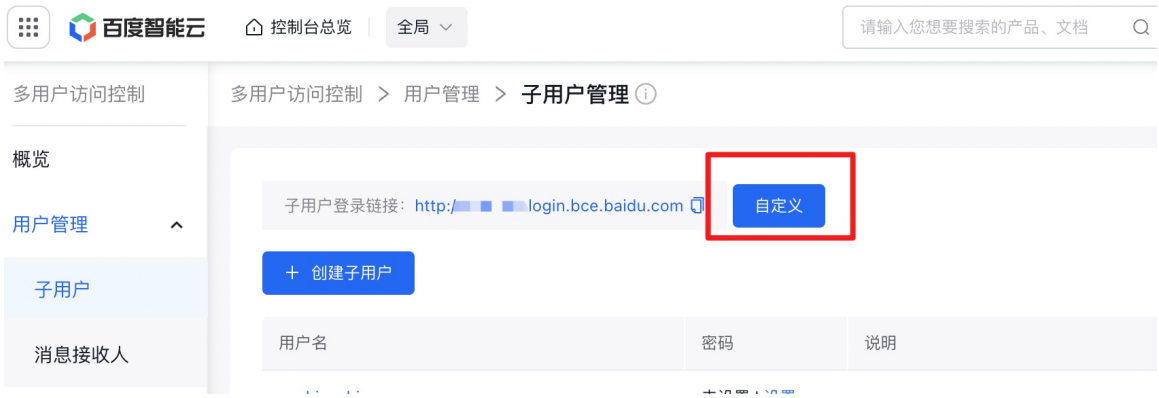
在某些场景下，你需要临时禁用子用户对云资源的访问权限，此时你可以使用子用户禁用功能。在“用户管理”的“子用户”页签中，选择对应的子用户，点击禁用，即可禁止该子用户使用。后续你也可以重新激活该子用户，恢复其访问权限。

禁用效果：子用户无法登录控制台，无法通过子用户的AKSK，API Key调用云账户的API。



主账号别名设置 您可以为主账号设置便于记忆的别名，以简化子用户登录链接和子用户登录时输入的主账号信息。

1. 在**多用户访问控制** > **用户管理** > **子用户**菜单子用户列表上方，点击”自定义“进行修改。



2. 输入自定义的主账号别名，点击确定并完成绑定手机的安全验证。

3. 修改完成后，子用户登录链接将进行简化，同时子用户登录时可输入主账号别名即可。

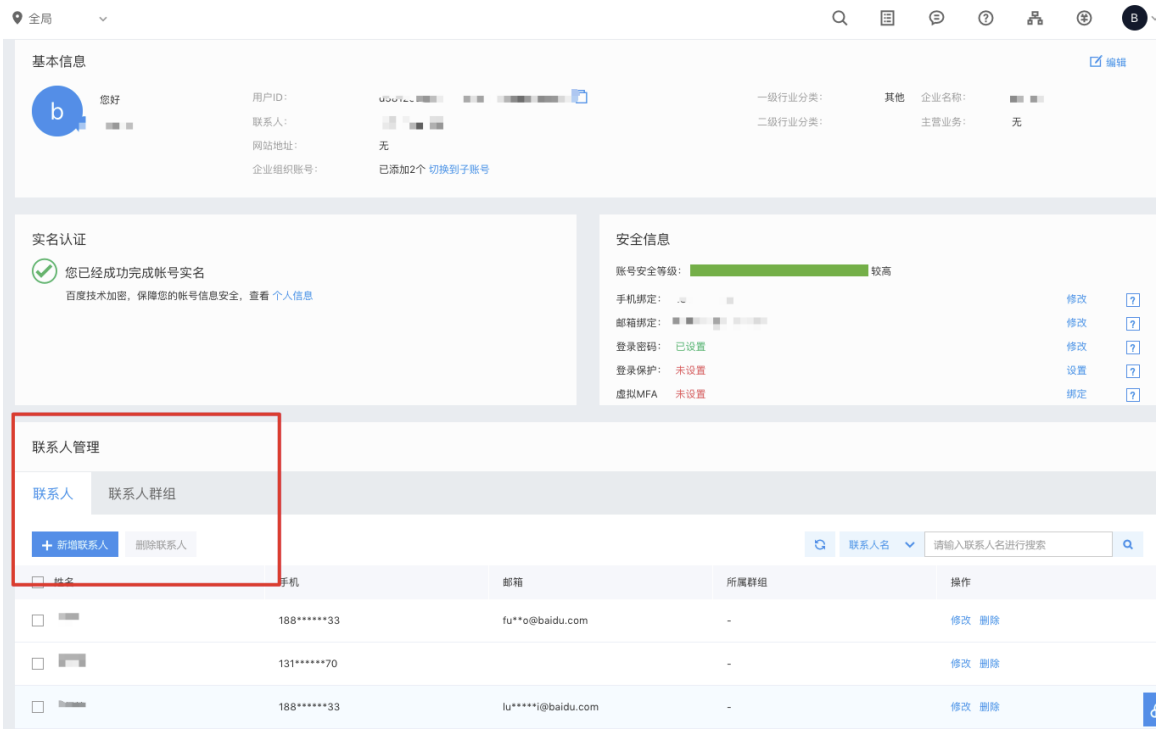
消息接收人管理

概述

消息接收人为用户的一种类型，它无法登录百度智能云控制台或者进行编程访问，仅可以通过主用户或管理员子用户设置用于接收来自账户的消息。

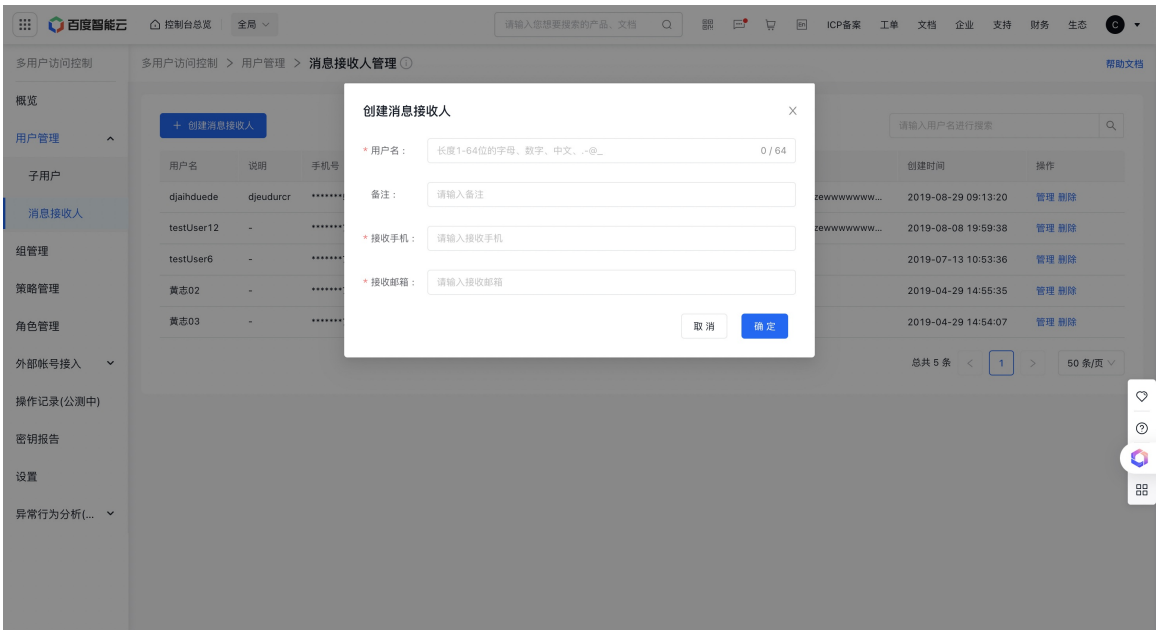
历史说明

原用户中心**联系人管理**迁移到**多用户访问控制 > 用户管理 > 消息接收人**进行统一管理。



创建消息接收人

1. 登录百度智能云管理控制台，选择**多用户访问控制**。
2. 选择**用户管理** > **消息接收人**，点击**创建消息接收人**。



> 注意：在同一账户下，消息接收人和子用户的用户名唯一；请填写有效的手机、邮箱，确保可以接收到消息通知，以验证消息接收人身份。

验证消息接收人

在接收来自账户的消息之前，消息接收人需要主动接收来自账户管理员发送的邀请：

1. 成功创建消息接收人，系统会自动发送短信和邮箱邀请给对应的消息接收人；
2. 你也可以在消息接收人列表中，通过查看消息接收人的短信/邮箱状态，确认其是否验证通过，对于未验证通过的用户，可以重发验证；

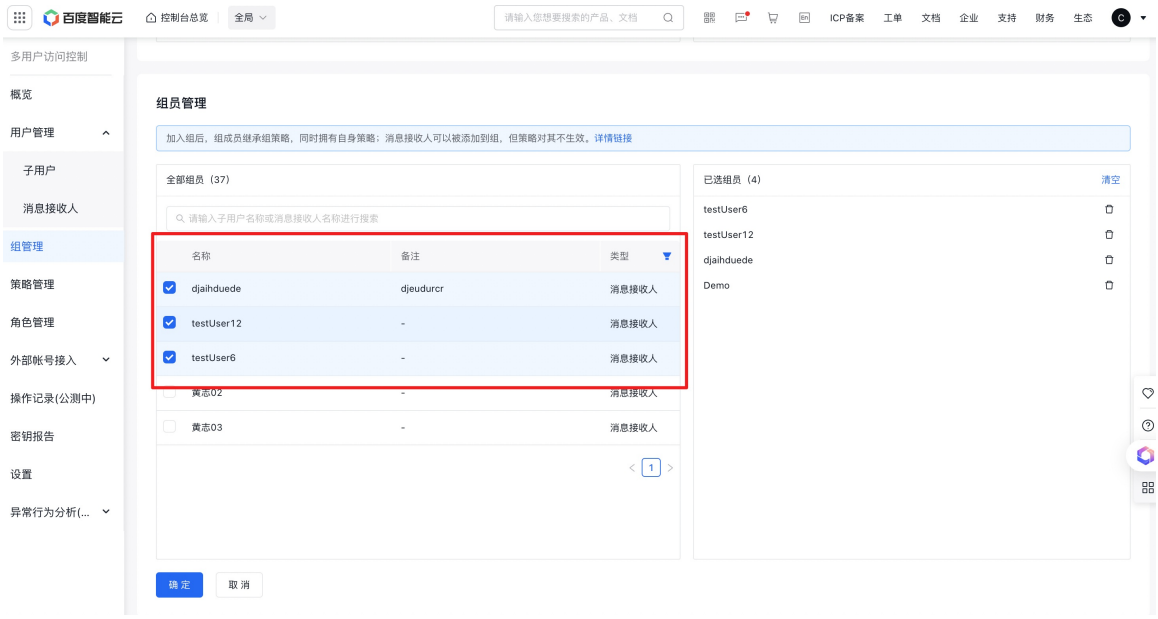
3. 消息接收人会通过短信或邮件的形式接收到来自账户的邀请，点击邀请链接，即可完成对其联系方式的验证。

为消息接收人订阅消息

点击**管理**，选择**编辑**，进入消息中心，为消息接收人设置订阅消息。

将消息接收人加入用户组

可以将消息接收人加入用户组，用户统一订阅消息，例如，可以在云监控BCM中设置用户组成员作为报警接收人等。



> 注意：消息接收人可以添加到用户组，权限策略对消息接收人不生效，如果希望一个用户可以拥有权限策略同时可以作为消息接收人，推荐使用**用户管理 > 子用户**。

双因素验证

什么是双因素验证

双因素验证(Two Factor Authentication)是在用户名和密码之外，额外增加的一层安全认证，用于确保登录和使用百度智能云资源的用户身份的安全性，业界也通常称为两步验证(2-Step Authentication)，或多因素验证(Multifactor Authentication)。

双因素验证方式 百度智能云支持手机短信、虚拟MFA等2种双因素验证方式：

- 短信验证：通过向云账号绑定的安全手机发送有时限要求的短信验证码，验证使用者身份；
- 虚拟MFA：通过云账号绑定的虚拟MFA App所产生的验证码，验证使用者身份，该验证方式安全级别较短信验证高，建议开启；

用户同时开启以上2种双因素认证方式时，只要任一验证通过即可。为保障你云上资源、资金等安全，建议主账号、管理员子用户为自身账号开启双因素验证，同时也可以为子用户开启双因素验证。

保护范围

百度智能云的双因素认证支持登录保护和操作保护：

- **登录保护：** 开启登录保护，登录账号时需要通过短信或虚拟MFA验证身份后才可登录成功。
- **操作保护：** 开启操作保护后，在用户进行敏感操作前，需要先完成身份的二次校验，以确保是本人操作；

开启双因素验证

任何百度智能云用户都可以为自身开启双因素验证，同时，为确保子用户访问云资源的安全，主账号以及管理员子用户可以为子用户开启双因素验证。

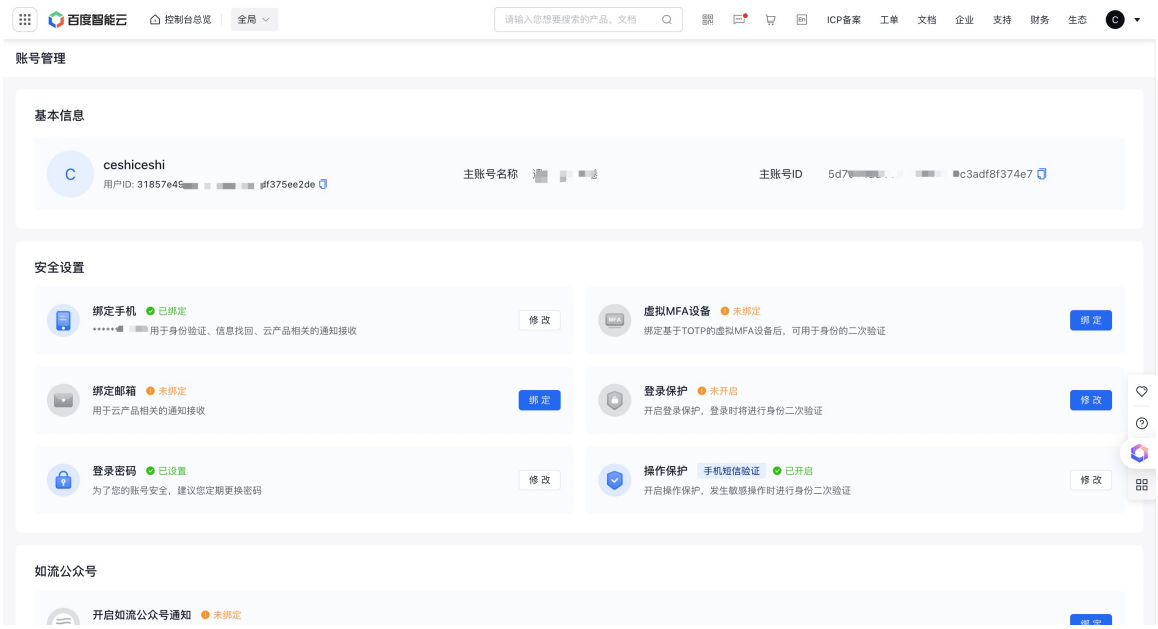
开启自身双因素验证

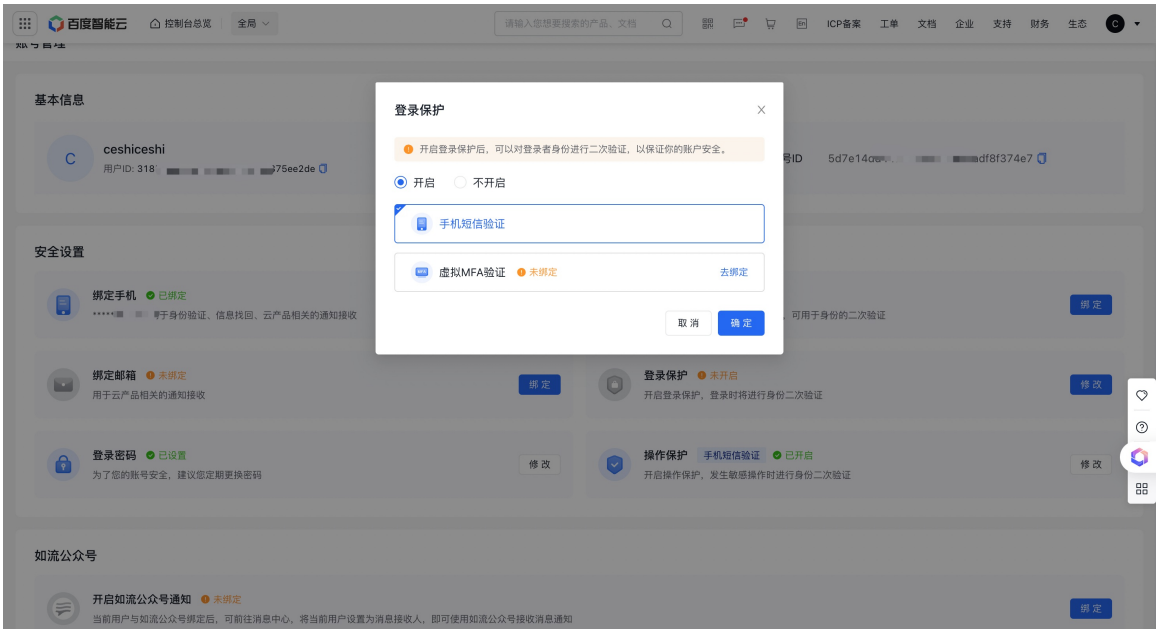
先决条件

拥有百度智能云账号：主账号或子用户。

操作步骤

1. 登录百度智能云控制台，或使用子用户登录链接完成登录；
2. 鼠标移动到右上角并点击用户头像，进入**用户中心**；
3. 在**安全设置**中的**登录保护**、**操作保护**中，点击**设置**按钮；
4. 在弹窗中，点击**开启**选项，你可以选择1种或2种双因素验证方式；
5. 点击**确定**完成设置。





> 主账号默认开启手机短信验证方式的操作保护

管理员为子用户开启双因素验证 先决条件

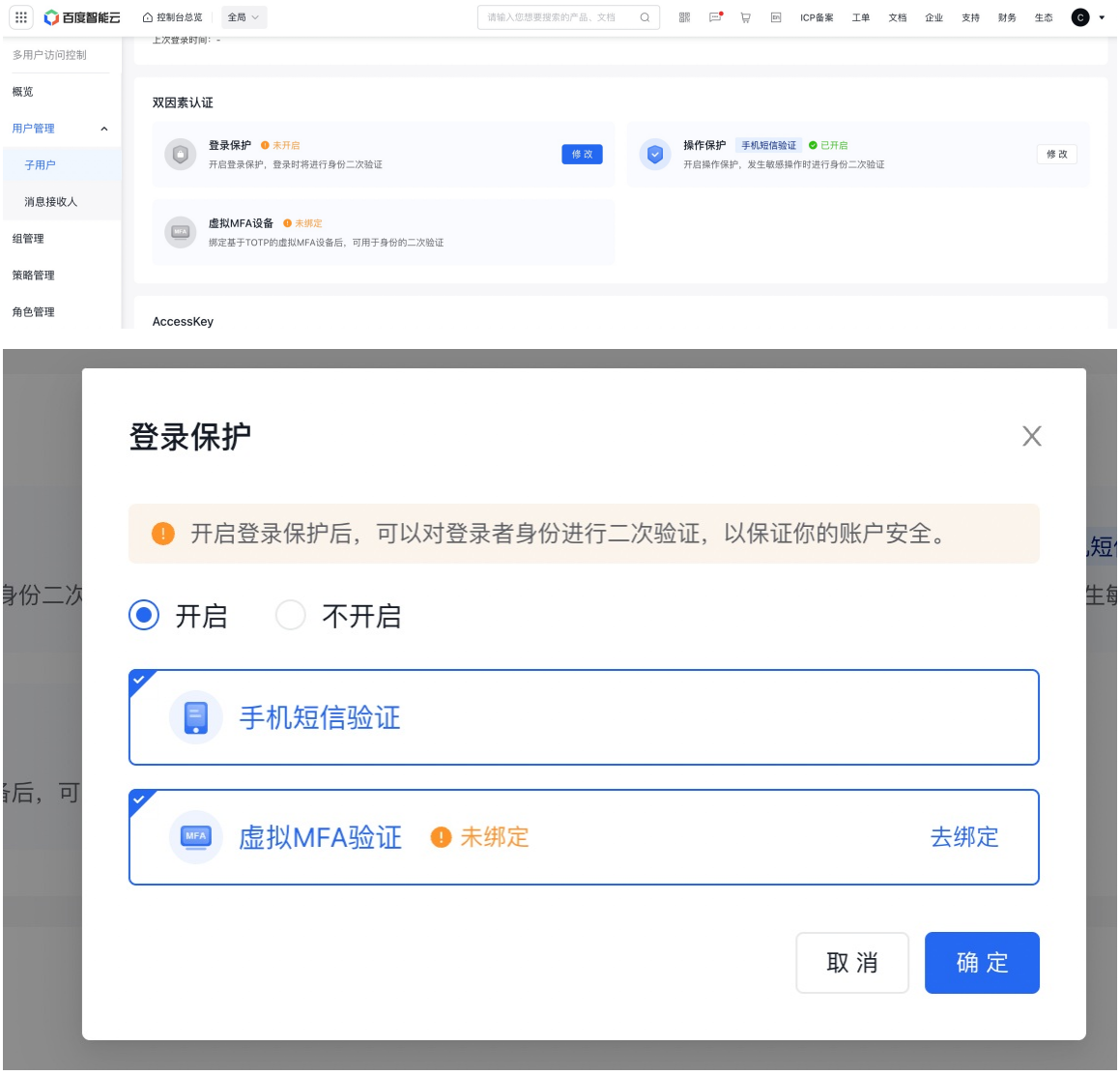
登录账号为主账号，或是管理员子用户。

操作步骤

1. 登录百度智能云控制台，或使用子用户登录链接完成登录；
2. 鼠标移动到右上角，点击**多用户访问控制**，进入**用户管理>子用户**；

3. 点击选中的子用户名称进入子用户详情页；

4. 在**双因素认证**模块，你可以为子用户选择开启**登录保护**和**操作保护**，**登录保护**和**操作保护**均可以选择开启1种或是同时开启2种双因素验证方式。



注意

1. 如果子用户被开启登录保护功能时，子用户尚未绑定手机号或绑定虚拟MFA，则在子用户下次登录控制台时，系统会自动跳转至绑定手机号/虚拟MFA的页面，进行手机/虚拟MFA绑定操作，详细请参考绑定虚拟MFA；
2. 如果管理员为子用户开启了操作保护功能，且子用户绑定了手机号或虚拟MFA，检测到账号异常或执行敏感操作时，需要子用户通过手机短信或软件动态码进行二次验证；
3. 如果管理员为子用户关闭了操作保护功能，检测到账号异常或执行敏感操作时，不需要子用户进行二次验证。

关闭双因素验证

操作方法和路径，请参考开启双因素验证，将原本已开启的登录保护或操作保护设置称不开启状态。

绑定虚拟MFA

用户为自身开启虚拟MFA验证、管理员用户为子用户开启虚拟MFA验证后，当使用虚拟MFA首次进行二次验证时，需要首先绑定虚拟MFA。

操作步骤

1. 下载身份验证器。您必须先在智能设备上安装一个 MFA 应用程序，才可继续进行操作，建议安装**百度智能云APP**或**Google Authenticator**。

> 注意：
>
> - ios版直接在苹果应用市场 (App Store) 搜索“百度智能云”或“Google Authenticator”进行安装。
> - 安卓版 Google Authenticator 还依赖外部二维码扫描组件，您还需要在应用市场中搜索安装“条码扫描器”。所以推荐安卓版用户使用“百度智能云APP”或您常用的其他身份验证器进行安装。

2. 绑定MFA。
   - 选择百度智能云APP：进入控制台页面，点击“虚拟MFA”，扫描电脑端上的二维码。
   - 选择其他身份验证器：点击身份验证器App上的扫一扫图标，扫描电脑端上的二维码）。

扫描成功后，系统会自动添加用户，**App**上会显示账号和密钥。进入**虚拟MFA**的动态口令页面。动态口令每30秒更新一次。

> 注意：
>
> - 如果您的账号为多人共享使用，当您成功绑定 MFA 之后，其他未绑定 MFA 的人将无法登录此账号。解决方法是让其他人也安装 MFA 应用程序并扫描此页的二维码，或者保存此二维码图片供其他人后续进行扫码。
> - 除在用户中心-安全信息-虚拟MFA中主动绑定虚拟MFA外，在未绑定虚拟MFA的情况下，使用虚拟MFA二次验证时，电脑

> 端同样会出现虚拟MFA的绑定二维码。

3. 输入连续的两组口令到电脑端页面，然后单击**确定**按钮，虚拟MFA设备绑定完成。

> 绑定虚拟MFA成功后，即可同时完成二次验证

解绑虚拟MFA

解绑虚拟MFA需要对已有绑定的虚拟MFA设备进行校验，如果你的手机误删了已安装虚拟MFA App，需要提交工单到后台解绑。

子用户操作

登录子用户控制台

登录子用户控制台之前，您需要为子用户设置密码，可以使用子用户名和密码登录；或为子用户绑定百度账号，绑定后使用该百度账号及密码登录。

子用户登录的两种方式，详细描述参见设置子用户密码：

1. 方式一：使用设置的用户名+设置的密码，进行登录操作。（推荐）
2. 方式二：绑定百度账号，使用百度账账号+对应的账号密码，进行登录操作。

登录操作

1. 进入**多用户访问控制** > **用户管理** > **子用户** 用户列表页，在列表上方，点击“子用户用户登录链接”，或者点击复制按钮，粘贴到浏览器进行访问。

2. 默认使用子用户用户名和密码，可以登录百度智能云控制台，如果该子用户绑定过百度账号或是百度员工账号，则可以点击**百度账号登录**，或是**百度员工登录**跳转登录。

修改子用户信息

1. 子用户登录成功后，进入子用户的管理控制台，点击头像进入用户中心。
2. 在**安全设置**栏，可以进行手机绑定、邮箱绑定、登录密码修改等操作

> 注意：子用户手机号、邮箱等信息，仅支持子用户登录后自行修改，主账号无法直接进行修改

用户组管理

概述

用户组是用户的集合，通常将具有相同职责的用户添加到一起，对应于企业内的研发小组、运维小组等，进行用户分类和权限区分，从而提高用户管理的效率。

常见场景

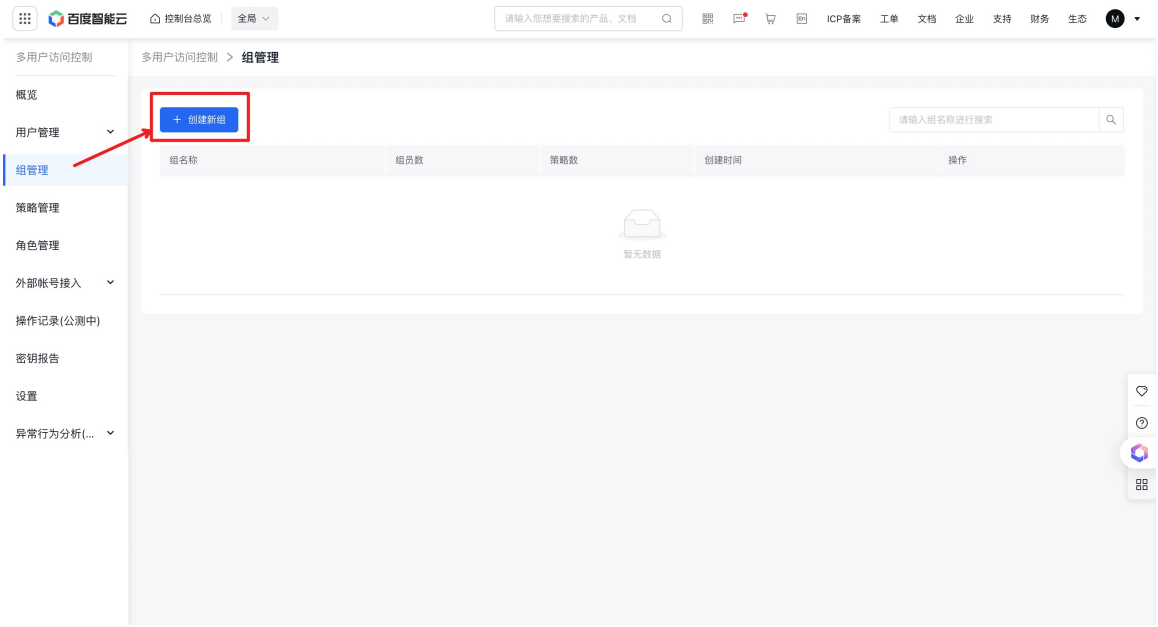
- 用户组内成员子用户发生职责变动，无需该子用户的权限进行操作，只需将该子用户从用户组中移除即可。
- 用户组所在团队的职责发生变化，无需为组内的成员子用户逐个编辑权限，只需对用户组本身进行权限策略的调整即可。

先决条件

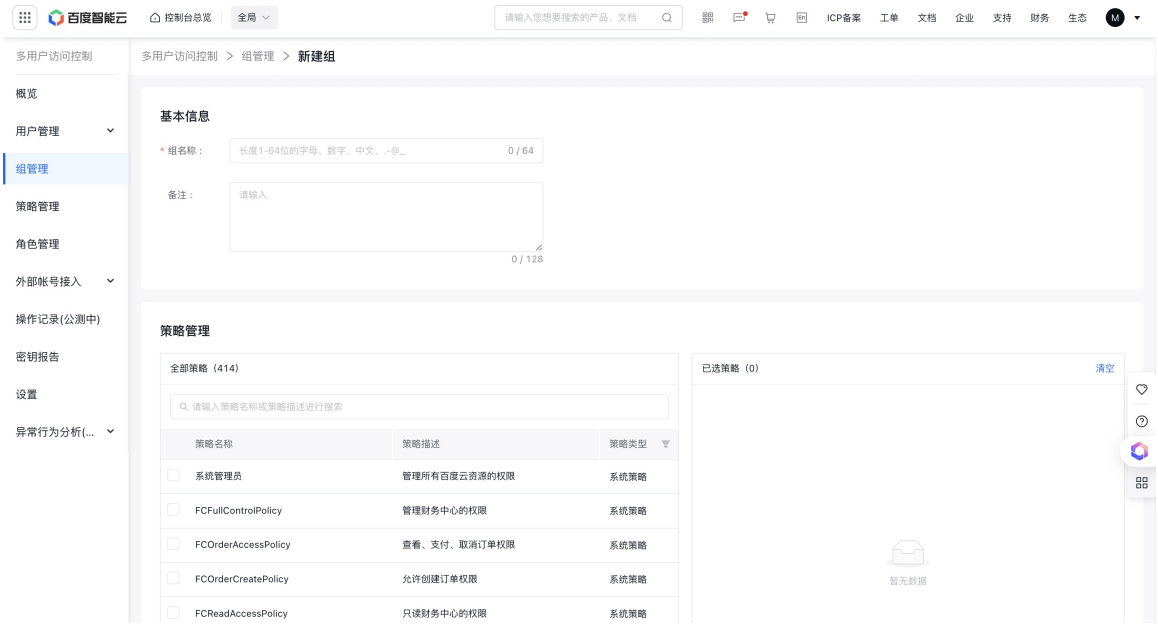
在使用用户组功能之前，你需要是百度智能云的主账号或者是拥有系统管理员权限的子用户，或是被授权了IAMFullControlAccessPolicy权限的子用户。

用户组创建与授权

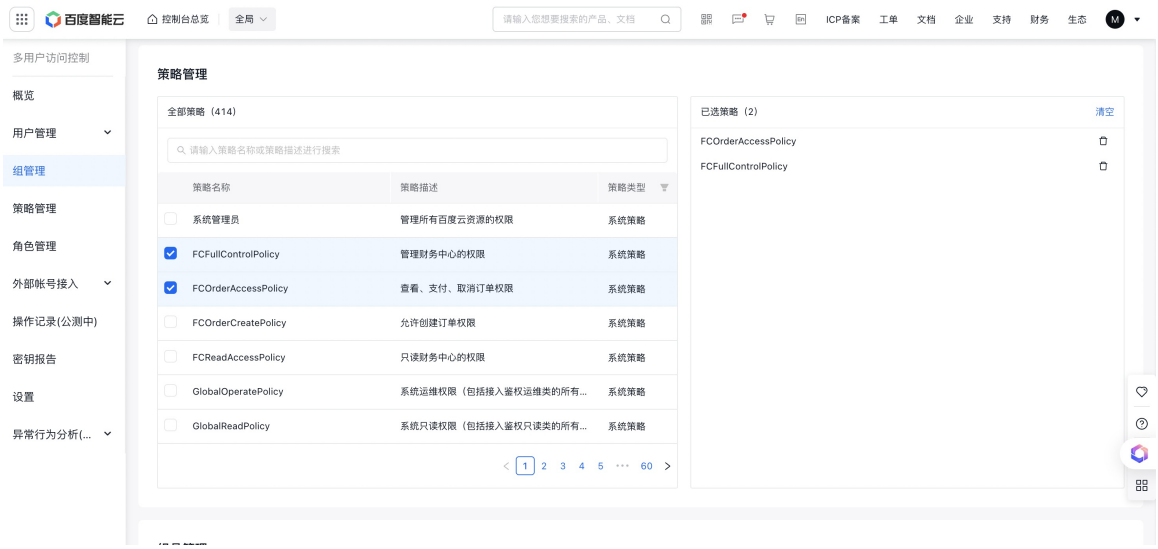
1. 登录百度智能云管理控制台,选择**多用户访问控制 > 组管理**, 点击**创建新组**。



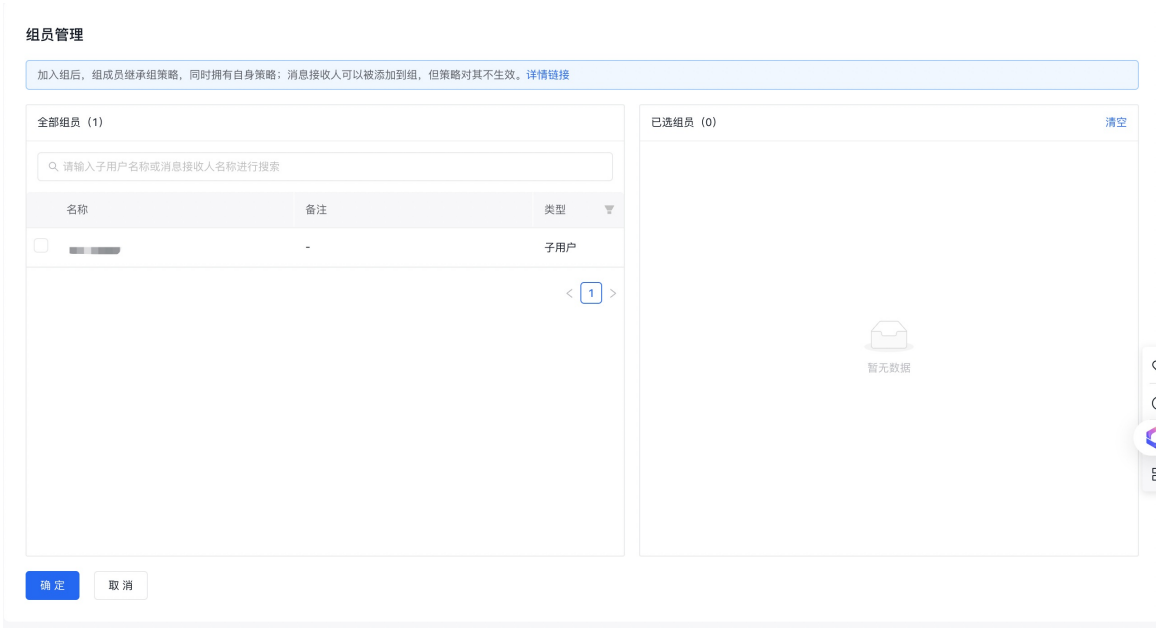
2. 填写待新建组的基本信息，需要填写组名称，允许英文字母、数字、特殊符号支持".-@_"，也可以填写备注。



3. 在策略管理中，为新建组添加权限策略，限制该组的访问权限。



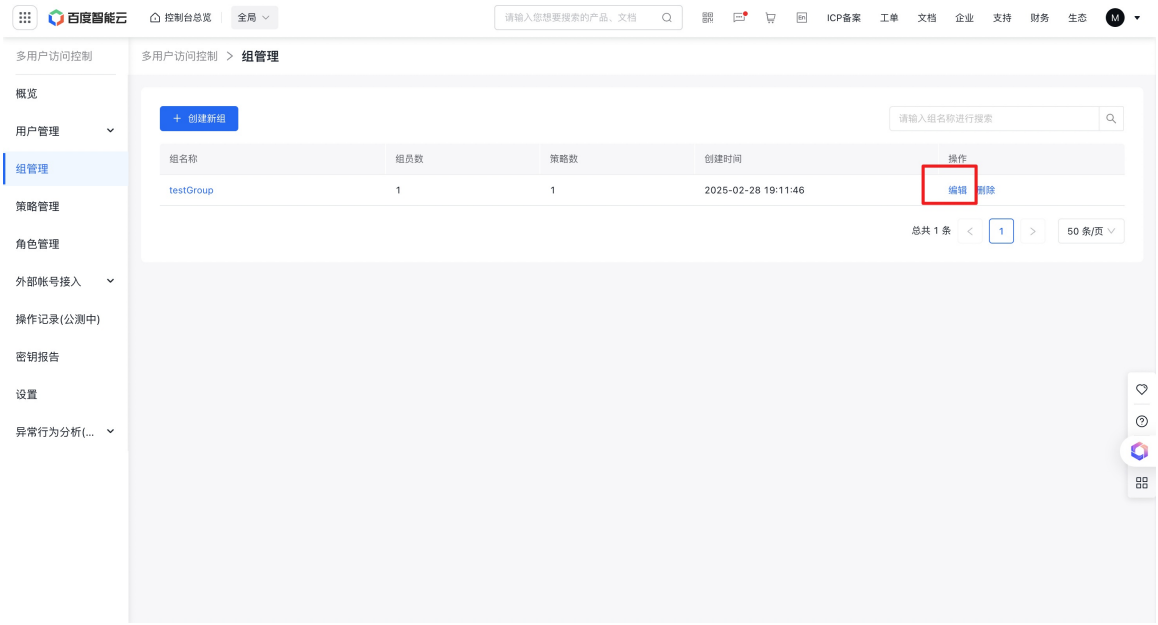
4. 在组员管理中，为新建组添加组员，组员继承组策略，同时拥有自身策略。可以为用户组添加子用户和消息接收人，消息jie shoujieshou



5. 点击**确认**，完成用户的新建。

为用户组修改策略

为已经建立好的用户组修改策略。点击**编辑**，在策略管理中，为组成员修改策略。



为用户组修改成员

为已经建立好的用户组修改成员。点击**编辑**，在组员管理中，添加或者减少组成员。

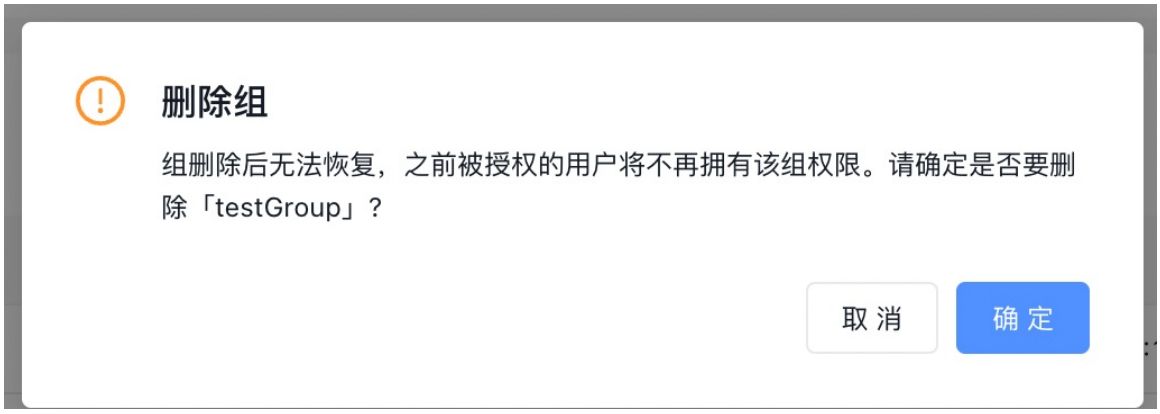
查看用户组信息

点击相应的**组名称**，进入所创建的用户组，可以查看用户组详细信息，例如用户组策略、用户组成员等。

删除用户组

点击**删除**，进行确认后，即可删除已经创建的用户组。

> 注意：组删除后无法恢复，之前被授权的子用户将不再拥有该组权限，可能线上业务中断，因此，在删除用户组之前，请务必确认组策略已经未被使用。



角色管理

概述

IAM角色(后文统一称角色)是一种虚拟身份，同用户身份一样，可以关联权限对资源进行操作，但它没有确定的身份认证密钥，且需要被一个受信的实体用户扮演才能正常使用。

你可以使用角色作为桥梁，向需要访问你云账户资源的用户、应用程序或服务提供访问权限。例如，你可以向其他云账户的用户、应用程序或服务授权临时访问你的账户资源，实现跨账户的资源访问，或是，向本账户内的子用户临时授权访问某些敏感资源。

本文将介绍角色的概念、工作原理，以及如何使用角色满足日常的账户和资源管理等需求。

相关概念

角色

可在百度智能云中账户中创建的拥有特定权限的IAM身份。角色与子用户类似，都是一种百度智能云身份，可以被授予**允许**或**拒绝**访问特定资源的权限。区别在于，角色是一种虚拟身份，没有确定的身份凭证(密码或密钥)，无法直接登录控制台或直接使用API访问你在百度智能云的资源。只有当角色被某一可信任的实体身份所代入时，通过提供角色会话的临时安全凭证，从而访问被授权的资源。

角色可以被如下类型的用户代入：

- 与角色在同一账户下的IAM用户
- 与角色不在同一账户下的IAM用户
- 由百度智能云提供的Web服务或产品
- 由与SAML2.0 兼容的身份提供商提供的外部用户

角色载体

指可以承担角色权限的对象。在信任策略中定义添加和管理特定角色的角色载体，以**允许**或**拒绝**这些角色载体访问你在百度智能云中的资源。目前支持成为角色载体的对象可以是主子用户、角色或用户组。

切换（代入）角色

指的是角色载体从自身用户空间切换到被授权的角色空间的操作。一个用户一旦被授予 *STSAssumeRoleAccess* 的权限，即可在其自身的用户空间和角色空间中进行切换，但一次只能切换到唯一的角色空间，切换到目标角色空间后，该用户当前只拥有目标角色权限。当前仅支持通过API的方式切换角色，详细操作请参考使用角色。

常见场景

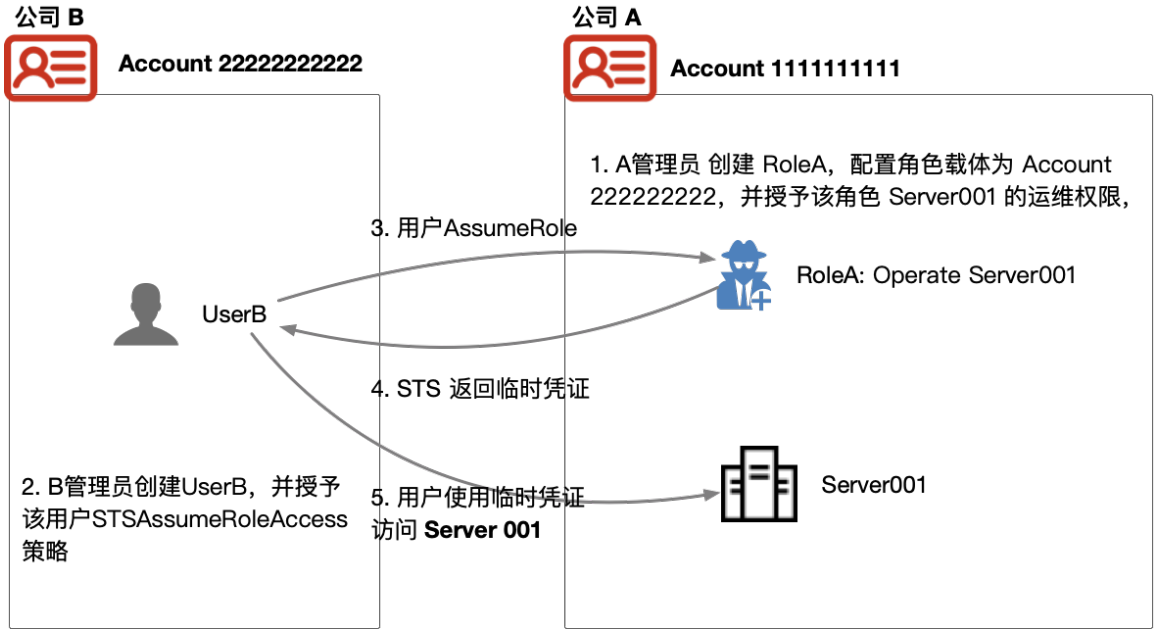
角色常用于解决跨账户的资源访问问题，或是同账户内子用户的临时授权问题，且不需要共享你账户内任何用户凭证信息，本节将介绍几种常见的业务场景。

给第三方账户授权访问你的云资源

你可以信任第三方账户，并为其账户的IAM用户或服务授予权限，以便其代入到你的云账户角色，并访问该角色被授权的云资源。你也可以切换到其他云账户给你授权的角色，以访问其他账户的云资源。

如下以一个实际案例来说明如何实现跨账户的资源访问：

公司A在百度智能云创建了账户Account111111111，并在该账户下创建了BCC服务器Server001，公司A的管理员希望将这台服务器交给公司B（拥有账户Account222222222）来运维。



在这个场景下，要满足公司A的管理需求，需要进行如下步骤：

1. 公司B提供其云账户ID或别名给到公司A的管理员；
2. 公司A的管理员在Account111111111中创建新角色RoleA，并将角色载体设置为**其他云账号**，并输入公司B的云账户ID或别名；
3. 公司A将Server001的运维策略授予RoleA；
4. 公司B的管理员创建子用户UserB，并授予该用户 _STSAssumeRoleAccess_ 策略；
5. UserB 可以通过调用临时安全凭证服务(STS)_AssumeRole_ API 切换到RoleA，即具备了运维Server001 的权限。

同一账户内的临时授权

你可以向IAM子用户授予权限，以便切换至你所在账户中的角色，或是信任你的第三方账户的角色。

如下以一个实例来说明如何使用角色来解决同账户内提升权限的问题：

假如在你的账户内有一台核心的BCC服务器实例，出于安全考虑，你不希望你的子用户通过控制台直接管理这台服务器（如删除操作），此时，你可以将这台服务器的管理权限授予给一个角色（详细操作参考创建角色），并授予特定的子用户使用角色的权限，详细操作请参考使用角色，这样可以对该服务器实例添加以下几层保护：

- 你必须显式地向子用户授予能够担任该角色的权限
- 当前子用户必须使用百度智能云的API代入角色

使用如上方式践行了最小权限原则 ，且只有用户需要执行特定任务时，才能使用安全级别较高的权限，以防止子用户对敏感的环境或服务进行了意外操作或更改。

创建角色

你可以为IAM角色授予对百度智能云资源的访问权限。利用IAM角色，你可以选择为自己账户下的子用户授予特定的访问权限，或与其他云账户建立信任关系，从而允许其他云账户下的子用户，利用你创建的IAM角色来访问你的云资源，比如，你可以允许第三方云账户访问你的对象存储BOS存储桶中的数据。

信任关系建立后，被信任账户的用户、服务或应用程序可以使用百度智能云的临时安全凭证(STS) 服务的 _AssumeRole_ API 来操作代入角色。此操作为被信任用户或服务提供临时安全凭证，以访问你的云资源。
有关IAM角色的相关概念，请参考角色相关概念。

本节将介绍如何使用控制台创建角色。

先决条件

在使用控制台创建角色前，你需要：

1. 拥有已激活的百度智能云账户，如何注册和激活云账户请参考注册；
2. 拥有该账户的系统管理员权限。

操作步骤

请参考如下步骤完成在控制台中创建你的角色：

1. 登录百度智能云控制台，鼠标移动到右上角用户头像，选择**多用户访问控制**；
2. 在左侧导航栏中选择**角色管理**，点击按钮**创建新角色**；
3. 填写角色名称、描述等基本信息，注意，角色名称在同一账户下必须唯一，名称不区分大小写，即**TESTROLE** 和 **testrole** 视为同一个角色；
4. 选择**角色载体**，你可以选择**当前云账号**，代表该角色为目前所在云账户创建，也仅支持当前云账户下的子用户或服务进行访问；如果你选择**其他云账号**，代表该角色为其他云账户所创建，此时需要填写其他云账户的**账户ID** (你可以在**用户中心 > 用户ID**找到该ID)，一个角色最多同时信任10个其他云账户；
5. 为角色授权。在**策略管理**中选择合适的系统策略或自定义策略授予该角色，如果已有策略都不符合要求，可以选择新增自定义策略，请参考权限策略。你也可以不选择之后再为该角色授权；
6. 点击 **完成**。

通过如上步骤，完成了百度智能云账户下一个全新角色的创建。

> **注意** 完成上述步骤仅算完成了所需配置的前半部分。你还必须为被信任账户中的各个用户授予 _STSAssumeRoleAccess_ 的策略权限。详细操作步骤请参考使用角色。

使用角色

通过创建角色操作，你拥有了一个被授予特定权限的角色，且该角色可以被某个云账户使用。本节将以被信任账户的角度，介绍如何作为被信任的用户，使用角色访问信任账户的云资源。为方便理解，这里假设云资源及角色所在账户ID为111111111，角色名称RoleA，拥有BCC服务器Server001的运维权限，被信任云账户为 222222222，希望为子用户UserB授予代入角色RoleA的权限。

先决条件

1. 拥有被信任账户222222222的`系统管理员`权限;
2. 子用户UserB拥有有效的获取AKSK;
3. 子用户UserB拥有目标账户的账户ID和角色名称。

操作步骤

控制台代入角色

1. 使用账户2222222管理员账户登录百度智能云控制台;
2. 为子用户UserB授 `STSAssumeRoleAccess`策略权限, 具体操作请参考用户授权;
3. 子用户UserB登录控制台,鼠标移动到页面右上角头像处,点击按钮**切换身份**跳转页面;
4. 在跳转页面中输入目标账户的账户ID、角色名称后,点击切换后进入目标角色空间;

5. 如需返回子用户UserB所在账户,需要将鼠标移动到页面右上角头像处,点击按钮**返回UserB**。

使用API代入角色

1. 使用账户222222222的管理员账户登录百度智能云控制台；
2. 为子用户UserB授予`STSAssumeRoleAccess`策略权限，具体操作请参考用户授权；
3. 子用户UserB通过AssumeRole API，替换参数`accountID`为`111111111`，`roleName`为RoleA，即可以代入账户111111111的角色RoleA。默认情况下，角色会话会持续2小时，你也可以在使用`AssumeRole`API时设定参数`durationSeconds`具体有效时间，这个时间不能超过2小时；
4. 此时，子用户UserB将得到角色RoleA的临时凭证，并暂时放弃其在账户222222222的权限，具备了运维服务器Server001的权限。在访问Server001时，需要将AK/SK 替换为`AssumeRole`API返回的临时`AK/SK`，并将Token更换为返回的`SessionToken`。

管理角色

本节将介绍如何管理你的百度智能云账户中的角色，涉及到角色编辑和修改，角色的删除操作等。

先决条件

需要管理你的角色，你需要拥有当前云账户的系统管理员权限

编辑角色

在角色使用过程中，你可能需要对现有角色信息进行修改，请参照如下步骤

1. 登录百度智能云控制台，鼠标移动到右上角用户头像，选择**多用户访问控制**；
2. 在左侧导航栏中选择**角色管理**，在角色列表中找到需要编辑角色项，点击按钮**编辑**；
3. 你可以编辑角色的基本信息、角色载体和角色所关联的策略。

删除角色

为保障你的云资源安全，对于不再使用的角色，请及时进行清理，按照如下步骤可以完成角色的删除：

1. 登录百度智能云控制台，鼠标移动到右上角用户头像，选择**多用户访问控制**；
2. 在左侧导航栏中选择**角色管理**，在角色列表中找到需要编辑角色项，点击**删除**；

3. 阅读弹窗提示后，点击**确认**，即完成角色的删除。

> **注意** 为避免线上使用的角色被用户误删除，系统层面限制了已关联权限策略的角色不能被直接删除，需要进入角色详情页，清空关联的策略后，才能进行删除操作。 另一方面为避免影响你的线上服务，在删除角色前，请检查该角色最近的活动状态，确保无线上用户或服务使用时，再进行删除操作。

常见问题

问：什么是IAM角色？

IAM用户是一种实体身份，有确定的身份凭证，它通常与某个确定的人或应用程序一一对应。IAM角色是一种虚拟身份，它关联一组权限，没有确定的身份凭证，必须关联到某个实体身份上才能使用。IAM角色与特定的用户或组没有关联。相反，可信实体可以担任角色，例如IAM用户、应用程序或BCC等服务。

问：IAM角色能帮助解决什么问题？

IAM角色使您能够通过已定义的权限来委派可信任实体的访问权限，而无需共享长期访问密钥。使用IAM角色，可以将访问权限委派给您账户内管理的IAM用户、其他账户下的IAM用户，以及BCC等服务。

问：如何担任IAM角色？

您可通过调用Security Token Service (STS) AssumeRole API（或者叫 AssumeRole、AssumeRoleWithWebIdentity 和 AssumeRoleWithSAML）担任IAM角色。这些API会返回一组临时安全凭证，应用程序之后可使用这些凭证来签署对服务API的请求。

问：可以担任多少个IAM角色？

您能担任的IAM角色的数量没有限制，但是当您向百度云服务提交请求时，只能用一个IAM角色进行操作。

问：IAM角色和IAM用户之间有什么区别？

IAM用户拥有永久性凭证，用来直接与云服务交互。IAM角色没有任何凭证，不能直接提出服务请求。IAM角色必须由获得授权的实体担任，如IAM用户、应用程序或bcc等服务。

问：是否支持向IAM组添加IAM角色？

目前不提供。

问：我可以创建多少个IAM角色？

您的账户最多只能创建100个IAM角色。

问：是否可以删除与服务关联的角色？ 可以。删除该角色之前，您必须解除该角色的授权。此步骤可以确保您不会意外删除您的正常运行所需的角色。

权限策略

权限策略概述

百度智能云IAM通过定义权限策略，将策略附加到用户，为用户提供灵活的权限管理和控制。管理员用户可以通过预定义的系统策略，直接为子用户进行授权，同时，也可以根据实际的业务需要，自定义子用户权限策略，为各个子用户授予不同的资源访问权限。本节将着重介绍IAM权限与策略的使用方法和可解决的问题。

策略类型

IAM多用户访问控制加上企业组织可将策略类型分为2类。

基于身份的策略

即将包含权限说明的策略附加到IAM身份上，比如给子用户、用户组或是角色授予的策略，这一类型的策略进一步可以分为以下2种类型：

- 系统策略
  系统策略为百度智能云预设策略，为用户常用权限集合，例如职能型的权限如系统管理员、财务人员，资源型的产品级的只读、运维权限等，通常系统策略权限颗粒度较粗，且不可被任何用户编辑和删除。

- 自定义策略
  由管理员用户创建的策略统称为自定义策略。用户可以根据实际的业务需要，配置特定产品服务、资源的操作权限，例如，可针对云服务器BCC的某个实例设置一条策略，将该策略关联给子用户，使其拥有BCC某个实例的操作权限。相比系统策略，自定义策略可控制的权限颗粒度更细，其更为灵活。
  自定义策略目前已支持同一条策略内配置多种服务权限、跨地域的资源实例，且支持无资源实例的操作权限，如创建/添加、页面/按钮的功能等。

服务控制策略
服务控制策略(SCP)属于IAM的扩展产品企业组织，它不直接控制用户的权限，而是用于控制其所附加的组织单元或是子账户的最大权限边界。有关企业组织和SCP的详细介绍，请参考企业组织。

管理IAM策略

查看系统策略

登录云控制台，鼠标移到右上角头像处，进入**多用户访问控制>策略管理**，默认展示系统策略列表，系统策略通常包含职能型策略如系统管理员，支持系统的所有策略，财务管理等；产品型策略，定义了分产品的通用策略合集，具体策略定义，请参考各产品线的权限说明文档。你可以点击**查看**按钮，查看系统策略的策略语法。

管理自定义策略

在**策略管理 > 自定义策略**中，你可以为你的账户定义业务相关的自定义策略，以实现细粒度地权限管控。你也可以基于标签为相同类型的资源（加了相同标签的资源）快速构建一条自定义策略，常适用于解决大量不同产品或服务组合授权的问题。

创建自定义策略 目前IAM支持2种方式创建自定义策略：按策略生成器创建和按标签创建。按策略生成器创建的方式，通过选择服务、权限以及区域下实例，以生成策略；按标签创建，根据你为服务实例创建的标签筛选资源，生成策略。

先决条件
拥有百度智能云系统管理员权限。

操作步骤：按策略生成器创建

1. 登录云控制台，鼠标移到右上角头像处，进入**多用户访问控制>策略管理**;
2. 点击**创建策略**，弹窗中选择**按策略生成器创建**；
3. 填写基本信息中的策略名称和描述；
4. 填写权限配置： 点击**添加权限**按钮，为当前策略添加一条权限，在弹窗中填写：
   - **选择服务**：即需要选择的产品名称。
   - **策略生成方式**：根据所选服务类型，支持使用策略生成器或编辑策略文件的方式生成最终策略，置灰则为默认为策略生成器方式。策略生成器为可视化的策略生成工具，通过按步骤地配置操作和资源实例，来最终生成策略；编辑策略文件，支持用户按照一定的策略语法，编辑JSON格式的策略文档，最终生成策略，详细请参考策略语法. 2种方式生成的最终策略，都会以ACL的格式存储到系统中。
   - **权限效力**：允许或拒绝，通常为选择允许，需要注意的是，拒绝权限效力强于允许，需要谨慎选择；
   - **权限选项**：由所选服务类型定义的业务相关的权限，支持多选；
   - **选择资源**：所选服务类型下的可选资源，可精细到资源实例。

- 选择**所有资源**，代表所有百度智能云支持的地域下的任意资源，包含未来添加的资源，在策略ACL描述中为“*”；
- 选择**特定资源**，即可通过筛选不同地域，选择特定的资源实例。

- **限制条件**：选择你需要为当前策略配置限制条件，配置完限制条件表示，只有符合该条件且满足权限策略的访问才会被允许。

5. 点击**完成**，回到创建策略页面，你可以继续按照第4步的操作步骤，进一步添加权限，也可以点击**完成**保存自定义策略。

操作步骤：按标签创建

1. 登录云控制台，鼠标移到右上角头像处，进入**多用户访问控制>策略管理**;
2. 点击**创建策略**，弹窗中选择**按标签创建**；
3. 填写基本信息中的策略名称和描述；
4. 填写权限配置：
   - **选择标签**：选择你需要的标签键值对，如果没有标签，可以选择点击**还没有标签？点击创建标签**链接跳转到标签管理；
   - **选择服务**：选择已支持标签的服务类型，查看哪些产品线已支持基于标签的授权，具体可参考使用IAM的产品；
   - **选择操作**：已选服务的权限操作，统一为只读、运维和管理权限；
   - **资源范围**：展示已选择服务的资源列表，如未命中任何实际资源，则代表全局所有资源，在未来的时间范围内，如果你将当前标签关联到实际的资源实例，该资源实例将受到当前自定义策略的控制。
5. 点击**完成**，保存已配置策略。

编辑自定义策略 在一些某些场景下，已有的自定义策略无法满足你的需求时，你可以创建一条全新的自定义策略，也可以编辑已有的自定义策略。登录云控制台，鼠标移到右上角头像处，进入**多用户访问控制>策略管理**，定位到需要编辑自定义策略，点击**编辑**按钮，即进入策略的编辑页面，策略编辑操作可直接参考**创建自定义策略**。

删除自定义策略 对于不再需要使用的自定义策略，你可以定位到需要被删除的自定义策略，点击**删除**按钮，并确认以完成策略删除操作。

> **重要提示:**
> 删除线上策略将有可能导致你的IAM用户或服务失去相应地操作权限，影响你的业务使用，所以在策略删除前，请务必确认当前策略已从所有的身份上移除.

授权

只有将策略授予到具体的用户身份时，权限控制才会生效。目前IAM支持对3种类型的身份进行授权操作。

给子用户授权

你可以将系统策略或是自定义策略直接授予子用户。在子用户管理列表中，点击**添加权限**，为子用户进行授权。子用户的权限受直接被赋予的策略控制，同时也会继承来自组的权限策略。

给用户组授权

推荐你将你的用户加入的用户组中，并给用户组授予权限策略。在组管理列表中，点击**编辑**，为用户组进行授权。

给角色授权

在角色管理中，定位到指定角色，点击**编辑**，在**权限信息**模块中，点击**授权**，为指定角色进行授权。

策略鉴权评估逻辑

概述

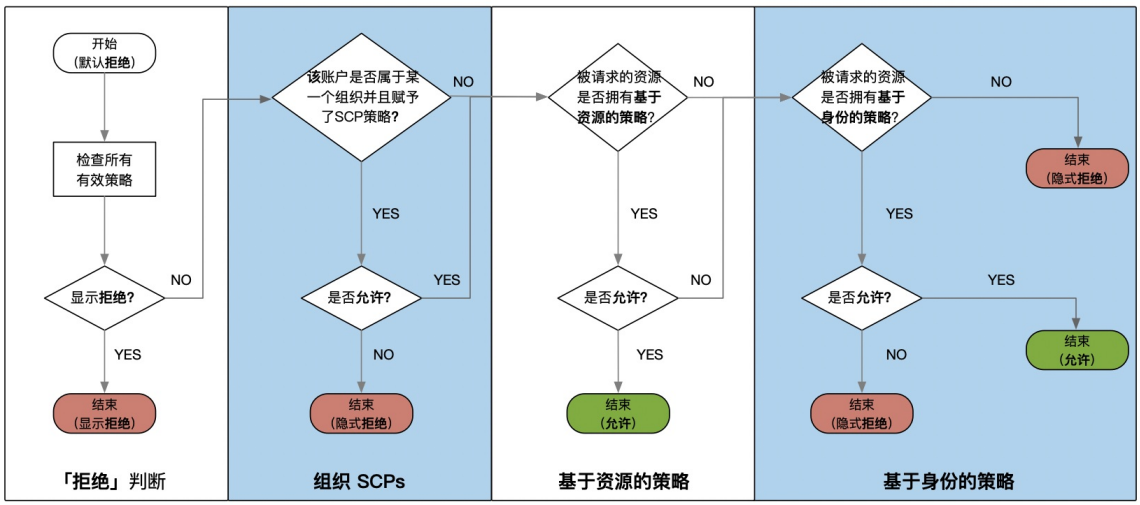
百度智能云IAM服务无缝集成了大量云服务，并为其提供资源的访问控制。当用户访问云服务时，云服务会将访问行为转化为鉴权上下文传递给IAM，IAM会根据用户拥有的权限策略进行权限鉴定，以返回成功或是失败。

IAM的权限策略类型包括企业组织的服务控制策略(SCP)、基于资源的策略和基于身份的策略，详细请参考策略类型。用户是否具备访问某具体云资源的权限，取决于用户拥有的权限策略的综合判定结果，本文将介绍策略鉴权的评估逻辑。

策略评估逻辑

百度智能云的策略评估逻辑遵循如下原则：

- 默认情况下，所有请求被隐式**拒绝**（仅主账户拥有所有权限）；
- 基于身份或基于资源的策略中的显式**允许**将覆盖隐式拒绝；
- 组织SCP的隐式**拒绝**会覆盖**允许**，即企业组织下的成员账户如果需要有某服务的权限，必须所在组织单元以及自身都被授予该服务权限；
- 任何策略中的显示**拒绝**会覆盖**允许**；



基于标签授权与鉴权

概述

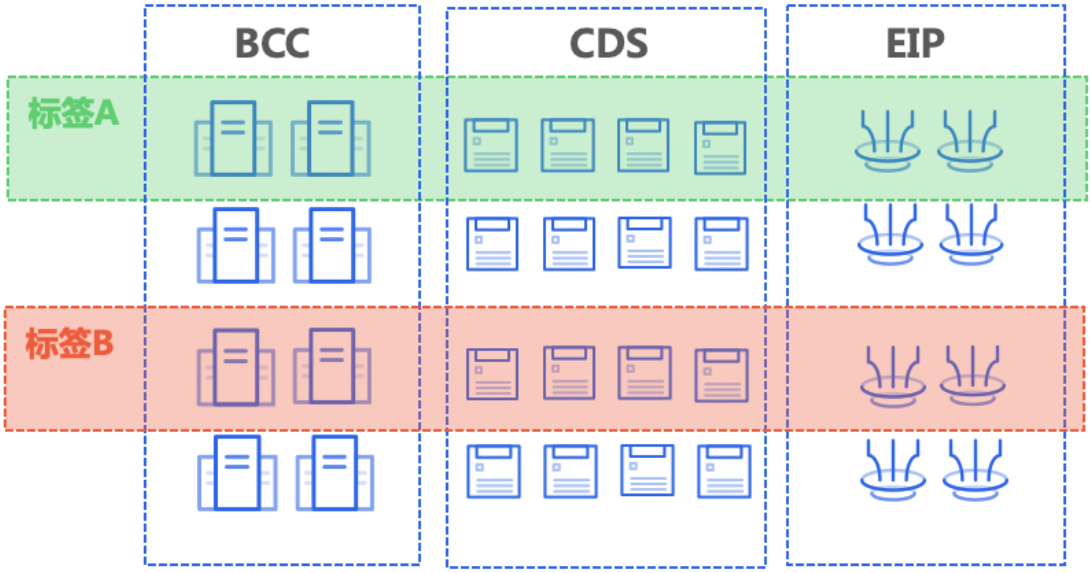
百度智能云提供了公共的标签管理服务，便于为云上的资源做分组管理，你可以为你的账户内资源规划好标签，对需要关联的资源或是同一项目内的资源加上相同的标签，以提升资源管理的效率。

对于已经加上标签的资源，你可以使用IAM的基于标签授权，快速为具备关联关系的资源配置自定义策略，并授予给需要该标签下资源权限的用户，当前已支持标签授权的服务请参考目前已支持的产品线

基于标签的权限特性

基于标签授权和鉴权可以为你提供如下权限管理特性，以帮助提升你的管理效率：

- 分组授权：具备关联关系的不同服务实例的快速授权；
- 在服务不具备“添加或创建”等无中生有创建资源的权限情况下，实现无中生有的操作并使被授权用户自动获得其所创建资源的权限；



常见的使用场景

场景一. 某大型企业客户不同服务资源的分组管理

某大型企业客户在新建项目的配置中，需要同时为60台BCC、120块CDS磁盘进行权限规划，在使用标签授权管理之前，只能通过手动维护BCC和CDS的磁盘关系，手动为多个用户分配资源权限，极其影响效率。使用标签授权后，用户只需在创建BCC、CDS时，为关联的资源加上相同的标签，通过标签授权的方式，快速为用户分配权限，同时，对于未来此项目中新增的BCC、CDS资源，只要加上相同的标签，其子用户可以自动获得新开机器的权限，极大提升了管理效率。

场景二. 某大型企业客户期望其子用户可以创建CDN域名并自动获得其所创建域名权限

某大型企业客户管理员，期望将域名管理的权限分配到子用户账号手中，详细需求如下：

1. 子用户能够自行添加域名，并自动拥有其添加的域名权限；
2. 子用户之间的域名资源相互隔离，如子用户A在不被授权的情况下，无法看到子用户B添加的域名；
3. 主用户可以查看和管理所有子用户添加的域名；

标签授权可以满足如上的产品需求，具体的操作方式如下：

Step1. **管理员用户**规划标签，比如给**子用户A**规划标签Key: department/Value: 123，给用户B规划标签Key: department/Value: 456，并在标签管理中建立对应标签;

Step2. **管理员用户**进入**IAM多用户访问控制 > 策略管理**，点击**创建策略**，选择创建策略的方式为**基于标签创建**；

Step3. **管理员用户**填写策略的基本信息，如命名策略为policy_for_user_A_with_tag123, 在权限配置内选择标签Key: department/Value: 123，选择服务 “内容分发网络 CDN”，选择操作**管理权限**，资源范围默认展示拥有该标签属性的所有资源，点击完成保存；

Step4. **管理员用户**按照Step3流程创建基于tag456的策略policy_for_user_B_with_tag456;

Step5. **管理员用户**创建**子用户A**，如UserA，并为UserA授予策略 policy_for_user_A_with_tag123的权限；**管理员用户**创建**子用户B**，如UserB，并为UserB授予策略 policy_for_user_B_with_tag456的权限；

Step6. **子用户UserA**登录控制台，进入CDN，选择域名管理 > 新添加域名，如添加域名为“cloud.baidu.com”, 在向导步骤3“标签”处，默认绑定标签Key: department/Value: 123，点击完成即完成域名创建，并且子用户UserA 拥有了管理域名 cloud.baidu.com 权限，且子用户UserA 无其他域名任何权限，如需给子用户UserA其他域名的访问权限，需要**管理员用户**在IAM中额外授权。子用户UserB同理。

基于标签鉴权

本质上标签是权限策略的一种条件或属性，基于标签的策略属于用户自定义策略，且在权限类型上，隶属于基于资源的策略，用户被授予标签资源权限时的鉴权评估逻辑符合策略鉴权评估逻辑。

策略语法

ACL语法简介

背景

百度智能云支持基于用户角色的权限校验和基于资源的权限校验。对用户请求，会根据请求身份得到与用户角色关联的accessControlList，进行鉴权；还会根据用户请求的资源路径，得到与资源关联的ACL（Access Control List 访问控制列表），进行鉴权。

名词解释

Access Control List (ACL)：附加在用户角色或资源上的一个权限控制列表。用户对于资源的权限控制，可以是resource-based也可以是user-based。用户可以通过设置ACL，来对各个服务进行访问上的细粒度控制。ACL是一个列表，由一条或多条entry组成。

User-based ACL：附加于用户的权限控制列表。添加用户或组到特定的权限策略，在策略中指定此角色对不同资源的权限。请求到来时，通过用户及其所在组所附加的权限列表集合，遍历权限描述进行鉴权。

Resource-based ACL：附加于资源的权限控制列表。对特定level的资源（如BOS的bucket级别），在创建资源的同时指定其适用的策略，并在策略描述中指定对不同用户的权限。用户请求资源时，得到资源上所关联的权限策略，遍历策略中的权限描述进行鉴权。

IAM ACL语法

IAM ACL使用JSON格式的策略描述语言基于user或resource进行细粒度的访问控制。命名方法使用首字母小写的驼峰命名法。

字段总览

字段名称	是否必选	字段含义
Id	可选	本ACL的标识符，用来标识本ACL的id或对ACL进行描述
accessControlList	必选	ACL的主元素，标识ACL主体的开始，由一组statement组成
eid	可选	标识子acl entry的id，或者对该entry进行描述
service	必选	本条entry影响的服务组件，"*"表示所有服务
region	必选	本entry影响的区域，"_"表示所有区域（对全局服务，强制为"_"）
effect	必选	指定能够与本条entry匹配上的request，是否允许其继续执行
grantee	可选	指明本entry的受影响人，只适用于resource ACL且必选
permission	必选	指明本entry所影响的权限
resource	必选	指明本entry所影响的资源
condition	可选	指明策略生效的条件，该字段下存在子字段

- Grantee子字段：

grantee子字段	字段含义	示例
id	授权（或禁止）的accountid	b124deeaf6f641c9ac27700b41a350a8
user	授权（或禁止）的用户	bob
group	授权（或禁止）的组	developers
saml-provider	上传的IDP名称	developers

- Condition子字段：

Condition子字段	字段含义
ipAddress	本statement生效的ip地址范围
Time	本statement生效的时间范围
Referer	本statement生效的referer

ACL与entry之间的关系:

- 每个resource具有一条独立的ACL
- 每条ACL由一条或多条entry组成

entry之间的关系:

- entry之间是相互独立的关系
- 相互独立的含义即：每个entry相互独立，entry逻辑判断与在ACL中的位置无关

综合示例

```
{
  "id": "id or description",
  "accessControlList": [
    {
      "eid": "eid or description",
      "service": "bce:bos",
      "region": "bj",
      "effect": "Allow",
      "permission": [
        "CreateBucket",
        "READ"
      ],
      "resource": [
        "bucketname/objectname"
      ],
      "grantee": [
        {
          "id": "accountid",
          "user": "bob",
          "saml-provider":"saml.xml"
        }
      ],
      "condition": {
        "ipAddress": [
          "192.168.0.0/16",
          "192.169.0.0/16"
        ],
        "time": {
        "in": [
            {
              "greaterThan":"2010-06-01T23:00:00Z",
              "lessThan":"2010-07-01T23:00:00Z "
            },
            {
              "greaterThan":"2010-08-01T23:00:00Z "
            }
          ]
        },
        "referer": {
          "stringEquals": [
            "www.abc.com",
            "www.example.com"
          ],
          "stringLike": [
            "www.baidu.com/*"
          ]
        }
      }
    }
  ]
}
```

外部账号接入

联合登录概览

概述

SAML（Security Assertion Markup Language）即安全标记语言，它是一个基于XML的通信协议，适用于在不同的安全域(Security Domain)之间交换认证和授权数据，以实现系统间账号的单点登录。 SAML协议的对象主要由身份提供方(IdP，Identity Provider), 服务提供方(SP, Service Provider)共同约束。

百度智能云支持基于SAML 2.0协议的单点登录。企业客户可以将支持SAML协议的账号系统作为IdP，将百度智能云作为SP，来打通企业已有账号系统与百度智能云的账号系统，实现单点登录（Single-sign on, 简称SSO）。

工作原理

百度智能云基于SAML 2.0协议提供了企业账号系统(IdP)直接接入，以实现单点登录的能力，其主要的工作原理如图所示：

1. 企业员工或其他用户通过客户端(通常是浏览器)登录企业IdP；
2. 企业IdP通过企业的身份存储目录认证用户身份；
3. 企业IdP将登录用户的信息以SAML断言的形式返回客户端；
4. 客户端将IdP返回的SAML断言传递到配置的百度智能云(SP)登录URL;
5. 百度智能云SSO节点根据SAML配置验证用户身份并到临时身份凭证服务STS获取身份凭证;
6. 返回客户端验证成功消息、身份凭证和回调地址；
7. 客户端跳转到百度智能云首页，完成单点登录SSO。

联合认证模式

百度智能云当前提供了2种基于SAML 2.0的联合认证方式：**IAM用户联合**和**IAM角色联合**。

- **IAM用户联合**：企业员工(客户)从外部身份源认证通过后，以IAM子用户身份访问云资源；
- **IAM角色联合**：企业员工(客户)从外部身份源认证通过后，以IAM角色身份访问云资源。

2种联合认证模式的区别如下：

区别点	IAM用户联合	IAM角色联合
以何种身份访问云资源	IAM子用户	IAM角色
映射关系	通常为一对一	通常为多对多，即企业内多个员工可共享同一个角色
可访问服务范围	支持主子用户的服务	支持STS的服务，详细请见目前支持的产品线
创建身份实体	需要为每个SSO的员工创建IAM子用户	仅需要创建有限的IAM角色
登录控制台	IAM子用户支持独立登录智能云控制台	IAM角色不支持独立登录控制台，需要由可信任的外部账号扮演

适用场景

不同的联合认证模式通常根据企业业务的实际需求进行选择，本节将介绍如何根据企业需求选择联合登录方式。

IAM用户联合

- 您希望将企业IdP的用户同步到百度智能云，并建立一一对应关系，以确保访问的可问责性；
- 部分您需要访问的服务，尚不支持通过角色访问(使用STS服务)；
- 您的企业IdP不支持较为复杂的属性配置；

IAM角色联合

- 您不希望为每个员工在百度智能云都同步创建子用户，以降低管理成本；
- 您希望在使用联合登录的同时，仍然可以使用IAM子用户的管理功能；
- 您希望根据员工在企业IdP中的某些属性，来区分云上拥有的权限。当进行权限调整时，只需要在本地进行属性的更改；
- 您拥有多个百度智能云账号，但使用统一的企业IdP，希望在企业IdP配置一次，就可以实现到多个百度智能云账号的联合登录；
- 您的企业内或是与合作伙伴间存在多个IdP，都需要访问同一个百度智能云账号，您需要在一个百度智能云内配置多个IdP进行联合登录；
- 除了控制台，您也希望使用API方式来进行联合登录。

IAM用户联合

配置IAM用户联合登录

你可以通过配置企业端IdP的SAML节点和百度智能云SP的 **外部账户接入** > **IAM用户联合** ，从而实现企业用户从企业应用到百度智能云IAM子用户的单点登录。

先决条件

1. 企业IdP支持SAML 2.0协议；
2. 拥有百度智能云的账号并激活。

配置流程 配置基于SAML的单点登录功能，需要同时完成IdP配置和SP配置，其中IdP配置包含基本配置、用户属性配置和下载元数据等，SP配置包含创建身份服务提供商和配置信任策略。本文以微软公司的Azure Active Directory（AAD）为IdP，介绍如何配置SAML IdP和百度智能云的SP。

配置IdP

1. 按照流程，注册Azure 账号；
2. 登录Azure门户，在左侧导航栏中，进入 **所有服务** > **Azure Active Directory**；
3. 点击 **企业应用程序** > **新建应用程序** ，并选择 **非库应用程序** ，填写应用名称，点击 **添加** 完成应用程序创建；

> 注意：非库应用程序要求开通AAD专业版，你可以选择开通免费试用版，并在完成配置后选择是否需要关闭该试用版。

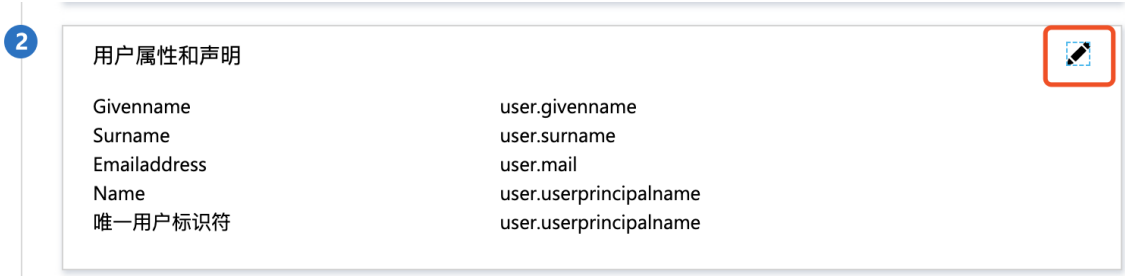
4. 进入应用程序，选择 **单一登录 > SAML** ；

5. 在基本SAML配置中，点击右上方编辑按钮，配置 **标识符(实体ID)** 字段为`urn:bce:baidu:webservices`或`https://login.bce.baidu.com/accountId/saml`，需要将accountId字段替换为百度智能云中的实际accountId(在百度智能云用户中心获取)；

> 注意：若希望使用从AAD侧发起登录 ，请配置 **标识符(实体ID)** 字段为`urn:bce:baidu:webservices`；若希望使用从百度智能云侧发起登录，请配置 **标识符(实体ID)** 字段为`https://login.bce.baidu.com/accountId/saml`。

6. 在基本SAML配置中，继续配置 **回复URL(断言使用者服务URL)** 字段为`https://login.bce.baidu.com/saml`；

7. 在用户属性和字段中，点击右上方编辑按钮，添加如下用户属性字段：



名称	源	源属性	说明
https://bce.baidu.com/SAML/Attributes/Subuser	属性	accountId:subuser-name/{subuser_name}, accountId:saml-provider/{providerName}	将accountId字段替换为百度智能云中的实际accountId(在百度智能云用户中心获取), subuser_name字段替换为子用户名称, providerName字段替换为IdP名称(有效字符串即可)，如azure

以上字段为必填字段，你可以根据需要添加额外的用户属性字段，详细配置参考本地身份服务的SAML断言配置

8. 在SAML签名证书中，下载IdP SAML元数据，IdP的配置到此结束。

配置SP

配置SP身份提供者

1. 登录百度智能云，鼠标移到右上角，进入**多用户访问控制 > 外部账号接入 > IAM用户联合**；

2. 在用户联合设置中，上传IdP配置步骤7中下载的SAML元数据，切换**功能状态开关**到打开状态 。

本地身份服务的SAML断言配置说明

基本配置

- SubjectConfirmationData中的`Recipient`字段必须配置为`https://login.bce.baidu.com/saml`
- AudienceRestriction中的Audience字段需要配置为`urn:bce:baidu:webservices`

- 属性中需要有名称为https://bce.baidu.com/SAML/Attributes/Subuser的断言，并且格式为“accountId:subuser-name/{subuser_name}, accountId:saml-provider/{providerName}”，其中accountId字段替换为百度智能云中的实际accountId(在百度智能云用户中心获取), subuser_name字段替换为子用户名称, providerName字段替换为IdP名称(有效字符串即可)，如azure。

SAML断言属性

SAML断言的名称和IDP信任策略的属性是一一对应的，目前百度智能云支持的属性包括： saml:iss, saml:aud, saml:cn, saml:eduPersonAffiliation, saml:eduPersonPrincipalName，它们对应的SAML属性分别是：

名称	属性含义
saml:iss	SAML断言的Issuer字段，非必填
saml:aud	SAML断言AudienceRestriction中的Audience字段
saml:cn	SAML断言中的urn:oid:2.5.4.3属性
saml: eduPersonAffiliation	SAML断言中的urn:oid:1.3.6.1.4.1.5923.1.1.1.1属性
saml: eduPersonPrincipalName	SAML断言中的urn:oid:1.3.6.1.4.1.5923.1.1.1.6属性

验证单点登录

从AAD侧发起登录

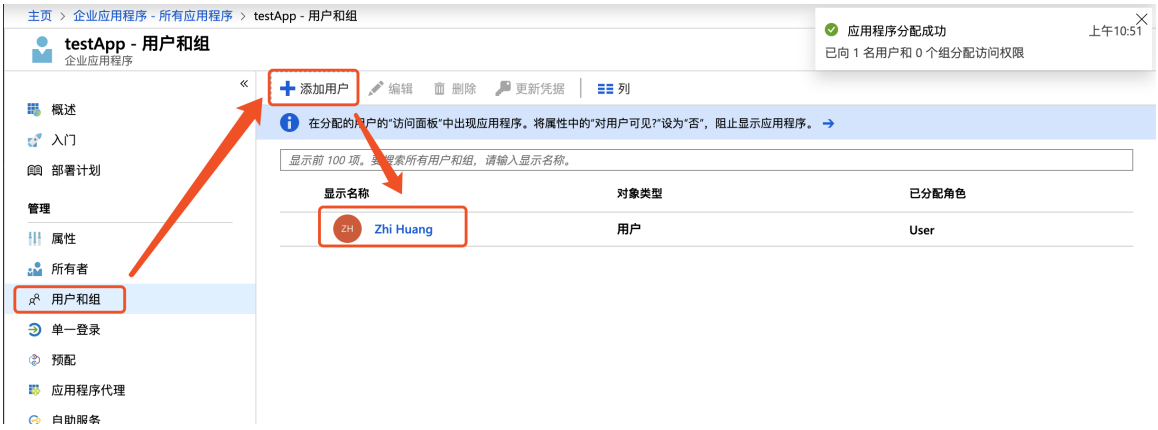
先决条件

已经完成IdP和SP的SAML配置。

操作指南

1. 登录Azure门户，导航到 **所有服务** > **Azure Active Directory** > **企业应用程序** > **testApp**；

2. 点击 **用户和组** > **新建用户** ，将需要授权单点登录的用户添加到该应用程序；



3. 点击 **单一登录** ，点击 **Validate**，选择使用 **作为当前用户登录** ，即可测试跳转到百度智能云页面；

4. 如果需要将登录链接嵌入到企业应用中，可以直接在如下位置获取：

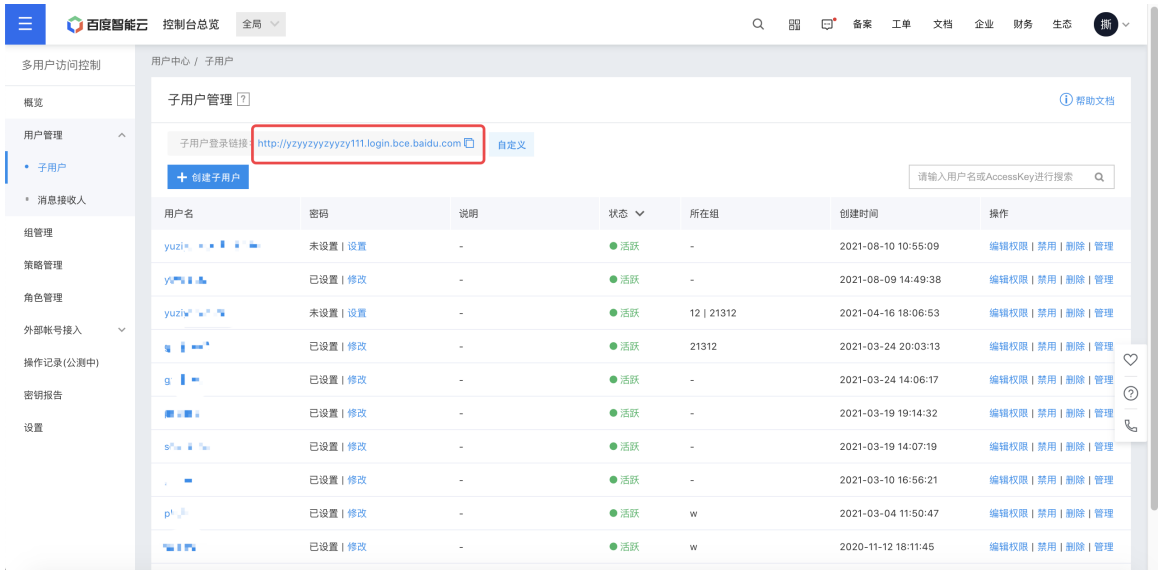
从百度智能云侧发起登录

先决条件

已经完成IdP和SP的SAML配置。

操作指南

1. 登录百度智能云控制台，在**多用户访问控制 > 用户管理 > 子用户**，点击子用户登录链接；



2. 点击使用企业账号登录，跳转至AAD登录界面，输入用户名密码，验证通过后，即可跳转到百度智能云页面；

> 注意

开发时，在重定向到https://login.bce.baidu.com/saml的时候需要携带SAMLResponse中的信息，此信息中需要标明用户具体的身份断言。

```
<RequestedAttribute isRequired="true" Name="https://bce.baidu.com/SAML/Attributes/Subuser"
FriendlyName="RoleEntitlement"/>
```

这两个属性是必传的属性，其中

属性`https://bce.baidu.com/SAML/Attributes/Subuser`用于表示用户当前访问的账户，子用户，和IdP名称，其格式为"accountId:subuser-name/{subuser_name}, accountId:saml-provider/{providerName}"，其中`accountId`为百度智能云中实际的账户ID, `subuser_name`传递SSO后的子用户名， `providerName`则为配置外部身份提供者的名称字段。

IAM角色联合

配置IAM角色联合

你可以通过配置企业端IdP的SAML节点和百度智能云SP的 **外部账户接入 > IAM角色SSO** ，从而实现以百度智能云的角色身份从企业应用到百度智能云的单点登录(SSO)。

先决条件

1. 企业IdP支持SAML 2.0协议；
2. 拥有百度智能云的账号并激活。

配置流程 配置基于SAML的单点登录功能，需要同时完成IdP配置和SP配置，其中IdP配置包含基本配置、用户属性配置和下载元数据等，SP配置包含创建身份服务提供商和配置**角色载体**(用于生成最终的信任策略)。本文以微软公司的Azure Active Directory（AAD）为IdP，介绍如何配置SAML IdP和百度智能云的SP。

配置IdP

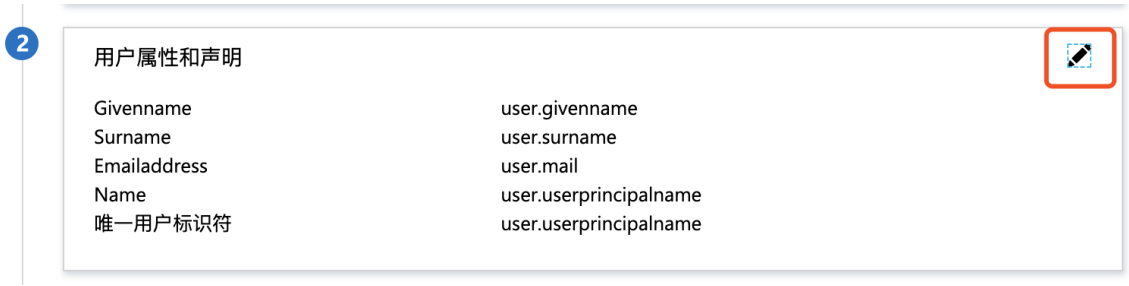
1. 按照流程，注册Azure 账号；
2. 登录Azure门户，在左侧导航栏中，进入 **所有服务 > Azure Active Directory**；
3. 点击 **企业应用程序 > 新建应用程序** ，并选择 **非库应用程序** ，填写应用名称，点击 **添加** 完成应用程序创建；

> 注意：非库应用程序要求开通AAD专业版，你可以选择开通免费试用版，并在完成配置后选择是否需要关闭该试用版。

4. 进入应用程序，选择 **单一登录 > SAML** ；

5. 在基本SAML配置中，点击右上方编辑按钮，配置 **标识符(实体ID)** 和 **回复URL(断言使用者服务URL)** 字段为`urn:bce:baidu:webservices`和`https://login.bce.baidu.com/saml`

6. 在用户属性和字段中，点击右上方编辑按钮，添加如下用户属性字段：



名称	源	源属性	说明
`https://bce.baidu.com/SAML/Attributes/Role`	属性	accountId:role/roleName, accountId:saml-provider/providerName	将accountId字段替换为百度智能云中的实际accountId(在百度智能云用户中心获取)，roleName字段替换为联合登录时的角色名称字段(SP角色管理中配置)，如“BCCAdmin”，providerName字段替换为IdP名称，如`azure`
`https://bce.baidu.com/SAML/Attributes/RoleSessionName`	属性	RoleSessionName	将RoleSessionName替换为你想要在百度智能云中显示的用户名，如`User01`

以上2个字段为必填字段，你可以根据需要添加额外的用户属性字段，详细配置参考本地身份服务的SAML断言配置说明

7. 在SAML签名证书中，下载IdP SAML元数据，IdP的配置到此结束。

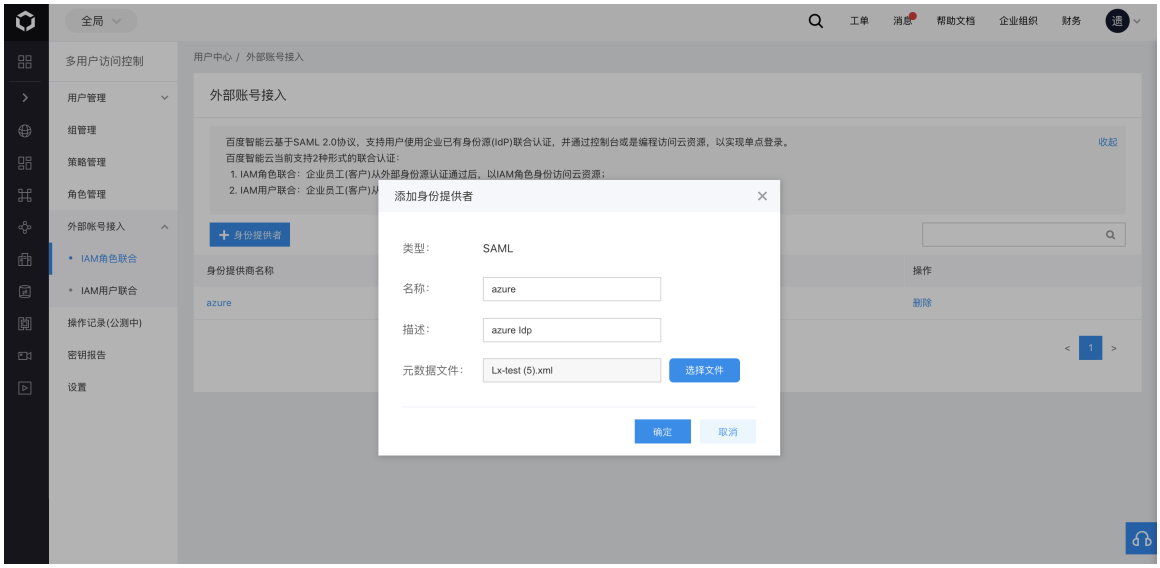
配置SP

新建SP身份提供者

1. 登录百度智能云，鼠标移到右上角，进入**多用户访问控制 > 外部账号接入 > IAM角色联合**；

2. 点击添加**身份提供者**，填写名称、描述，并将上述IdP配置步骤7中下载的SAML元数据导入，点击**确定** 保存；

> 注意：这里的身份提供者名称必须与IdP中配置的用户属性`providerName`保持一致，案例中填写`azure`;



配置IAM角色并授权

从左侧导航栏中点击**角色管理**，配置当以外部身份单点登录到百度智能云时可配置的角色，这里以虚拟机BCC的管理人员角色为例

1. 点击**创建新角色**，填写角色名称如“BCCAdmin”，描述为“BCC管理人员角色”；

2. 在**角色载体**中，选择载体类型为**外部账号**，载体实例选择上一步中添加的“azure”，

3. 你还可以为IdP设定可切换到百度智能云的条件限制，当前百度智能云支持的属性字段有：saml:iss, saml:aud, saml:cn, saml:eduPersonAffiliation, eduPersonPrincipalName。添加限制条件的作用在于更精细化的权限控制，只有在某些条件匹配下，才可以以*BCCAdmin*角色的形式SSO到百度智能云；

4. 在**策略管理**中，将“BCCFullControlAccessPolicy”授予给当前角色，点击完成；

5. 如果你需要为当前IdP设置不同的角色，可以按照如上步骤继续配置，IdP内身份SSO到百度智能云时，通过`https://bce.baidu.com/SAML/Attributes/Role`属性字段*roleName*进行切换。

到此，你已经完成了百度智能云SAML SP的配置工作，下一步你可以回到Azure AD中测试单点登录的效果。

本地身份服务的SAML断言配置说明

基本配置

- SubjectConfirmationData中的Recipient字段必须配置为https://login.bce.baidu.com/saml
- AudienceRestriction中的Audience字段需要配置为urn:bce:baidu:webservices
- 属性中需要有名称为https://bce.baidu.com/SAML/Attributes/Role的断言，并且格式为“accountId:role/roleName, accountId:saml-provider/providerName”，其中accountId需要替换为目标账户的实际AccountId, roleName配置为你期望SSO到百度智能云时扮演的角色, providerName配置为**IAM角色联合**中配置的身份提供商名称；
- 属性中需要有名称为https://bce.baidu.com/SAML/Attributes/RoleSessionName的断言，RoleSessionName会显示在百度智能云，格式为字符串。

SAML断言属性

SAML断言的名称和IdP信任策略的属性是一一对应的，目前百度智能云支持的属性包括： saml:iss, saml:aud, saml:cn, saml:eduPersonAffiliation, saml:eduPersonPrincipalName，它们对应的SAML属性分别是：

名称	属性含义
saml:iss	SAML断言的Issuer字段，非必填
saml:aud	SAML断言AudienceRestriction中的Audience字段
saml:cn	SAML断言中的urn:oid:2.5.4.3属性
saml: eduPersonAffiliation	SAML断言中的urn:oid:1.3.6.1.4.1.5923.1.1.1.1属性
saml: eduPersonPrincipalName	SAML断言中的urn:oid:1.3.6.1.4.1.5923.1.1.1.6属性

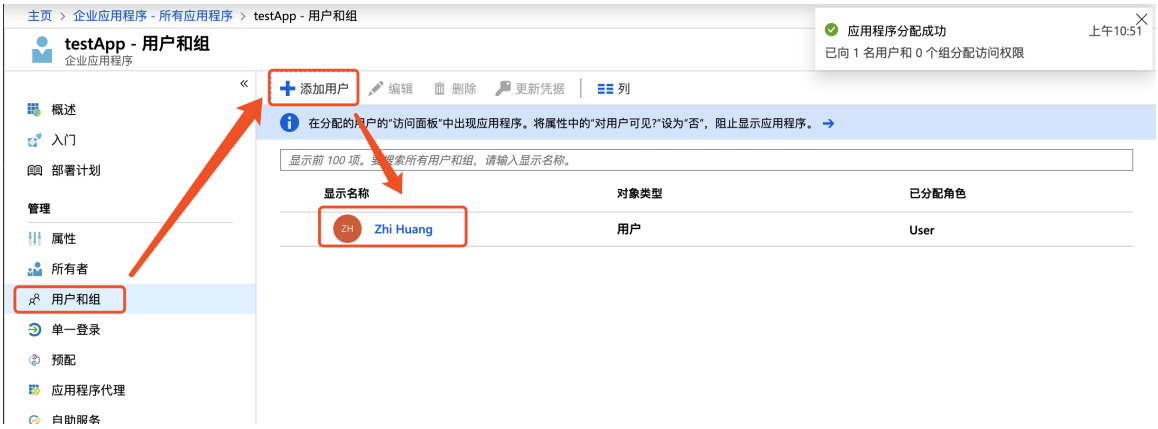
验证单点登录

先决条件

已经完成IdP和SP的SAML配置。

操作指南

1. 登录Azure门户，导航到 **所有服务** > **Azure Active Directory** > **企业应用程序** > **testApp**；
2. 点击 **用户和组** > **新建用户** ，将需要授权单点登录的用户授权到该应用程序；



3. 点击 **单一登录** ，点击 Validate，选择使用 **作为当前用户登录** ，即可测试跳转到百度智能云页面；

4. 如果需要将登录链接嵌入到企业应用中，可以直接在如下位置获取：

> 注意

开发时，在重定向到`https://login.bce.baidu.com/saml`的时候需要携带`SAMLResponse`中的信息，此信息中需要标明用户具体的身份断言。

```
 <RequestedAttribute isRequired="true" Name="https://bce.baidu.com/SAML/Attributes/Role"
FriendlyName="RoleEntitlement"/>
<RequestedAttribute isRequired="true" Name="https://bce.baidu.com/SAML/Attributes/RoleSessionName"
FriendlyName="RoleSessionName"/>
```

这两个属性是必传的属性，其中

1. 属性`https://bce.baidu.com/SAML/Attributes/Role`用于表示用户当前访问的账户，角色，和IdP名称，其格式为"accountId:role/roleName, accountId:saml-provider/providerName"，其中`accountId`为百度智能云中实际的账户ID，`roleName`填写上文配置的*BCCAdmin*，表示通过SAML登录的用户拥有管理员权限，`providerName`则为配置外部身份提供者的名称字段；
2. 属性`https://bce.baidu.com/SAML/Attributes/RoleSessionName`为用户百度智能云在控制台中显示的相应用户名称，此名称也会记录到操作日志中。

消息中心

消息中心介绍

消息中心是百度智能云的一站式消息平台，承担百度智能云统一的消息配置和调用。作为发送消息的统一调度中心，以短信、邮件、站内信等方式向用户发送消息。

点击**消息-进入消息中心**即可访问百度智能云的消息中心。

站内信

在消息中心可以查看站内信消息，并按照消息类型对站内信进行筛选。点击站内信的消息标题可以查看站内信的详细内容。

接收设置

编辑接收人

消息中心接收系统和产品发送给主账号的消息通知，并且默认主账号为接收人。

1、点击**编辑**，对接收人进行修改。

2、点击**快捷添加消息接收人**。

3、验证消息接收人。

成功创建消息接收人，系统会自动发送短信和邮箱邀请给对应的消息接收人，点击邀请链接，即可完成对其联系方式的验证；你可以重新打开“编辑接收人”窗口，通过查看消息接收人的短信/邮箱状态，确认其是否验证通过，对于未验证通过的用户，可以重发验证。

消息接收人管理

在消息中心，点击消息接收人管理，对消息接收人进行统一管理。

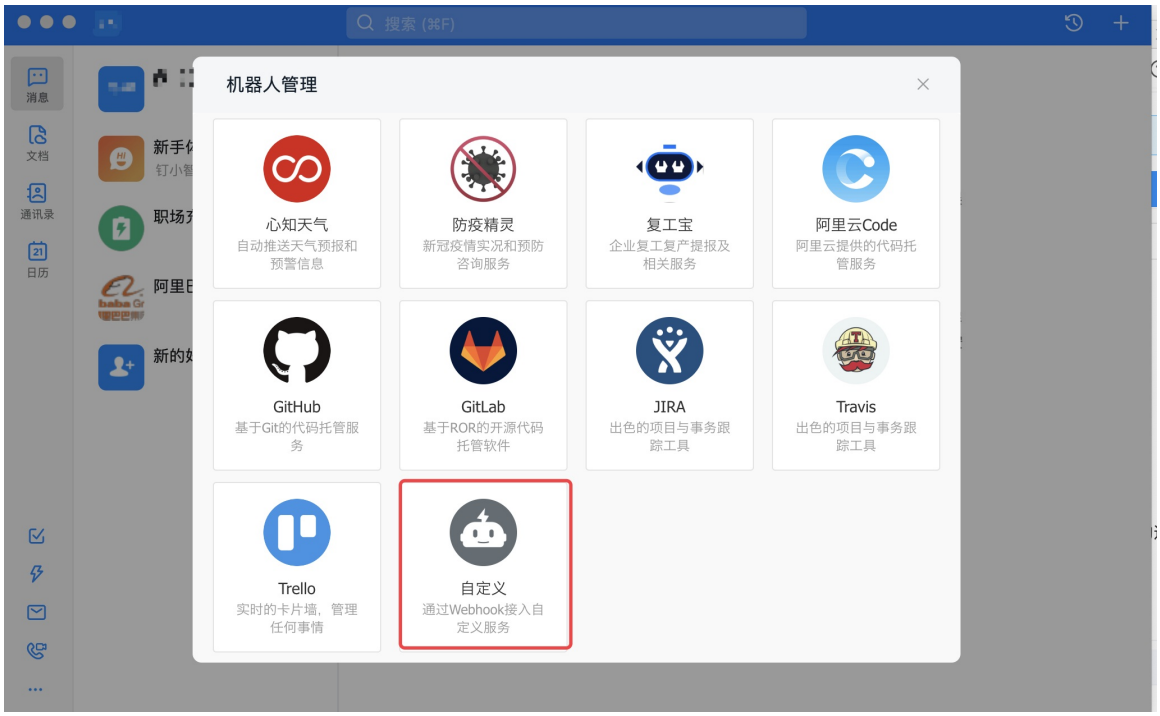
> 温馨提示： 在消息接收人管理中添加消息接收人后，还需要将其在消息中心设置为接收人，才能够接收消息中心发送的消息通知。

编辑钉钉机器人

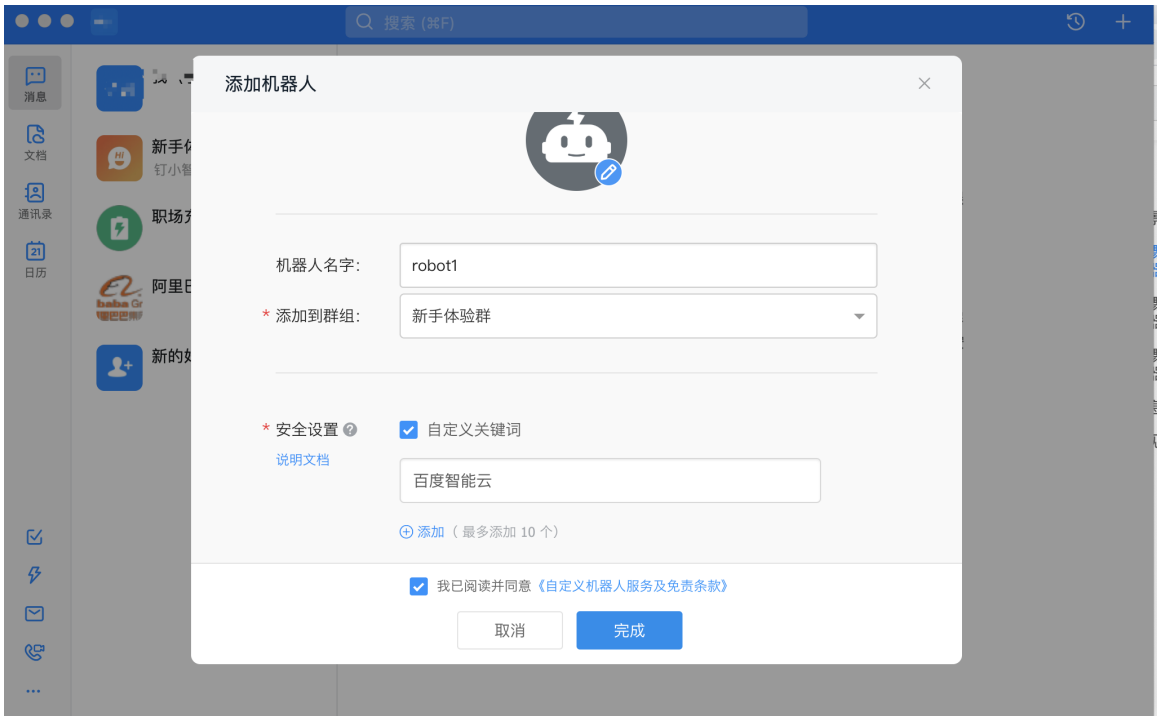
消息中心可以通过钉钉的自定义机器人，将系统和产品发送的消息通知同步到钉钉的聊天群。

1、打开PC端钉钉（以PC端为例），点击左上角头像，选择**机器人管理**。

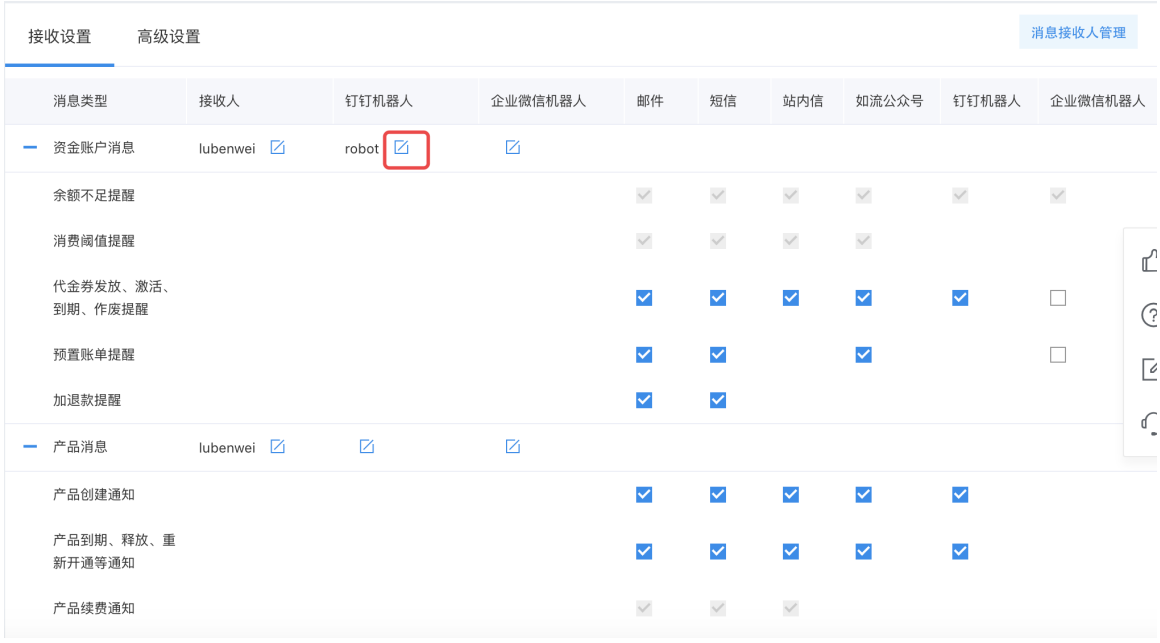
2、在机器人管理界面，选择自定义机器人。



3、输入**机器人名字**并选择要发送消息通知的群，同时可以为机器人设置机器人头像；在**安全设置**中，选择**自定义关键词**并添加**百度智能云**为关键词。勾选**我已阅读并同意《自定义机器人服务及免责条款》**，点击**完成**，获取钉钉机器人Webhook地址。



4、进入消息中心-消息接收设置-接收设置，点击**编辑**，对钉钉机器人进行修改。



5、点击**添加钉钉机器人**，输入钉钉机器人名称和获取的钉钉机器人Webhook地址，点击**确认**。

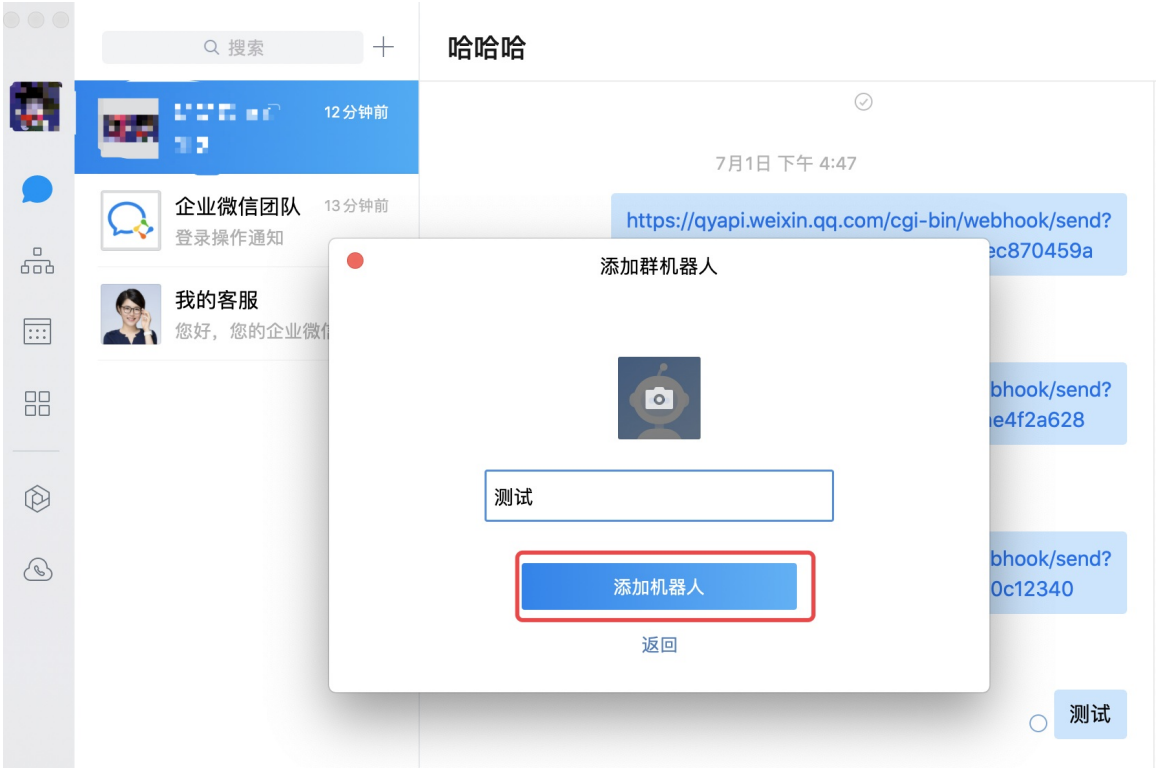
编辑企业微信机器人

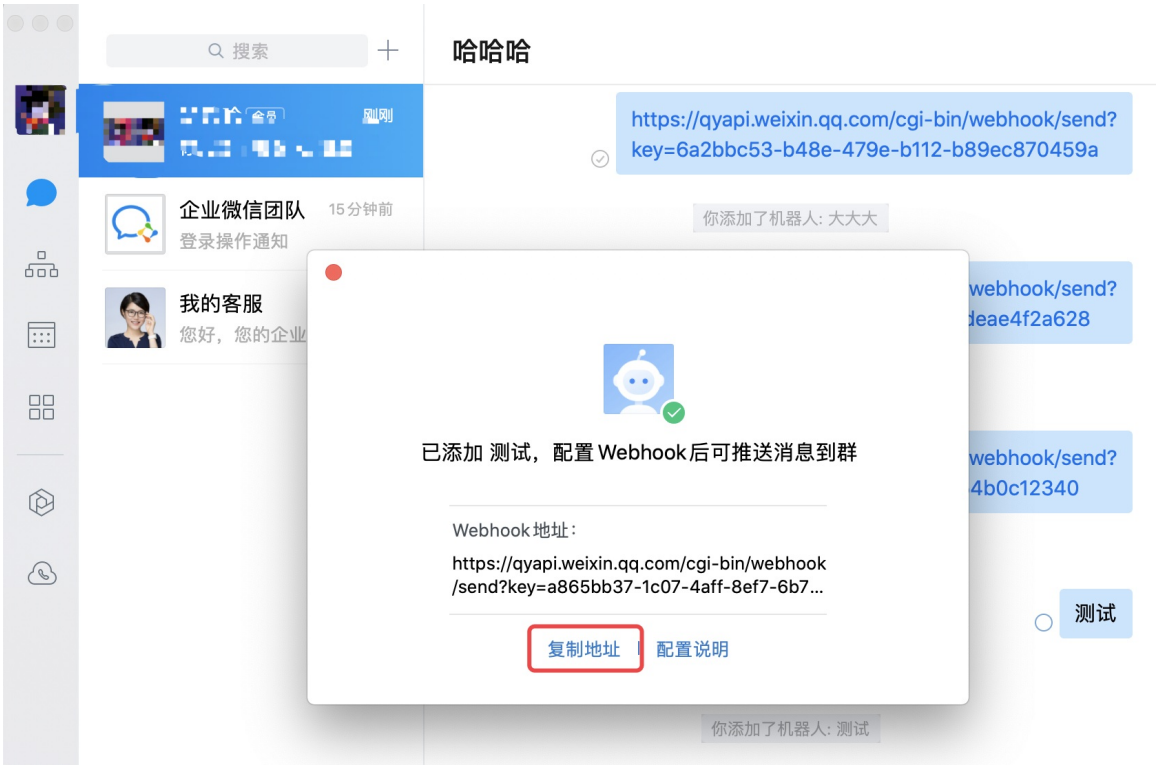
消息中心可以通过企业微信的群机器人，将系统和产品发送的消息通知同步到企业微信的聊天群。

1、打开PC端企业微信（以PC端为例），右键点击聊天列表中的企业微信群，点击**添加群机器人**。

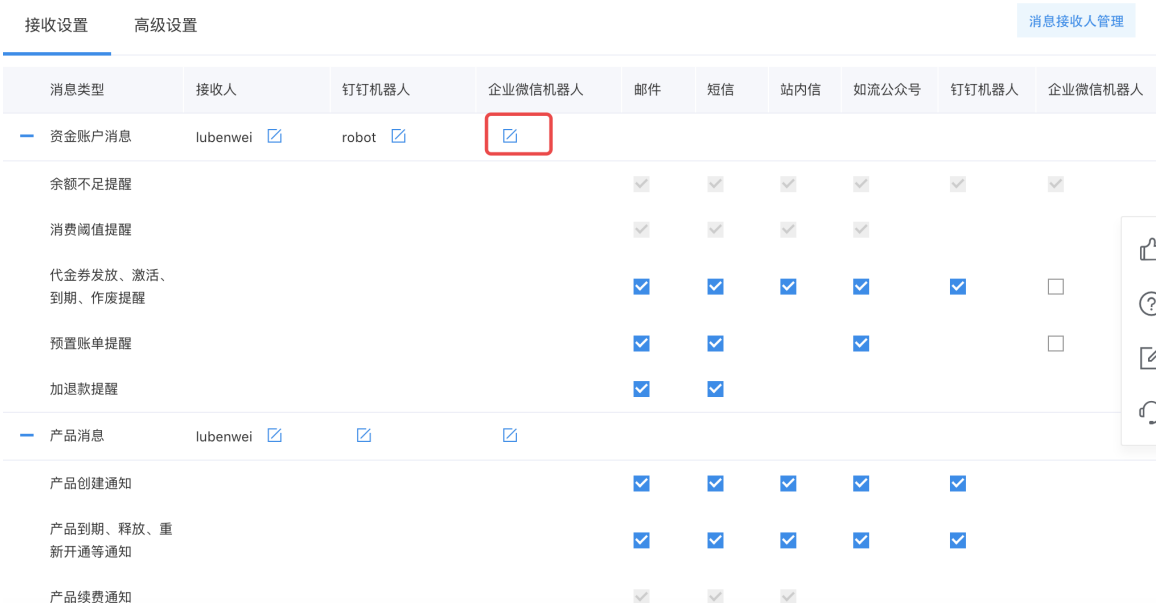
2、点击**新创建一个机器人**。

3、输入机器人名称，点击**添加机器人**，复制Webhook地址。





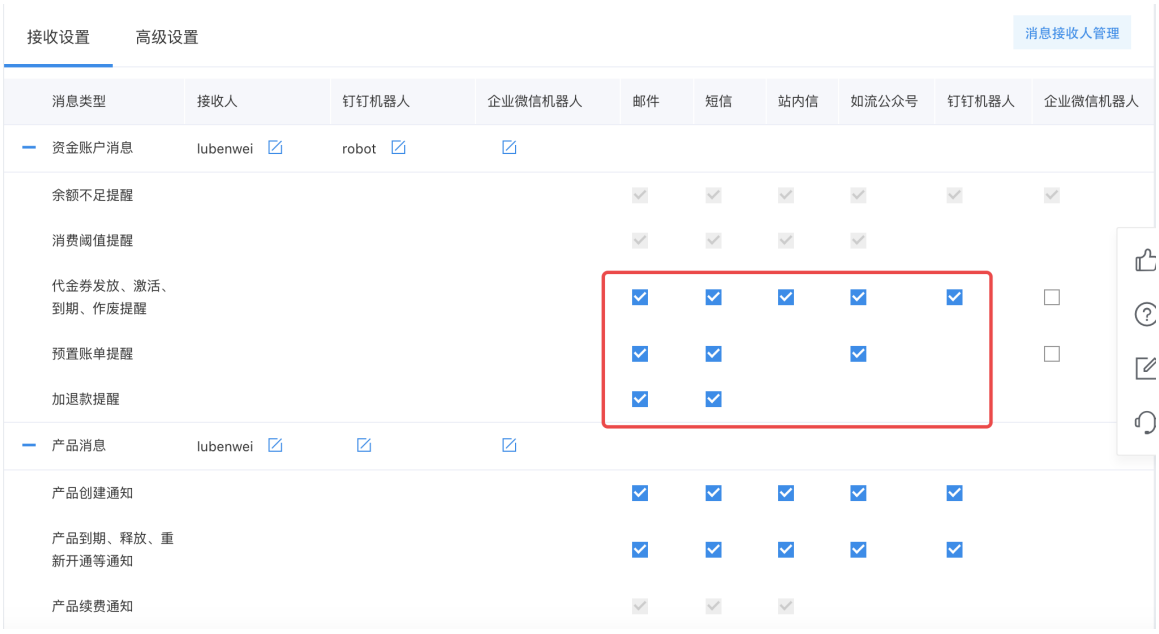
4、进入消息中心-消息接收设置-接收设置，点击**编辑**，对企业微信机器人进行修改。



5、点击**添加企业微信机器人**，输入企业微信机器人名称和获取的企业微信机器人Webhook地址，点击**确认**。

消息订阅

在消息中心，通过“**勾选**”，对不同消息类型的消息通知完成订阅；置灰的勾选框表示必须订阅的消息通知。



高级设置

在消息中心的高级设置中，可以选择是否以窗口提醒的形式接收站内信以及消息通知的语言种类。

消息中心常见问题

1、如何取消全部的消息通知？

无法如何取消全部的消息通知，某些类型的消息通知为必选项。

2、如何取消某一类消息通知？

对于可勾选的消息类型，取消邮件、短信、站内信、如流公众号等消息通知的方式即可。

3、如何设置多人同时接收消息通知？

点击**编辑接收人**，选择多位用户作为消息接收人。

4、为避免因账户欠费导致业务中断，应设置何种消息通知以能够及时提醒充值？

余额不足提醒为默认必选项，用户只需要设置所需的消息接收人即可。

5、为什么开通了消息中心提醒，却并未收到短信或邮件通知？

请查看是否在消息中心设置了正确的消息接收人。

6、百度智能云APP能否进行消息接收设置？

不能

账户安全审计

操作记录

IAM提供主账号／子用户对云资源所进行的运维管控类操作记录，用于用户合规审计。目前IAM的关键行为如主子用户的登入登出、管理员用户的管理操作与行为，已经全部接入到云审计。

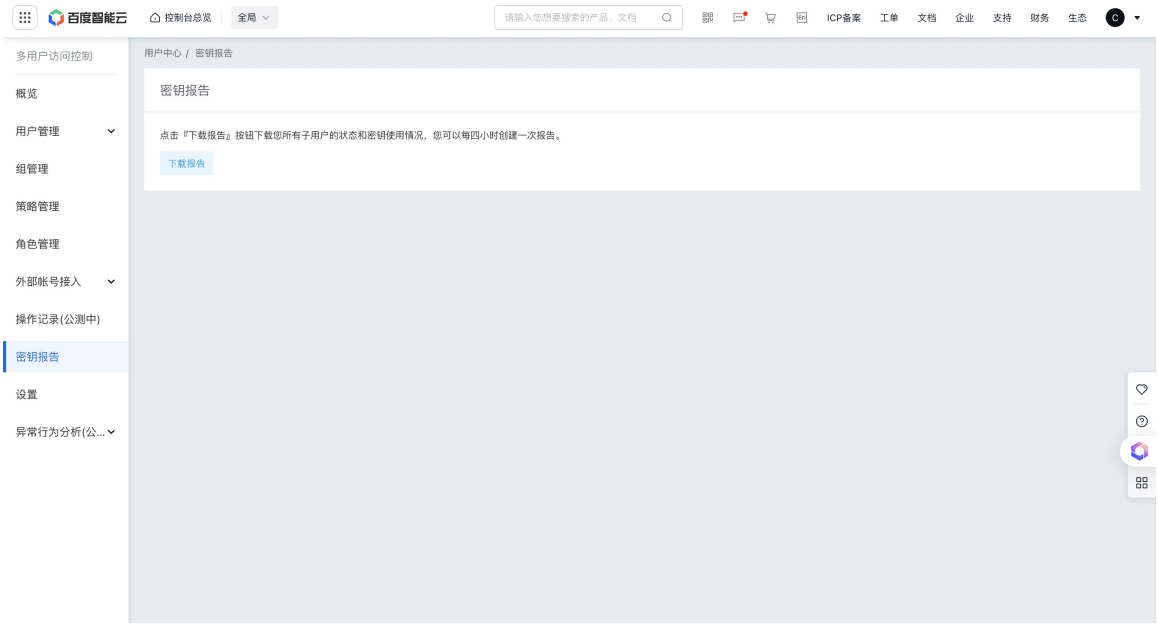
你可以点击**多用户访问控制 > 操作记录** 进入云审计中，查看用户的关键行为和事件。

账号管理和多用户访问控制支持的云审计的事件列表可参考： 事件列表

密钥报告

密钥报告记录子用户的状态和密钥使用情况，您可以每四小时创建一次。及时关注子用户使用情况，减少安全隐患。

主要记录信息包括：用户名,创建时间,是否启用MFA,是否启用密码,上次修改密码时间,上次使用密码时间,ak上次使用时间,ak上次使用的服务,ak上次使用区域。



设置

设置账户别名

- 为你的账户设置账户别名，从而得到一个简单易懂的IAM用户登录链接。
- 如果选择APP登录，对应的主账号部分填写的也是在此处自定义的账户别名。

操作步骤

1. 进入控制台，鼠标移动到页面右上角头像处，进入**多用户访问控制 > 设置**。
2. 编辑**账户别名设置**部分

例如： 账户别名设置为 `test_123`，则 PC 登录链接会简化为`http://test_123.login.bce.baidu.com`，APP登录时主账号别名部分填写 `test_123` 即可。

子用户安全设置

设置密码策略

自定义密码策略用于加强IAM子用户的密码强度、有效期等，以保证你的账户安全。目前百度智能云支持配置的密码策略有：

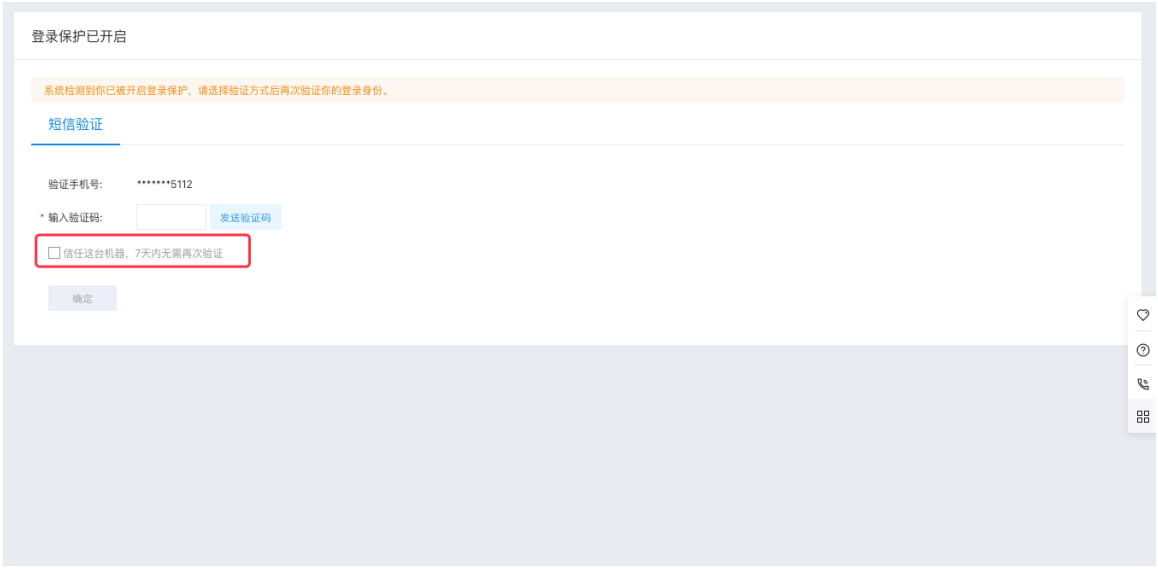
1. 密码长度：默认填写8位，范围8-32位；
2. 至少包含：多选，可选“大写字母”“小写字母”“数字”“特殊字符(!”#$%&’()*+,-./:;<=>?@[]^_>{|}<)”；选择某类元素则表示密码中至少要包含一个此元素，如选择“大写字母”和“小写字母”，则表示密码中必须要至少包含一个大写字母和至少一个小写字母；
3. 密码有效期：默认0天，表示永久有效，范围0-1095；密码过期3天内控制台会弹窗提示，请及时修改密码；
4. 密码过期后策略：

- 不限制登录，表示密码过期后子用户仍然可以登录控制台，但必须重置密码；
- 限制登录，表示密码过期后子用户无法登录控制台，必须由管理员用户重置密码；

5. 历史密码检查：默认1次，表示不能与上一次密码重复，范围0-24；
6. 密码重试约束：一小时内使用错误密码登录超过一定次数，则锁定一小时，默认5次，范围0-32，0表示无约束；

保存MFA状态7天

在开启登录保护双因素认证的前提下，开启**保存MFA状态7天**，即可在登录保护验证时勾选"信任这台机器，7天内无需再次验证"，验证通过后，7天内使用相同的浏览器登录时，无需进行登录保护验证



> 说明：
>
> - 此功能对主子用户同时生效。

设置会话过期时间

登录会话过期是指用户登录后，如果在有效时间内不进行任何操作，系统自动清理当前用户的会话信息，以保证账户安全。会话过期后，需要重新登录控制台。

> 说明：
>
> - 时间设置范围为15分钟~23小时59分钟，默认过期时间为1小时，用户可以根据实际情况自行设置。
> - 此功能对主子用户同时生效。

IP白名单

在IAM上对所有使用服务的设置IP白名单，IAM用户在直接登录控制台或是使用OpenAPI或SDK访问百度智能云资源时，来源IP必须在IP白名单内才能进行访问。

当前**IP白名单**功能支持：

- 限制子用户登录控制台，即限制非白名单内的IP登录控制台进行操作；
- 限制编程访问云资源，即限制非白名单内的IP通过OpenAPI或SDK访问云资源，当前已支持编程访问限制的云服务清单见下文。

你可以选择只限制子用户登录或是限制编程访问，也可以同时限制2种方式。

IP白名单填写要求 填写允许访问的 IP 地址或者 IP 段。

- IP 白名单不设置则默认整个网路。
- 若需要添加多个 IP，用英文逗号或者空格隔开。
- 若填写 IP 段，如10.10.10.0/24，则表示10.10.10.X的IP地址都可以访问，支持CIDR模式。

AccessKey泄漏监测与告警 在用户配置了IP白名单的基础上，可以进一步选择打开用户AccessKey泄漏监测与告警功能，对于来自非白名单内IP的AccessKey访问，在一定时间内达到一定次数会触发BCM提供的事件告警，从而帮助客户及时发现潜在AccessKey泄漏的情况，有效保障云账号的资产安全。

AccessKey(后文统一简称AK)泄漏监测与告警使用**BCM**事件监控能力，具体开通需要完成如下2个步骤：

1. 在多用户访问控制的设置模块，打开AK泄漏监测告警；
2. 在云监控BCM的事件监控模块，配置BCT服务的事件监控；

打开AK泄漏监测告警

开启IP白名单功能后，在**AK泄漏监测告警**选项处，点击**开启**按钮，即可完成在多用户访问控制的配置，下一步需要跳转到BCM的事件监控模块，进行相应配置。

配置事件告警

BCM的事件监控中，需要完成报警策略配置、报警动作配置：

1. 点击**创建报警策略**；
2. 在策略信息模块，填写策略名称，选择服务为**云审计BCT**，地域默认为全局，其他保持默认选项；
3. 在报警动作模块中，选择已配置的报警动作，将告警信息推送到相应的联系人或组，如你尚未配置任何报警，可先**添加报警动作**，详细可参考配置报警动作

配置完成后，对于来自非IP白名单内的AK调用，只要符合报警触发条件，即会按照在BCM中配置的报警动作，对相应的联系人或是群组进行告警通知。收到告警后，建议相关人员按照通知提示，到云审计服务中查看详细的访问记录，以确认是否为安全访问。

支持编程访问云资源IP限制的服务清单 计算

产品名称
云服务器 BCC
物理服务器 BBC
应用引擎 BAEPRO
容器实例 BCI
容器引擎 CCE
函数计算 CFC
专属服务器 DCC

网络

产品名称
弹性公网IP EIP
负载均衡 BLB
智能云解析 DNS
私有网络 VPC

存储与CDN

产品名称
内容分发网络 CDN
对象存储 BOS
文件存储 CFS
存储网关 BSG

安全与管理

产品名称
DDoS防护服务 ADAS
云监控 BCM
云安全 BSS

数据分析

产品名称
百度MapReduce BMR
百度流式计算 BSC
百度数据科学平台 JARVIS
百度消息服务 KAFKA

数据库

产品名称
云数据库 RDS
云数据库 SCS
云数据库 HTAP for CockroachDB
云数据库 DocDB for MongoDB

网站服务

产品名称
云主机管家 HME
云虚拟主机 BCH
域名服务 BCD

物联网服务

产品名称
时序数据库 TSDB
规则引擎 Rule Engine
物解析 IoT Parser
物管理 IoT Device
智能调度 ILS
智能调度 ILS
函谷物联安全系统 HISK

智能多媒体服务

产品名称
文档服务 DOC
音视频直播 LSS
实时音视频通信 RTC
音视频点播 VOD

人工智能

产品名称
OCR能力引擎 AI_OCR
人脸能力引擎 AI_FACE
百度机器学习 AI_BML

区块链

产品名称
百度区块链引擎 BBE

用户异常行为分析（公测中）

风险行为管理

概述

用户实体异常行为分析(User Entity Behavior Analysis, UEBA)，是一种基于用户(包含自然人和设备实体等，后续统称用户)在系统的行为模式，分析是否存在异常，并提示给系统管理人员的高级安全功能。

用户异常行为的场景通常包括：账号异常、AK异常、设备异常、操作行为异常、主机失陷、横向移动等。本文将介绍如何创建、管理不同场景下的风险行为，以及如何接收风险行为的告警通知。

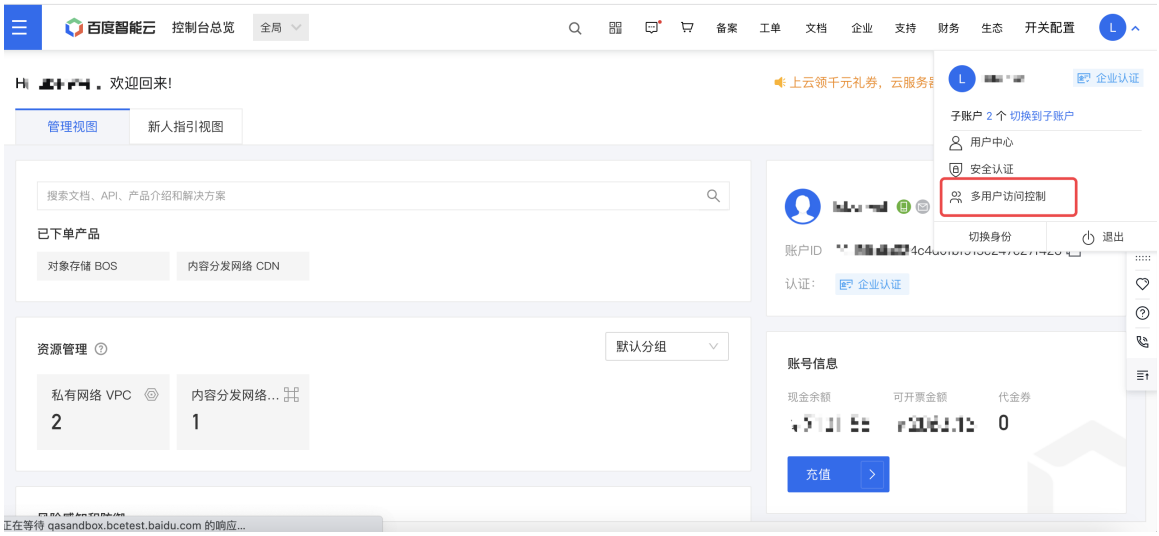
> 注意：本期仅支持AKSK未使用、AKSK调用异常两种风险行为

公测说明

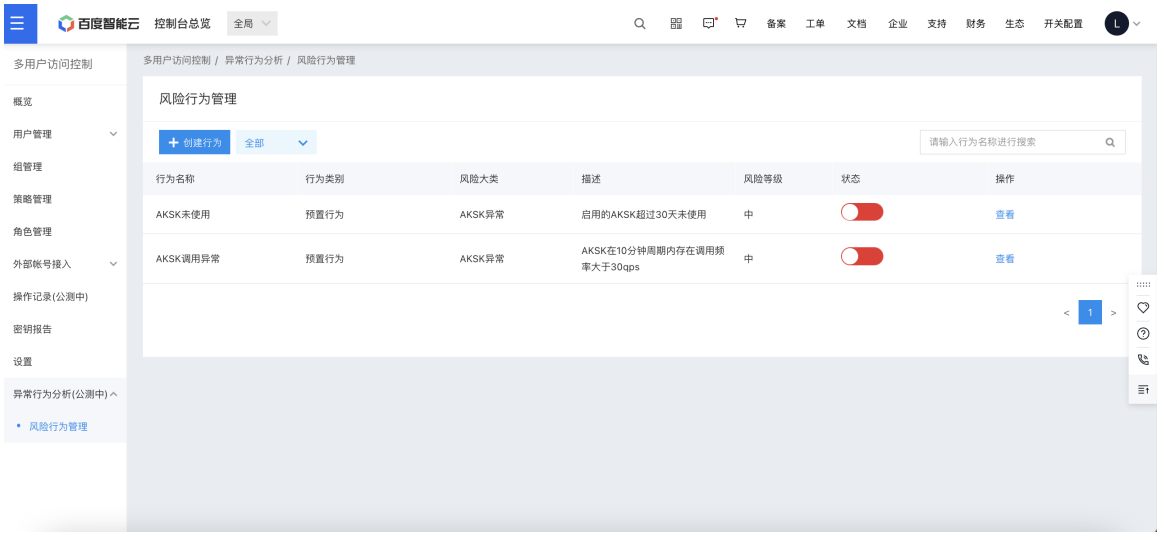
用户实体异常行为分析功能目前处于公测状态，仅对白名单内的账户开放。开通公测请联系大客户服务经理。

创建风险行为

1、登录百度智能云管理控制台，点击右上角头像，点击**多用户访问控制**。

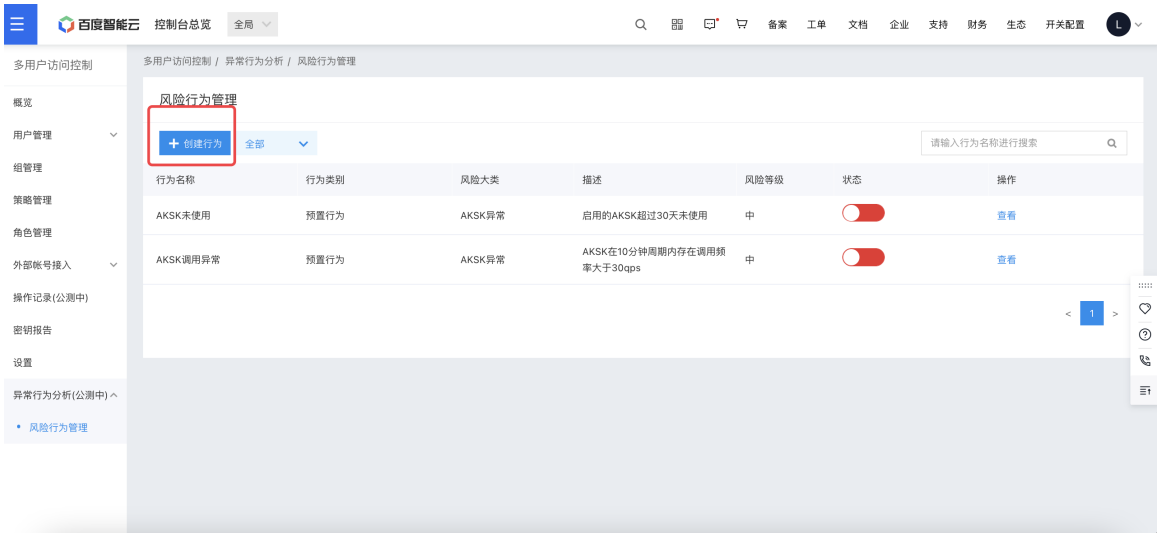


2、进入**多用户访问控制 > 异常行为分析 > 风险行为管理**。

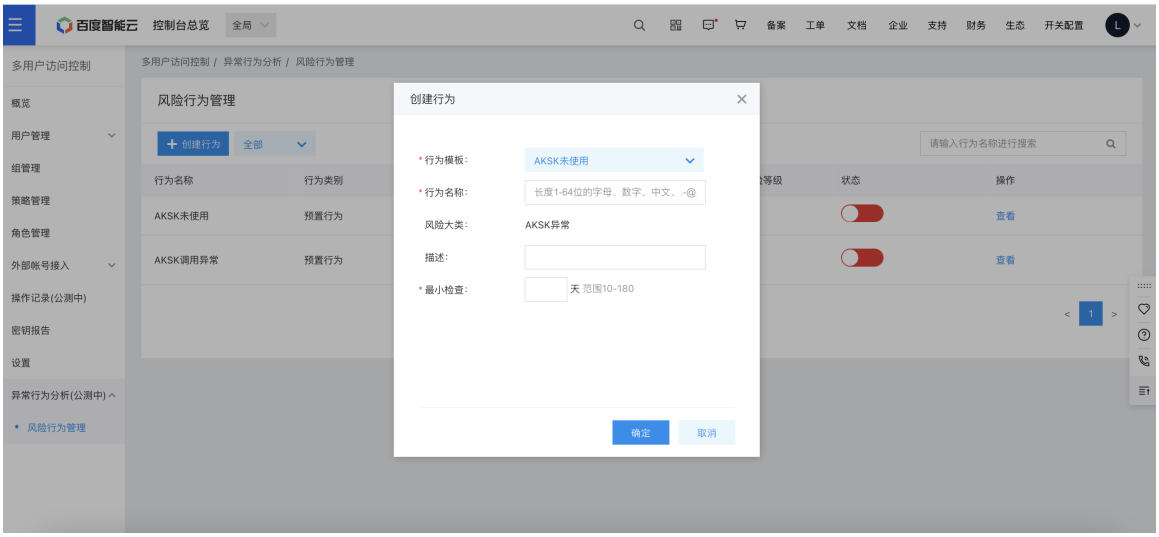


> 风险行为包括预置行为、自定义行为：
>
> 1、预置行为是百度智能云预设风险行为，不支持编辑或删除；2、自定义行为是管理员根据实际的业务需要，配置出的特定场景下的风险行为，相比于预置行为，自定义行为监测的粒度更细，风险行为内容配置更加灵活。

3、点击**创建行为**。

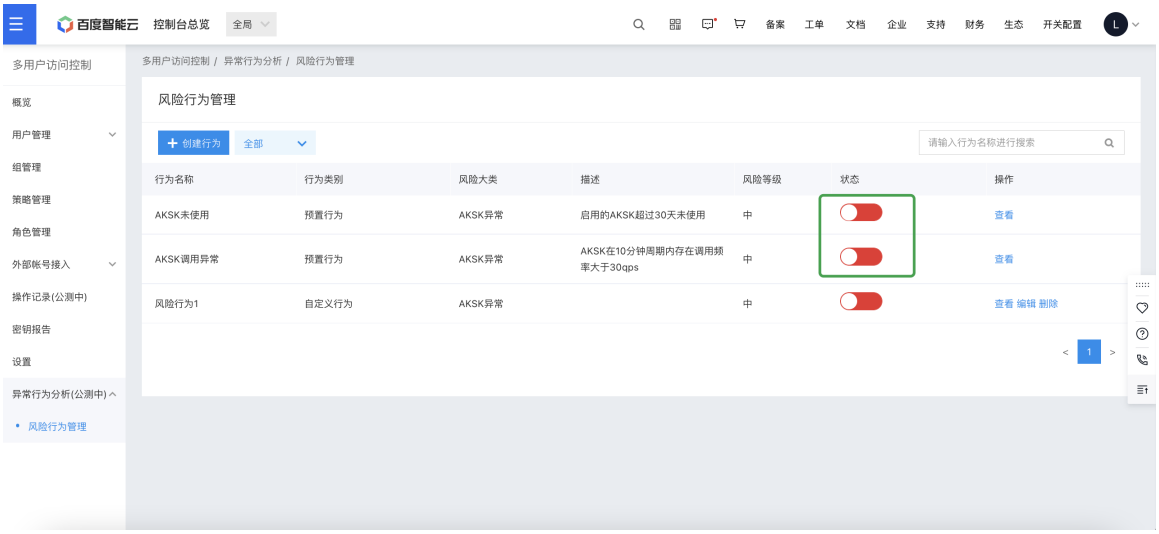


4、选择行为模板，输入风险行为配置内容，点击**确定**，完成自定义行为创建。

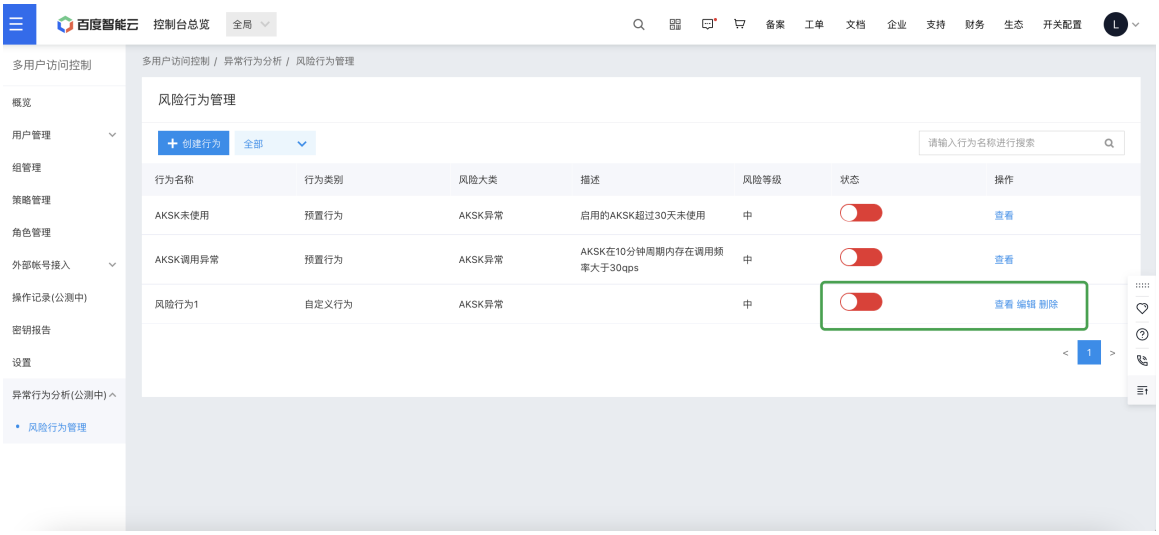


管理风险行为

1、针对预置行为，仅支持风险行为的启用、禁用。

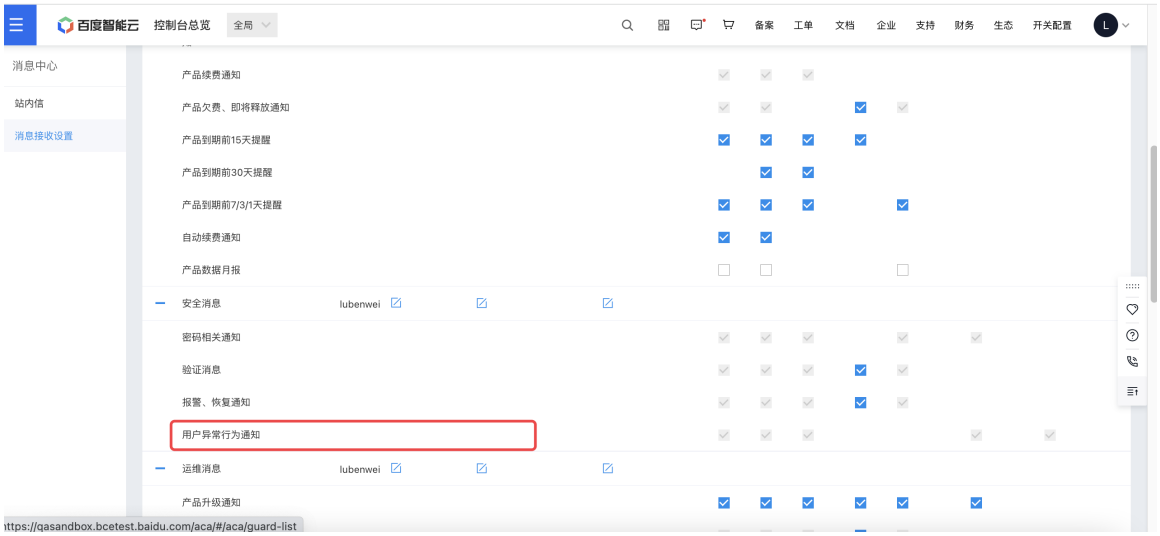


2、针对自定义行为，可以对风险行为进行编辑、删除、启用、禁用。



告警通知

在**消息中心 > 消息接收设置 > 安全消息**中，已为用户设置好了用户异常行为通知，并支持短信、邮件、站内信、钉钉机器人、企业微信机器人等五种通知渠道，通知渠道的配置请参考消息中心。



当风险行为出现异常时，会根据配置好的通知渠道发送告警通知。

> 注意：对于AKSK异常的告警通知，仅会在邮件中提示用户具体发生异常的AK。

API参考（IAM）

简介

概述

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，适用于企业组织内的不同角色，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。

用户可以通过本文档提供的简单的REST API接口，进行用户和组的管理，及其权限授予等操作。

当前多用户访问控制开放API的服务域名为：iam.bj.baidubce.com，支持HTTP和HTTPS两种方式访问。

接口概览

本节汇总了多用户访问控制IAM可调用的API，具体接口信息请点击链接查看详细内容。

接口	描述
用户管理接口	创建、查询、更新、删除、列举子用户，管理子用户的登录配置以及AccessKey。
组管理接口	创建、查询、更新、删除、列举用户组，管理用户组成员。
策略管理接口	创建、查询、更新、删除、列举权限策略，为用户、组关联或解除权限。

通用说明

认证机制

所有API的安全认证一律采用Access Key与请求签名机制。 Access Key由Access Key ID和Secret Access Key组成，均为字符串。 对于每个HTTP请求，使用下面所描述的算法生成一个认证字符串。提交认证字符串放在Authorization头域里。服务端根据生成算法验证认证字符串的正确性。 认证字符串的格式为`bce-auth-v{version}/{accessKeyId}/{timestamp}/{expirationPeriodInSeconds}/{signedHeaders}/{signature}`。

- version是正整数。
- timestamp是生成签名时的UTC时间。
- expirationPeriodInSeconds表示签名有效期限。
- signedHeaders是签名算法中涉及到的头域列表。头域名之间用分号（;）分隔，如host;x-bce-date。列表按照字典序排列。（本API签名仅使用host和x-bce-date两个header）
- signature是256位签名的十六进制表示，由64个小写字母组成。

当百度智能云接收到用户的请求后，系统将使用相同的SK和同样的认证机制生成认证字符串，并与用户请求中包含的认证字符串进行比对。如果认证字符串相同，系统认为用户拥有指定的操作权限，并执行相关操作；如果认证字符串不同，系统将忽略该操作并返回错误码。鉴权认证机制的详细内容请参见鉴权认证机制。

通信协议

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。

请求结构说明

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

请求参数包括如下4种：

参数类型	说明
URI	用于指明操作实体，如:PUT /v1/user/{userName}
Query参数	URL中携带的请求参数
HEADER	通过HTTP头域传入，如:x-bce-date
RequestBody	通过JSON格式组织的请求数据体

响应结构说明

响应值分为两种形式：

响应内容	说明
HTTP STATUS CODE	如200,400,403,404等
ResponseBody	JSON格式组织的响应数据体

版本号

参数	类型	参数位置	描述	是否必须
version	String	URI参数	API版本号	必须

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。

规范化字符串

通常一个字符串中可以包含任何Unicode字符。在编程中这种灵活性会带来不少困扰。因此引入“规范字符串”的概念。一个规范字符串只包含百分号编码字符以及URI（Uniform Resource Identifier）非保留字符（Unreserved Characters）。 RFC 3986规定URI非保留字符包括以下字符：字母（A-Z，a-z）、数字（0-9）、连字号（-）、点号（.）、下划线（_）、波浪线（~）。 将任意一个字符串转换为规范字符串的方式是:

- 将字符串转换成UTF-8编码的字节流。
- 保留所有URI非保留字符原样不变。
- 对其余字节做一次RFC 3986中规定的百分号编码（Percent-Encoding），即一个%后面跟着两个表示该字节值的十六进制字母。字母一律采用大写形式。

示例： 原字符串：`this is an example for 测试`，对应的规范字符串：`this%20is%20an%20example%20for%20%E6%B5%8B%E8%AF%95`。

服务域名

区域	服务端点Endpoint	协议
北京	iam.bj.baidubce.com	HTTP and HTTPS

> 为了进一步加强了对安全性的支持，TLS协议仅支持1.2+版本

公共请求头与公共响应头

公共请求头

头域	说明	是否必须
Authorization	包含Access Key与请求签名。	必须
x-bce-date	该请求创建的时间，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。如果用户使用了标准的Date域，该头域可以不填。当两者同时存在时，以x-bce-date为准。	可选

公共响应头

头域	说明
Content-Length	RFC2616中定义的HTTP请求内容的类型。
x-bce-request-id	对应请求的requestId。

错误码

错误码格式

当用户访问API出现错误时，会返回给用户相应的错误码和错误信息，便于定位问题，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	表示具体错误类型。
message	String	有关该错误的详细说明。
requestId	String	导致该错误的requestId。

例如：

```
 {
   "code":"IllegalRequestUrl",
   "message":"The requested url belongs to domain which is not under acceleration",
   "requestId":" 81d0b05f-5ad4-1f22-8068-d5c9de60a1d7"
}
```

公共错误码

错误码	错误消息	HTTP状态码	描述
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。
InternalError	We encountered an internal error. Please try again.	500Internal 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。
InvalidHTTPAuthHeader	The HTTP authorization header is invalid. Consult the service documentation for details.	400 Bad Request	Authorization头域格式错误。
InvalidHTTPRequest	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	详见ETag。
RequestExpired	Request has expired. Timestamp date is XXX.	400 Bad Request	请求超时。XXX要改成x-bce-date的值。如果请求中只有Date，则需要将Date转换为datetime。
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。

用户管理接口

创建用户

接口描述

创建用户。

请求结构

```
 POST /v1/user HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
name	String	RequestBody参数	用户名	是
description	String	RequestBody参数	用户的描述	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

UserModel对象

请求示例

```
 POST /v1/user HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 20
Authorization: AuthorizationString

{"name":"test-user"}
```

响应示例

```
 HTTP/1.1 201 Created
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
{
    "id": "f4225c39ba2247b692eec7b461835aa1",
    "createTime": "2019-06-06T03:42:13Z",
    "name": "test-user"
}
```

查询用户

接口描述

查询用户。

请求结构

```
 GET /v1/user/{userName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

UserModel对象

请求示例

```
 GET /v1/user/test-user HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
 Content-Type: application/json;charset=UTF-8
 X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
 Server: BWS

{
    "id": "f4225c39ba2247b692eec7b461835aa1",
    "createTime": "2019-06-06T03:42:13Z",
    "name": "test-user"
}
```

更新用户

接口描述

更新用户。

请求结构

```
 PUT /v1/user/{userName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
name	String	RequestBody参数	更新后的用户名	否
description	String	RequestBody参数	用户的描述	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

UserModel对象

请求示例

```
 PUT /v1/user/test-user HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 34
Authorization: AuthorizationString

{"description":"update user demo"}
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "id": "f4225c39ba2247b692eec7b461835aa1",
    "createTime": "2019-06-06T03:42:13Z",
    "name": "test-user"
}
```

删除用户

接口描述

删除用户。

> **说明**：删除子用户前用户需要解除子用户关联的其他所有对象，包括：权限策略、用户组、AccessKey。

请求结构

```
 DELETE /v1/user/{userName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

user对象

请求示例

```
DELETE /v1/user/test-user HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
HTTP/1.1 204 No Content
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举用户

接口描述

列举用户。

请求结构

```
GET /v1/user HTTP/1.1
Host: iam.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

无。 **响应头域**

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
users	List<UserModel>	用户对象的列表

请求示例

```
GET /v1/user HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
   "users": [
      {
         "id": "f4225c39ba2247b692eec7b461835aa1",
         "createTime": "2019-06-06T03:42:13Z",
         "name": "test-user"
      }
   ]
}
```

配置用户的控制台登录

接口描述

配置用户的控制台登录，为其配置登录密码、开启登录MFA、配置第三方账号绑定等。

请求结构

```
 PUT /v1/user/{userName}/loginProfile HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
loginProfile	LoginProfile	RequestBody参数	控制台登录的相关配置	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

LoginProfile对象

请求示例

```
 PUT /v1/user/test-user/loginProfile HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 76
Authorization: AuthorizationString

{"password":"Pa$$word4Demo", "enabledLoginMfa":true, "loginMfaType":"PHONE" ,
"thirdPartyType":"PASSPORT", "thirdPartyAccount":"test-passportAccount"}
```

> 注意：若设置`enabledLoginMfa":true`，那么必须要指定`loginMfaType`。

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "enabledLogin": true,
    "enabledLoginMfa": true,
    "loginMfaType": "PHONE" ,
    "thirdPartyType":"PASSPORT",
    "thirdPartyAccount":"test-passportAccount"
}
```

查询控制台登录配置

接口描述

查询用户的控制台登录配置。

请求结构

```
 GET /v1/user/{userName}/loginProfile HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

LoginProfile对象

请求示例

```
 GET /v1/user/test-user/loginProfile HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "enabledLogin": true,
    "enabledLoginMfa": true,
    "loginMfaType": "PHONE",
    "thirdPartyType":"PASSPORT",
    "thirdPartyAccount":"test-passportAccount"
}
```

关闭控制台登录配置

接口描述

关闭用户的控制台登录配置，即关闭用户的控制台登录。

请求结构

```
 DELETE /v1/user/{userName}/loginProfile HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 DELETE /v1/user/test-user/loginProfile HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
 HTTP/1.1 200 OK
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

修改子用户操作保护

接口描述

修改子用户操作保护，即开启关闭子用户操作保护。

请求结构

```
POST /v1/user/operation/mfa/switch HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	RequestBody参数	需要更新的子用户名称	是
enabledMfa	Boolean	RequestBody参数	开启或关闭子用户操作保护	是
mfaType	String	RequestBody参数	子用户操作保护类型	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
POST /v1/user/operation/mfa/switch HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
{"userName":"test-username", "enabledMfa":true, "mfaType":"PHONE,TOTP"}
```

> 注意：若设置`"enabledMfa":true`，那么必须要指定`mfaType`。

响应示例

```
HTTP/1.1 200 OK
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

修改子用户密码

接口描述

修改子用户密码，不更新当前子用户登录控制台配置。

请求结构

```
PUT /v1/subUser/{userName}/update HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
password	String	RequestBody参数	更新后的密码	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

UserModel对象

请求示例

```
 PUT /v1/subUser/test-username/update HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
{
    "password": "passwordDemo@$$"
}
```

响应示例

```
 HTTP/1.1 200 OK
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "id": "f4225c39ba2247b692eec7b461835aa1",
    "name": test-username,
    "createTime": "2023-02-19T03:42:13Z",
    "enable":"true",
    "description":"test update sub user password"
}
```

创建用户的AccessKey

接口描述

为用户创建一组AccessKey访问密钥。

请求结构

```
 POST /v1/user/{userName}/accesskey HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

AccessKey对象

请求示例

```
 POST /v1/user/test-user/accesskey HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 201 Created
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "secret": "0f005450fd4e4b329050edc5955da477",
    "id": "e767c68b72194dba9ed71883c7b284b3",
    "createTime": "2019-06-06T07:33:51Z"
}
```

禁用用户的AccessKey

接口描述

将用户的指定一组AccessKey访问密钥设置为禁用状态。

请求结构

```
 PUT /user/{userName}/accesskey/{accessKeyId}?disable HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
accessKeyId	String	URL参数	访问密钥id	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

AccessKey对象

请求示例

```
 PUT /v1/user/test-user/accesskey/e767c68b72194dba9ed71883c7b284b3?disable HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
{
    "enabled": false,
    "id": "e767c68b72194dba9ed71883c7b284b3",
    "createTime": "2019-06-06T07:33:51Z"
}
```

启用用户的AccessKey

接口描述

将用户的指定一组AccessKey访问密钥恢复为启用状态。

请求结构

```
 PUT /user/{userName}/accesskey/{accessKeyId}?enable HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
accessKeyId	String	URL参数	访问密钥id	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

AccessKey对象

请求示例

```
 PUT /v1/user/test-user/accesskey/e767c68b72194dba9ed71883c7b284b3?enable HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
{
    "enabled": true,
    "id": "e767c68b72194dba9ed71883c7b284b3",
    "createTime": "2019-06-06T07:33:51Z"
}
```

删除用户的AccessKey

接口描述

删除用户的指定一组AccessKey访问密钥。

请求结构

```
 DELETE /user/{userName}/accesskey/{accessKeyId} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
accessKeyId	String	URL参数	访问密钥id	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

无

请求示例

```
 DELETE /v1/user/test-user/accesskey/e767c68b72194dba9ed71883c7b284b3 HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 204 No Content
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举用户的AccessKey

接口描述

列举用户的全部AccessKey访问密钥。

请求结构

```
 GET /v1/user/{userName}/accesskey HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
accessKeys	List<AccessKey>	用户访问密钥列表

请求示例

```
 GET /v1/user/test-user/accesskey HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
 HTTP/1.1 200 OK
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "accessKeys": [
        {
            "id": "e0eedb7fc40c45a597c4c80516a918bd",
            "createTime": "2019-06-06T07:37:21Z",
            "lastUsedTime": "2021-10-14T07:37:21Z",
            "enabled": true,
            "description": "测试"
        }
    ]
}
```

查询用户的AccessKey上次使用时间

接口描述

查询用户提供的AccessKey访问密钥上次使用时间。

> **说明**：拥有系统策略：IAMManageAccessKeyPolicy 权限的子用户可以查看自身AccessKey的上次使用时间；拥有系统策略：IAMFullControlAccessPolicy 、IAMReadAccessPolicy 权限的子用户可以查看所在账号下所有子用户AccessKey的上次使用时间

请求结构

```
 GET /v1/accesskey/{accessKeyId}/lastusedtime HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
accessKeyId	String	URL参数	访问密钥id	是

响应参数

名称	类型	描述
accessKeyId	String	访问密钥id
lastUsedTime	String	访问密钥id的上次使用时间

请求示例

```
 GET /v1/accesskey/e767c68b72194dba9ed71883c7b284b3/lastusedtime HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "accessKeyId": "e767c68b72194dba9ed71883c7b284b3",
    "lastUsedTime": "2024-10-25T08:27:43.000Z"
}
```

解绑子用户虚拟 MFA

接口描述

解绑子用户虚拟 MFA

请求结构

```
 DELETE /v1/user/{userName}/mfaType/{mfaType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
mfaType	String	URL参数	MFA 类型，固定为 TOTP	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

无

请求示例

```
 DELETE /v1/user/test2/mfaType/TOTP HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 204 No Content
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

查询IAM用户联合信息

接口描述

查询IAM用户联合信息，对应页面菜单：多用户访问控制->外部账号接入->IAM用户联合页面的内容。

请求结构

```
 GET /v1/subUser/idp/query HTTP/1.1
Host: iam.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

无

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
idp	Idp	当前账户IAM用户联合配置

请求示例

```
GET /v1/subUser/idp/query HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
HTTP/1.1 200 OK
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "status": "enable",
    "auxiliaryDomain": "www.baidu.com",
    "domainId": "91b4ee8f04f84c7dae51bc159c9cabcd",
    "encodeMetadata": "xxx",
    "fileName": "xxx.xml",
    "createTime": "2024-01-30T11:57:11.000Z",
    "updateTime": "2024-01-30T11:57:11.000Z"
}
```

更新IAM用户联合功能状态

接口描述

更新IAM用户联合功能状态。

请求结构

```
POST /v1/subUser/idp/updateStatus HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
status	String	RequestParam参数	功能状态，打开为enable，关闭为disable	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
idp	Idp	更新后的当前账户IAM用户联合配置

请求示例

```
POST /v1/subUser/idp/updateStatus?status=enable HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
 HTTP/1.1 200 OK
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "status": "enable",
    "auxiliaryDomain": "www.baidu.com",
    "domainId": "91b4ee8f04f84c7dae51bc159c9cabcd",
    "encodeMetadata": "xxx",
    "fileName": "xxx.xml",
    "createTime": "2024-01-30T11:57:11.000Z",
    "updateTime": "2024-01-30T11:57:11.000Z"
}
```

更新IAM用户联合功能配置

接口描述

更新IAM用户联合功能配置。

请求结构

```
 POST /v1/subUser/idp/update HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: authorization string
```

请求参数

名称	类型	位置	描述	是否必须
updateSubUserIdpRequest	UpdateSubUserIdpRequest	RequestBody参数	更新IAM用户联合功能配置请求体	是

请求头域

除公共头域外，无其它特殊头域。

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
idp	Idp	更新后的当前账户IAM用户联合配置

请求示例

```
 POST /v1/subUser/idp/update HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
 HTTP/1.1 200 OK
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "status": "enable",
    "auxiliaryDomain": "www.baidu.com",
    "domainId": "91b4ee8f04f84c7dae51bc159c9cabcd",
    "encodeMetadata": "xxx",
    "fileName": "xxx.xml",
    "createTime": "2024-01-30T11:57:11.000Z",
    "updateTime": "2024-01-30T11:57:11.000Z"
}
```

删除IAM用户联合配置

接口描述

删除IAM用户联合功能配置。

请求结构

```
 POST /v1/subUser/idp/delete HTTP/1.1
Host: iam.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

无

响应头域

除公共头域外，无其它特殊头域。

响应参数

无

请求示例

```
 POST /v1/subUser/idp/delete HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: authorization string
```

响应示例

```
 HTTP/1.1 200 OK
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

组管理接口

创建组

接口描述

创建组。

请求结构

```
 POST /v1/group HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
name	String	RequestBody参数	组名	是
description	String	RequestBody参数	组的描述	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

GroupModel对象

请求示例

```
 POST /v1/group HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 20
Authorization: AuthorizationString

{"name":"test_group"}
```

响应示例

```
 HTTP/1.1 201 Created
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "description": "",
    "id": "d3f6fd54fe3643459109ea0675c295a8",
    "createTime": "2019-06-06T08:18:34Z",
    "name": "test_group"
}
```

查询组

接口描述

查询组。

请求结构

```
GET /v1/group/{groupName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
groupName	String	URL参数	组名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

GroupModel对象

请求示例

```
GET /v1/group/test_group HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "description": "",
    "id": "d3f6fd54fe3643459109ea0675c295a8",
    "createTime": "2019-06-06T08:18:34Z",
    "name": "test_group"
}
```

更新组

接口描述

更新组。

请求结构

```
PUT /v1/group/{groupName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
groupName	String	URL参数	组名	是
name	String	RequestBody参数	更新后的组名	否
description	String	RequestBody参数	组的描述	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

GroupModel对象

请求示例

```
 PUT /v1/group/test_group HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 39
Authorization: AuthorizationString

{"description":"update for test group"}
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "description": "update for test group",
    "id": "d3f6fd54fe3643459109ea0675c295a8",
    "createTime": "2019-06-06T08:18:34Z",
    "name": "test_group"
}
```

删除组

接口描述

删除组。

> **说明**：删除组前需要将子用户从组内全部移除，并解除所有权限策略的关联。

请求结构

```
 DELETE /v1/group/{groupName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
groupName	String	URL参数	组名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 DELETE /v1/group/test_group HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 204 No Content
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举组

接口描述

列举组。

请求结构

```
 GET /v1/group HTTP/1.1
Host: iam.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

无。

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
groups	List<GroupModel>	组对象的列表

请求示例

```
 GET /v1/group HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "groups": [
        {
            "description": "update for test group",
            "id": "d3f6fd54fe3643459109ea0675c295a8",
            "createTime": "2019-06-06T08:18:34Z",
            "name": "test_group"
        }
    ]
}
```

添加用户到组

接口描述

添加用户到组。

请求结构

```
 PUT /v1/group/{groupName}/user/{userName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
groupName	String	URL参数	组名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 PUT /v1/group/test_group/user/test-user HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 204 No Content
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

从组内移除用户

接口描述

从组内移除用户。

请求结构

```
 DELETE /v1/group/{groupName}/user/{userName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
groupName	String	URL参数	组名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 DELETE /v1/group/test_group/user/test-user HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 204 No Content
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举用户的组

接口描述

列举用户所在的组。

请求结构

```
GET /v1/user/{userName}/group HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
groups	List<GroupModel>	组对象的列表

请求示例

```
GET /v1/user/test-user/group HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "groups": [
        {
            "description": "",
            "id": "5cf62eee890146ed8116f6b449268f33",
            "name": "test_group"
        }
    ]
}
```

列举组内用户

接口描述

列举组内的用户。

请求结构

```
GET /v1/group/{groupName}/user HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
groupName	String	URL参数	组名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
users	List<UserModel>	用户对象的列表

请求示例

```
 GET /v1/group/test_group/user
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
   "users": [
      {
         "id": "f4225c39ba2247b692eec7b461835aa1",
         "createTime": "2019-06-06T03:42:13Z",
         "name": "test-user"
      }
   ]
}
```

角色管理接口

创建角色

接口描述

创建角色。

请求结构

```
 POST /v1/role HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
name	String	RequestBody参数	角色名	是
description	String	RequestBody参数	角色的描述	否
assumeRolePolicyDocument	String	RequestBody参数	指定允许扮演角色的载体	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

RoleModel对象

请求示例

```
 POST /v1/user HTTP/1.1h
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 20
Authorization: Authorization String

{
 "name":"test-role",
 "description":"test role create",
 "assumeRolePolicyDocument": "{\"version\":\"v1\",\"accessControlList\":[{\"service\":\"bce:iam\",\"permission\":
[\"AssumeRole\"],\"region\":\"*\",\"grantee\":[{\"id\":\"grantee-id\"}],\"effect\":\"Allow\"}]}"
}
```

响应示例

```
 HTTP/1.1 201 Created
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
{
    "id": "9c039e6385854e07a5a31e254eab21ea",
    "createTime": "2019-06-06T03:42:13Z",
    "name": "test-role",
    "assumeRolePolicyDocument": "{\"id\":\"sts role acl\",\"version\":\"v1\",\"accessControlList\":
[{\"service\":\"bce:iam\",\"region\":\"*\",\"resource\":
[\"role/9c039e6385854e07a5a31e254eab21ea\"],\"grantee\":[{\"id\":\"grantee-
id\"}],\"effect\":\"Allow\",\"permission\":[\"AssumeRole\"],\"eid\":\"role acl entry\"}]}"
}
```

查询角色

接口描述

查询角色。

请求结构

```
 GET /v1/role/{roleName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
roleName	String	URL参数	角色名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

RoleModel对象

请求示例

```
 GET /v1/role/test-role HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "id": "9c039e6385854e07a5a31e254eab21ea",
    "createTime": "2019-06-06T03:42:13Z",
    "name": "test-role",
    "assumeRolePolicyDocument": "{\"id\":\"sts role acl\",\"version\":\"v1\",\"accessControlList\":
[{\"service\":\"bce:iam\",\"region\":\"*\",\"resource\":
[\"role/9c039e6385854e07a5a31e254eab21ea\"],\"grantee\":[{\"id\":\"grantee-
id\"}],\"effect\":\"Allow\",\"permission\":[\"AssumeRole\"],\"eid\":\"role acl entry\"}]}"
}
```

更新角色

接口描述

更新角色。

请求结构

```
 PUT /v1/role/{roleName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
roleName	String	URL参数	角色名	是
name	String	RequestBody参数	更新后的角色名	否
description	String	RequestBody参数	角色的描述	否
assumeRolePolicyDocument	String	RequestBody参数	指定可以扮演此角色的身份	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

RoleModel对象

请求示例

```
 PUT /v1/role/test-role HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 34
Authorization: AuthorizationString

{
 "description":"update role demo",
 "assumeRolePolicyDocument": "{\"version\":\"v1\",\"accessControlList\":[{\"service\":\"bce:iam\",\"permission\":
[\"AssumeRole\"],\"region\":\"*\",\"grantee\":[{\"id\":\"new-grantee-id\"}],\"effect\":\"Allow\"}]}"
 }
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "id": "9c039e6385854e07a5a31e254eab21ea",
    "createTime": "2019-06-06T03:42:13Z",
    "name": "test-role",
    "description": "update role demo",
    "assumeRolePolicyDocument": "{\"id\":\"sts role acl\",\"version\":\"v1\",\"accessControlList\":
[{\"service\":\"bce:iam\",\"region\":\"*\",\"resource\":
[\"role/9c039e6385854e07a5a31e254eab21ea\"],\"grantee\":[{\"id\":\"new-grantee-
id\"}],\"effect\":\"Allow\",\"permission\":[\"AssumeRole\"],\"eid\":\"role acl entry\"}]}"
}
```

删除角色

接口描述

删除角色。

请求结构

```
 DELETE /v1/role/{roleName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
roleName	String	URL参数	角色名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 DELETE /v1/role/test-role HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: Authorization String
```

响应示例

```
 HTTP/1.1 204 No Content
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举角色

接口描述

列举角色。

请求结构

```
 GET /v1/role HTTP/1.1
Host: iam.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

无

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
roles	List<RoleModel>	角色对象的列表

请求示例

```
 GET /v1/role HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: Authorization String
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "roles": [
        {
            "id": "9c039e6385854e07a5a31e254eab21ea",
            "createTime": "2019-06-06T03:42:13Z",
            "name": "test-role",
            “description": "update role demo",
       "assumeRolePolicyDocument": "{\"id\":\"sts role acl\",\"version\":\"v1\",\"accessControlList\":
[{\"service\":\"bce:iam\",\"region\":\"*\",\"resource\":
[\"role/9c039e6385854e07a5a31e254eab21ea\"],\"grantee\":[{\"id\":\"grantee-
id\"}],\"effect\":\"Allow\",\"permission\":[\"AssumeRole\"],\"eid\":\"role acl entry\"}]}"
        }
    ]
}
```

策略管理接口

> **说明**：权限策略分为自定义策略和系统策略。 自定义策略是用户自行创建管理的权限策略，类型为Custom；系统策略是云平台统一管理的内置策略，类型为System。 在下文API中，如未指定policyType参数，则默认为自定义策略类型。

创建策略

接口描述

创建权限策略。

请求结构

```
 POST /v1/policy HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
name	String	RequestBody参数	组名	是
description	String	RequestBody参数	组的描述	否
document	String	RequestBody参数	策略内容，ACL格式序列化后得到的String	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

PolicyModel对象

请求示例

```
 POST /v1/policy HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 20
Authorization: AuthorizationString

{"name":"test_policy", "document":"{\"accessControlList\": [{\"region\":\"bj\",\"service\":\"bcc\",\"resource\":
[\"*\"],\"permission\":[\"*\"],\"effect\":\"Allow\"}]}"}
```

响应示例

```
 HTTP/1.1 201 Created
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "description": "",
    "createTime": "2019-06-06T09:13:50Z",
    "document": "{\"id\":\"policy_d19f78b0595242b5a8c3419c09c81b40\",\"accessControlList\":
[{\"service\":\"bcc\",\"region\":\"bj\",\"resource\":[\"*\"],\"effect\":\"Allow\",\"permission\":[\"*\"]}]}",
    "type": "Custom",
    "id": "d19f78b0595242b5a8c3419c09c81b40",
    "name": "test_policy"
}
```

更新策略

接口描述

仅支持更按【策略语法自定义创建】的权限策略。

请求结构

```
 POST /v1/policy/{policyName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
policyName	String	url参数	原策略名	是
name	String	RequestBody参数	新策略名	否
description	String	RequestBody参数	策略的描述	否
document	String	RequestBody参数	策略内容，ACL格式序列化后得到的String	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

PolicyModel对象

请求示例

```
 POST /v1/policy/test_policy HTTP/1.1
Host: iam.bj.baidubce.com
content-type: application/json
Content-Length: 20
Authorization: AuthorizationString

{"name":"test_policy", "document":"{\"accessControlList\": [{\"region\":\"bj\",\"service\":\"bcc\",\"resource\":
[\"*\"],\"permission\":[\"*\"],\"effect\":\"Allow\"}]}"}
```

响应示例

```
 HTTP/1.1 200
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "description": "",
    "createTime": "2019-06-06T09:13:50Z",
    "document": "{\"id\":\"policy_d19f78b0595242b5a8c3419c09c81b40\",\"accessControlList\":
[{\"service\":\"bcc\",\"region\":\"bj\",\"resource\":[\"*\"],\"effect\":\"Allow\",\"permission\":[\"*\"]}]}",
    "type": "Custom",
    "id": "d19f78b0595242b5a8c3419c09c81b40",
    "name": "test_policy"
}
```

查询策略

接口描述

查询权限策略。

请求结构

```
 GET /v1/policy/{policyName}?policyType={policyType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
policyName	String	URL参数	策略名	是
policyType	String	Query参数	要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

PolicyModel对象

请求示例

```
 GET /v1/policy/test_policy HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "description": "",
    "createTime": "2019-06-06T09:13:50Z",
    "document": "{\"id\":\"policy_d19f78b0595242b5a8c3419c09c81b40\",\"accessControlList\":
[{\"service\":\"bcc\",\"region\":\"bj\",\"resource\":[\"*\"],\"effect\":\"Allow\",\"permission\":[\"*\"]}]}",
    "type": "Custom",
    "id": "d19f78b0595242b5a8c3419c09c81b40",
    "name": "test_policy"
}
```

删除策略

接口描述

删除权限策略。

> **说明**：删除组前需要需要先解除到该策略的全部权限关联。

请求结构

```
 DELETE /v1/policy/{policyName} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
policyName	String	URL参数	组名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 DELETE /v1/policy/test_policy HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 204 No Content
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举策略

接口描述

列举权限策略。

当policyType为System时，可列举系统内置的策略列表。

请求结构

```
 GET /v1/policy?policyType={policyType}&nameFilter=${nameFilter} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
policyType	String	Query参数	要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略	否
nameFilter	String	Query参数	策略名字的过滤条件；不为空时仅返回名字包含此关键字的结果	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
policies	List<PolicyModel>	策略对象的列表

请求示例

```
 GET /v1/policy HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "policies": [
        {
            "description": "",
            "createTime": "2019-06-06T09:13:50Z",
            "document": "{\"id\":\"policy_d19f78b0595242b5a8c3419c09c81b40\",\"accessControlList\":
[{\"service\":\"bcc\",\"region\":\"bj\",\"resource\":[\"*\"],\"effect\":\"Allow\",\"permission\":[\"*\"]}]}",
            "type": "Custom",
            "id": "d19f78b0595242b5a8c3419c09c81b40",
            "name": "test_policy"
        }
    ]
}
```

关联用户权限

接口描述

为用户关联权限策略。

请求结构

```
 PUT /v1/user/{userName}/policy/{policyName}?policyType={policyType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
policyName	String	URL参数	策略名	是
policyType	String	Query参数	要关联的策略类型，为System时查询系统策略；为Custom时查询自定义策略	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 PUT /v1/user/test-user/policy/test_policy
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

解除用户权限

接口描述

解除子用户关联的权限策略。

请求结构

```
 DELETE /v1/user/{userName}/policy/{policyName}policyType={policyType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是
policyName	String	URL参数	策略名	是
policyType	String	Query参数	关联的策略类型，为System时查询系统策略；为Custom时查询自定义策略	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 DELETE /v1/user/test-user/policy/test_policy
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 204 No Content
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举用户的权限

接口描述

列举用户关联的权限策略。

请求结构

```
GET /v1/user/{userName}/policy HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
userName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
policies	List<PolicyModel>	策略对象的列表

请求示例

```
GET /v1/user/test-user/policy HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "policies": [
        {
            "description": "",
            "createTime": "2019-06-06T09:13:50Z",
            "document": "{\"id\":\"policy_d19f78b0595242b5a8c3419c09c81b40\",\"accessControlList\":
[{\"service\":\"bcc\",\"region\":\"bj\",\"resource\":[\"*\"],\"effect\":\"Allow\",\"permission\":[\"*\"]}]}",
            "type": "Custom",
            "id": "d19f78b0595242b5a8c3419c09c81b40",
            "name": "test_policy"
        }
    ]
}
```

关联组权限

接口描述

为用户组关联权限策略。

请求结构

```
PUT /v1/group/{groupName}/policy/{policyName}?policyType={policyType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
groupName	String	URL参数	组名	是
policyName	String	URL参数	策略名	是
policyType	String	Query参数	要关联的策略类型，为System时查询系统策略；为Custom时查询自定义策略	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
PUT /v1/group/test_group/policy/test_policy
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

解除组权限

接口描述

解除用户组关联的权限策略。

请求结构

```
DELETE /v1/group/{groupName}/policy/{policyName}policyType={policyType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
groupName	String	URL参数	组名	是
policyName	String	URL参数	策略名	是
policyType	String	Query参数	关联的策略类型，为System时查询系统策略；为Custom时查询自定义策略	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
 DELETE /v1/group/test_group/policy/test_policy
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 204 No Content
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举组的权限

接口描述

列举用户组关联的权限策略。

请求结构

```
 GET /v1/group/{groupName}/policy HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
groupName	String	URL参数	用户名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
policies	List<PolicyModel>	策略对象的列表

请求示例

```
 GET /v1/group/test_group/policy HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "policies": [
        {
            "description": "",
            "createTime": "2019-06-06T09:13:50Z",
            "document": "{\"id\":\"policy_d19f78b0595242b5a8c3419c09c81b40\",\"accessControlList\":
[{\"service\":\"bcc\",\"region\":\"bj\",\"resource\":[\"*\"],\"effect\":\"Allow\",\"permission\":[\"*\"]}]}",
            "type": "Custom",
            "id": "d19f78b0595242b5a8c3419c09c81b40",
            "name": "test_policy"
        }
    ]
}
```

关联角色权限

接口描述

为角色关联权限策略。

请求结构

```
 PUT /v1/role/{roleName}/policy/{policyName}?policyType={policyType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
roleName	String	URL参数	角色名	是
policyName	String	URL参数	策略名	是
policyType	String	Query参数	要关联的策略类型，为System时查询系统策略；为Custom时查询自定义策略	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
PUT /v1/role/test-role/policy/test_policy
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

解除角色权限

接口描述

解除角色关联的权限策略。

请求结构

```
DELETE /v1/role/{roleName}/policy/{policyName}?policyType={policyType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
roleName	String	URL参数	角色名	是
policyName	String	URL参数	策略名	是
policyType	String	Query参数	关联的策略类型，为System时查询系统策略；为Custom时查询自定义策略	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

无。

请求示例

```
DELETE /v1/role/test-role/policy/test_policy
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
HTTP/1.1 204 No Content
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS
```

列举角色的权限

接口描述

列举角色关联的权限策略。

请求结构

```
 GET /v1/role/{roleName}/policy HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
roleName	String	URL参数	角色名	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
policies	List<PolicyModel>	策略对象的列表

请求示例

```
 GET /v1/role/test-role/policy HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "policies": [
        {
            "description": "",
            "createTime": "2019-06-06T09:13:50Z",
            "document": "{\"id\":\"policy_d19f78b0595242b5a8c3419c09c81b40\",\"accessControlList\":
[{\"service\":\"bcc\",\"region\":\"bj\",\"resource\":[\"*\"],\"effect\":\"Allow\",\"permission\":[\"*\"]}]}",
            "type": "Custom",
            "id": "d19f78b0595242b5a8c3419c09c81b40",
            "name": "test_policy"
        }
    ]
}
```

列举权限授予的主体

接口描述

列举权限策略被授予的主体。

请求结构

```
GET /v1/policy/{policyId}/grant/{grantType} HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
policyId	String	URL参数	策略ID	是
grantType	String	URL参数	授予类型, UserPolicy或GroupPolicy	是

响应头域

除公共头域外，无其它特殊头域。

响应参数

名称	类型	描述
entities	List	策略被授予主体对象的列表
id	String	主体 id
name	String	主体名称
type	String	UserPolicy或 GroupPolicy
attach_time	DateTime	策略被授予时间

请求示例

```
GET /v1/policy/8e7cb46773944da8aa3b351e1dfe1c59/grant/UserPolicy HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "entities": [
        {
            "id": "4b284b9ea9724fd4af7c45986a447876",
            "name": "test10",
            "type": "UserPolicy",
            "attach_time": "2023-10-11T10:05:14.000Z"
        }
    ]
}
```

列举权限授予的所有主体

接口描述

列举权限策略被授予的所有主体。

请求结构

```
 GET /v1/policy/{policyId}/entity HTTP/1.1
Host: iam.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

名称	类型	位置	描述	是否必须
policyId	String	URL参数	策略ID	是

响应参数

名称	类型	描述
entities	List	策略被授予主体对象的列表
id	String	主体 id
name	String	主体名称
type	String	UserPolicy、GroupPolicy、RolePolicy
attach_time	DateTime	策略被授予时间

请求示例

```
 GET /v1/policy/8e7cb46773944da8aa3b351e1dfe1c59/entity HTTP/1.1
Host: iam.bj.baidubce.com
Authorization: AuthorizationString
```

响应示例

```
 HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
X-Bce-Request-Id: 8d51a788-e79d-4ced-a9e0-0662ec85b7f7
Server: BWS

{
    "entities": [
        {
            "id": "36b6fb0121a84c65b437fe7df0341234",
            "name": "testUser",
            "type": "UserPolicy",
            "attachTime": "2024-04-22T10:57:08.000Z"
        },
        {
            "id": "fbe4f761c3f340c3a29022c62f042345",
            "name": "testGroup",
            "type": "GroupPolicy",
            "attachTime": "2024-04-22T10:57:22.000Z"
        },
        {
            "id": "91a247bb83ec44b0950ee452db9f4567",
            "name": "testRole",
            "type": "RolePolicy",
            "attachTime": "2024-04-22T10:57:32.000Z"
        }
    ]
}
```

数据类型

Model对象定义

UserModel

参数名称	类型	描述
id	String	用户id
name	String	用户名称
createTime	DateTime	创建时间
description	String	用户描述
enabled	Boolean	用户启用状态

LoginProfile

参数名称	类型	描述
password	String	用户密码；作为响应时不显示
needResetPassword	Boolean	下次登录时是否需要重置密码
enabledLogin	Boolean	是否允许子用户控制台登录
enabledLoginMfa	Boolean	是否要求绑定二次验证设备
loginMfaType	enum	二次验证类型，可选：PHONE-手机号，TOTP虚拟MFA设备
thirdPartyType	enum	绑定的第三方登录类型，可选：UUAP-内网账号，PASSPORT-百度账号
thirdPartyAccount	String	绑定的第三方登录账号。绑定类型为PASSPORT时可以是手机、邮箱以及账号名称

说明：thirdPartyType与thirdPartyAccount均为空字符串时解除绑定

AccessKey

参数名称	类型	描述
id	String	AccessKey的公开Id，即AK
secret	String	AccessKey的密钥，即SK
createTime	DateTime	AccessKey的创建时间
description	String	AccessKey的描述

GroupModel

参数名称	类型	描述
id	String	组id
name	String	组名称
createTime	DateTime	创建时间
description	String	组描述

PolicyModel

参数名称	类型	描述
id	String	策略id
name	String	策略名称
type	String	策略类型，可选：Custom - 自定义策略；System - 系统内置策略
createTime	DateTime	创建时间
description	String	策略描述
document	String	策略内容，要求为ACL格式序列化后得到的String

RoleModel

参数名称	类型	描述
id	String	角色id
name	String	角色名称
createTime	DateTime	创建时间
description	String	角色描述
assumeRolePolicyDocument	String	指定允许扮演此角色的实体

Idp

参数名称	类型	描述
status	String	用户联合功能状态，开启状态返回enable，关闭状态为disable
domainId	String	账户ID
encodeMetadata	String	Base64编码后的IdP元数据
fileName	String	IdP元数据文件名称
auxiliaryDomain	String	辅助域名

UpdateSubUserIdpRequest

参数名称	类型	描述
fileName	String	文件名称，必须为xml格式文件
encodeMetadata	String	Base64编码后的IdP元数据
auxiliaryDomain	String	辅助域名

TokenResponse（原BearTokenResponse）

参数名称	类型	描述
userId	String	用户id
token	String	短期API Key
status	enum	升级至TokenResponse后不再返回 Token状态，可选： DISABLE-禁用 ENABLE -启用
createTime	DateTime	创建时间，utc时区，tz格式
expireTime	DateTime	过期时间，utc时区，tz格式

API参考（STS）

简介

概述

多用户访问控制IAM (Identity and Access Management)是由百度智能云提供的身份管理和访问控制服务。IAM帮助智能云用户解决身份标识、验证、鉴权和授权等问题，通过定义谁(身份)在什么条件下从哪里可以对云端什么资源做何种操作，来控制用户对你的云资源访问。你可以通过云控制台，使用用户名密码的方式访问IAM，也可以通过本文档提供的符合REST API规范的接口来进行身份管理和访问控制。

接口概览

本节汇总了多用户访问控制IAM当前可调用API，具体接口信息请点击链接查看详细内容。

接口	描述
代入角色	又称AssumeRole，通过此 API 切换到用户需要使用的目标角色，返回一组关联到指定角色的临时身份凭证。
获取会话令牌	通过此接口返回主账户的临时身份凭证。

产品限制

目前产品仅**白名单**开放，需要申请试用，请联系百度智能云IAM团队。

通用说明

实名认证

百度智能云提供个人认证、企业认证两种认证方式，您可以选择任意一种方式进行认证。

认证机制

所有API的安全认证一律采用Access Key与请求签名机制。 Access Key由Access Key ID和Secret Access Key组成，均为字符串。 对于每个HTTP请求，使用下面所描述的算法生成一个认证字符串。提交认证字符串放在Authorization头域里。服务端根据生成算法验证认证字符串的正确性。 认证字符串的格式为bce-auth-v{version}/{accessKeyId}/{timestamp}/{expirationPeriodInSeconds}/{signedHeaders}/{signature}`。

- version是正整数。
- timestamp是生成签名时的UTC时间。
- expirationPeriodInSeconds表示签名有效期限。
- signedHeaders是签名算法中涉及到的头域列表。头域名之间用分号（;）分隔，如host;x-bce-date。列表按照字典序排列。（本API签名仅使用host和x-bce-date两个header）
- signature是256位签名的十六进制表示，由64个小写字母组成。

当百度智能云接收到用户的请求后，系统将使用相同的SK和同样的认证机制生成认证字符串，并与用户请求中包含的认证字符串进行比对。如果认证字符串相同，系统认为用户拥有指定的操作权限，并执行相关操作；如果认证字符串不同，系统将忽略该操作并返回错误码。鉴权认证机制的详细内容请参见鉴权认证机制。

通信协议

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。

请求结构说明

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

请求参数包括如下4种：

参数类型	说明
URI	用于指明操作实体，如:PUT /v1/cluster/{clusterUuid}
Query参数	URL中携带的请求参数
HEADER	通过HTTP头域传入，如:x-bce-date
RequestBody	通过JSON格式组织的请求数据体

响应结构说明

响应值分为两种形式：

响应内容	说明
HTTP STATUS CODE	如200,400,403,404等
ResponseBody	JSON格式组织的响应数据体

版本号

参数	类型	参数位置	描述	是否必须
v1	String	URI参数	API版本号	必须

幂等性

当调用创建接口时如果遇到了请求超时或服务器内部错误，用户可能会尝试重发请求，这时用户通过clientToken参数避免创建出比预期要多的资源，即保证请求的幂等性。

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

如果用户使用同一个clientToken值调用创建接口，则服务端会返回相同的请求结果。因此用户在遇到错误进行重试的时候，可以通过提供相同的clientToken值，来确保只创建一个资源；如果用户提供了一个已经使用过的clientToken，但其他请求参数（包括queryString和requestBody）不同甚至url Path不同，则会返回IdempotentParameterMismatch的错误代码。

clientToken的有效期为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。

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。

规范化字符串

通常一个字符串中可以包含任何Unicode字符。在编程中这种灵活性会带来不少困扰。因此引入“规范字符串”的概念。一个规范字符串只包含百分号编码字符以及URI（Uniform Resource Identifier）非保留字符（Unreserved Characters）。 RFC 3986规定URI非保留字符包括以下字符：字母（A-Z，a-z）、数字（0-9）、连字号（-）、点号（.）、下划线（_）、波浪线（~）。 将任意一个字符串转换为规范字符串的方式是:

- 将字符串转换成UTF-8编码的字节流。
- 保留所有URI非保留字符原样不变。
- 对其余字节做一次RFC 3986中规定的百分号编码（Percent-Encoding），即一个%后面跟着两个表示该字节值的十六进制字母。字母一律采用大写形式。

示例： 原字符串：`this is an example for 测试`，对应的规范字符串：`this%20is%20an%20example%20for%20%E6%B5%8B%E8%AF%95`。

服务域名

区域	服务端点Endpoint	协议
北京	sts.bj.baidubce.com	HTTP and HTTPS

公共请求头与公共响应头

公共请求头

头域	说明	是否必须
Authorization	包含Access Key与请求签名。	必须
x-bce-date	该请求创建的时间，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。如果用户使用了标准的Date域，该头域可以不填。当两者同时存在时，以x-bce-date为准。	可选
x-bce-content-sha256	表示内容部分的SHA256签名的十六进制字符串，其中内容指HTTP Request Payload Body，即Content部分在被HTTP encode之前的原始数据。	可选

公共响应头

头域	说明
Content-Length	RFC2616中定义的HTTP请求内容的类型。
x-bce-request-id	对应请求的requestId。

错误码

错误码格式

当用户访问API出现错误时，会返回给用户相应的错误码和错误信息，便于定位问题，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	表示具体错误类型。
message	String	有关该错误的详细说明。
requestId	String	导致该错误的requestId。

例如：

```
 {
   "code":"IllegalRequestUrl",
   "message":"The requested url belongs to domain which is not under acceleration",
   "requestId":" 81d0b05f-5ad4-1f22-8068-d5c9de60a1d7"
}
```

公共错误码

错误码	错误消息	HTTP状态码	描述
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。
InternalError	We encountered an internal error. Please try again.	500Internal 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。
InvalidHTTPAuthHeader	The HTTP authorization header is invalid. Consult the service documentation for details.	400 Bad Request	Authorization头域格式错误。
InvalidHTTPRequest	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。
		400	

MalformedJSON	The JSON you provided was not well-formed.	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	详见ETag。
RequestExpired	Request has expired. Timestamp date is XXX.	400 Bad Request	请求超时。XXX要改成x-bce-date的值。如果请求中只有Date，则需要将Date转换为datetime。
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。

STS 错误码

错误码	错误消息	HTTP状态码	描述
InvalidRequestBody	Could not read document	400	请求的request body不是有效的json。
NotFound	could not found:accessKeyId.	404	请求的AccessKeyId或user/role不存在。
BadRequest	missing required query_string parameter:	400	请求不合法：缺少必填的参数、参数超限等等
Forbidden	you have no permission to assume role	403	没有权限发起请求

STS相关接口

目前STS相关接口仅限**白名单开放**，需要申请试用，请联系百度智能云IAM团队。

AssumeRole

接口描述

获取关联到指定角色的临时身份凭证。

权限说明

请求发起人需要具有合法的AccessKeyID和SecretAccessKey才能发起请求，请参考鉴权认证。

请求结构

```
POST /v1/credential?assumeRole&accountId=dc9b5191440d4f93851ddffb4e942b75&roleName=testRole HTTP/1.1
Content-Type: application/json
Authorization: bce-auth-v1/04795b2a1d12490bbee94511b7b78d7e/2016-11-11T10:34:26Z/1800/host/9e5619677b3a461f5b95d061b88823634e8bbc0e6b46e5b3cecbb75f21906a3d
Host: sts.bj.baidubce.com
```

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	描述	参数位置	是否必须
durationSeconds	String	临时AK的有效期，不能超过2h	Query参数	否
accountId	String	角色所在的账户的Id	Query参数	是
userId	String	用户的Id，非必填普通场景无需关注	Query参数	否
roleName	String	角色的名称	Query参数	是
authorization	String	认证字符串，参考生成认证字符串	header	是
accessControlList	String	为临时身份凭证绑定的权限	body	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

参数名称	类型	描述	参数位置
credential	String	生成的临时身份凭证	body

请求示例

```
POST /v1/credential?assumeRole&accountId=dc9b5191440d4f93851ddffb4e942b75&roleName=testRole HTTP/1.1
Content-Type: application/json
Authorization: bce-auth-v1/04795b2a1d12490bbee94511b7b78d7e/2016-11-11T10:34:26Z/1800/host/9e5619677b3a461f5b95d061b88823634e8bbc0e6b46e5b3cecbb75f21906a3d
Host: sts.bj.baidubce.com
```

响应示例

```
{
  "accessKeyId": "785e811dec6a47308ce6da241b25b501",
  "secretAccessKey": "5a57d543676d46b9952e7354e3ac1060",
  "sessionToken":
"NmViZDI1Nzg1NTM0NGExNmFlYzM2YjJiOGFkNDc2MWZ8AAAAAFgBAAD6a2SBhmOXBRCNraVyOShGIR5/j+CE4V5VLv9zmFM5
",
  "createTime": "2016-11-11T10:34:53Z",
  "expiration": "2016-11-11T11:34:53Z",
  "userId": "dc9b5191440d4f93851ddffb4e942b75",
  "roleId": "aed8e56613f24e0e97597c1b0d29fcba"
}
```

GetSessionToken

接口描述

返回主账户的临时身份凭证。

权限说明

请求发起人需要具有合法的AccessKeyID和SecretAccessKey才能发起请求，请参考鉴权认证。

请求结构

```
 POST /v1/sessionToken?durationSeconds=43200 HTTP/1.1
Content-Type: application/json
Authorization: bce-auth-v1/c1b351e39aaa464d9ee9038ff6cea440/2016-12-
07T02:56:37Z/1800/host/3a55417c1f94c47829ca935103f5a3686bacfefddd80d18608dcabe4ade1d63e
Host: sts.bj.baidubce.com
```

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	描述	参数位置	是否必须
durationSeconds	String	临时AK的有效期，不能超过36h，默认为43200（12h）	Query参数	否
authorization	String	主账户的认证字符串	header	是
accessControlList	String	为临时身份凭证绑定的权限	body	否
attachment	String	业务方绑定在credential上的一些信息	body	否

响应头域

除公共头域外，无其它特殊头域。

响应参数

参数名称	类型	描述	参数位置
credential	String	生成的临时身份凭证	body

请求示例

```
 POST /v1/sessionToken?durationSeconds=43200 HTTP/1.1
Content-Type: application/json
Authorization: bce-auth-v1/c1b351e39aaa464d9ee9038ff6cea440/2016-12-
07T02:56:37Z/1800/host/3a55417c1f94c47829ca935103f5a3686bacfefddd80d18608dcabe4ade1d63e
Host: sts.bj.baidubce.com
```

响应示例

```
 {
  “accessKeyId”: “1eccc6374fde485085c127d52a65b15e”,
  “secretAccessKey”: “30839b659fb74f1cbd9c581c07e68e33”,
  “sessionToken”:
“NmViZDI1Nzg1NTM0NGExNmFlYzM2YjJiOGFkNDc2MWZ8AAAAADgBAADs/9cMh8x/tKTMErRX40h3MVvz0kaH2PM5rCoVhryOI
,
  “createTime”: “2016-12-07T02:58:05Z”,
  “expiration”: “2016-12-07T14:58:05Z”,
  “userId”: “dc9b5191440d4f93851ddffb4e942b75”
}
```

数据类型

返回参数对象定义

credentials

参数名称	类型	描述
accessKeyId	String	返回临时ak
accessKeySecret	String	返回临时sk
sessionToken	String	session token,用户使用临时sk签名请求时必须携带
expiration	datetime	临时身份凭证的失效时间
userId	String	所关联的用户的Id（GetSessionToken返回）
roleId	String	所关联的角色的Id（AssumeRole返回）

功能更新记录

2019-04-30

- 开放AssumeRole、GetSessionToken接口内测

SDK

Java-SDK

概述

本文档主要介绍IAM Java SDK的安装和使用。在使用本文档前，您需要先了解IAM的一些基本知识。若您还不了解IAM，可以参考产品描述和快速入门。

安装SDK工具包

运行环境

Java SDK工具包可在JDK1.7、JDK1.8环境下运行。

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SDK下载页面找到。

方式二：直接使用JAR包安装

1. 下载最新版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

```
com.baidubce
├─── auth                          //BCE签名相关类
├─── http                          //BCE的Http通信相关类
├─── internal                      //SDK内部类
├─── model                         //BCE公用model类
├─── services
│    └── iam                      //iam服务相关类
│        ├── model                //iam内部model，如Request或Response
│        ├── IamClient.class      //iam客户端入口类
│        └── api                  //iam api配置，不需要修改
├─── util                          //BCE公用工具类
├─── BceClientConfiguration.class  //对BCE的HttpClient的配置
├─── BceClientException.class      //BCE客户端的异常类
├─── BceServiceException.class     //与BCE服务端交互后的异常类
├─── ErrorCode.class               //BCE通用的错误码
└─── Region.class                  //BCE提供服务的区域
```

卸载SDK

预期卸载 SDK 时，删除 pom 依赖或源码即可。

初始化

确认Endpoint

百度智能云目前开放了多区域支持，请参考区域选择说明。

目前只支持“华北-北京”一个区域。 北京区域：http://iam.bj.baidubce.com 对应信息为：

访问区域	对应Endpoint
BJ	iam.bj.baidubce.com

获取密钥

要使用百度智能云IAM，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IAM做签名验证。 可以通过如下步骤获得并了解您的AK/SK信息：

注册百度智能云账号

创建AK/SK

新建IamClient

IamClient是IAM服务的客户端，为开发者与IAM服务进行交互提供了一系列的方法。

使用AK/SK新建IamClient

通过AK/SK方式访问IAM，用户可以参考如下代码新建一个IamClient：

```
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IamClient
    IamClient client = new IamClient(ACCESS_KEY_ID, SECRET_ACCESS_KEY);
  }
}
```

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》。

配置IamClient

如果用户需要配置IamClient的一些细节的参数，可以在构造IamClient的时候传入BceClientConfiguration对象。BceClientConfiguration是IAM服务的配置类，可以为客户端配置代理，最大连接数等参数。

使用代理

下面一段代码可以让客户端使用代理访问IAM服务：

```
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

// 创建BceClientConfiguration实例
BceClientConfiguration config = new BceClientConfiguration();

// 配置代理为本地8080端口
config.setProxyHost("127.0.0.1");
config.setProxyPort(8080);

// 创建IAM客户端
config.setCredentials(new DefaultBceCredentials(ACCESS_KEY_ID,SECRET_ACCESS_KEY));
config.setEndpoint(ENDPOINT);
IamClient client = new IamClient(config);
```

使用上面的代码段，客户端的所有操作都会通过127.0.0.1地址的8080端口做代理执行。

对于有用户验证的代理，可以通过下面的代码段配置用户名和密码：

```
// 创建BceClientConfiguration实例
BceClientConfiguration config = new BceClientConfiguration();

// 配置代理为本地8080端口
config.setProxyHost("127.0.0.1");
config.setProxyPort(8080);

//设置用户名和密码
config.setProxyUsername(<username>);                        //用户名
config.setProxyPassword(<password>);                        //密码
```

设置网络参数

用户可以用BceClientConfiguration对基本网络参数进行设置：

```
 BceClientConfiguration config = new BceClientConfiguration();

// 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BceClientConfiguration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

用户管理接口

创建用户

支持子用户的创建，请参考如下代码：

```
  public void createUser(IamClient client) {
    CreateUserRequest createUserRequest = new CreateUserRequest();
    //设置用户名 长度1-64位的字母、数字或"_"
    createUserRequest.setName("test_user_name");
    //设置用户描述
    createUserRequest.setDescription("test_user_description");

    client.createUser(createUserRequest);
  }
```

查询用户

子用户查询，请参考如下代码：

```
public void getUser(IamClient client) {
  //设置用户名
  String userName = "test_user_name";

  client.getUser(userName);
}
```

更新用户

支持子用户的创建，请参考如下代码：

```
public void updateUser(IamClient client) {
  // 目前的用户名
  String userName = "test_user_name";
  // 设置更新的用户信息
  UpdateUserRequest updateUserRequest = new UpdateUserRequest();
  // 设置更新后的用户名
  updateUserRequest.setName("new_user_name");
  // 设置更新后的用户描述
  updateUserRequest.setDescription("new_user_description");

  client.updateUser(userName, updateUserRequest);
}
```

删除用户

支持子用户的删除，请参考如下代码：

```
public void deleteUser(IamClient client) {
  // 目前的用户名
  String userName = "test_user_name";

  client.deleteUser(userName);
}
```

列举用户

列举用户，请参考如下代码：

```
public void listUser(IamClient client) {
  client.listUser();
}
```

配置用户的控制台登录

配置用户的控制台登录，请参考如下代码：

```
public void updateLoginProfile(IamClient client) {
    // 用户名
    String userName = "test_user_name";
    UpdateLoginProfileRequest updateLoginProfileRequest = new UpdateLoginProfileRequest();
    // 设置用户密码；作为响应时不显示
    updateLoginProfileRequest.setPassword("new_passwd");
    // 下次登录时是否需要重置密码
    updateLoginProfileRequest.setNeedResetPassword(true);
    // 是否要求绑定二次验证设备
    updateLoginProfileRequest.setEnabledLoginMfa(true);
    // 二次验证类型，可选：PHONE-手机号，TOTP虚拟MFA设备
    updateLoginProfileRequest.setLoginMfaType("PHONE");
    // 绑定的第三方登录类型，可选：UUAP-内网账号，PASSPORT-百度账号
    updateLoginProfileRequest.setThirdPartyType("UUAP");
    // 绑定的第三方登录账号。绑定类型为PASSPORT时可以是手机、邮箱以及账号名称
    updateLoginProfileRequest.setThirdPartyAccount("PASSPORT");
    client.updateLoginProfile(userName，updateLoginProfileRequest)
}
```

> 说明：1、`thirdPartyType`与`thirdPartyAccount`均为空字符串时解除绑定；2、若设置`enabledLoginMfa":true`，那么必须要指定`loginMfaType`

查询控制台登录配置

查询用户的控制台登录配置，请参考如下代码：

```
public void getLoginProfile(IamClient client) {
    // 用户名
    String userName = "test_user_name";
    client.getLoginProfile(userName);
}
```

关闭控制台登录配置

关闭用户的控制台登录配置，即关闭用户的控制台登录，请参考如下代码：

```
public void deleteLoginProfile(IamClient client) {
    // 用户名
    String userName = "test_user_name";
    client.deleteLoginProfile(userName);
}
```

创建用户的AccessKey

为用户创建一组AccessKey访问密钥，请参考如下代码：

```
public void createAccessKey(IamClient client) {
    // 用户名
    String userName = "test_user_name";

    client.createAccessKey(userName);
}
```

禁用用户的AccessKey

将用户的指定一组AccessKey访问密钥设置为禁用状态，请参考如下代码：

```
public void disableAccessKey(IamClient client) {
    // 用户名
    String userName = "test_user_name";
    // ak
    String accessKeyId = "test_user_ak";

    client.disableAccessKey(userName, accessKeyId);
}
```

启用用户的AccessKey

将用户的指定一组AccessKey访问密钥恢复为启用状态，请参考如下代码：

```
public void enableAccessKey(IamClient client) {
    // 用户名
    String userName = "test_user_name";
    // ak
    String accessKeyId = "test_user_ak";

    client.enableAccessKey(userName, accessKeyId);
}
```

删除用户的AccessKey

删除用户的指定一组AccessKey访问密钥，请参考如下代码：

```
public void deleteAccessKey(IamClient client) {
    // 用户名
    String userName = "test_user_name";
    // ak
    String accessKeyId = "test_user_ak";

    client.deleteAccessKey(userName, accessKeyId);
}
```

列举用户的AccessKey

列举用户的全部AccessKey访问密钥，请参考如下代码：

```
public void listAccessKey(IamClient client) {
    // 用户名
    String userName = "test_user_name";

    client.listAccessKey(userName);
}
```

角色管理接口

创建角色

创建角色，请参考如下代码：

```
  public void createRole(IamClient client) {

    CreateRoleRequest createRoleRequest = new CreateRoleRequest();
    // 设置角色名
    createRoleRequest.setName("test_role_name");
    // 设置角色描述
    createRoleRequest.setDescription("test_description");
    // 指定允许扮演角色的载体
    createRoleRequest.setAssumeRolePolicyDocument("{\"version\":\"v1\",\"accessControlList\":
[{\"service\":\"bce:iam\"," +
            "\"permission\":[\"AssumeRole\"],\"region\":\"*\",\"grantee\":[{\"id\":\"grantee-
id\"}],\"effect\":\"Allow\"}]}");

    client.createRole(createRoleRequest);
  }
```

查询角色

查询角色，请参考如下代码：

```
  public void getRole(IamClient client) {
    // 角色名
    String roleName = "test_role_name";
    client.getRole(roleName);
  }
```

更新角色

更新角色，请参考如下代码：

```
  public void updateRole(IamClient client) {
    // 旧的角色名
    String roleName = "old_role_name";
    UpdateRoleRequest updateRoleRequest = new UpdateRoleRequest();
    // 设置更新的角色名
    updateRoleRequest.setName("new_role_name");
    // 更新的角色描述
    updateRoleRequest.setDescription("new_role_description");
    // 更新的允许扮演角色的载体
    updateRoleRequest.setAssumeRolePolicyDocument("{\"version\":\"v1\",\"accessControlList\":
[{\"service\":\"bce:iam\"," +
            "\"permission\":[\"AssumeRole\"],\"region\":\"*\",\"grantee\":[{\"id\":\"grantee-
id\"}],\"effect\":\"Allow\"}]}");

    client.updateUser(roleName, updateRoleRequest);
  }
```

删除角色

删除角色，请参考如下代码：

```
  public void deleteRole(IamClient client) {
    // 角色名
    String roleName = "test_role_name";

    client.deleteRole(roleName);
  }
```

列举角色

列举角色，请参考如下代码：

```
public void listUser(IamClient client) {
    client.listUser();
}
```

策略管理接口

创建策略

创建权限策略，请参考如下代码：

```
public CreatePolicyResponse createPolicy(IamClient client) {
    CreatePolicyRequest createPolicyRequest = new CreatePolicyRequest();
    // 设置策略名
    createPolicyRequest.setName("test_policy_name");
    // 设置策略描述
    createPolicyRequest.setDescription("test_policy_description");
    // 策略内容，ACL格式序列化后得到的String
    createPolicyRequest.setDocument("{\"accessControlList\": [{\"region\":\"bj\",\"service\":\"bcc\",\"resource\":
[\"*\"]," +
            "\"permission\":[\"*\"],\"effect\":\"Allow\"}]}");

    client.createPolicy(createPolicyRequest);
}
```

查询策略

查询权限策略，请参考如下代码：

```
public void getPolicy(IamClient client) {
    // 策略名
    String policyName = "test_policy_name";
    // 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    String policyType = "System";

    client.getPolicy(policyName, policyType);
}
```

删除策略

删除权限策略，请参考如下代码：

```
public void deletePolicy(IamClient client) {
    // 策略名
    String policyName = "test_policy_name";
    client.deletePolicy(policyName);
}
```

列举策略

列举权限策略，当policyType为System时，可列举系统内置的策略列表，请参考如下代码：

```
public void listPolicy(IamClient client) {
    // 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    String policyType = "System";

    client.listPolicy(policyType);
}
```

关联用户权限

为用户关联权限策略，请参考如下代码：

```
public void attachPolicyToUser(IamClient client) {
    // 用户名
    String userName = "test_user_name";
    // 策略名
    String policyName = "test_policy_name";
    // 策略类型
    String policyType = "System";

    client.attachPolicyToUser(userName, policyName, policyType);
}
```

解除用户权限

解除子用户关联的权限策略，请参考如下代码：

```
public void detachPolicyFromUser(IamClient client) {
    // 用户名
    String userName = "test_user_name";
    // 策略名
    String policyName = "test_policy_name";
    // 策略类型
    String policyType = "System";

    client.detachPolicyFromUser(userName，policyName, policyType)
}
```

列举用户的权限

列举用户关联的权限策略，请参考如下代码：

```
public void listPoliciesForUser(IamClient client) {
    // 用户名
    String userName = "test_user_name";

    return client.listPoliciesForUser(userName)
}
```

关联组权限

为用户组关联权限策略，请参考如下代码：

```
public void attachPolicyToGroup(IamClient client) {
    // 组名
    String groupName = "test_group_name";
    // 策略名
    String policyName = "test_policy_name";
    // 策略类型
    String policyType = "System";

    client.attachPolicyToGroup(userName, policyName, policyType);
}
```

解除组权限

解除用户组关联的权限策略，请参考如下代码：

```
public void detachPolicyFromGroup(IamClient client) {
    // 组名
    String groupName = "test_group_name";
    // 策略名
    String policyName = "test_policy_name";
    // 策略类型
    String policyType = "System";

    client.detachPolicyFromGroup(groupName, policyName, policyType);
}
```

列举组关联的权限

列举组关联的权限策略，请参考如下代码：

```
public void listPoliciesForGroup(IamClient client) {
    // 组名
    String groupName = "test_group_name";

    client.listPoliciesForUser(groupName);
}
```

关联角色权限

为角色关联权限策略，请参考如下代码：

```
public void attachPolicyToRole(IamClient client) {
    // 角色名
    String roleName = "test_role_name";
    // 策略名
    String policyName = "test_policy_name";
    // 策略类型
    String policyType = "System";

    client.attachPolicyToRole(roleName, policyName, policyType);
}
```

解除角色权限

解除角色关联的权限策略，请参考如下代码：

```
public void detachPolicyFromRole(IamClient client) {
    // 角色名
    String roleName = "test_role_name";
    // 策略名
    String policyName = "test_policy_name";
    // 策略类型
    String policyType = "System";

    client.detachPolicyFromRole(roleName, policyName, policyType);
}
```

列举角色的权限

列举角色关联的权限策略，请参考如下代码：

```
public void listPoliciesForRole(IamClient client) {
    // 角色名
    String roleName = "test_role_name";
    client.listPoliciesForRole(roleName);
}
```

组管理接口

创建组

创建组，请参考如下代码：

```
public void createGroup(IamClient client) {
    CreateGroupRequest createGroupRequest = new CreateGroupRequest();
    // 组名
    createGroupRequest.setName("test_group_name");
    // 组描述
    createGroupRequest.setDescription("test_group_description");

    client.createGroup(createGroupRequest);
}
```

查询组

查询组，请参考如下代码：

```
public void getGroup(IamClient client) {
    // 组名
    String groupName = "test_group_name";

    client.getGroup(groupName);
}
```

更新组

更新组，请参考如下代码：

```
public void updateGroup(IamClient client) {
    // 组名
    String groupName = "test_group_name";
    // 设置更新的组信息
    UpdateGroupRequest updateGroupRequest = new UpdateGroupRequest();
    // 设置更新的组名
    updateGroupRequest.setName("new_group_name");
    // 设置更新的组描述
    updateGroupRequest.setDescription("new_group_description");

    client.updateGroup(groupName, updateGroupRequest);
}
```

删除组

删除组，请参考如下代码：

```
public void deleteGroup(IamClient client) {
    // 组名
    String groupName = "test_group_name";

    client.deleteGroup(groupName);
}
```

列举组

列举组，请参考如下代码：

```
public void listGroup(IamClient client) {
    client.listGroup();
}
```

添加用户到组

添加用户到组，请参考如下代码：

```
public void addUserToGroup(IamClient client) {
    // 组名
    String groupName = "test_group_name";
    // 用户名
    String userName = "test_user_name";

    client.addUserToGroup(userName ， groupName);
}
```

从组内移除用户

从组内移除用户，请参考如下代码：

```
public void removeUserFromGroup(IamClient client) {
    // 组名
    String groupName = "test_group_name";
    // 用户名
    String userName = "test_user_name";

    client.removeUserFromGroup(userName, groupName)
}
```

列举用户的组

列举用户所在的组，请参考如下代码：

```
public void listGroupsForUser(IamClient client) {
  // 用户名
  String userName = "test_user_name";

  client.listGroupsForUser(userName);
}
```

列举组内用户

列举组内用户，请参考如下代码：

```
public void listUsersInGroup(IamClient client) {
  // 组名
  String groupName = "test_group_name";

  client.listUsersInGroup(groupName);
}
```

错误码

错误码格式

当用户访问API出现错误时，会返回给用户相应的错误码和错误信息，便于定位问题，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	表示具体错误类型。
message	String	有关该错误的详细说明。
requestId	String	导致该错误的requestId。

例如：

```
{
  "code":"IllegalRequestUrl",
  "message":"The requested url belongs to domain which is not under acceleration",
  "requestId":" 81d0b05f-5ad4-1f22-8068-d5c9de60a1d7"
}
```

公共错误码

错误码	错误消息	HTTP状态码	描述
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。
		500Inte	所有未定义的其他错误。在有明确对应的其他类型

InternalError	We encountered an internal error. Please try again.	rnal Server Error	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。
InvalidHTTPAuthHeader	The HTTP authorization header is invalid. Consult the service documentation for details.	400 Bad Request	Authorization头域格式错误。
InvalidHTTPRequest	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	详见ETag。
RequestExpired	Request has expired. Timestamp date is XXX.	400 Bad Request	请求超时。XXX要改成x-bce-date的值。如果请求中只有Date，则需要将Date转换为datetime。
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。

版本变更记录

- Java SDK开发包[2020-07-06] 版本号 v0.10.115 首次发布：

- 支持创建、查询、更新、删除、列举用户
- 支持配置用户的控制台登录，支持查询、关闭控制台登录
- 支持创建、禁用、启用、删除、列举用户的AccessKey
- 支持创建、查询、更新、删除、列举用户组
- 支持添加用户到组，从组内移除用户，列举用户的组和组内用户
- 支持创建、查询、更新、删除、列举角色
- 支持创建、查询、删除、列举策略
- 支持关联、解除、列举用户的权限、组权限、角色权限

Python-SDK

概述

本文档主要介绍IAM Python SDK的安装和使用。在使用本文档前，您需要先了解IAM的一些基本知识。若您还不了解IAM，可以参考产品描述和快速入门。

安装SDK工具包

环境准备

1、运行环境

Python SDK支持在Python2.7和Python3.x 的环境下运行。

2、安装pycrypto依赖

安装SDK之前，需要先执行命令`pip install pycrypto`安装pycrypto依赖。

如果安装失败，请执行`pip install pycryptodome`。

下载和安装SDK

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）来完成SDK的安装：

```
cd bce-python-sdk-version

python setup.py install
```

SDK目录结构

```
baidubce
  ├─── auth                        //公共权限目录
  ├─── http                        //Http请求模块
  ├─── services                     //服务公共目录
  │     └─── iam                   //IAM目录
  │          ├──iam_client.py      //IAM客户端入口类
  ├─── bce_base_client.py           //BCE客户端入口类的基类
  ├─── bce_client_configuration.py    //HttpClient的配置类
  ├─── 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

百度智能云目前开放了多区域支持，请参考区域选择说明。

目前只支持“华北-北京”一个区域。 北京区域：`https://iam.bj.baidubce.com` 对应信息为：

访问区域	对应Endpoint	支持协议
BJ	iam.bj.baidubce.com	HTTP,HTTPS

获取密钥

要使用百度智能云IAM，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IAM做签名验证。 可以通过如下步骤获得并了解您的AK/SK信息：

注册百度智能云账号

创建AK/SK

新建IamClient

IamClient是IAM服务的客户端，为开发者与IAM服务进行交互提供了一系列的方法。

1、在新建IamClient之前，需要先创建配置文件对IamClient进行配置，以下将此配置文件命名为 iam_sample_conf.py ，具体配置信息如下所示：

```
##### 引入标准日志模块
import logging

##### 引入配置管理模块以及安全认证模块
from baidubce.bce_client_configuration import BceClientConfiguration
from baidubce.auth.bce_credentials import BceCredentials

##### 设置Host，Access Key ID和Secret Access Key
HOST = ''
AK = ''
SK = ''

##### 设置日志文件的句柄和日志级别
logger = logging.getLogger('baidubce.services.iam.iamclient')
fh = logging.FileHandler('sample.log')
fh.setLevel(logging.DEBUG)

##### 设置日志文件输出的顺序、结构和内容
formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')
fh.setFormatter(formatter)
logger.setLevel(logging.DEBUG)
logger.addHandler(fh)

##### 创建BceClientConfiguration
config = BceClientConfiguration(credentials=BceCredentials(AK, SK), endpoint=HOST)
```

2、创建IamClient，参考sample/iam/iam_sample.py

```
##### 引入配置模块
import iam_sample_conf

##### 引入IamClient模块
from baidubce.services.iam.iam_client import IamClient

##### 创建IamClient
iam_client = IamClient(iam_sample_conf.config)
```

配置IamClient

设置网络参数

用户可以设置一些网络参数：

```
##### 设置请求超时时间
iam_sample_conf.config.connection_timeout_in_mills = TIMEOUT

##### 设置接收缓冲区大小
iam_sample_conf.config.recv_buf_size(BUF_SIZE)

##### 设置发送缓冲区大小
iam_sample_conf.config.send_buf_size(BUF_SIZE)

##### 设置连接重试策略
##### 三次指数退避重试
iam_sample_conf.config.retry_policy = BackOffRetryPolicy()
```

参数说明

通过BceClientConfiguration能指定的所有参数如下表所示：

参数	说明
send_buf_size	发送缓冲区大小
recv_buf_size	接收缓冲区大小
connection_timeout_in_mills	请求超时时间（单位：毫秒）
retry_policy	连接重试策略，初始化Client时默认为三次指数退避

用户管理接口

创建用户

支持子用户的创建，请参考如下代码：

```
def create_user():

    iam_client = IamClient(iam_sample_conf.config)

    # 创建用户请求为dict
    # 设置用户名name 长度1-64位的字母、数字或"_"
    # 设置用户描述description
    create_user_request = {"name": "test_user", "description": "create user: test_user"}
    response = iam_client.create_user(create_user_request)

    print(response)
```

查询用户

子用户查询，请参考如下代码：

```
def get_user():

    iam_client = IamClient(iam_sample_conf.config)

    # 设置用户名
    user_name = b'test_user'
    response = iam_client.get_user(user_name)

    print(response)
```

更新用户

支持子用户的更新，请参考如下代码：

```
def update_user():

    iam_client = IamClient(iam_sample_conf.config)

    # 目前的用户名
    user_name = b"test_user"

    # 更新用户请求为dict
    # 设置更新后的用户名
    # 设置更新后的用户描述
    update_user_request = {"name": "test_user_new", "description": "test-new"}
    response = iam_client.update_user(user_name, update_user_request)

    print(response)
```

删除用户

支持子用户的删除，请参考如下代码：

```
def delete_user():

    iam_client = IamClient(iam_sample_conf.config)

    # 目前的用户名
    user_name = b"test_user"
    response = iam_client.delete_user(user_name)

    print(response)
```

列举用户

列举用户，请参考如下代码：

```
def list_user():

    iam_client = IamClient(iam_sample_conf.config)

    response = iam_client.list_user()

    print(response)
```

配置用户的控制台登录

配置用户的控制台登录，请参考如下代码：

```
def update_user_login_profile():

    iam_client = IamClient(iam_sample_conf.config)

    # 配置用户的控制台登录的请求为dict
    # 设置用户密码password，作为响应时不显示
    # 是否要求绑定二次验证设备enabledLoginMfa
    # 二次验证类型loginMfaType，可选：PHONE-手机号，TOTP虚拟MFA设备
    # 绑定的第三方登录类型thirdPartyType，可选：UUAP-内网账号，PASSPORT-百度账号
    # 绑定的第三方登录账号thirdPartyAccount，绑定类型为PASSPORT时可以是手机、邮箱以及账号名称
    update_user_login_profile_request = {"password":"Pa$$word4Demo", "enabledLoginMfa": True, "loginMfaType":
"PHONE",
                                         "thirdPartyType": "PASSPORT", "thirdPartyAccount": "testPassportAccount"}

    # 目前的用户名
    user_name = b"test_user"
    response = iam_client.update_user_login_profile(user_name, update_user_login_profile_request)

    print(response)
}
```

> 说明：1、thirdPartyType与thirdPartyAccount均为空字符串时解除绑定；2、若设置enabledLoginMfa":true，那么必须要指定loginMfaType

查询控制台登录配置

查询用户的控制台登录配置，请参考如下代码：

```
def get_user_login_profile():

    iam_client = IamClient(iam_sample_conf.config)

    # 目前的用户名
    user_name = b"test_user"
    response = iam_client.get_user_login_profile(user_name)

    print(response)
```

关闭控制台登录配置

关闭用户的控制台登录配置，即关闭用户的控制台登录，请参考如下代码：

```
def delete_user_login_profile():

    iam_client = IamClient(iam_sample_conf.config)

    # 设置目前的用户名
    user_name = b"test_user"
    response = iam_client.delete_user_login_profile(user_name)

    print(response)
```

创建用户的AccessKey

为用户创建一组AccessKey访问密钥，请参考如下代码：

```
def create_user_accesskey():

    iam_client = IamClient(iam_sample_conf.config)

    # 设置目前的用户名
    user_name = b"test_user"
    response = iam_client.create_user_accesskey(user_name)

    print(response)
```

禁用用户的AccessKey

将用户的指定一组AccessKey访问密钥设置为禁用状态，请参考如下代码：

```
def disable_user_accesskey():

    iam_client = IamClient(iam_sample_conf.config)

    # 设置目前的用户名
    user_name = b"test_user"

    # 设置需要禁用的ak
    accesskey_id = b"test_access_key_id"

    response = iam_client.disable_user_accesskey(user_name, accesskey_id)
    print(response)
```

启用用户的AccessKey

将用户的指定一组AccessKey访问密钥恢复为启用状态，请参考如下代码：

```
def enable_user_accesskey():

    iam_client = IamClient(iam_sample_conf.config)

    # 设置目前的用户名
    user_name = b"test_user"

    # 设置需要启用的ak
    accesskey_id = b"test_access_key_id"
    response = iam_client.enable_user_accesskey(user_name, accesskey_id)

    print(response)
```

删除用户的AccessKey

删除用户的指定一组AccessKey访问密钥，请参考如下代码：

```
def delete_user_accesskey():

    iam_client = IamClient(iam_sample_conf.config)

    # 设置目前的用户名
    user_name = b"test_user"

    # 设置需要删除的ak
    accesskey_id = b"test_access_key_id"
    response = iam_client.delete_user_accesskey(user_name, accesskey_id)

    print(response)
```

列举用户的AccessKey

列举用户的全部AccessKey访问密钥，请参考如下代码：

```
def list_user_accesskey():

    iam_client = IamClient(iam_sample_conf.config)

    # 设置目前的用户名
    user_name = b"test_user"
    response = iam_client.list_user_accesskey(user_name)

    print(response)
```

角色管理接口

创建角色

创建角色，请参考如下代码：

```
def create_role():

    iam_client = IamClient(iam_sample_conf.config)

    # 创建角色的请求为dict
    # 设置角色名name
    # 设置角色描述description
    # 指定允许扮演角色的载体assumeRolePolicyDocument
    create_role_request = {"name": "test_role", "description": "create role: test_role",
                           "assumeRolePolicyDocument": "{\"version\":\"v1\",\"accessControlList\":[{"
                                                       "\"service\":\"bce:iam\",\"permission\":[\"AssumeRole\"],"
                                                       "\"region\":\"*\",\"grantee\":[{"
                                                       "\"id\":\"test_account_id\"}],"
                                                       "\"effect\":\"Allow\"}]}"}
    response = iam_client.create_role(create_role_request)

    print(response)
```

查询角色

查询角色，请参考如下代码：

```
def get_role():

    iam_client = IamClient(iam_sample_conf.config)

    # 设置角色名
    role_name = b"test_role"
    response = iam_client.get_role(role_name)

    print(response)
```

更新角色

更新角色，请参考如下代码：

```
def update_role():

    iam_client = IamClient(iam_sample_conf.config)

    # 旧的角色名
    role_name = b"test_role"

    # 更新角色的请求为dict
    # 设置更新的角色名 name
    # 设置更新的角色描述 description
    # 设置更新的允许扮演角色的载体 assumeRolePolicyDocument
    update_role_request = {"name": "test_role_new", "description": "update role: test_role",
                           "assumeRolePolicyDocument": "{\"version\":\"v1\",\"accessControlList\":[{"
                                                       "\"service\":\"bce:iam\",\"permission\":[\"AssumeRole\"],"
                                                       "\"region\":\"*\",\"grantee\":[{"
                                                       "\"id\":\"test_account_id\"}],"
                                                       "\"effect\":\"Allow\"}]}"}
    response = iam_client.update_role(role_name, update_role_request)

    print(response)
```

删除角色

删除角色，请参考如下代码：

```
def delete_role():

    iam_client = IamClient(iam_sample_conf.config)

    # 需要删除的角色名
    role_name = b"test_role"
    response = iam_client.delete_role(role_name=role_name)

    print(response)
```

列举角色

列举角色，请参考如下代码：

```
def list_role():

    iam_client = IamClient(iam_sample_conf.config)

    response = iam_client.list_role()

    print(response)
```

策略管理接口

创建策略

创建权限策略，请参考如下代码：

```
def create_policy():

    iam_client = IamClient(iam_sample_conf.config)

    # 创建策略的请求为dict
    # 设置策略名name
    # 设置策略描述description
    # 设置策略内容document，ACL格式序列化后得到的String
    create_policy_request = {"name": "test_policy", "description": "create policy: test_policy_1",
                             "document": '{ "accessControlList": [ { "region": "bj", "resource": [ "*" ], "effect":'
                                         '"Allow", "service": "bce:bos", "permission": [ "READ" ] } ] } '}
    response = iam_client.create_policy(create_policy_request)

    print(response)
```

查询策略

查询权限策略，请参考如下代码：

```
def get_policy():

    iam_client = IamClient(iam_sample_conf.config)

    # 策略名
    policy_name = b"test_policy"
    # 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    policy_type = b"Custom"
    response = iam_client.get_policy(policy_name=policy_name, policy_type=policy_type)

    print(response)
```

删除策略

删除权限策略，请参考如下代码：

```
def delete_policy():

    iam_client = IamClient(iam_sample_conf.config)

    # 策略名
    policy_name = b"test_policy"
    response = iam_client.delete_policy(policy_name=policy_name)

    print(response)(policyName);
}
```

列举策略

列举权限策略，当policyType为System时，可列举系统内置的策略列表，请参考如下代码：

```
def list_policy():

    iam_client = IamClient(iam_sample_conf.config)

    # 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    policy_type = b"Custom"
    # 关键字过滤
    name_filter = b"test"
    response = iam_client.list_policy(policy_type=policy_type, name_filter=name_filter)

    print(response)
```

关联用户权限

为用户关联权限策略，请参考如下代码：

```
def attach_policy_to_user():

    iam_client = IamClient(iam_sample_conf.config)

    # 需要关联策略的用户
    user_name = b"test_user"
    # 关联的策略
    policy_name = b"test_policy"
    # 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    policy_type = b"Custom"
    response = iam_client.attach_policy_to_user(user_name, policy_name, policy_type)

    print(response)
```

解除用户权限

解除子用户关联的权限策略，请参考如下代码：

```
def detach_policy_from_user():

    iam_client = IamClient(iam_sample_conf.config)

    # 需要关联策略的用户
    user_name = b"test_user"
    # 关联的策略
    policy_name = b"test_policy"
    # 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    policy_type = b"Custom"
    response = iam_client.detach_policy_from_user(user_name, policy_name, policy_type)

    print(response)
```

列举用户的权限

列举用户关联的权限策略，请参考如下代码：

```
def list_policies_from_user():

    iam_client = IamClient(iam_sample_conf.config)

    # 用户名
    user_name = b"test_user"
    response = iam_client.list_policies_from_user(user_name)

    print(response)
```

关联组权限

为用户组关联权限策略，请参考如下代码：

```
def attach_policy_to_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 需要关联策略的用户组
    group_name = b"test_group"
    # 关联的策略
    policy_name = b"test_policy"
    # 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    policy_type = b"Custom"
    response = iam_client.attach_policy_to_group(group_name, policy_name, policy_type)

    print(response)
```

解除组权限

解除用户组关联的权限策略，请参考如下代码：

```
def detach_policy_from_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 需要关联策略的用户组
    group_name = b"test_group"
    # 关联的策略
    policy_name = b"test_policy"
    # 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    policy_type = b"Custom"
    response = iam_client.detach_policy_from_group(group_name, policy_name, policy_type)

    print(response)
```

列举组关联的权限

列举组关联的权限策略，请参考如下代码：

```
def list_policies_from_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 用户组名
    group_name = b"test_group"
    response = iam_client.list_policies_from_group(group_name)

    print(response)
```

关联角色权限

为角色关联权限策略，请参考如下代码：

```
def attach_policy_to_role():

    iam_client = IamClient(iam_sample_conf.config)

    # 需要关联策略的角色
    role_name = b"test_role"
    # 关联的策略
    policy_name = b"test_policy"
    # 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    policy_type = b"Custom"
    response = iam_client.attach_policy_to_role(role_name, policy_name, policy_type)

    print(response)
```

解除角色权限

解除角色关联的权限策略，请参考如下代码：

```
def detach_policy_from_role():

    iam_client = IamClient(iam_sample_conf.config)

    # 需要关联策略的角色
    role_name = b"test_role"
    # 关联的策略
    policy_name = b"test_policy"
    # 要查询的策略类型，为System时查询系统策略；为Custom时查询自定义策略
    policy_type = b"Custom"
    response = iam_client.detach_policy_from_role(role_name, policy_name, policy_type)

    print(response)
```

列举角色的权限

列举角色关联的权限策略，请参考如下代码：

```
def list_policies_from_role():

    iam_client = IamClient(iam_sample_conf.config)

    # 角色名
    role_name = b"test_role"
    response = iam_client.list_policies_from_role(role_name)

    print(response)
```

组管理接口

创建组

创建组，请参考如下代码：

```
def create_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 创建用户组的请求为dict
    # 设置用户组名name
    # 设置用户组描述description
    create_group_request = {"name": "test_group", "description": "create test_group"}
    response = iam_client.create_group(create_group_request)

    print(response)
```

查询组

查询组，请参考如下代码：

```
def get_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 用户组名
    group_name = b"test_group"
    response = iam_client.get_group(group_name)

    print(response)
```

更新组

更新组，请参考如下代码：

```
def update_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 目前的用户组名
    group_name = b"test_group"

    # 更新用户组的请求为dict
    # 设置新的用户组名name
    # 设置用户组描述description
    update_group_request = {"name": "test_group_new", "description": "update test_group"}
    response = iam_client.update_group(group_name, update_group_request)

    print(response)
```

删除组

删除组，请参考如下代码：

```
def delete_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 目前的用户组名
    group_name = b"test_group"
    response = iam_client.delete_group(group_name)

    print(response)
```

列举组

列举组，请参考如下代码：

```
def list_group():

    iam_client = IamClient(iam_sample_conf.config)

    response = iam_client.list_group()

    print(response)
```

添加用户到组

添加用户到组，请参考如下代码：

```
def add_user_to_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 用户组名
    group_name = b"test_group"
    # 需要加入用户组中的用户
    user_name = b"test_user"
    response = iam_client.add_user_to_group(group_name, user_name)

    print(response)
```

从组内移除用户

从组内移除用户，请参考如下代码：

```
def remove_user_from_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 用户组名
    group_name = b"test_group"

    # 需要从用户组中删除的用户
    user_name = b"test_user"
    response = iam_client.remove_user_from_group(group_name, user_name)

    print(response)
```

列举用户的组

列举用户所在的组，请参考如下代码：

```
def list_user_group():

    iam_client = IamClient(iam_sample_conf.config)

    # 用户名
    user_name = b"test_user"
    response = iam_client.list_user_group(user_name)

    print(response)
```

列举组内用户

列举组内用户，请参考如下代码：

```
def list_group_user():

    iam_client = IamClient(iam_sample_conf.config)

    group_name = b"test_group"
    response = iam_client.list_group_user(group_name)

    print(response)
```

错误码

错误码格式

当用户访问API出现错误时，会返回给用户相应的错误码和错误信息，便于定位问题，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	表示具体错误类型。
message	String	有关该错误的详细说明。
requestId	String	导致该错误的requestId。

例如：

```
{
    "code":"IllegalRequestUrl",
    "message":"The requested url belongs to domain which is not under acceleration",
    "requestId":" 81d0b05f-5ad4-1f22-8068-d5c9de60a1d7"
}
```

公共错误码

错误码	错误消息	HTTP状态码	描述
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。
InternalError	We encountered an internal error. Please try again.	500Internal 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。
InvalidH	The HTTP authorization header is invalid. Consult the	400 Bad	

TTPAuthHeader	service documentation for details.	Request	Authorization头域格式错误。
InvalidHTTPRequest	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	详见ETag。
RequestExpired	Request has expired. Timestamp date is XXX.	400 Bad Request	请求超时。XXX要改成x-bce-date的值。如果请求中只有Date，则需要将Date转换为datetime。
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。

版本变更记录

- Python SDK开发包[2021-11-24] 版本号 v0.8.63 首次发布：：
  - 支持创建、查询、更新、删除、列举用户
  - 支持配置用户的控制台登录，支持查询、关闭控制台登录
  - 支持创建、禁用、启用、删除、列举用户的AccessKey
  - 支持创建、查询、更新、删除、列举用户组
  - 支持添加用户到组，从组内移除用户，列举用户的组和组内用户

- 支持创建、查询、更新、删除、列举角色
- 支持创建、查询、删除、列举策略
- 支持关联、解除、列举用户的权限、组权限、角色权限

Go-SDK

概述

本文档主要介绍IAM GO SDK的安装和使用。在使用本文档前，您需要先了解IAM的一些基本知识。若您还不了解IAM，可以参考产品描述和快速入门。

安装SDK工具包

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

```
 bce-sdk-go
|--auth                 //BCE签名和权限认证
|--bce                  //BCE公用基础组件
|--http                 //BCE的http通信模块
|--services              //BCE相关服务目录
| |--appblb             //应用型负载均衡服务目录
| |--bbc                //物理服务器
| |--bcc                //云服务器
| |--bec                //百度边缘计算
| |--bie                //百度智能边缘
| |--bls                //日志服务
| |--bos                //BOS服务目录
| | |--bos_client.go    //BOS客户端入口
| | |--api              //BOS相关API目录
| |    |--bucket.go     //BOS的Bucket相关API实现
| |    |--object.go     //BOS的Object相关API实现
| |    |--multipart.go  //BOS的Multipart相关API实现
| |    |--module.go     //BOS相关API的数据模型
| |    |--util.go       //BOS相关API实现使用的工具
| |--cce                //容器引擎
| |--cdn                //内容分布网络
| |--cert               //SSL证书服务
| |--cfc                //函数计算
| |--cfs                //CFS文件存储服务
| |--cfw                //云防火墙
| |--csn                //云智能网
| |--ddc                //DDC数据库专属集群
| |--dts                //数据传输服务
| |--eip                //弹性公网IP
| |--endpoint            //SNIC服务网卡
| |--eni                //ENIC弹性网卡
| |--esg                //企业安全组
| |--etGateway           //专线网关
| |--iam                //身份管理
| |--rds                //云数据库
| |--scs                //SCS服务目录
| |--sms                //SMS服务目录
| |--sts                //STS服务目录
| |--vca                //VCA服务目录
| |--vcr                //VCR服务目录
| |--vpc                //私有网络
| |--vpn                //VPN网关
| |--mms                //多模态媒资检索目录
|--util                 //BCE公用的工具实现
```

卸载SDK

预期卸载SDK时，删除下载的源码即可。

初始化sdk

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IAM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。

获取密钥

要使用百度云IAM，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IAM做签名验证。

可以通过如下步骤获得并了解您的AK/SK信息：

注册百度云账号

创建AK/SK

新建IAM Client

普通型IAM Client是IAM服务的客户端，为开发者与IAM服务进行交互提供了一系列的方法。

使用AK/SK新建IAM Client

通过AK/SK方式访问IAM，用户可以参考如下代码新建一个IAM Client：

```
import (
"github.com/baidubce/bce-sdk-go/services/iam"
)

func main() {
// 用户的Access Key ID和Secret Access Key
ACCESS_KEY_ID, SECRET_ACCESS_KEY := <your-access-key-id>, <your-secret-access-key>

// 初始化一个IAMClient
iamClient, err := iam.NewClient(AK, SK)
}
```

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 如何获取AKSK》。

使用STS创建IAM Client

申请STS token

IAM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云的API或SDK访问百度云资源。

通过STS方式访问IAM，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云STS使用介绍。

用STS token新建IAM Client

申请好STS后，可将STS Token配置到IAM Client中，从而实现通过STS Token创建IAM Client。

代码示例

GO SDK实现了STS服务的接口，用户可以参考如下完整代码，实现申请STS Token和创建IAM Client对象：

```
 import (
 "fmt"

 "github.com/baidubce/bce-sdk-go/auth"            //导入认证模块
 "github.com/baidubce/bce-sdk-go/services/iam"    //导入IAM服务模块
 "github.com/baidubce/bce-sdk-go/services/sts"    //导入STS服务模块
)

func main() {
 // 创建STS服务的Client对象，Endpoint使用默认值
 AK, SK := <your-access-key-id>, <your-secret-access-key>
 stsClient, err := sts.NewClient(AK, SK)
 if err != nil {
  fmt.Println("create sts client object :", err)
  return
 }

 // 获取临时认证token，有效期为60秒，ACL为空
 stsObj, err := stsClient.GetSessionToken(60, "")
 if err != nil {
  fmt.Println("get session token failed:", err)
  return
    }
 fmt.Println("GetSessionToken result:")
 fmt.Println("  accessKeyId:", stsObj.AccessKeyId)
 fmt.Println("  secretAccessKey:", stsObj.SecretAccessKey)
 fmt.Println("  sessionToken:", stsObj.SessionToken)
 fmt.Println("  createTime:", stsObj.CreateTime)
 fmt.Println("  expiration:", stsObj.Expiration)
 fmt.Println("  userId:", stsObj.UserId)

 // 使用申请的临时STS创建IAM服务的Client对象，Endpoint使用默认值
 iamClient, err := iam.NewClient(stsObj.AccessKeyId, stsObj.SecretAccessKey, "")
 if err != nil {
  fmt.Println("create iam client failed:", err)
  return
 }
 stsCredential, err := auth.NewSessionBceCredentials(
  stsObj.AccessKeyId,
  stsObj.SecretAccessKey,
  stsObj.SessionToken)
 if err != nil {
  fmt.Println("create sts credential object failed:", err)
  return
 }
 iamClient.Config.Credentials = stsCredential
}
```

> 注意： 目前使用STS配置IAM Client时，无论对应IAM服务的Endpoint在哪里，STS的Endpoint都需配置为`http://sts.bj.baidubce.com`。 上述代码中创建STS对象时使用此默认值。

配置HTTPS协议访问IAM

IAM支持HTTPS传输协议，您可以通过在创建IAM Client对象时指定的Endpoint中指明HTTPS的方式，在IAM GO SDK中使用HTTPS访问IAM服务：

```
// import "github.com/baidubce/bce-sdk-go/services/iam"

ENDPOINT := "https://iam.bj.baidubce.com" //指明使用HTTPS协议
AK, SK := <your-access-key-id>, <your-secret-access-key>
iamClient, _ := iam.NewClientWithEndpoint(AK, SK, ENDPOINT)
```

配置IAM Client

如果用户需要配置IAM Client的一些细节的参数，可以在创建IAM Client对象之后，使用该对象的导出字段Config进行自定义配置，可以为客户端配置代理，最大连接数等参数。

使用代理

下面一段代码可以让客户端使用代理访问IAM服务：

```
// import "github.com/baidubce/bce-sdk-go/services/iam"

//创建IAM Client对象
AK, SK := <your-access-key-id>, <your-secret-access-key>
ENDPOINT := "iam.bj.baidubce.com
client, _ := iam.NewClient(AK, SK, ENDPOINT)

//代理使用本地的8080端口
client.Config.ProxyUrl = "127.0.0.1:8080"
```

设置网络参数

用户可以通过如下的示例代码进行网络参数的设置：

```
// import "github.com/baidubce/bce-sdk-go/services/iam"

AK, SK := <your-access-key-id>, <your-secret-access-key>
client, _ := iam.NewClient(AK, SK)

// 配置不进行重试，默认为Back Off重试
client.Config.Retry = bce.NewNoRetryPolicy()

// 配置连接超时时间为30秒
client.Config.ConnectionTimeoutInMillis = 30 * 1000
```

配置生成签名字符串选项

```
// import "github.com/baidubce/bce-sdk-go/services/iam"

AK, SK := <your-access-key-id>, <your-secret-access-key>
client, _ := iam.NewClient(AK, SK)

// 配置签名使用的HTTP请求头为`Host`
headersToSign := map[string]struct{}{"Host": struct{}{}}
client.Config.SignOption.HeadersToSign = HeadersToSign

// 配置签名的有效期为30秒
client.Config.SignOption.ExpireSeconds = 30
```

参数说明

用户使用GO SDK访问IAM时，创建的IAM Client对象的Config字段支持的所有参数如下表所示：

配置项名称	类型	含义
Endpoint	string	请求服务的域名
ProxyUrl	string	客户端请求的代理地址
Region	string	请求资源的区域
UserAgent	string	用户名称，HTTP请求的User-Agent头
Credentials	*auth.BceCredentials	请求的鉴权对象，分为普通AK/SK与STS两种
SignOption	*auth.SignOptions	认证字符串签名选项
Retry	RetryPolicy	连接重试策略
ConnectionTimeoutInMillis	int	连接超时时间，单位毫秒，默认20分钟

说明：

1. Credentials字段使用auth.NewBceCredentials与auth.NewSessionBceCredentials函数创建，默认使用前者，后者为使用STS鉴权时使用，详见“使用STS创建IAM Client”小节。
2. SignOption字段为生成签名字符串时的选项，详见下表说明：

名称	类型	含义
HeadersToSign	map[string]struct{}	生成签名字符串时使用的HTTP头
Timestamp	int64	生成的签名字符串中使用的时间戳，默认使用请求发送时的值
ExpireSeconds	int	签名字符串的有效期

> 其中，HeadersToSign默认为`Host`，`Content-Type`，`Content-Length`，`Content-MD5`；TimeStamp一般为零值，表示使用调用生成认证字符串时的时间戳，用户一般不应该明确指定该字段的值；ExpireSeconds默认为1800秒即30分钟。

3. Retry字段指定重试策略，目前支持两种：NoRetryPolicy和BackOffRetryPolicy。默认使用后者，该重试策略是指定最大重试次数、最长重试时间和重试基数，按照重试基数乘以2的指数级增长的方式进行重试，直到达到最大重试测试或者最长重试时间为止。

用户管理接口

创建用户 通过以下代码可以创建子用户。

```
    name := "test-user-sdk-go"
  args := &api.CreateUserArgs{
Name:        name,
    Description: "description",
  }

  result, err := client.CreateUser(args)
  if err != nil {
    fmt.Println("Create iam user failed", err)
  } else {
    fmt.Println("Create iam user success", result)
  }
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档CreateUser创建用户。

查询用户 通过以下代码可以查询单个子用户。

```
 name := "test-user-sdk-go"
result, err := client.GetUser(name)
if err != nil {
 fmt.Println("Get iam user failed", err)
} else {
 fmt.Println("Get iam user success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档GetUser查询用户。

更新用户 通过以下代码可以更新子用户。

```
    name := "test-user-sdk-go"
args := &api.UpdateUserArgs{
 Description: "newDescription",
}

result, err := client.UpdateUser(name, args)
if err != nil {
 fmt.Println("Update iam user failed", err)
} else {
 fmt.Println("Update iam user success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档UpdateUser更新用户。

列举用户 通过以下代码可以列举子用户。

```
 result, err := client.ListUser()
if err != nil {
 fmt.Println("List iam user failed", err)
} else {
 fmt.Println("List iam user success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档ListUser创建用户。

删除用户 通过以下代码可以更新子用户。

```
 name := "test-user-sdk-go"
err = client.DeleteUser(name)
if err != nil {
 fmt.Println("Delete iam user failed", err)
} else {
 fmt.Println("Delete iam user success", name)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档DeleteUser删除用户。

配置用户控制台登录 通过以下代码可以配置用户的控制台登录，为其配置登录密码、开启登录MFA、配置第三方账号绑定等。

```
 name := "test-user-sdk-go-login-profile"
args := &api.UpdateUserLoginProfileArgs{
 Password:        "1@3Qwe4f",
 EnabledLoginMfa: false,
 LoginMfaType:    "PHONE",
}

result, err := client.UpdateUserLoginProfile(name, args)
if err != nil {
 fmt.Println("Update iam user login profile failed", err)
} else {
 fmt.Println("Update iam user login profile success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档UpdateUserLoginProfile配置用户控制台登录。

查询控制台登录配置 通过以下代码可以查询用户的控制台登录配置。

```
 name := "test-user-sdk-go-login-profile"
result, err := client.GetUserLoginProfile(name)
if err != nil {
 fmt.Println("Get iam user login profile failed", err)
} else {
 fmt.Println("Get iam user login profile success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档GetUserLoginProfile查询用户控制台登录。

关闭控制台登录配置 关闭用户的控制台登录配置，即关闭用户的控制台登录。

```
 name := "test-user-sdk-go-login-profile"
err = client.DeleteUserLoginProfile(name)
if err != nil {
 fmt.Println("Delete iam user login profile failed", err)
} else {
 fmt.Println("Delete iam user login profile success", name)
}
```

提示：

- 详细的参数配置及限制条件，可以参考IAM API文档DeleteUserLoginProfile关闭用户控制台登录。

修改子用户操作保护 通过以下代码修改子用户操作保护。

```
  userName := "test-user-sdk-go-switch-operation-mfa"
 enableMfa := true
 mfaType := "PHONE,TOTP"
 args := &api.UserSwitchMfaArgs{
    UserName:   userName,
    EnabledMfa: enableMfa,
    MfaType:    mfaType,
 }
 err := IAM_CLIENT.UserOperationMfaSwitch(args)
 if err != nil {
    fmt.Println("switch user mfa failed", err)
 } else {
    fmt.Println("switch user mfa success", result)
 }
```

提示：

- 详细的参数配置及限制条件，可以参考IAM API文档修改子用户操作保护。

修改子用户密码 通过以下代码修改子用户密码。

```
  userName := "test-user-name-sdk-go-sub-update"
 Password := "Baidu@123"
 args := &api.UpdateSubUserArgs{
    Password: Password,
 }
 res, err := IAM_CLIENT.SubUserUpdate(userName, args)
 if err != nil {
    fmt.Println("update sub user failed", err)
 } else {
    fmt.Println("update sub user success", result)
 }
```

提示：

- 详细的参数配置及限制条件，可以参考IAM API文档修改子用户密码。

创建用户的AccessKey 通过以下代码为用户创建一组AccessKey访问密钥。

```
 name := "test-user-sdk-go-accessKey"
result, err := client.CreateAccessKey(name)
if err != nil {
 fmt.Println("Create accessKey failed", err)
} else {
 fmt.Println("Create accessKey success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档创建用户的AccessKey。

禁用用户的AccessKey 通过以下代码为禁用用户的AccessKey。

```
 name := "test-user-sdk-go-accessKey"
accessKeyId := "<your-access-key-id>"
result, err := client.DisableAccessKey(name, accessKeyId)
if err != nil {
 fmt.Println("Disable accessKey failed", err)
} else {
 fmt.Println("Disable accessKey success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档禁用用户的AccessKey。

启用用户的AccessKey 通过以下代码为启用用户的AccessKey。

```
 name := "test-user-sdk-go-accessKey"
accessKeyId := "<your-access-key-id>"
result, err := client.EnableAccessKey(name, accessKeyId)
if err != nil {
 fmt.Println("Enable accessKey failed", err)
} else {
 fmt.Println("Enable accessKey success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档启用用户的AccessKey。

删除用户的AccessKey 删除用户的指定一组AccessKey访问密钥。

```
 name := "test-user-sdk-go-accessKey"
accessKeyId := "<your-access-key-id>"
err = client.DeleteAccessKey(name, accessKeyId)
if err != nil {
 fmt.Println("Delete accessKey failed", err)
} else {
 fmt.Println("Delete accessKey success", accessKeyId)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档删除用户的AccessKey。

列举用户的AccessKey 列举用户的全部AccessKey访问密钥。

```
 name := "test-user-sdk-go-accessKey"
result, err := client.ListAccessKey(name)
if err != nil {
 fmt.Println("List accessKey failed", err)
} else {
 fmt.Println("List accessKey success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档列举用户的AccessKey。
>
> **查询IAM用户联合信息** 查询IAM用户联合信息，对应页面菜单：多用户访问控制->外部账号接入->IAM用户联合页面的内容。

```
   result, err := client.GetSubUserIdpConfig()
  if err != nil {
 fmt.Println("Get subUser idp config failed", err)
} else {
 fmt.Println("Get subUser idp config success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档查询IAM用户联合信息。

更新IAM用户联合功能状态 更新IAM用户联合功能状态。

```
   status := "disable"
  result, err := client.UpdateSubUserIdpStatus(status)
  if err != nil {
 fmt.Println("Updte subUser idp config status failed", err)
} else {
 fmt.Println("Updte subUser idp config status success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档更新IAM用户联合功能状态。

更新IAM用户联合功能配置 更新IAM用户联合功能配置。

```
    var request api.UpdateSubUserIdpRequest

  fileName := "ok.xml"
  request.AuxiliaryDomain = "<your-auxiliary-domain>"
  request.FileName = "<your-file-name>"
  request..EncodeMetadata = "<your-encode-metadata>"

  result, err := client.UpdateSubUserIdp(&request)
  if err != nil {
 fmt.Println("Updte subUser idp config failed", err)
} else {
 fmt.Println("Updte subUser idp config success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档更新IAM用户联合功能配置。

删除IAM用户联合配置 删除IAM用户联合功能配置。

```
    err := IAM_CLIENT.DeleteSubUserIdp()
  if err != nil {
 fmt.Println("Delete subUser idp config failed", err)
} else {
 fmt.Println("Updte subUser idp config success")
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API文档删除IAM用户联合配置。

角色管理接口

创建角色 通过以下代码创建角色

```
    roleName := "test_role_sdk_go"
args := &api.CreateRoleArgs{
 Name:        roleName,
 Description: "description",
 AssumeRolePolicyDocument: "{\"version\":\"v1\",\"accessControlList\":[{\"service\":\"bce:iam\",\"permission\"" +
  ":[\"AssumeRole\"],\"region\":\"*\",\"grantee\":[{\"id\":\"grantee-id\"}],\"effect\":\"Allow\"}]}",
}

result, err := client.CreateRole(args)
if err != nil {
 fmt.Println("Create role failed", err)
} else {
 fmt.Println("Create role success", result)
}
```

提示：

- 详细的参数配置及限制条件，可以参考IAM API 文档创建角色

查询角色 通过以下代码查询角色

```
 roleName := "test_role_sdk_go"
result, err := client.GetRole(roleName)
if err != nil {
 fmt.Println("Get role failed", err)
} else {
 fmt.Println("Get role success", result)
}
```

提示：

- 详细的参数配置及限制条件，可以参考IAM API 文档查询角色

更新角色 通过以下代码查询角色

```
 args := &api.UpdateRoleArgs{
 Description: "newDescription",
 AssumeRolePolicyDocument: "{\"version\":\"v1\",\"accessControlList\":[{\"service\":\"bce:iam\",\"permission\"" +
  ":[\"AssumeRole\"],\"region\":\"*\",\"grantee\":[{\"id\":\"grantee-id\"}],\"effect\":\"Allow\"}]}",
}

roleName := "test_role_sdk_go"
result, err := client.UpdateRole(roleName, args)
if err != nil {
 fmt.Println("Update role failed", err)
} else {
 fmt.Println("Update role success", result)
}
```

提示：

- 详细的参数配置及限制条件，可以参考IAM API 文档更新角色

删除角色 通过以下代码查询角色

```
 roleName := "test_role_sdk_go"
err = client.DeleteRole(roleName)
if err != nil {
 fmt.Println("Delete role failed", err)
} else {
 fmt.Println("Delete role success", roleName)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档删除角色

列举角色 通过以下代码列举角色

```
 result, err := client.ListRole()
if err != nil {
 fmt.Println("List role failed", err)
} else {
 fmt.Println("List role success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举角色

策略管理接口

创建策略 通过以下代码创建策略

```
 name := "test_sdk_go_policy"
args := &api.CreatePolicyArgs{
    Name:        name,
    Description: "description",
    Document:    "{\"accessControlList\": [{\"region\":\"bj\",\"service\":\"bcc\"," +
"\"resource\":[\"*\"],\"permission\":[\"*\"],\"effect\":\"Allow\"}]}",
}

result, err := client.CreatePolicy(args)
if err != nil {
    fmt.Println("Update policy failed", err)
} else {
    fmt.Println("Update policy success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档创建策略

查询策略 通过以下代码查询策略

```
    name := "test_sdk_go_policy"
policyType := "Custom"
result, err := client.GetPolicy(name, policyType)
if err != nil {
 fmt.Println("Update policy failed", err)
} else {
 fmt.Println("Update policy success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档创建策略

删除策略 通过以下代码删除策略

```
 name := "test_sdk_go_policy"
err = client.DeletePolicy(name)
if err != nil {
 fmt.Println("List policy failed", err)
} else {
 fmt.Println("List policy success", name)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档删除策略

列举策略 通过以下代码列举策略

```
 name := "test_sdk_go_policy"
policyType := "Custom"
result, err := client.ListPolicy(name, policyType)
if err != nil {
 fmt.Println("List policy failed", err)
} else {
 fmt.Println("List policy success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举策略

关联用户权限 通过以下代码关联用户权限

```
 userName := "test_sdk_go_user"
policyName := "test_sdk_go_policy"
args := &api.AttachPolicyToUserArgs{
 UserName:   userName,
 PolicyName: policyName,
}
err = client.AttachPolicyToUser(args)
if err != nil {
 fmt.Println("Attach policy to user failed", err)
} else {
 fmt.Println("Attach policy to user success", args)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档关联用户权限

解除用户权限 通过以下代码解除用户权限

```
 userName := "test_sdk_go_user"
policyName := "test_sdk_go_policy"
args := &api.DetachPolicyFromUserArgs{
 UserName:   userName,
 PolicyName: policyName,
}
err = client.DetachPolicyFromUser(args)
if err != nil {
 fmt.Println("Detach policy to user failed", err)
} else {
 fmt.Println("Detach policy to user success", args)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档解除用户权限

列举用户的权限 通过以下代码列举用户的权限

```
 userName := "test_sdk_go_user"
result, err := client.ListUserAttachedPolicies(userName)
if err != nil {
 fmt.Println("List user attached policy failed", err)
} else {
 fmt.Println("List user attached policy success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举用户的权限

关联组权限 通过以下代码关联组权限

```
 groupName := "test_sdk_go_group"
policyName := "test_sdk_go_policy"
args := &api.AttachPolicyToGroupArgs{
 GroupName:  groupName,
 PolicyName: policyName,
}
err = client.AttachPolicyToGroup(args)
if err != nil {
 fmt.Println("Attach policy to group failed", err)
} else {
 fmt.Println("Attach policy to group success", args)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档关联组权限

解除组权限 通过以下代码解除组权限

```
 groupName := "test_sdk_go_group"
policyName := "test_sdk_go_policy"
args := &api.DetachPolicyFromGroupArgs{
 GroupName:  groupName,
 PolicyName: policyName,
}
err = client.DetachPolicyFromGroup(args)
if err != nil {
 fmt.Println("Detach policy to group failed", err)
} else {
 fmt.Println("Detach policy to group success", args)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档解除组权限

列举组权限 通过以下代码列举组权限

```
 groupName := "test_sdk_go_group"
result, err := client.ListGroupAttachedPolicies(groupName)
if err != nil {
 fmt.Println("List group attached policy failed", err)
} else {
 fmt.Println("List group attached policy success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举组权限

关联角色权限 通过以下代码关联角色权限

```
 roleName := "test_sdk_go_group"
policyName := "test_sdk_go_policy"
args := &api.AttachPolicyToRoleArgs{
 RoleName:   roleName,
 PolicyName: policyName,
}
err = client.AttachPolicyToRole(args)
if err != nil {
 fmt.Println("Attach policy to role failed", err)
} else {
 fmt.Println("Attach policy to role success", args)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档关联角色权限

解除角色权限 通过以下代码关联角色权限

```
 roleName := "test_sdk_go_group"
policyName := "test_sdk_go_policy"
args := &api.DetachPolicyToRoleArgs{
 RoleName:   roleName,
 PolicyName: policyName,
}
err = client.DetachPolicyFromRole(args)
if err != nil {
 fmt.Println("Detach policy to role failed", err)
} else {
 fmt.Println("Detach policy to role success", args)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档解除角色权限

列举角色的权限 通过以下代码列举角色权限

```
 roleName := "test_sdk_go_group"
result, err := client.ListRoleAttachedPolicies(roleName)
if err != nil {
 fmt.Println("List role attached policy failed", err)
} else {
 fmt.Println("List role attached policy success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举角色的权限

列举权限绑定的实体 通过以下代码列举权限绑定实体

```
 policyId := "test_policy_id"
entities, err := client.ListPolicyAttachedEntities(policyId)
if err != nil {
 fmt.Println("List policy attached entities failed", err)
} else {
 fmt.Println("List policy attached entities success", entities)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举权限授予的所有主体

组管理接口

创建组 通过以下代码创建组

```
 name := "test_group_sdk_go"
args := &api.CreateGroupArgs{
 Name:        name,
 Description: name,
}

result, err := client.CreateGroup(args)
if err != nil {
 fmt.Println("Create group failed", err)
} else {
 fmt.Println("Create group success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档创建组

查询组 通过以下代码查询组

```
 name := "test_group_sdk_go"
result, err := client.GetGroup(name)
if err != nil {
 fmt.Println("Get group failed", err)
} else {
 fmt.Println("Get group success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档查询组

更新组 通过以下代码更新组

```
 name := "test_group_sdk_go"
args := &api.UpdateGroupArgs{
 Description: "newDes",
}
result, err := client.UpdateGroup(name, args)
if err != nil {
 fmt.Println("Update group failed", err)
} else {
 fmt.Println("Update group success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档更新组

删除组 通过以下代码删除组

```
 name := "test_group_sdk_go"
err = client.DeleteGroup(name)
if err != nil {
 fmt.Println("Delete group failed", err)
} else {
 fmt.Println("Delete group success", name)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档删除组

列举组 通过以下代码删除组

```
 result, err := client.ListGroup()
if err != nil {
 fmt.Println("Delete group failed", err)
} else {
 fmt.Println("Delete group success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举组

添加用户到组 通过以下代码添加用户到组

```
 userName := "test_user_sdk_go"
groupName := "test_user_sdk_go"
err = client.AddUserToGroup(userName, groupName)
if err != nil {
 fmt.Println("Add user to group failed", err)
} else {
 fmt.Println("Add user to group success", userName)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档添加用户到组

从组内移除用户 通过以下代码从组内移除用户

```
 userName := "test_user_sdk_go"
  groupName := "test_user_sdk_go"
  err = client.DeleteUserFromGroup(userName, groupName)
  if err != nil {
      fmt.Println("Add user to group failed", err)
  } else {
      fmt.Println("Add user to group success", userName)
  }
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档从组内移除用户

列举用户的组 通过以下代码列举用户的组

```
    userName := "test_user_sdk_go"
result, err := client.ListGroupsForUser(userName)
if err != nil {
 fmt.Println("List groups for user failed", err)
} else {
 fmt.Println("List groups for user success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举用户的组

列举组内用户 通过以下代码列举组内用户

```
 groupName := "test_user_sdk_go"
result, err := client.ListUsersInGroup(groupName)
if err != nil {
 fmt.Println("List user in group failed", err)
} else {
 fmt.Println("List user in group success", result)
}
```

> **提示：**
>
> - 详细的参数配置及限制条件，可以参考IAM API 文档列举组内用户

错误处理

GO语言以error类型标识错误，IAM支持两种错误见下表：

错误类型	说明
BceClientError	用户操作产生的错误
BceServiceError	IAM服务返回的错误

客户端异常

客户端异常表示客户端尝试向IAM发送请求以及数据传输时遇到的异常。例如，当发送请求时网络连接不可用时，则会返回BceClientError；当上传文件时发生IO异常时，也会抛出BceClientError。

服务端异常

当IAM服务端出现异常时，IAM服务端会返回给用户相应的错误信息，以便定位问题。常见服务端异常可参见IAM错误返回

版本变更记录

IAM GO sdk于[2022-10-13]首次发布

首次发布内容：

- 创建、查看、列表、更新、删除IAM用户
- 配置、查询、关闭用户控制台配置
- 创建、查看、列表、删除、启用、禁用AccessKey
- 创建、查看、列表、更新、删除IAM用户组
- 创建、列表、列表、更新、删除角色
- 创建、查看、列表、更新、删除、关联用户权限、解除用户权限、列举用户权限、关联组权限、解除组权限、列举组权限、关联角色权限、解除角色权限、列举角色的权限

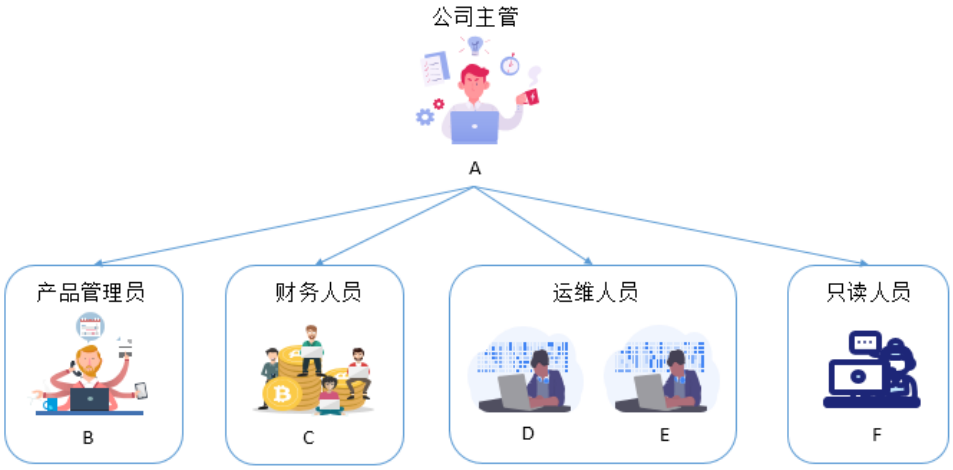
典型实践

用户管理与权限分配

场景描述

假如您是 A 公司的管理者，在创建公司时注册了云账号 Company-A，购买了多种云资源（如BCC、BOS、CDN、RDS等）。假设 A 公司的人员结构如下，公司里有多个员工需要操作这些云资源，比如有的负责购买，有的负责运维，还有的负责资源查看

及使用。由于每个员工的工作职责不一样，需要的权限也不一样。



对上述场景存在如下需求：

- 出于安全或信任的考虑，A 不希望将云账号密钥直接透露给员工，而希望能给员工创建独立的账号和权限。
- 所有员工账号进行的操作行为可审计。
- 不需要分别核算每个操作人员的成本，所发生费用统一计入主账号账单。

解决方案

针对以上场景，我们看看如何使用 IAM 来帮助您实现用户管理和资源权限的的分配。

第1步：给主账号开启登录保护

鉴于您之前可能与他人共用一个账号，账号泄露的可能性很高。建议您开启账号登录保护功能。

在您进行登录操作的时候，会给用户增加一层保护。该保护的措施是在正确输入账号和密码的前提下，还需要额外输入一种能证明身份的凭证。这样即使他人盗取用户密码，也无法登录用户账号，能够最大限度地保证账号安全。

第2步：创建子用户及用户组

根据上述企业场景，您需要分别给A、B、C、D、E、F用户分别创建不同的子用户账号，然后创建对应的产品管理组、财务组、运维组和只读组，再将所有涉及的用户添加到对应的组织中去，创建详情请参考用户管理和用户组管理。

第3步：为子用户开通双因素验证

考虑到管理和财务相关操作一般都较为敏感，您可能会担心管理员的账号密码泄露会带来巨大的风险，那么您可以为这些子用户开启双因素验证，而且可以将账号密码和双因素认证设备交给不同的人员分开保管，这样可以做到必须两人同时在场时才能完成账户登录和某些敏感操作。

第4步：给不同的用户组分配权限

IAM 提供了多种系统策略供您选择使用。根据上述场景，可以分配如下权限：

- 公司主管 A：授予系统管理员权限，拥有管理所有百度智能云资源的权限；
- 产品管理员 B： 授予产品的管理权限，拥有创建、删除实例，并且对实例进行相关操作的权限；
- 财务人员 C：授予财务权限（FCFullControlPolicy），拥有管理财务中心的权限；
- 运维人员 D、E：授予相关产品的运维权限，可以进行资源的相关配置和查看，但是不可以进行资源的创建、购买和删除；
- 只读人员 F：可以授予HR或者其他参与人员只读权限，可以用来查看相关产品的资源列表、日志等，但无操作权限。

如果您觉得IAM提供的系统策略不能满足您的细粒度需求，还可以设置自定义策略，来实现细粒度的资源、实例级权限设置。比如，授予子用户某个BCC实例的的管理权限，而不具有所有BCC实例的管理权限等。

子用户操作审计

公司主管A为了能够了解资源被使用的情况，可以通过操作记录查询每个子用户的操作访问情况，以便及时掌握子用户对资源实例所进行的操作记录，用于进行安全分析、资源变更以及合规性审计。详情参考操作记录。

员工岗位变动

如果有员工转岗，想要更换权限，则只需将该员工更换所在的分组即可；如果有员工离职或者出现问题，则只需要禁用该员工的账号即可，IAM会自动禁用该子用户的所有权限；如果有新员工入职，则需要为新员工开通一个子用户账号并为该账号授予对应权限即可。

百度智能云合作伙伴开通子用户教程

面向对象：

百度智能云注册合作伙伴

使用场景：

需要开通子用户以访问合作伙伴课程中心

开通子用户具体步骤：

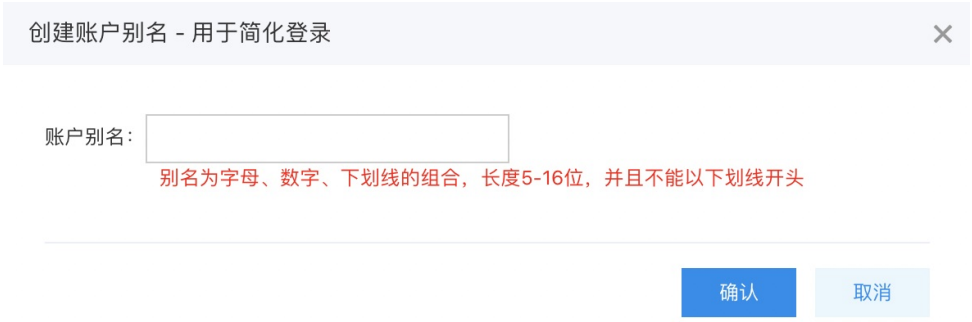
1. 登录百度智能云。

2. 鼠标移到管理控制台右上角的账户图标，选择“多用户访问控制”。

3. 在页面左侧“用户管理”下选择“子用户”。

4. 为了方便员工登录，强烈建议自定义一个账户别名，点击“子用户”页签中的“自定义”。

5. 设置“账户别名”，例如设置成公司的英文名，员工在登录时需要输入该账户别名。举例设置一个名字叫『mycompany』。



6. 点击“创建子用户”。

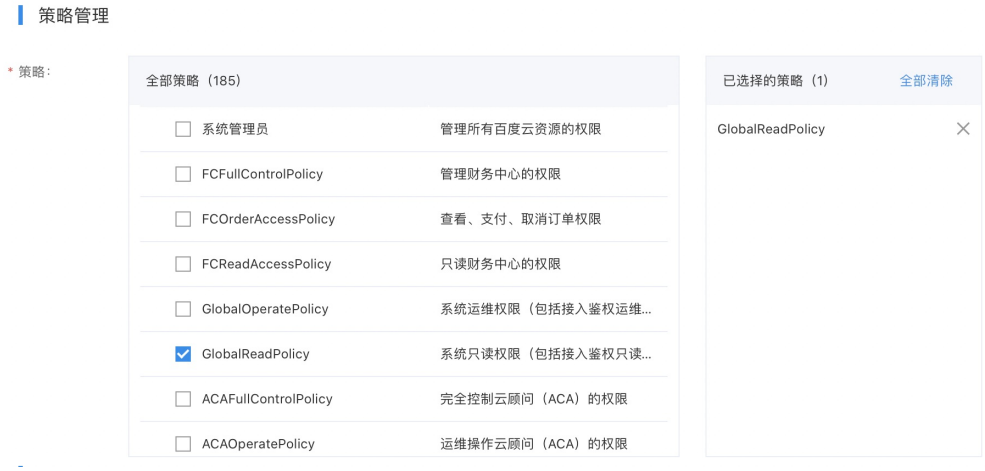
7. 在弹出的新建用户界面设置子用户的用户名和密码，您可以选择勾选“要求用户下次登录时必须重置密码”，则子用户在下次登录时，系统会要求子用户重设密码。

8. 在子用户管理列表中，点击”编辑权限”，为子用户进行授权。

9. 建议为子用户设置『系统只读权限』，完成后用子用户登录即可访问合作伙伴课程中心。

10. 如果子用户较多，可以在组管理批量设置权限，在“组管理”页签点击“创建新组”。

11. 设置组名，在策略管理中，为新建组的成员添加系统策略。



12. 在组员管理中，为新建组添加组员，组员继承组策略，同时拥有自身策略。

【开通子用户完成】

子用户登录具体步骤：

1. 在百度智能云登录界面选择『子用户登录』。

2. 在主账号位置填入账户别名（上文主账户管理员设置的mycompany），填入子用户登录名和密码（上文主账户管理员新增的子用户名称和密码）。

【子用户登录完成，课程可正常访问】

操作记录

操作记录（公测中）

用户可以通过Baidu CloudTrail对云账户进行监管、合规性检查、操作审核和风险审核。借助 CloudTrail，可以记录日志、持续监控并保留与整个云基础设施中的操作相关的账户活动。有关CloudTrail详情请参考云审计BCT文档。

常见问题

常见问题总览

子用户问题

- 子用户如何登录？
- 子用户如何修改密码？

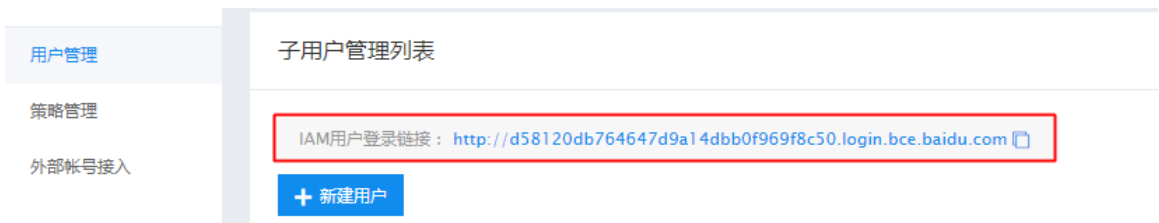
产品权限问题

- 权限介绍汇总
- 访问被拒绝

子用户问题

子用户如何登录？

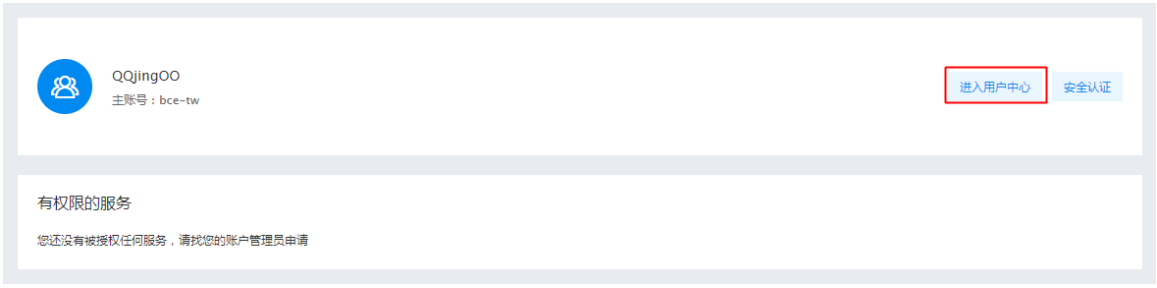
1. 【主账户】->【多用户访问控制】->选择“用户管理”页签，点击“IAM用户登录链接”。



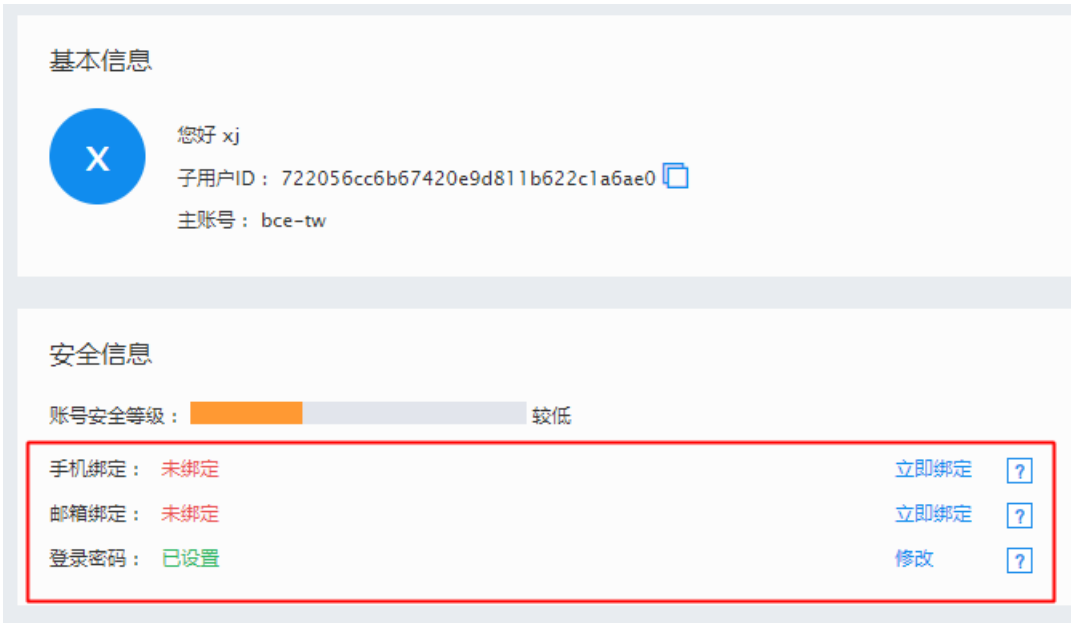
2. 选择登录方式，使用子用户登录百度智能云控制台。

子用户如何修改密码？

1. 子用户登录成功后，进入子用户的管理控制台，点击“进入用户中心”，来到用户信息页。



2. 在“安全信息”栏，可以进行手机绑定、邮箱绑定、登录密码修改等操作。



产品权限问题

权限介绍汇总

更多权限产品参考：目前支持产品线

- 了解BAE应用和环境的权限范围可参考BAE多用户协作。
- 了解BCC的权限范围可参考BCC多用户协作。

- 了解BOS的权限范围可参考BOS多用户协作。
- 了解CDN的权限范围可参考CDN多用户协作。
- 了解物接入IOT的权限范围可参考物接入IOT多用户协作。
- 了解TSDB的权限范围可参考时序数据库TSDB多用户协作。
- 了解RDS的权限范围可参考关系型数据库RDS多用户协作。
- 了解AI产品的权限范围可参考AI产品多用户协作。

访问被拒绝

问题描述：当我的子用户访问某个云产品服务时候，提示我“没有权限”。

1. 可以查看百度智能云目前支持产品线列表，确认该产品支持子用户权限使用，并详细了解权限力度。目前支持产品线
2. 验证您是否具有调用您请求的操作和资源的许可。
    - 子用户权限
    - 用户组权限

> 注意：在组员管理中，为新建组添加组员，组员继承组策略，同时拥有自身策略。