百度智能云

ECS 文档

【版权声明】

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

【商标声明】

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

【免责声明】

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

目录

目录 2
功能发布记录 4
产品描述 5
产品介绍 5
核心概念 5
特性 6
优势 7
应用场景 7
硬件规格 8
快速入门 9
安装激活 11
安全说明 11
激活边缘服务器 11
需要激活的边缘服务器使用方法 11
免激活的边缘服务器使用方法 13
硬件配置 16
网络配置 16
操作指南 17
概览 17
概览页面 17
服务器管理 18
概览 18
服务器上云 19
查看详情 23
应用部署 25
进入终端 27
更多操作 30
应用管理 33
官方应用 33
自定义应用 35
算法仓库 40
算法管理 40
算法编排 46
套件管理 63
套件概览 63
运维与支持 64
告警管理 64
应用日志 65
售后服务 66
边缘控制台 69

接入边缘控制台 69
使用边缘控制台 69
配置多用户访问控制 72
逻辑集群 76
概览 76
新建逻辑集群 77
应用部署 80
查看详情 82
扩展服务 86
Lucky平台 86
Lucky平台介绍 86
操作指南 87
API参考 93
EventHub事件中心 154
事件中心操作指南 154
EdgeInfer边缘视频推理引擎 157
API参考 157
AI算法 166
头部识别 166
打电话检测 167
人脸算法 168

功能发布记录

新功能发布

发布时间	功能概述
2023-04	• 扩展服务——EventHub事件中心上线 • 扩展服务——边缘视频推理引擎上线
2023-03	• 扩展服务——AI算法上线
2023-02	• 扩展服务——Lucky平台上线
2023-01	• 其他服务器上云功能上线 • ECS服务器上云功能更新
2022-12	• 支持服务器线上下单购买 • 新增售后服务功能
2022-11	• 批量激活服务器功能上线 • 支持批量部署应用
2022-10	• 新增算法编排功能 • 支持添加多合一算法 • 上线官方子算法列表
2022-09	• 新增进入终端功能 • 上线远程服务功能 • 支持分享终端功能
2022-08	• 新增算法仓库模块 • 官方算法、自定义算法功能迁移至算法仓库模块
2022-08	• 上线多款一体机套件
2022-07	• 上线多款官方算法
2022-06	• 上线自定义算法功能
2022-05	• 上线多款官方应用

2022-04	• 上线套件管理功能 • 支持一体机套件部署能力
2022-03	• 上线算法部署功能
2022-03	• 上线边缘控制台，支持应用管理、监控告警、网络配置等功能
2022-02	• 增加运维与支持模块 • 新增资源监控，以及告警管理功能 • 新增应用日志功能
2022-01	• 上线自定义应用功能

产品描述

产品介绍

百度智能云边缘服务器（Edge Computing Server，简称ECS）是一种基于多种规格的硬件服务器和功能丰富的软件管理平台的软硬一体化服务，为企事业单位的边缘业务提供一站式、快速落地的行业智能化解决方案和边缘计算能力，助力产业降本增效和智能化转型。

在车路协同、智慧工地、工业互联网等多种场景下，ECS都有着广泛的应用。凭借高性价比、低能耗、类型丰富的硬件，适应复杂极端的边缘环境、满足本地计算需求。通过功能丰富的云端、边缘管控平台，支持本地业务的快速、安全部署。同时提供场景化的AI能力，支持第三方业务系统和应用的部署。

核心概念

边缘服务器

与服务配套的一体机产品，用于承载边缘计算业务的应用和算法等。

边缘服务器管理平台

边缘服务器ECS的云端管理平台，提供边缘服务器资源的云上管理、控制、监控等能力。

ECS边缘控制台

边缘服务器ECS的本地管理平台，可离线管理边缘服务器的网络配置、应用等。

应用

边缘服务器为边缘计算业务提供的应用软件。

官方应用

ECS为用户提供的官方应用软件。

自定义应用

由用户自助创建的应用软件。

一体机套件

由多款应用、算法构成的软件组合，用于支持特定场景的业务需求。

官方套件

ECS为用户提供的官方一体机套件。

部署

将应用装载到边缘服务器上的动作称为部署。

监控与告警

ECS拥有对于边缘服务器资源使用情况的监控能力，方便用户对重要资源进行管理。并可以通过设置告警策略实现自动告警。

资源使用监控

ECS平台对服务器的状态、内存、磁盘、CPU、带宽、GPU、显存等资源进行监控，并展示具体的使用情况。

警戒线

支持用户自主设置各类资源的警戒红线、警戒黄线，作为紧急、重要告警的触发依据。

告警管理

支持用户查看全部服务器的当前及历史告警情况。

应用日志

提供对服务器上应用、容器的日志查询功能。

特性

云边容器引擎

- 提供轻量、易交付、易维护、易使用的云边容器引擎，用云原生的技术提升运维和应用管理效率。

算力资源调度

- 在算力资源有限的情况下，通过算子编排、算力调度，实现分时调度、动态启停等功能，最大化利用算力。

边缘自治

- 在断网或弱网环境下依然支持边缘服务器本地计算、分析等。

边云协同

- 采用百度自有引擎，支持边缘集群管理，实现云-边-端三位一体的边云协同。

智能运维

- ECS平台支持快速部署、监控告警、官方及自定义应用远程下发等，实现智能运维。

快速交付

- 边缘服务器硬件支持开箱即用，实现解决方案快速集成和落地。

硬件维保

- 硬件设备客户全程免维护，由百度提供全面的现场级运维和维保服务。

优势

- **完整的硬件生态**
  - 边缘盒子、边缘服务器、边缘一体机，灵活满足不同算力需求的的边缘场景。
  - 硬件系列产品高性能、易拓展配置，满足高算力需求。
  - 短机型、低能耗、耐极端环境，支持各种极端条件下的部署。
- **软硬件一体化交付**
  - 硬件产品+边缘服务平台一体化交付，支持完全私有化部署和运营。
  - 结合不同应用场景的具体需求，进行不同类型的硬件组合，形成完整解决方案，低成本快速集成和落地。
- **边云协同，支持算力资源调度**
  - 采用百度自有的算力资源优化调度引擎AI-Edge和单机容器引擎，支持边缘集群管理，实现边云协同。
  - AI能力业内领先，内置多种应用及算子，支持算法定制服务，并提供多种算子套餐，根据应用场景及具体需求自由选配。
  - 在资源有限的情况下，通过算子编排、算力调度，实现智能资源调度，最大化利用算力。
- **极致的运维服务**
  - 软硬件快速交付、快速部署、开箱即用，网络一键配置、主机一键纳管。
  - 软件平台由百度团队进行日常运维和升级服务，用户也可以远程查看管理。
  - 硬件设备的交付、安装及出现损坏等问题，客户全程免维护，由百度提供全面的现场级运维和维保服务。

应用场景

智慧工地

传统工地施工现场的违规操作缺少监管手段，现场设施、明火烟雾等无法实时监控。基于边缘服务器硬件+AI算法，构建智慧工地解决方案，实现工地的监控告警和闭环处理。

优势

- 本地快速部署，搭建本地计算单元；

- 边缘运维平台，支持云边运维、云边数据监控、应用远程下发、远程策略更新；
- 成熟算法和方案，支持监控告警闭环处理；
- 对卸料平台、配电箱、危大工程等进行实时监控、现场报警，保证工地安全。

智慧灯杆

传统灯杆功能单一、改造难度大。城市停车难，违停需要依靠人力巡检治理，空闲车位也无法无法实现统一管理和利用。智慧灯杆融合AI算力，实现灯杆的功能多样化，缓解城市停车压力。

优势

- 借助边缘服务器，支持现有灯杆利旧改造；
- 提供违停监管、智能车位推荐等算法方案，助力城市停车管理；
- 成熟的算法和方案，助力智慧城市管理，如城市环境监测、人流量监测等。

智慧零售

通过对零售店铺的实时数据进行智能分析，得出人脸+行为+轨迹等统计数据，管理端可以即时查看客流情况、客户属性、趋势及繁忙情况，基于分析结果来优化服务流程，提升运营效率和服务能力。

优势

- 通过人脸识别技术对进店人员身份进行会员识别，提供会员身份针对性服务；
- 捕捉用户停留时长、智能分析其行进轨迹，辅助推断潜在需求点；
- 对店员规范着装、店员在岗情况识别等，助力门店规范管理；
- 提供智能客流量分析、商品热力图等，辅助分析决策。

智慧加油站

传统加油站采用人工监控，存在监管不及时、成本高、易疏漏等问题。百度智慧加油站基于边缘服务器，提供智能监管、作业督查的一体化方案，助力风险防控和规范作业管理。

优势

- 提供吸烟、烟火、消防通道占用等安全隐患监测；
- 通过视频智能分析技术，实现监控场景智能防范；
- 高效、精确、不间断地识别出各类安全事故，实现提前预警。

硬件规格

边缘服务器硬件分为边缘盒子系列、边缘服务器、边缘整机柜。

目前，ECS提供的硬件规格如下表所示。

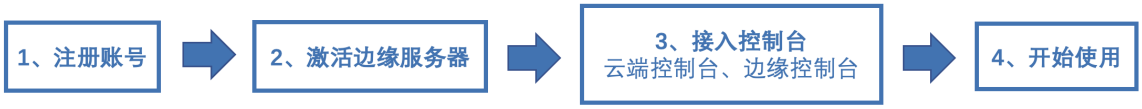
名称	特性	规格	使用方式
ECS-N-B01	• -40℃低温缓起 • 支持宽温工作 • 支持Zigbee通信	• 模组：Jetson Nano/TX2 NX/Xavier NX • 网络：4G/5G/Wi-Fi/Zigbee • IO：1 RS232/485+4 USB • 工作环温：-40-65℃ • 防护等级：IP40	需要激活
ECS-N-B02	• 无风扇设计 • 支持多种无线网络 • 2.5寸硬盘扩展	• 模组：Jetson Xavier NX • 网络：4G/5G/Wi-Fi • IO：2 RJ45+2 USB • 工作环温：-25-60℃ • 防护等级：IP50	免激活
ECS-N-B03	• 4路千兆PoE+接口 • 高效直触式被动散热 • 高防护等级，满足工业场景需求	• 模组：Jetson AGX Xavier • 存储：32 GB 256-bit LPDDR4 • IO：2 RS232 + 2 RS485 + 2 USB • 工作环温：-20-60℃	免激活
ECS-N-B04	• 高效主动散热设计 • 工业级防静电高抗振等级 • 体积小，高性能	• 模组：Jetson Nano • 存储：4 GB 64 bit LPDDR4 • IO：1 RS232 + 3 USB • 工作环温：0-40℃	免激活

快速入门

快速入门流程

本文帮助您快速了解百度智能云边缘服务器ECS的基础能力，包括注册账号、激活边缘服务器、接入控制台等。

基本操作流程如下图所示：



准备工作

1. 需要准备以下物品： 盒子、电源线、鼠标键盘等外接设备。
2. 将盒子正常拿出后接电和其他外接设备。

注意：

1.盒子准备好后需要连接网络。

2.使用网线的话可以直接连接LAN和WAN口后连接上显示屏登录入机器后打开黑窗口输入ifconfig查看IP即可。

注册与登录

（1） 注册并登录百度智能云平台，请参考 注册 和 登录。

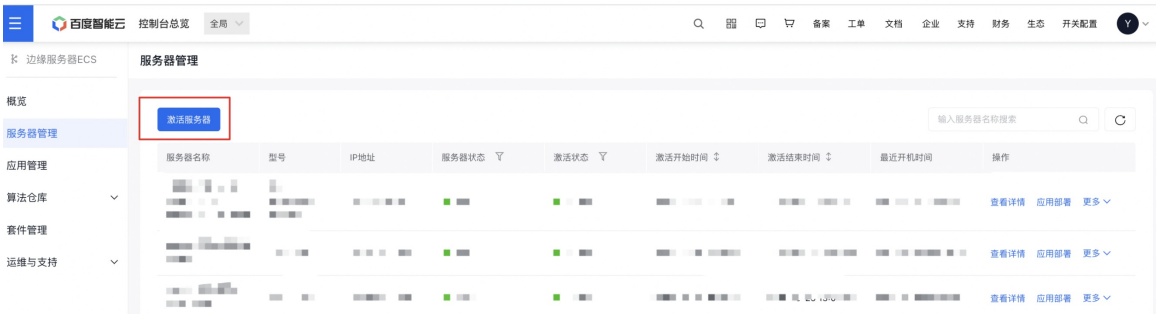
（2） 完成实名认证，操作细节请参考 实名认证，实名认证后才可购买百度智能云边缘服务器ECS。

激活边缘服务器

对于需要进行激活才能使用的服务器（如ECS-N-B01），请参考激活流程：需要激活的边缘服务器使用方法；

将盒子激活到ECS中心平台

1.登录平台进入"服务器管理"页面后点击“激活服务器”，进入“服务器激活”页面。



2.按照提示选择服务器型号，并填入服务器名称、服务器序列号，对服务器硬件进行设置和安装,完成点击"立即激活"按钮，激活服务器。

3.对于支持免激活开箱即用的服务器（如ECS-N-B02），请参考使用流程：免激活的边缘服务器使用方法；

接入控制台

ECS拥有两种控制台：云端控制台、边缘控制台。

接入云端控制台

（1） 安装边缘服务器并完成网络配置；

（2） 在工作电脑的浏览器中，输入边缘服务器网口IP地址192.168.100.198，访问ECS边缘控制台，即可开始使用ECS的边缘控制台服务；

（3） 首次登录ECS边缘控制台，输入默认的用户名和密码（联系客服获取）即可登录。后续若需要修改登录密码，可以进入边缘控制台进行修改。

接入边缘控制台

登录成功后，在 云管理控制台 导航栏中选择“产品服务 > 存储和CDN > 边缘服务器ECS”，进入 ECS云端管理平台 ，即可开始使用ECS的云端控制台服务。

开始使用

完成接入控制台的操作后，可以参考操作指南，开始使用服务器与控制台的各种功能。

安装激活

安全说明

在安装所有型号的边缘服务器之前，请先阅读并遵循安全说明。

通用安全注意事项

1. 为保障人身和设备安全，在安装、操作和维护边缘服务器时，请根据随机附带的安装说明书安装。
2. 接触任何导体表面或接线端子之前应使用电压表测量接触点的电压，确认无电击危险。
3. 请使用随机配备的电源线等供电组件，不要随意更换电源线或插头。

安装注意事项

1. 边缘服务器应在符合安装环境要求的环境下使用，否则可能造成设备故障，由此引发的设备功能异常或部件损坏、人身安全事故、财产损失等不在设备质量保证范围之内。
2. 切勿将任何物体塞入边缘服务器的开孔处。
3. 安装边缘服务器时，请确保安装牢固，避免跌落。
4. 如果出现以下任何情况，请从电源插座拔下电源线插头，并与售后服务部门联系：

- 电源线或电源插头已损坏。
- 边缘服务器被水淋湿。
- 边缘服务器损坏。
- 按照操作说明进行操作时，边缘服务器不能正常工作。

激活边缘服务器

需要激活的边缘服务器使用方法

百度智能云拥有多种型号的边缘服务器，部分边缘服务器在使用前，需要对服务器进行激活。

此处以ECS-N-B01边缘服务器为例，说明边缘服务器的激活方式。

准备工作

- 边缘服务器已经安装完成。
- 安装电源线，确认电源灯常亮。
- 注册百度智能云账号。

- 开通边缘服务器服务。

激活流程

边缘服务器的激活流程包含两个步骤：配置服务器网络、激活服务器。

配置服务器网络

服务器网络有两种配置方式：通过Wi-Fi热点配置、通过网线进行配置。

通过Wi-Fi热点进行配置

1. 准备一台可以具备网络功能的手机（或工作电脑），打开手机（或工作电脑）的Wi-Fi列表，连接边缘服务器产生的Wi-Fi热点；
2. 打开浏览器，输入网址：ecslogin.cn，输入默认的账号和密码（联系客服获取），进入网络配置页面；
3. 根据提示，对边缘服务器的网络进行配置；
4. 输入相关信息，验证通过后，即为网络配置完成。

通过网线进行配置

1. 准备一台可以正常使用的工作电脑，通过网线，将工作电脑与边缘微服务器连接。
2. 在工作电脑上，进入网络修改页面，并对IP地址、子网掩码进行如下修改：

IP地址：192.168.100.100

子网掩码：255.255.255.0

3. 设置完成后，在工作电脑的浏览器中，输入边缘服务器网口IP地址192.168.100.198，进入ECS边缘控制台，对边缘服务器的网络进行配置。

4.输入相关信息，验证通过后，即为网络配置完成。

激活服务器

完成网络配置后，可以激活边缘服务器。您可以直接扫描边缘服务器壳体标签上的二维码，激活边缘服务器，也可以登录边缘服务器管理平台，根据提示信息，激活边缘服务器。

扫描二维码激活

1. 找到张贴在边缘服务器壳体上的硬件标签，使用已连网且具有扫描功能的手机，扫描二维码。

以ECS-N-B01型号边缘服务器为例，硬件标签张贴在一体机底部。

2. 使用百度智能云账号，登录激活流程页面。
3. 在激活流程页面，按照提示信息，开始激活流程。
4. 根据提示，检查边缘服务器，确认电源线连接正常、电源灯常亮，确认网线连接正常、网口灯闪烁。
5. 确认服务器信息，然后单击"立即激活"，完成边缘服务器的激活。

在边缘服务器管理平台激活

1. 使用百度智能云账号，登录边缘服务器管理平台。
2. 在左侧导航栏点击"服务器管理"，进入管理页面后，点击"激活服务器"。
3. 在激活服务器页面，根据提示信息，选择您要激活的边缘服务器型号、为您的边缘服务器命名，并填写张贴在边缘服务器壳体标签上的服务器序列号。
4. 检查边缘服务器，确认电源线连接正常、电源灯常亮，确认网线连接正常、网口灯闪烁。
5. 点击"立即激活"，完成边缘服务器的激活。

免激活的边缘服务器使用方法

百度智能云拥有多种型号的边缘服务器，部分边缘服务器支持开箱即用，不需要在使用前对服务器进行激活。

此处以ECS-N-B02边缘服务器为例，说明免激活边缘服务器的使用方式。

准备工作

- 边缘服务器已经安装完成。
- 安装电源线，确认电源灯常亮。
- 注册百度智能云账号。
- 开通边缘服务器服务。

使用流程

免激活边缘服务器的使用流程分为三个步骤：配置服务器网络、登录ECS边缘控制台、绑定至边缘服务器管理平台。

配置服务器网络

服务器网络有两种配置方式：通过Wi-Fi热点配置、通过网线进行配置。

通过Wi-Fi热点进行配置

1. 准备一台可以具备网络功能的手机（或工作电脑），打开手机（或工作电脑）的Wi-Fi列表，连接边缘服务器产生的Wi-Fi热点；
2. 打开浏览器，输入网址：ecslogin.cn，输入默认的账号和密码，进入网络配置页面；
3. 根据提示，对边缘服务器的网络进行配置；
4. 输入相关信息，验证通过后，即为网络配置完成。

通过网线进行配置

1. 准备一台可以正常使用的工作电脑，通过网线，将工作电脑与边缘服务器连接。
2. 在工作电脑上，进入网络修改页面，并对IP地址、子网掩码进行如下修改：

IP地址：192.168.100.100

子网掩码：255.255.255.0

3.输入相关信息，验证通过后，即为网络配置完成。

登录ECS边缘控制台

1.在工作电脑的浏览器中，输入边缘服务器网口IP地址192.168.100.198，访问ECS边缘控制台。

2.首次登录ECS边缘控制台，输入默认的用户名和密码（联系客服获取）即可登录。后续若需要修改登录密码，可以进入边缘控制台进行修改。

3.登录ECS边缘控制台，开始使用您的边缘服务器。

绑定至边缘服务器管理平台

在完成以上两个步骤之后，您已经可以在边缘网络下正常使用您的边缘服务器。

为了使您得到更加丰富的体验，推荐将边缘服务器绑定至边缘服务器管理平台，以获取更多服务。

您可以直接扫描边缘服务器壳体标签上的二维码进行绑定，也可以登录边缘服务器管理平台，根据提示信息进行绑定。

扫描二维码绑定

1. 找到张贴在边缘服务器壳体上的硬件标签，使用已连网且具有扫描功能的手机，扫描二维码。

以ECS-N-B02型号边缘服务器为例，硬件标签张贴在一体机底部。

2. 使用百度智能云账号，登录激活流程页面。
3. 在激活流程页面，按照提示信息，开始绑定流程。
4. 根据提示，检查边缘服务器，确认电源线连接正常、电源灯常亮，确认网线连接正常、网口灯闪烁。
5. 确认服务器信息，然后单击"立即激活"，完成边缘服务器的绑定。

在边缘服务器管理平台绑定

1. 使用百度智能云账号，登录边缘服务器管理平台。
2. 在左侧导航栏点击"服务器管理"，进入管理页面后，点击"激活服务器"。
3. 在激活服务器页面，根据提示信息，选择您要绑定的边缘服务器型号、为您的边缘服务器命名，并填写张贴在边缘服务器壳体标签上的服务器序列号。
4. 检查边缘服务器，确认电源线连接正常、电源灯常亮，确认网线连接正常、网口灯闪烁。
5. 点击"立即激活"，完成边缘服务器的绑定。

硬件配置

网络配置

有线网络配置

可以参照以下步骤，进行有线网络配置。

1. 当前默认IP是192.168.1.237
2. 使用电脑直连，或者通过IP网段配置相同的路由器连接盒子网络端口，使用SSH登录SSH root@192.168.1.237
3. 也可以直接外接显示器和键盘，直接使用用户名和密码进入
4. 用户名和密码是root和baidu_ecs
5. 使用vim或其他文本编辑器，修改/etc/network/interfaces.d/eth0

auto eth0

iface eth0 inet static

address 192.168.1.180 改成对应的IP

netmask 255.255.0.0

gateway 192.168.1.1 改成对应的网关

6. 直接重启盒子即可

或者重新启动网卡，需要外接显示器和键盘

Ifconfig eth0 down

Ifconfig eth0 up

操作指南

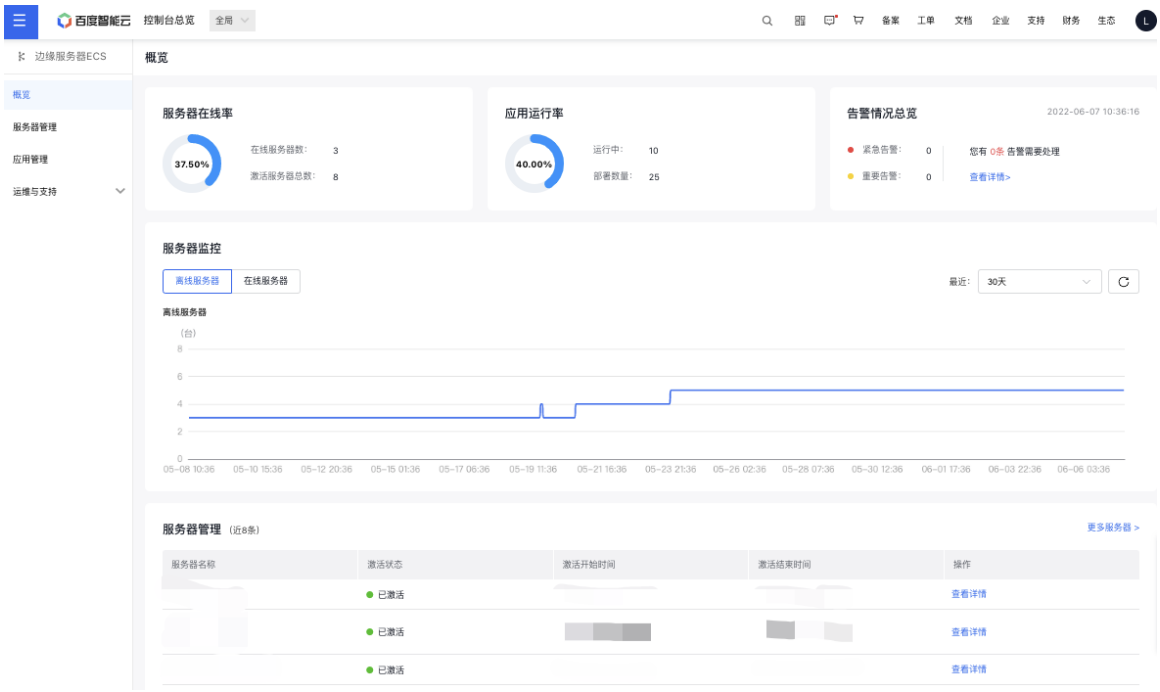
概览

概览页面

在概览页面，可以查看账号名下已激活的全部边缘服务器的在离线状态、应用运行率、告警情况、部署信息等。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“概览”，进入服务器信息概览页面。



服务器在线率

展示账号名下已激活的服务器总数、在线的服务器数量，以及服务器在线率。

应用运行率

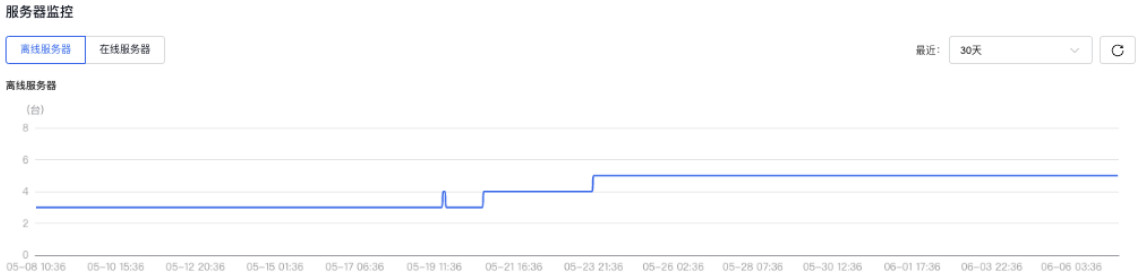
展示已部署在服务器上的应用总数、运行中的应用数量，以及应用运行率。

告警情况

展示当前的紧急告警、重要告警数量，也可以点击"立即处理"按钮，对当前告警进行查看处理。

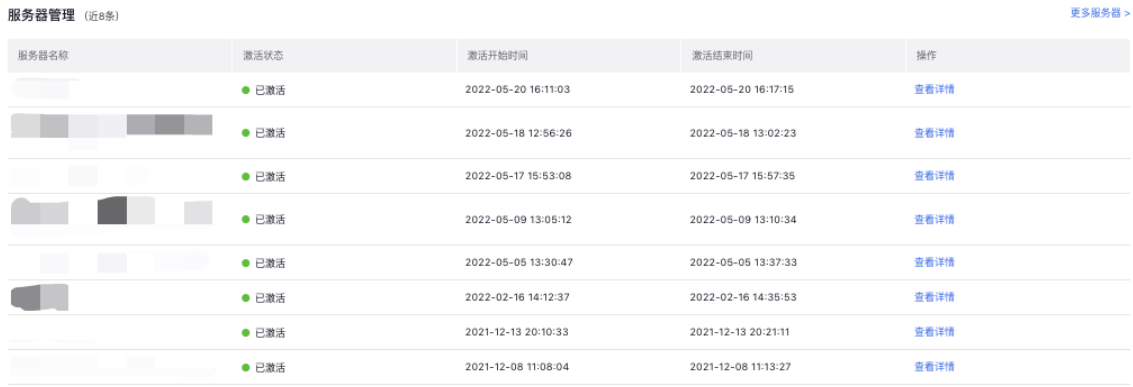
服务器监控

展示选定时间段内的离线服务器数量、在线服务器数量。



服务器管理

展示已激活的服务器信息，也可以点击"查看详情"，以查看每台服务器的具体情况。



部署信息

展示服务器的部署地址等部署情况。

服务器管理

概览

服务器管理主要分为三个功能模块，分别为：**概览**、**上云**和**操作**。

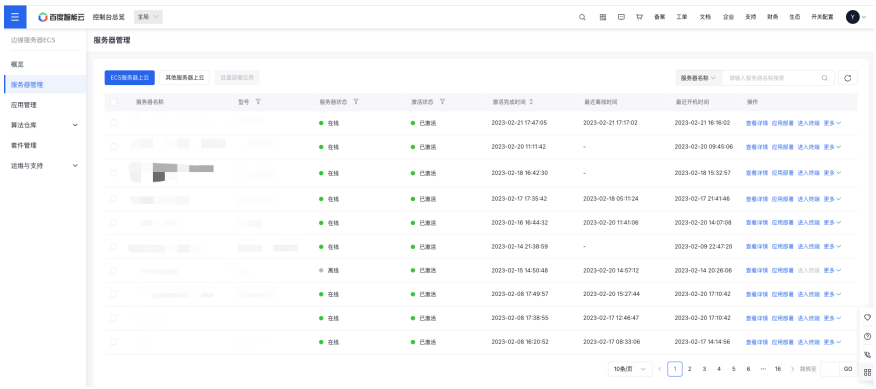
概览展示用户全部服务器的名称、型号、状态等信息；

上云用于进行ECS服务器、其他服务器的上云纳管；

操作包括查看详情、应用部署、进入终端，以及批量部署应用等多种操作。

概览

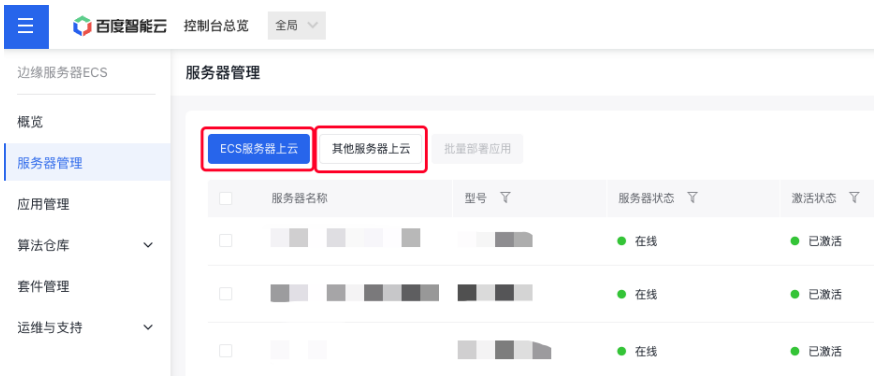
通过服务器列表，展示用户全部服务器的名称、型号、状态等信息。



平台支持根据服务器名称、服务器序列号进行搜索。选定条件、输入关键字后，可以进行对应搜索。

上云

平台支持ECS服务器、其他服务器的上云纳管。



操作

平台支持多种服务器操作，包括查看详情、应用部署、进入终端，以及批量部署应用等多种操作。

服务器上云

ECS服务器上云

对于ECS服务器（百度智能云边缘服务器，已预装ECS固件），可以使用"ECS服务器上云"功能，完成服务器的上云管理。

准备工作

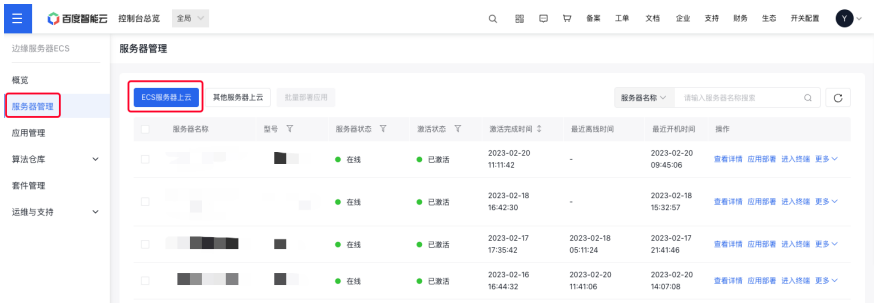
1. 将待上云的ECS服务器拆封取出，并完成安装；
2. 插入电源线，确认电源线已正确安装，且电源灯常亮；
3. 将服务器联网，确认服务器一端的网线已插入LAN1网口，并且网口灯闪烁；
4. 准备工作完成，开始进行上云操作。

上云步骤

1. 登录ECS控制台；

2. 点击左侧导航栏“服务器管理”，进入服务器管理页面；

3. 点击“ECS服务器上云”按钮，进入操作页面；



4. 选择对应的服务器型号；

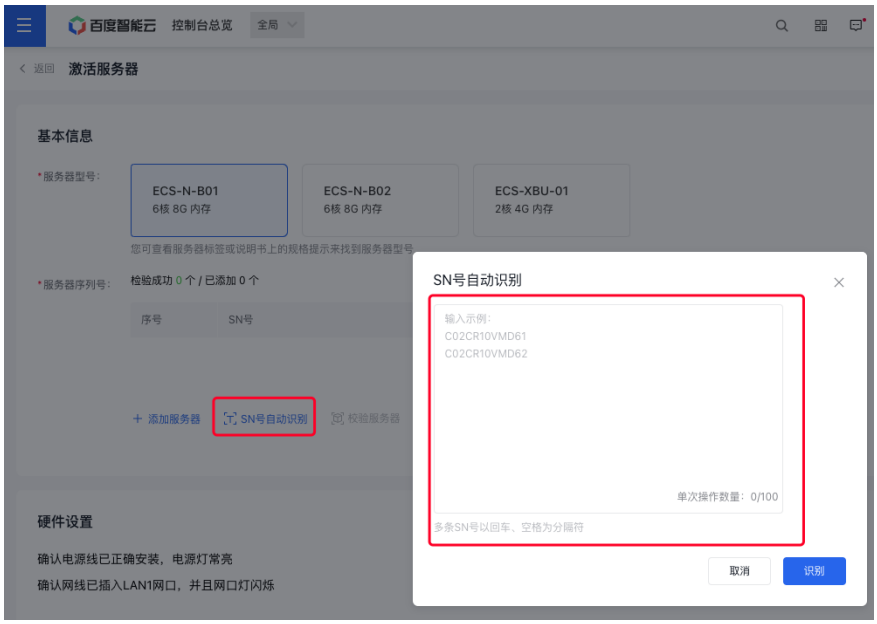
5. 填入服务器SN号；

平台支持两种填入SN号的方式：

（1）逐台添加：点击"添加服务器"按钮，服务器序列号列表中将自动增加一行，在空格中填入SN号即可，点击"删除"按钮可以删除整行内容；

（2）批量添加：点击"SN号自动识别"按钮，弹出"SN号自动识别"弹窗，输入SN号，或从文档、表格中将SN号复制粘贴到输入框，点击"识别"按钮，即可自动识别全部SN号。

注意：识别过程中，多条SN号以回车、空格为分隔符；单次识别上限为100条。



6. 校验服务器：SN号填好后，点击"校验服务器"按钮，对SN号的合规性进行校验；

注意：需要SN号、服务器型号均正确，才会判定为序列号合规。

7. 当校验结果全部为合规时，页面左下角"立即激活"按钮亮起，可以对全部已填入SN号的服务器进行批量激活；

8. 点击"立即激活"按钮，开始激活服务器，等待3-10分钟，完成服务器上云。

可以在"服务器管理" - "服务器列表"中，查看所有服务器的状态和上云过程。

其他服务器上云

对于非ECS服务器（其他厂商的服务器，未预装ECS固件），可以使用"其他服务器上云"功能，完成服务器的上云管理。

通过"其他服务器上云"功能，用户可以免费使用百度智能云边缘服务器管理平台，对任意品牌、任意厂商的边缘服务器进行纳管。

本功能可以帮助用户进行服务器存量管理、利旧升级等。

准备工作

1. 将待上云的ECS服务器拆封取出，并完成安装；

2. 插入电源线，确认电源线已正确安装，且电源灯常亮；

3. 将服务器联网，确认服务器一端的网线已插入LAN1网口，并且网口灯闪烁；

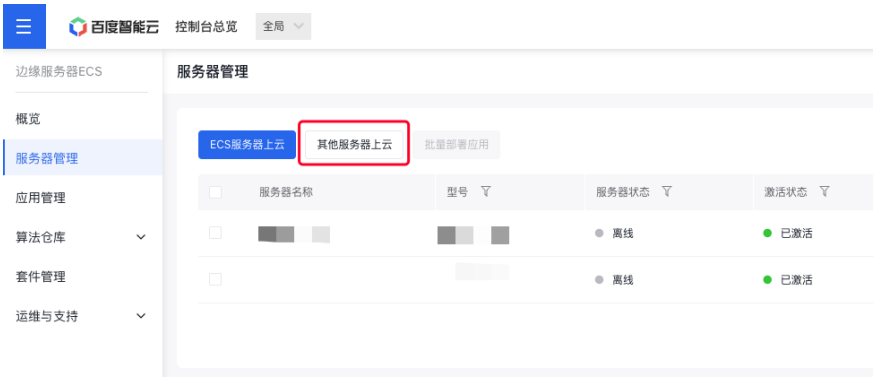
4. 准备工作完成，开始进行上云操作。

操作步骤

1. 登录ECS控制台；

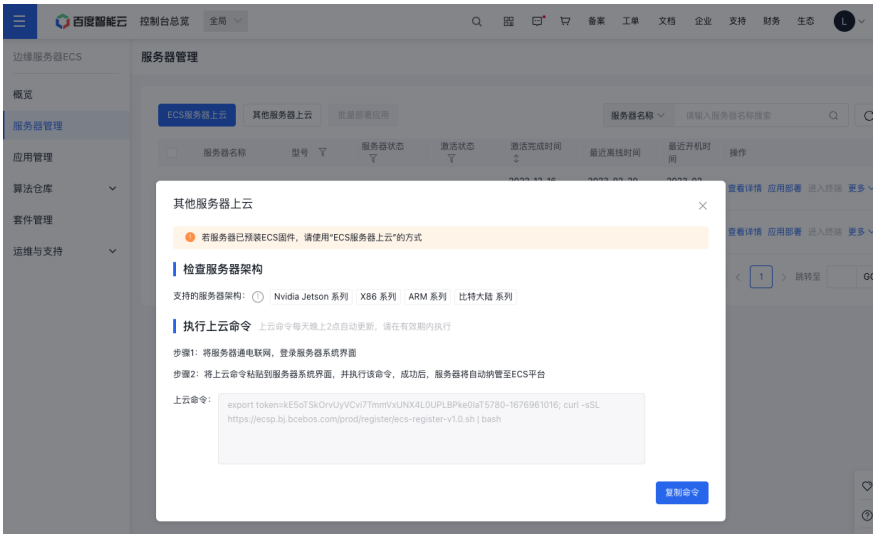
2. 点击左侧导航栏“服务器管理”，进入服务器管理页面；

3. 点击“其他服务器上云”按钮，弹出提示弹窗；



4. 根据弹窗中的提示，检查服务器架构是否支持；

当前支持纳管的服务器架构：Nvidia Jetson 系列、X86 系列、ARM 系列、比特大陆 系列。



5. 根据弹窗提示，执行上云命令；

步骤1：将服务器通电联网，登录服务器系统界面；

步骤2：将上云命令粘贴到服务器系统界面，并执行该命令，成功后，服务器将自动纳管至ECS平台；

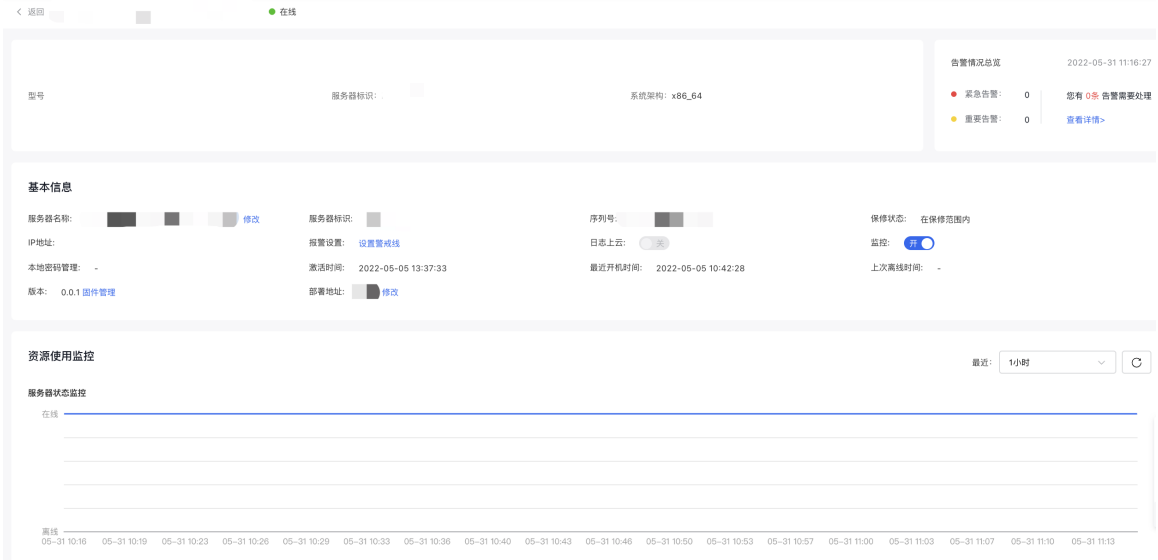
6. 可以在"服务器管理" - "服务器列表"中，查看所有服务器的状态和上云过程。

查看详情

进入单台服务器的详情页，可以查看该服务器的基本信息、资源使用情况等。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“服务器管理”，进入服务器管理页面。
3. 点击"查看详情"按钮，查看该服务器详情。



基本信息

展示该服务器的名称、IP地址、固件版本、部署地址等基本信息，部分信息支持用户自定义修改。

设置警戒线

支持自定义设置警戒线。警戒线设置后，警戒红线对应紧急告警，警戒黄线对应重要告警。

资源类型包括内存、磁盘、显存、CPU、带宽等等。

固件管理

支持固件版本的查看、升级、回滚等。

资源使用监控

平台对该服务器的资源使用情况进行实时记录，并根据用户设置的时间条件进行展示。

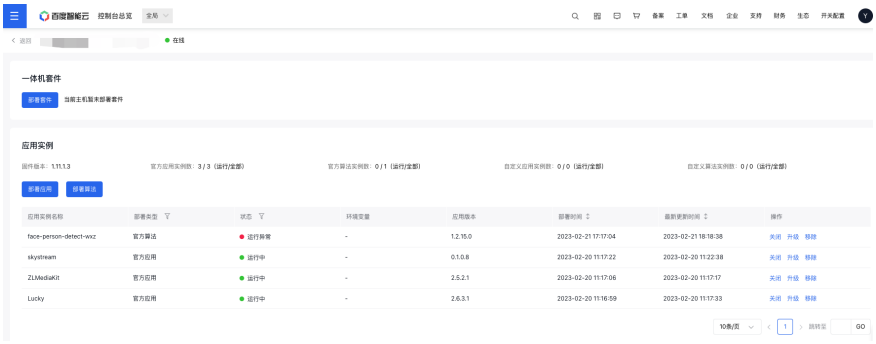
展示内容包括：服务器在离线状态、内存使用量（使用率）、磁盘使用量（使用率）、CPU使用量（使用率）、带宽使用量（使用率）等。

应用部署

通过ECS管理平台，用户可以在服务器上部署应用、算法、套件等，并支持批量部署应用功能。

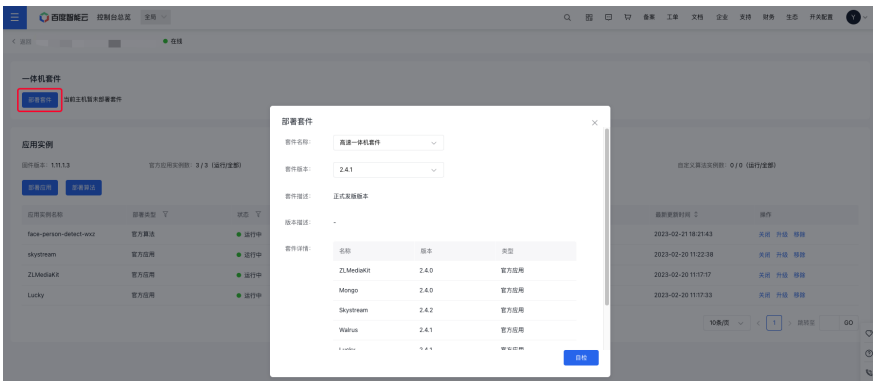
路径

1. 登录ECS控制台。
2. 点击左侧导航栏“服务器管理”，进入服务器管理页面。
3. 点击"应用部署"按钮，管理该服务器上的套件、应用、算法。



部署套件

支持部署套件，选择想要部署的套件名称、套件版本，自检通过后即可完成部署。



套件列表

展示本服务器上已部署的套件信息，包括名称、状态、版本等。

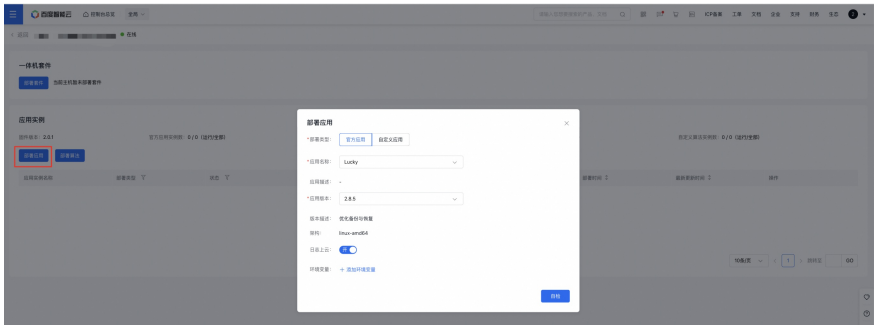
同时支持启动、关闭、升级、卸载等操作。

部署应用

点击"部署应用"按钮，开始在该服务器上进行应用部署。

选择想要部署的应用类型、应用名称、应用版本，是否开启日志上云，自检通过后即可完成部署。

注意：只有在线服务器，才能够进行应用部署。

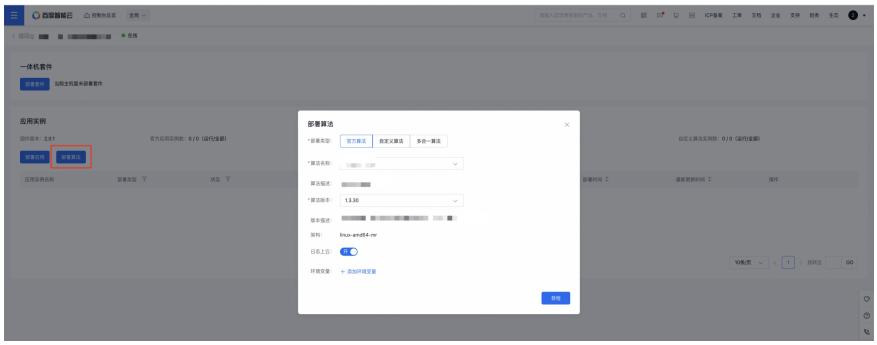


部署算法

点击"部署算法"按钮，开始在该服务器上进行算法部署。

选择想要部署的算法类型、算法名称、算法版本，是否开启日志上云，自检通过后即可完成部署。

注意：只有在线服务器，才能够进行算法部署。



应用实例

应用实例列表，展示本服务器上已部署的应用和算法的数量、名称、状态、版本等各类信息。

支持启动、关闭、升级、移除等操作。

批量部署应用

平台支持对服务器批量进行应用部署，操作步骤如下：

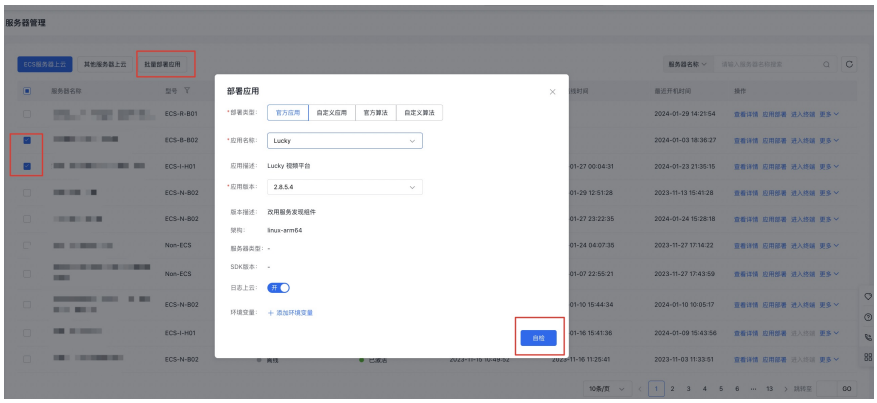
1. 登录ECS控制台；

2. 点击左侧导航栏“服务器管理”，进入服务器管理页面；

3. 在列表左侧勾选框中，勾选想要部署应用的服务器，可以使用筛选和搜索组件，快速选择对应服务器；

注意：所选服务器必须为在线服务器，否则"批量部署应用"按钮将不会亮起。

4. 勾选完毕后，点击"批量部署应用"按钮，选择想要部署的应用类型、应用名称、应用版本，然后点击"自检"按钮；



5. 查看自检结果，只有全部所选服务器的自检结果均为"通过"，"部署"按钮才会亮起，点击"部署"按钮，可以进行批量部署。

对于自检结果不通过的服务器，可以点击"删除"按钮，将该服务器从本次的部署列表中删掉。

6. 点击"部署"按钮后，平台将自动把所选应用部署在所选服务器上。

可以点击服务器"应用部署"按钮，进入对应的服务器应用详情页，查看部署进程。

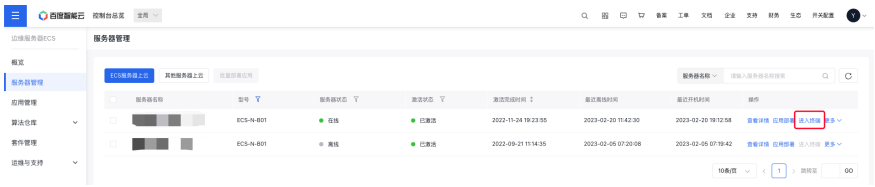
进入终端

ECS管理平台基于百度自有的云边容器引擎，打通云边通道，为用户提供服务器远程访问、远程操作、远程文件传输等服务。

通过"进入终端"功能，用户可以从云端平台，直接访问服务器主机，对服务器进行各项操作，并支持远程文件管理。

路径

1. 登录ECS控制台；
2. 点击左侧导航栏“服务器管理”，进入服务器管理页面；
3. 点击"进入终端"按钮，进入终端登录页面；

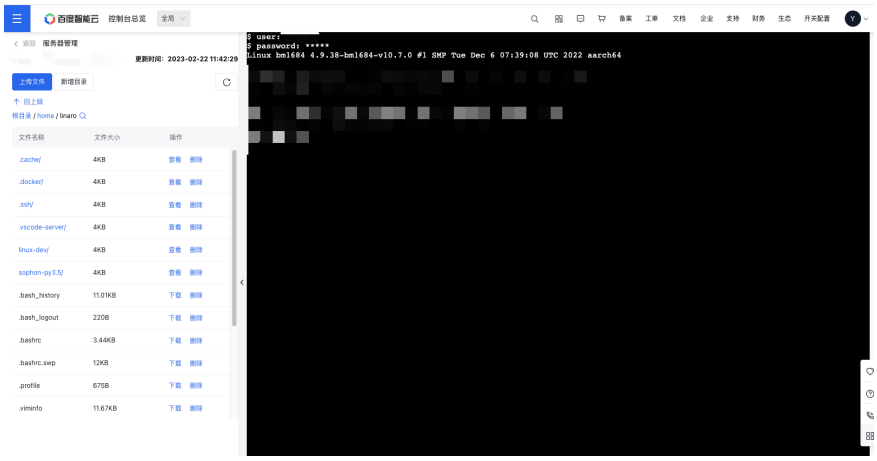


登录

1. 进入终端登录页面后，在右侧操作区，输入服务器对应的SSH账号密码进行登录；

2. 先输入账号，按回车键确认，再输入密码，再次按回车键确认，验证正确后，将自动进入服务器终端操作页面。

如果不知道服务器SSH账号密码，可以联系ECS团队或提交工单，以获取账号密码。



远程操作

登录成功后，用户可以在右侧操作区输入各类指令，对服务器进行远程操作、管理。

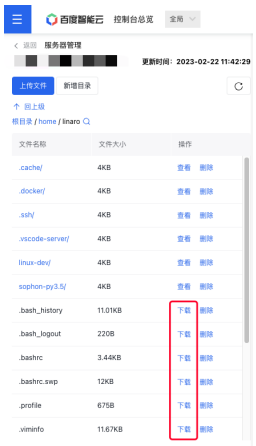
文件管理

文件上传

登录成功后，点击"上传文件"按钮，可以选择文件上传至边缘服务器。

文件下载

点击文件对应的"下载"按钮，可以直接将该文件下载至电脑端。



查看文件夹

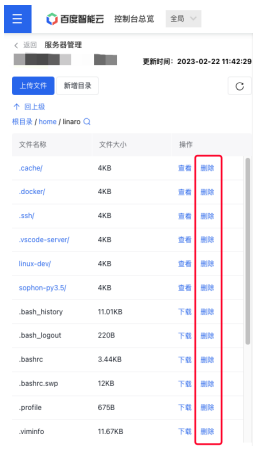
点击"查看"按钮，可以进入文件夹，查看文件夹内的文件。

新增目录

点击"新增目录"按钮，可以在服务器内增加一个新的文件夹。

文件删除

点击文件对应的"删除"按钮，可以直接将该文件或文件夹从服务器内删除。



更多操作

ECS管理平台基于百度自有的云边容器引擎，打通云边通道，为用户提供各类远程服务器管理功能。

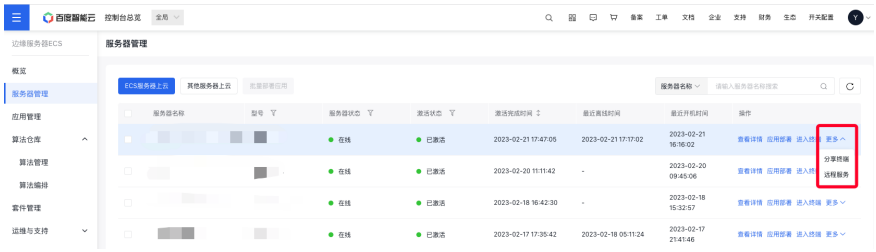
通过"**分享终端**"功能，用户可以将服务器的操作权限分享给第三方用户。第三方用户使用访问链接和密码，可以在规定时间内访问、操作服务器，获得"进入终端"相同的功能权限。

通过"**远程服务**"功能，用户可以从云端平台远程访问边缘服务器的边缘控制台、Lucky平台等各种IP地址，并进行远程操作。

"更多操作"中的功能，一般只面向白名单用户开放，如有需要，请联系ECS团队或提交工单，以获取白名单权限。

路径

1. 登录ECS控制台；。
2. 点击左侧导航栏“服务器管理”，进入服务器管理页面。
3. 点击“更多”按钮，进行更多服务器相关操作。



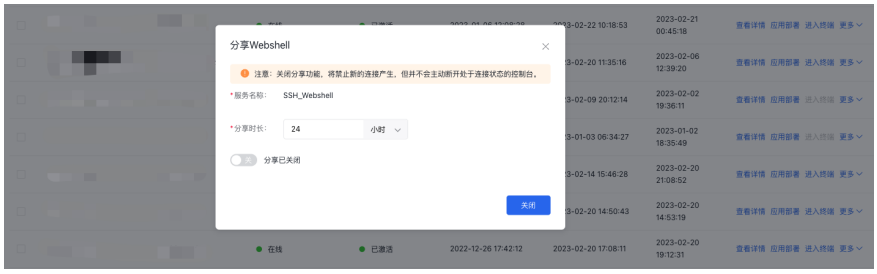
分享终端

通过"**分享终端**"功能，用户可以将服务器的操作权限分享给第三方用户。第三方用户使用访问链接和密码，可以在规定时间内访问、操作服务器，获得"进入终端"相同的功能权限。

注意：只有在线服务器，才能使用"分享终端"功能。

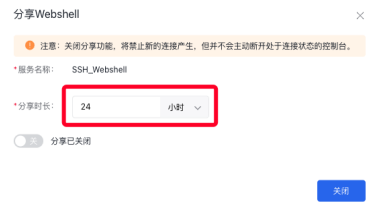
使用方法

1. 点击"更多"-"分享终端"，弹出分享终端弹窗；

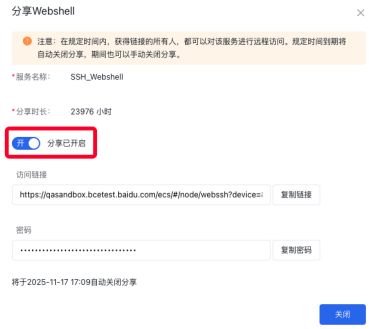


2. 输入分享时长，最长支持999天或23976小时；

在规定时间内，获得链接的所有人，都可以对该服务进行远程访问。规定时间到期将自动关闭分享，期间也可以手动关闭分享；

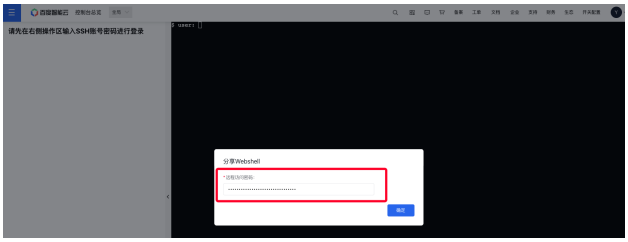


3. 启动分享开关，自动展示访问链接、访问密码；



4. 复制访问链接、访问密码，并分享给第三方用户；

5. 第三方用户点击访问链接，输入访问密码，可以进入服务器终端，使用与"进入终端"相同的功能；



6. 关闭分享开关，将立即关闭分享。

关闭分享功能，将禁止新的连接产生，但并不会主动断开处于连接状态的控制台。

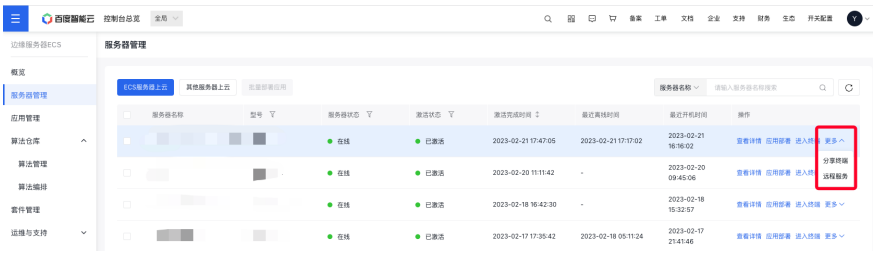
远程服务

通过"**远程服务**"功能，用户可以从云端平台远程访问边缘服务器的边缘控制台、Lucky平台等各种IP地址，并进行远程操作。

注意：只有在线服务器，才能使用"远程服务"功能。

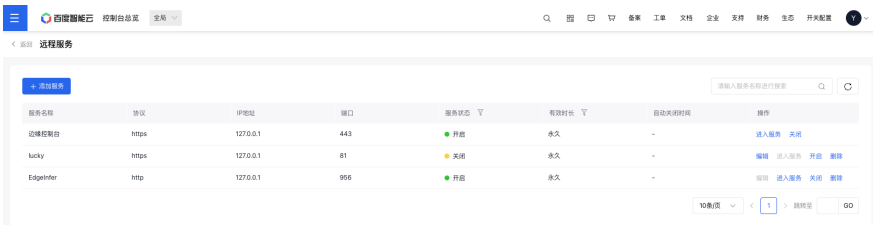
使用方法

1. 点击"更多"-"远程服务"，进入远程服务页面；

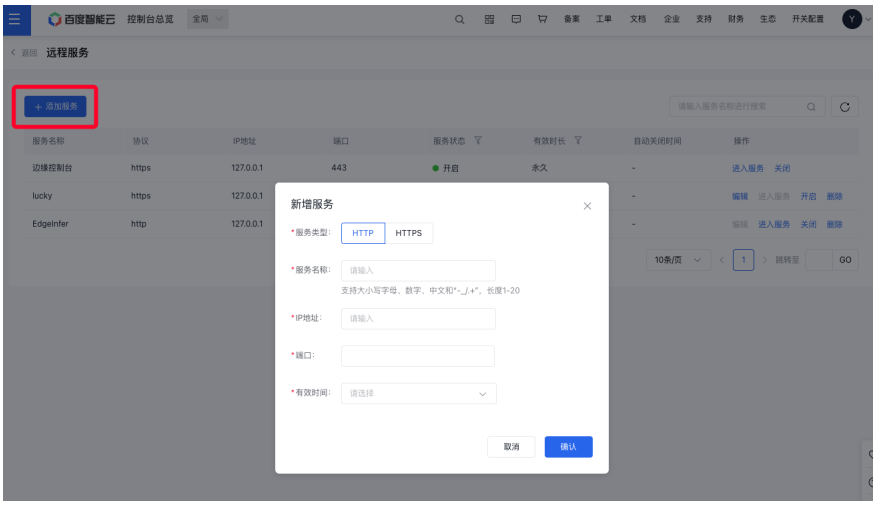


2. 在远程服务页面，展示服务器当前已有的全部服务名称、IP地址、端口等信息；

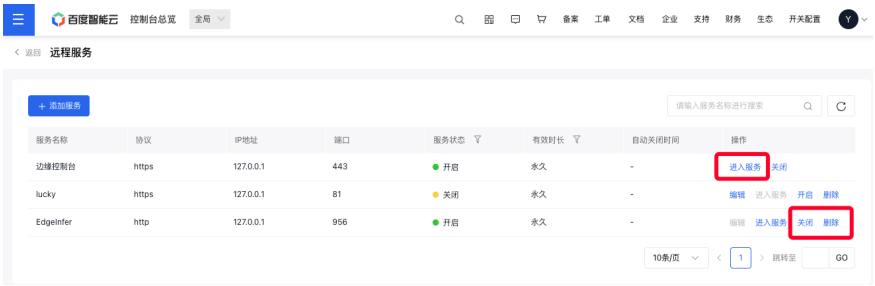
"边缘控制台"为平台自带的远程服务，且不能删除。用户可以通过云端控制台，直接进入边缘控制台，对服务器进行远程操作；



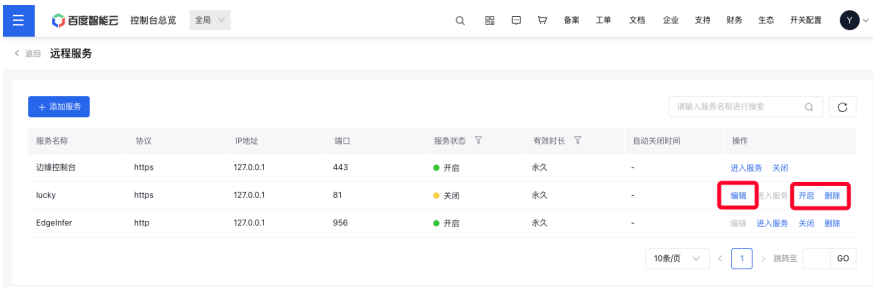
3. 用户可以添加更多远程服务，点击"添加服务"按钮，选择服务类型，填入服务名称、IP地址、端口等各项参数，设置有效时间，即可创建自定义的远程服务；



4. 对于开启中的远程服务，点击"进入服务"，可以进入该服务的对应地址，并支持关闭、删除服务等操作；



5. 对于已关闭的远程服务，点击"编辑"按钮，可以编辑更新该服务的各项参数，并支持开启、删除服务等操作。



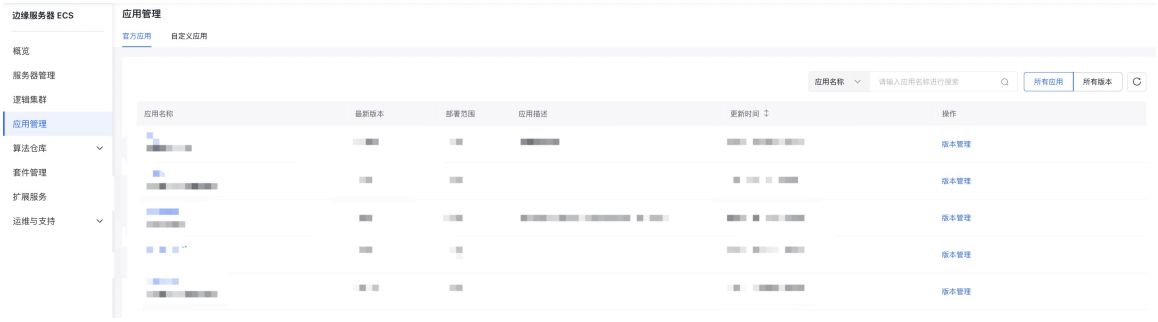
应用管理

官方应用

边缘服务器ECS为用户提供丰富的官方应用和增值服务，涵盖资源调度、视频处理、图像处理等多种功能。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“官方应用”，查看、管理本账户所拥有的官方应用。



官方应用列表

展示该账号已购买的全部官方应用，展示方式包括：所有应用、所有版本。

所有应用

在"所有应用"模式下，列表按照应用名称展示账号已购买的所有官方应用，每款应用展示一次，同时展示应用的最新版本、应用描述、更新时间，并支持"版本管理"操作。

- 版本管理

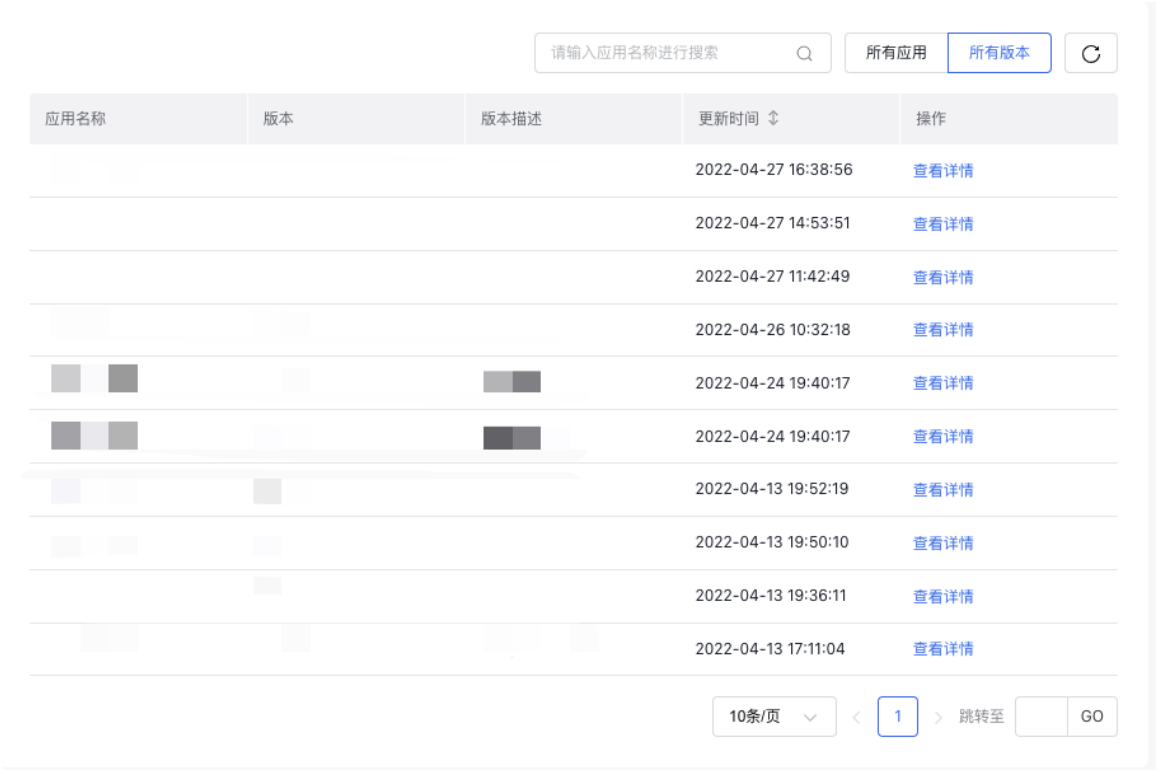
在"版本管理"页面，展示该应用所包含的全部版本，包括版本名称、版本描述、更新时间等，并支持"查看详情"操作。

- 查看详情

在"查看详情"页面，展示该应用、该版本的详细信息，包含应用介绍、版本信息等。

所有版本

在"所有版本"模式下，列表展示账号已购买的所有官方应用的所有版本，同时展示应用名称、版本号、版本描述、更新时间，并支持"查看详情"操作。



- 查看详情

在"查看详情"页面，展示该应用、该版本的详细信息，包含应用介绍、版本信息等。

自定义应用

边缘服务器ECS提供自定义应用功能，支持用户按需添加、编辑、维护自定义应用。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“服务器管理”，进入服务器管理页面。
3. 点击"自定义应用"，添加、管理、查看本账户的自定义应用。

添加自定义应用

支持用户自主添加自定义应用，添加方式包括：表单创建、Yaml创建。

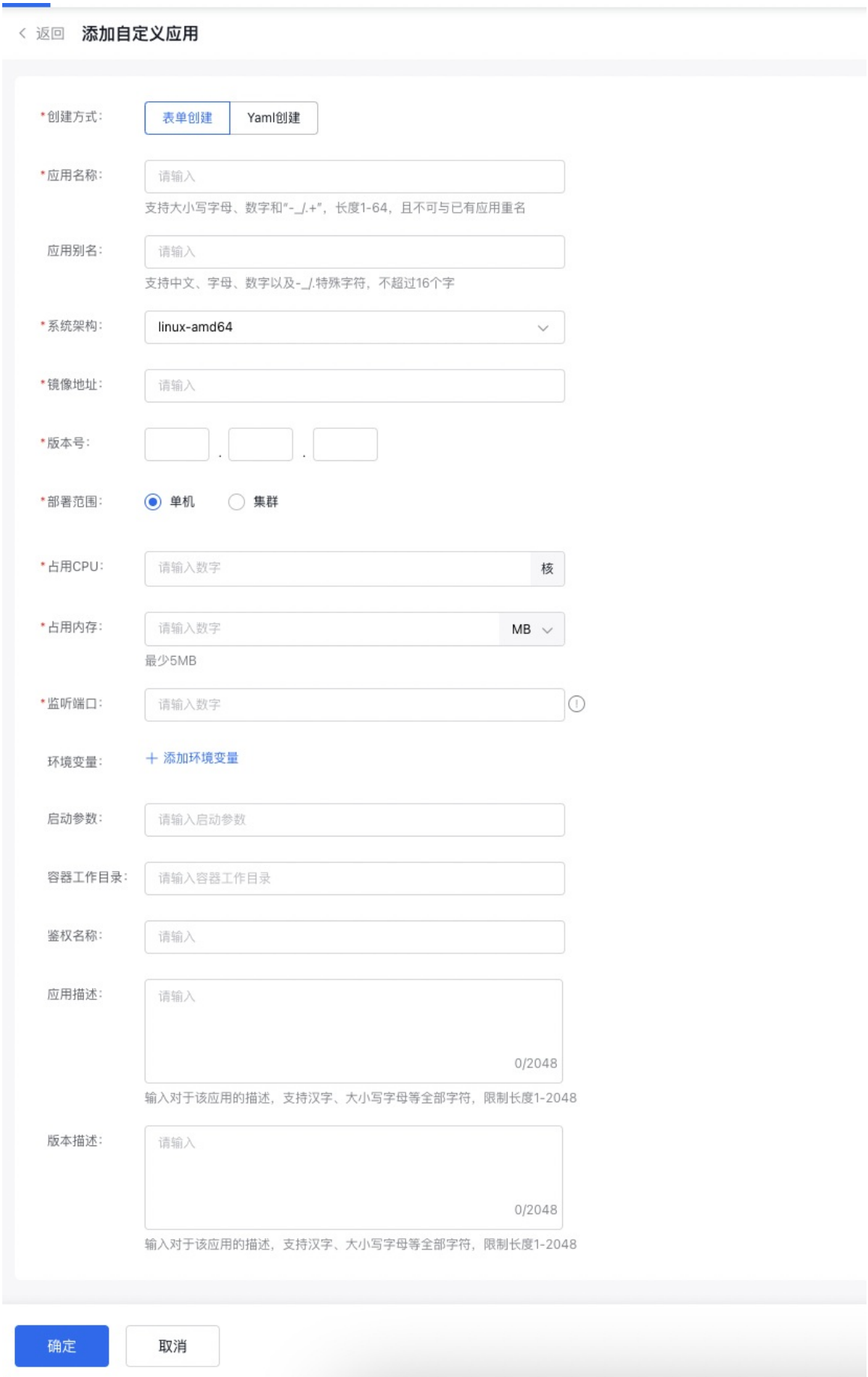
表单创建

以"表单创建"的模式创建自定义应用，用户需要在表单中填入：

必填项：应用名称、系统架构、镜像地址、版本号、部署范围、占用CPU、占用内存、监听端口；

选填项：应用别名、环境变量、启动参数、容器工作目录、应用描述、版本描述、鉴权名称；

填写完成，并满足填写规则检验后，即可创建完成。



11

Yaml创建

以"Yaml创建"的模式创建自定义应用，用户需要填入：

必填项：应用名称、系统架构、版本号、部署范围、Yaml文件；

选填项：应用别名、鉴权名称、应用描述、版本描述；

填写完成，并满足填写规则检验后，即可创建完成。

- 模板管理

可以查看现有的全部模板，并支持查看Yaml、更新Yaml、删除等操作。

- 另存为模板

已填写完成的内容，可以被另存为模板，以供创建时参考。

- 选择模板

用户可以从模板库中选择模板，快速创建自定义应用。

自定义应用列表

展示该账号已创建的全部自定义应用，展示方式包括：所有应用、所有版本。

所有应用

在"所有应用"模式下，列表按照应用名称展示账号已创建的所有自定义应用，每款应用展示一次，同时展示应用的最新版本、创建方式、应用描述、更新时间，并支持"版本管理"及"删除"操作。

注：已部署的自定义应用不支持删除。

- 版本管理

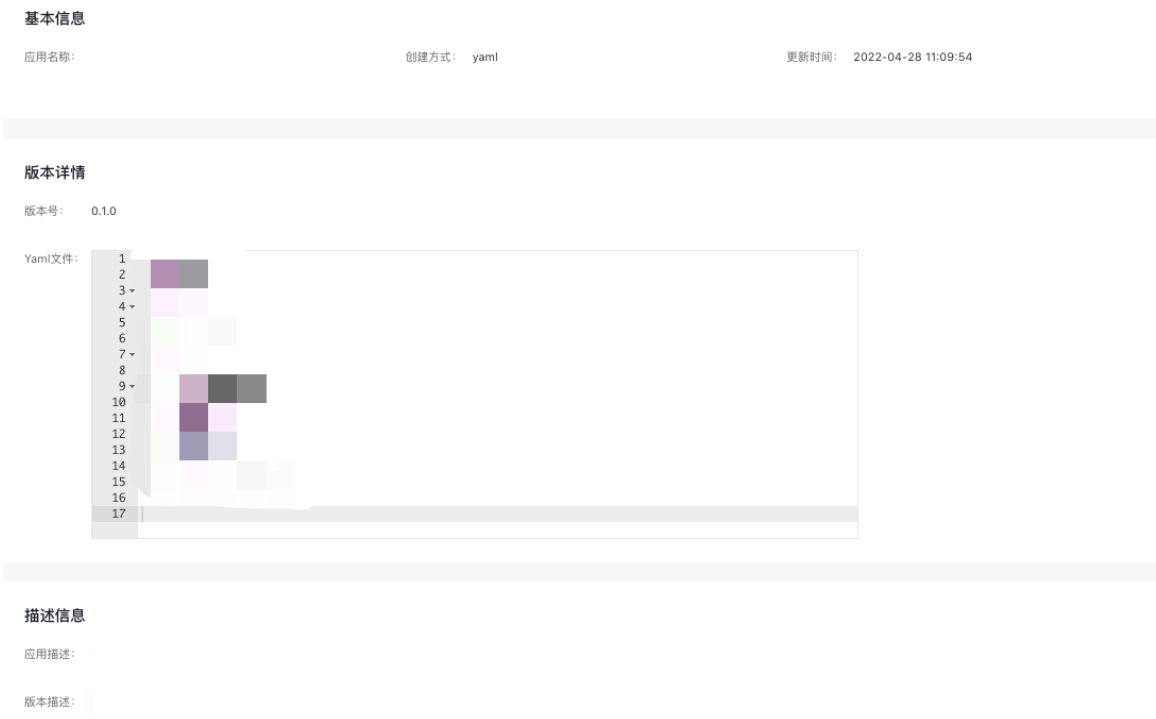
在"版本管理"页面，支持为该自定义应用添加新的版本，添加方式与创建时相同。

注：同一个自定义应用的各个版本，仅支持使用相同的模式进行创建。

同时，展示该应用所包含的全部版本，包括版本名称、版本描述、创建时间等，并支持"查看详情"、"复制"、"删除"操作。

- 查看详情

在"查看详情"页面，展示该应用、该版本的详细信息，包含应用介绍、版本信息、部署信息等。

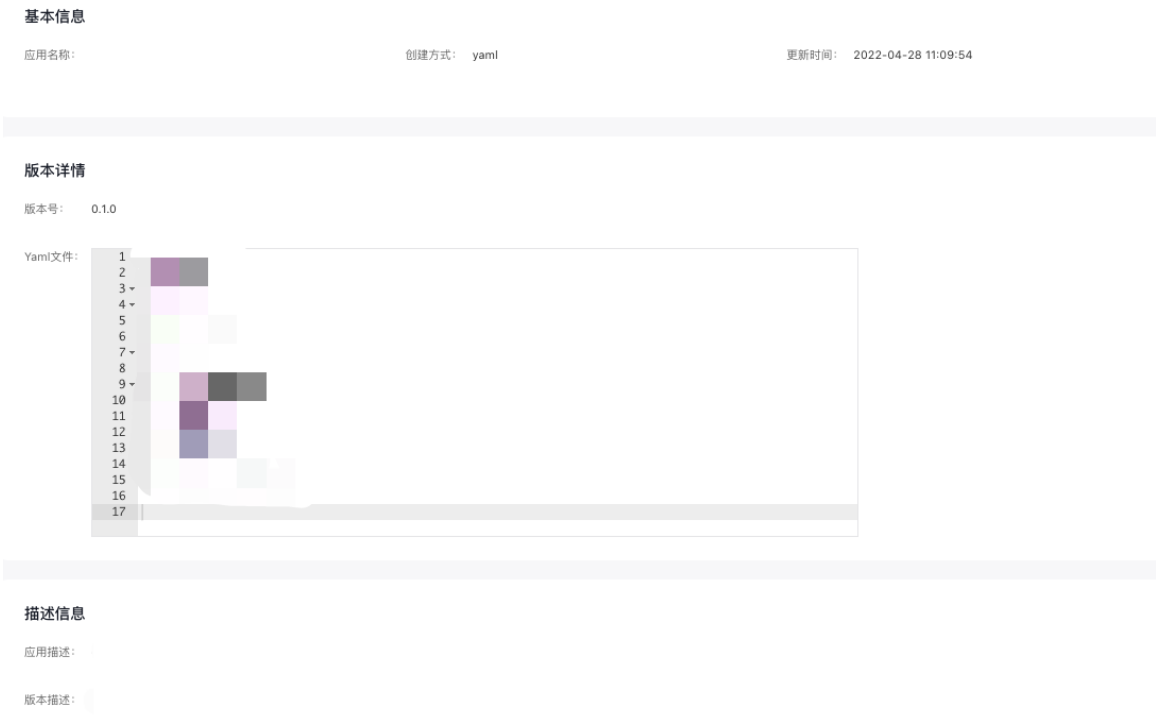


所有版本

在"所有版本"模式下，列表展示账号已创建的所有自定义应用的所有版本，同时展示应用名称、版本号、来源、创建方式、版本描述、更新时间，并支持"查看详情"、"删除"操作。

- 查看详情

在"查看详情"页面，展示该应用、该版本的详细信息，包含应用介绍、版本信息、部署信息等。



算法仓库

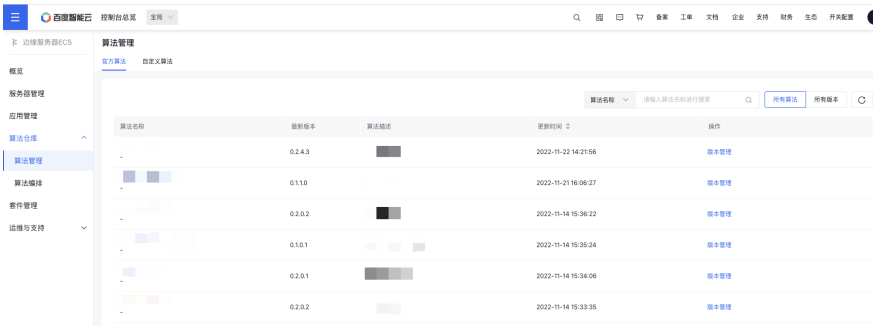
算法管理

官方算法

边缘服务器ECS为用户提供丰富的官方算法，适用于智慧零售、智慧工地、智慧营业厅等多种应用场景。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“算法仓库”，选择"算法管理"。
3. 点击"官方算法"，查看、管理本账户所拥有的官方算法。

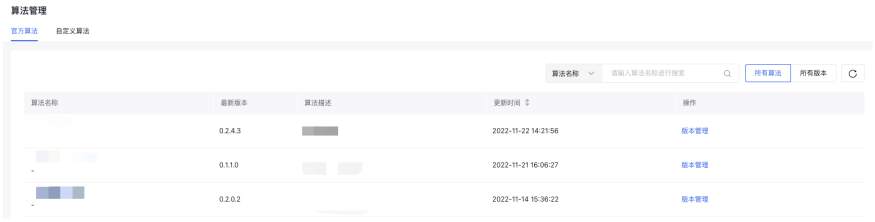


官方算法列表

展示该账号已购买的全部官方算法，展示方式包括：所有算法、所有版本。

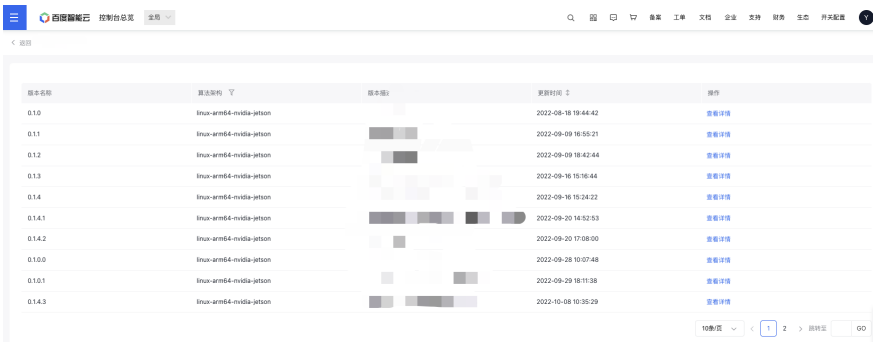
所有算法

在"所有算法"模式下，列表按照算法名称展示账号已购买的所有官方算法，每款算法展示一次，同时展示算法的最新版本、算法描述、更新时间，并支持"版本管理"操作。



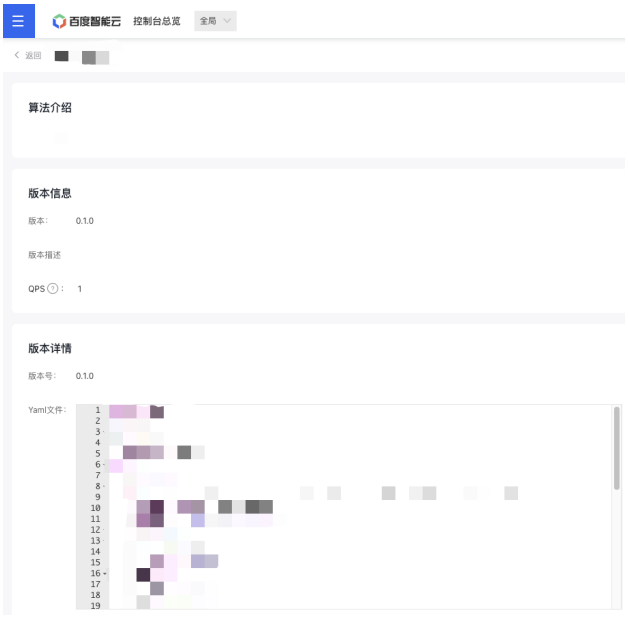
- 版本管理

在"版本管理"页面，展示该算法所包含的全部版本，包括版本名称、版本描述、更新时间等，并支持"查看详情"操作。



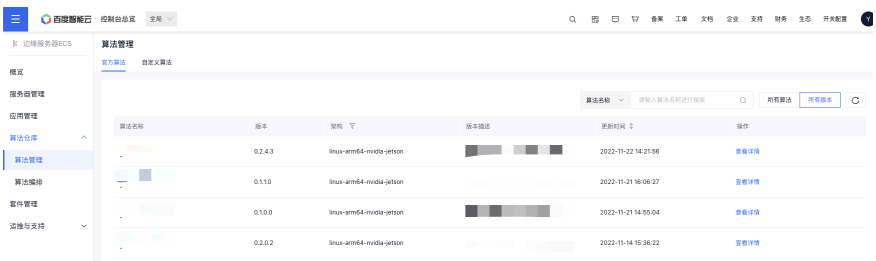
- 查看详情

在"查看详情"页面，展示该算法、该版本的详细信息，包含算法介绍、版本信息等。



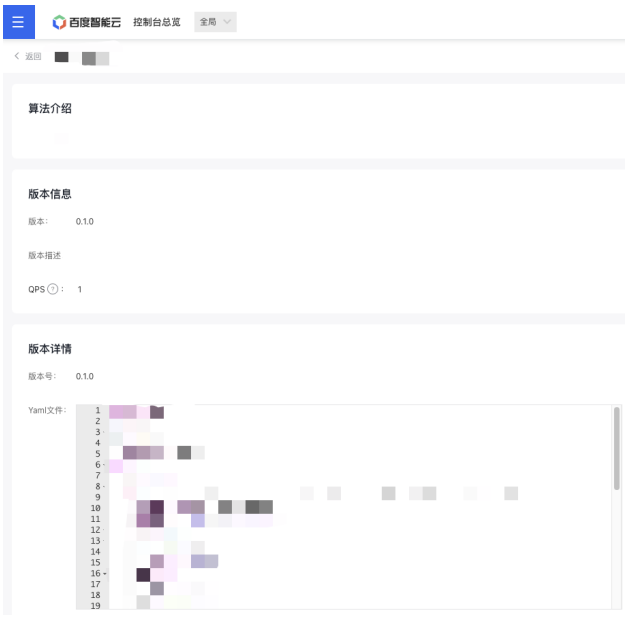
所有版本

在"所有版本"模式下，列表展示账号已购买的所有官方算法的所有版本，同时展示算法名称、版本号、版本描述、更新时间，并支持"查看详情"操作。



- 查看详情

在"查看详情"页面，展示该算法、该版本的详细信息，包含算法介绍、版本信息等。

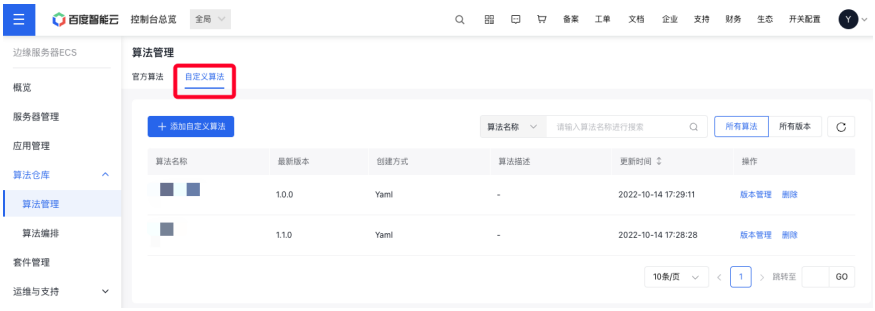


自定义算法

边缘服务器ECS提供自定义算法功能，支持用户按需添加、编辑、维护自定义算法。

路径

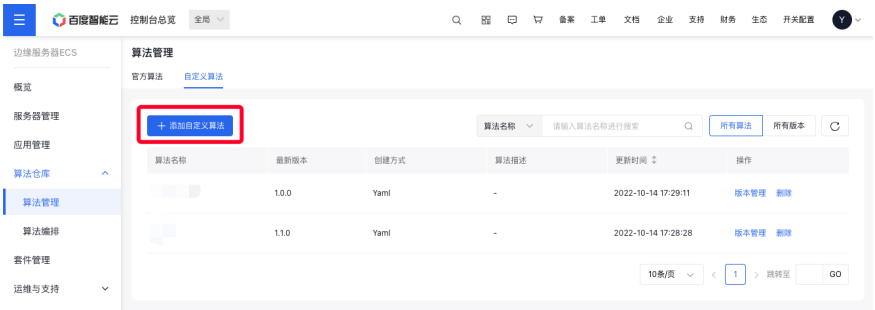
1. 登录ECS控制台。
2. 点击左侧导航栏“算法仓库”，选择"算法管理"。
3. 点击"自定义算法"，添加、管理、查看本账户的自定义算法。



添加自定义算法

支持用户自主添加自定义算法，添加方式包括：表单创建、Yaml创建。

点击"添加自定义算法"按钮，进入自定义算法添加页面。



表单创建

以"表单创建"的模式创建自定义算法，用户需要在表单中填入：

必填项：算法名称、算法架构、镜像地址、版本号、对外端口、占用CPU、占用内存、监听端口等；

选填项：算法别名、环境变量、启动参数、容器工作目录、算法描述、版本描述等；

填写完成，并满足填写规则检验后，即可创建完成。

Yaml创建

以"Yaml创建"的模式创建自定义算法，用户需要填入：

必填项：算法名称、算法架构、版本号、Yaml文件等；

选填项：算法别名、算法描述、版本描述等；

填写完成，并满足填写规则检验后，即可创建完成。

- 模板管理

可以查看现有的全部模板，并支持查看Yaml、更新Yaml、删除等操作。

- 另存为模板

已填写完成的内容，可以被另存为模板，以供创建时参考。

- 选择模板

用户可以从模板库中选择模板，快速创建自定义算法。

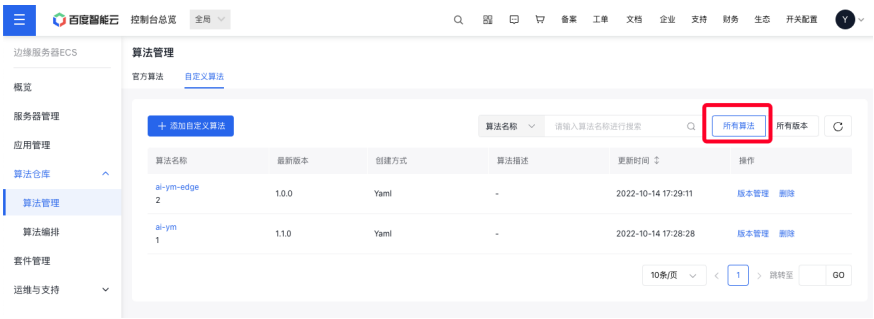
自定义算法列表

展示该账号已创建的全部自定义算法，展示方式包括：所有算法、所有版本。

所有算法

在"所有算法"模式下，列表按照算法名称展示账号已创建的所有自定义算法，每款算法展示一次，同时展示算法的最新版本、创建方式、算法描述、更新时间，并支持"版本管理"及"删除"操作。

注：已部署的自定义算法不支持删除。

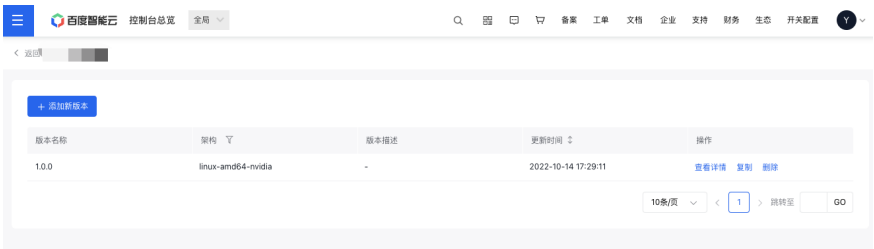


- 版本管理

在"版本管理"页面，支持为该自定义算法添加新的版本，添加方式与创建时相同。

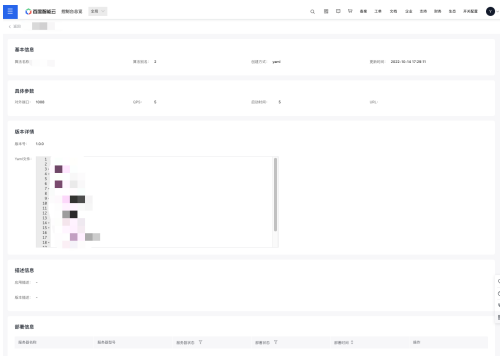
注：同一个自定义算法的各个版本，仅支持使用相同的模式进行创建。

同时，展示该算法所包含的全部版本，包括版本名称、版本描述、创建时间等，并支持"查看详情"、"复制"、"删除"操作。



- 查看详情

在"查看详情"页面，展示该应用、该版本的详细信息，包含应用介绍、版本信息、部署信息等。

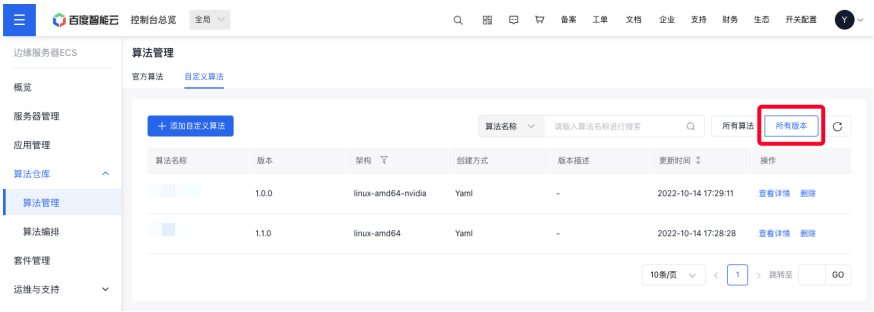


- 复制

点击"复制"按钮，可以复制该算法版本的参数信息，帮助用户添加、编辑新版本，进行版本管理。

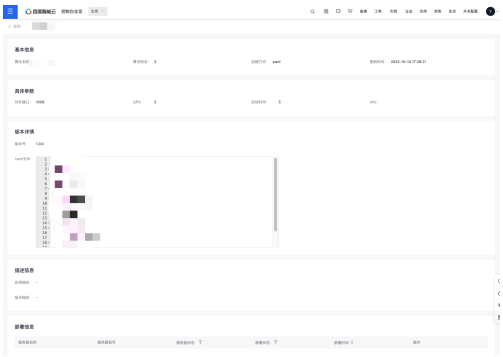
所有版本

在"所有版本"模式下，列表展示账号已创建的所有自定义算法的所有版本，同时展示算法名称、版本号、来源、创建方式、版本描述、更新时间，并支持"查看详情"、"删除"操作。



- 查看详情

在"查看详情"页面，展示该算法、该版本的详细信息，包含算法介绍、版本信息、部署信息等。



算法编排

多合一算法

边缘服务器ECS为用户提供算法编排服务，用户可以依照需求，自定义添加、编辑、管理多合一算法。

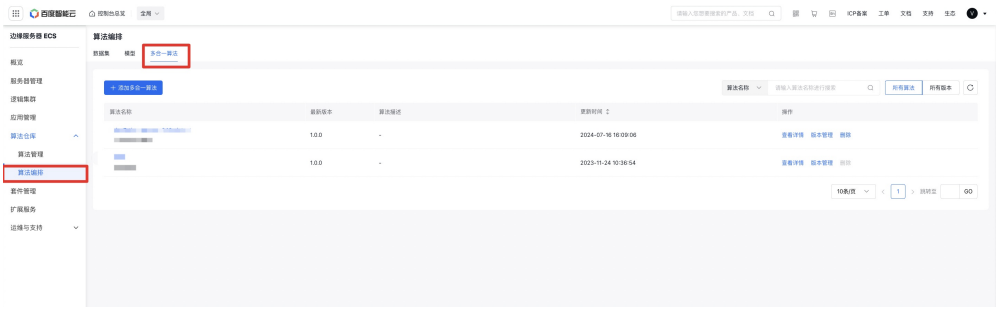
多合一算法，将多个子算法合成为一个完整算法，大幅节约算法所占用的各项资源。帮助用户在有限的算力下同时使用更多算法，增强算法能力、提高算力利用率、实现更多的业务需求。

路径

1. 登录ECS控制台；

2. 点击左侧导航栏“算法仓库”，选择"算法编排"。

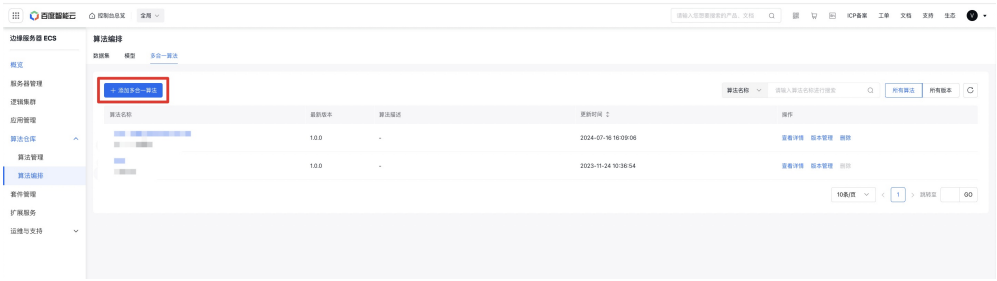
3. 点击"多合一算法"，查看、管理全部的多合一算法。



添加多合一算法

支持用户自主添加多合一算法，添加方法如下：

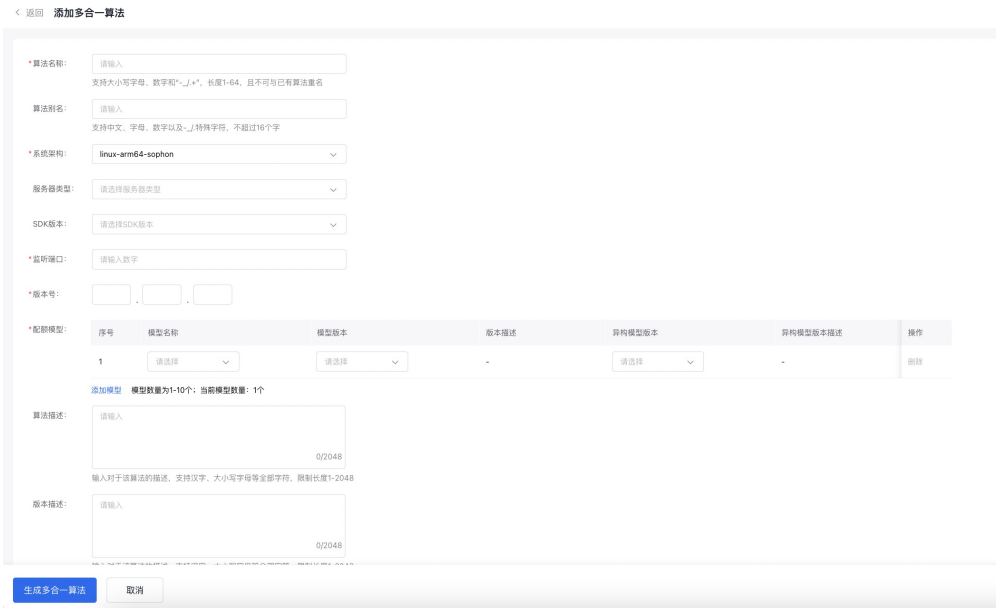
1. 点击"添加多合一算法"按钮，进入多合一算法添加页面。



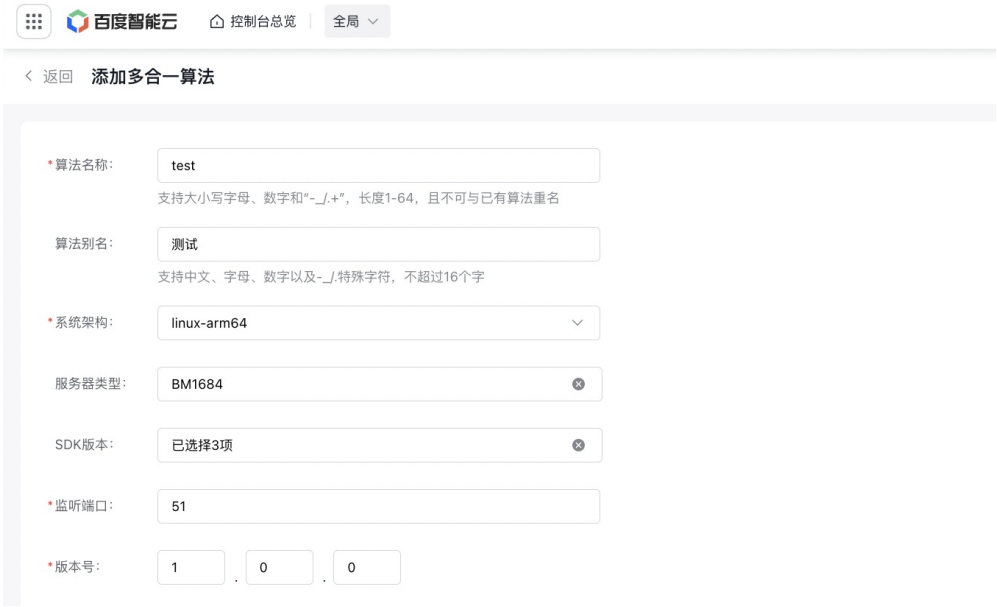
2. 在添加多合一算法页面，填入各项参数；

必填项：算法名称、系统架构、监听端口、版本号、配额模型等；

选填项：算法别名、服务器类型、SDK版本、算法描述、版本描述等；



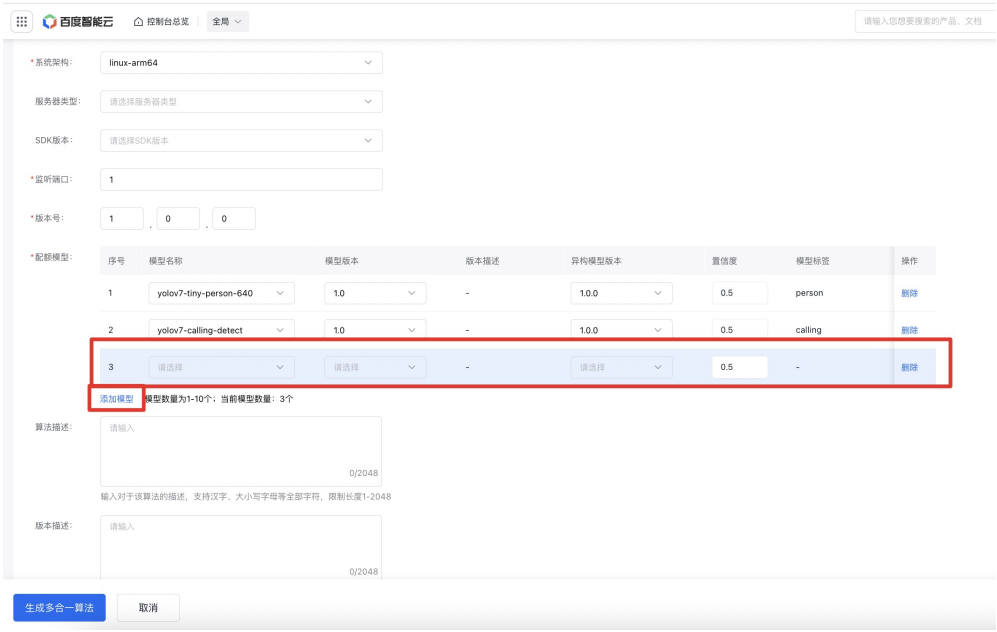
3.算法名称、算法别名、系统架构、服务器类型、SDK版本、监听端口、版本号，这些可以根据算法自身需求或部署算法的服务器的具体情况来进行填写；



4.点击"添加模型"按钮， 选择想要使用的算法模型的名称、模型版本、异构模型版本，自定义置信度，选择并填写好信息后即完成配额模型的添加；

注：只有在模型模块中创建好模型后，并完成转换模型才可以在此"模型名称"中看到模型。

每个多合一算法，最多使用10个子算法；



5. 点击"生产多合一算法"按钮，已选择的全部子算法将自动合成为一个多合一算法。

在"生成多合一算法"弹窗中，会再次展示在多合一算法时选择的各项参数，如果发现版本或置信度等等填写错误可以取消创建重新填写。

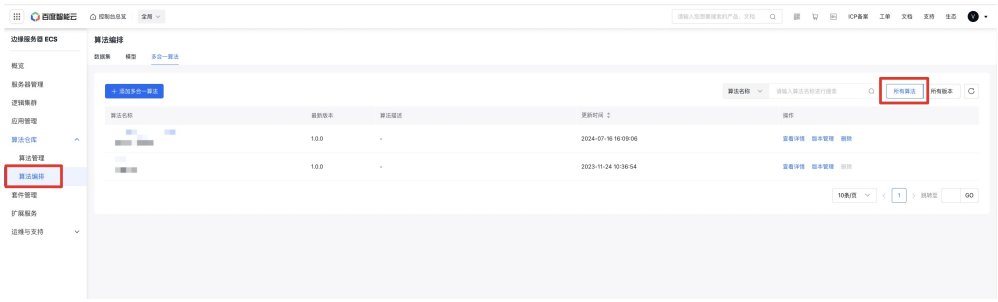
多合一算法列表

展示该账号已创建的全部多合一算法，展示方式包括：所有算法、所有版本。

所有算法

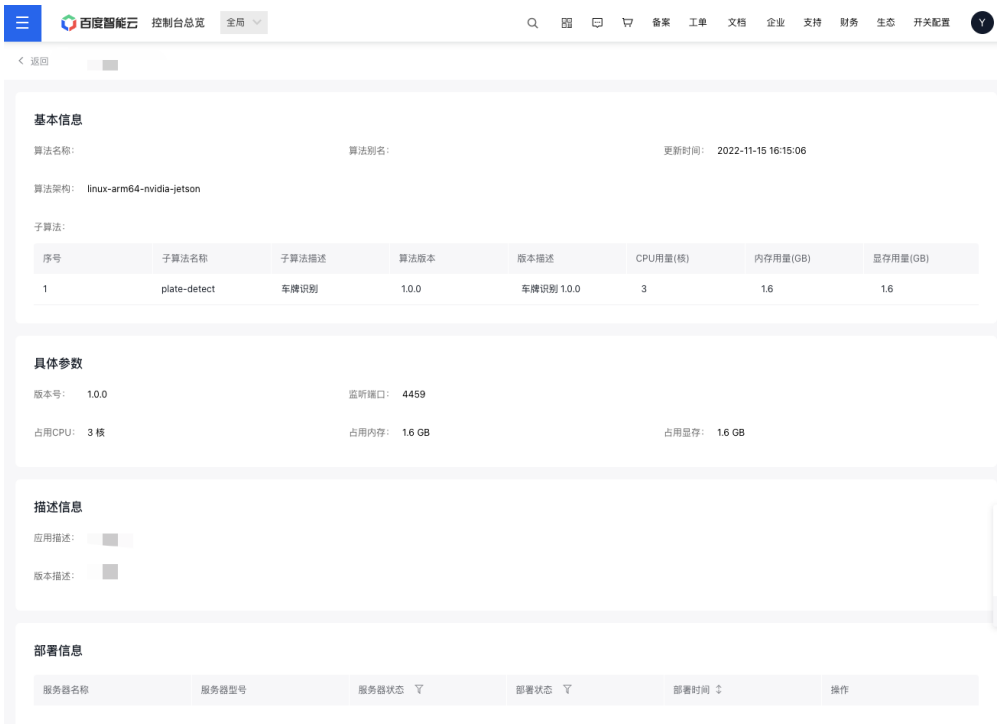
在"所有算法"模式下，列表按照算法名称展示账号已创建的所有多合一算法，每款算法展示一次，同时展示算法的最新版本、创建方式、算法描述、更新时间，并支持"查看详情"和"版本管理"及"删除"操作。

注：已部署的多合一算法不支持删除。



- 查看详情

在"查看详情"页面，展示该应用、该版本的详细信息，包含应用介绍、版本信息、部署信息等。

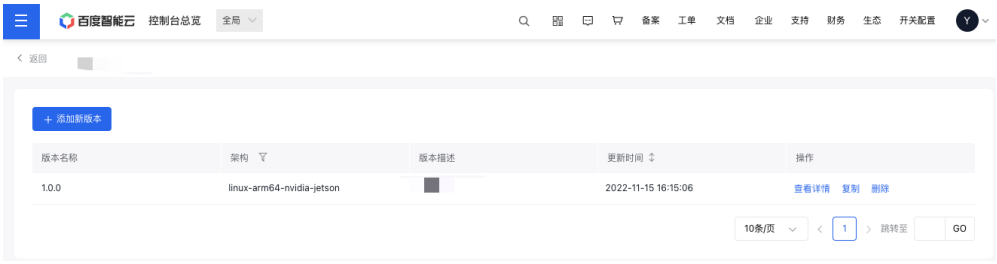


- 版本管理

在"版本管理"页面，支持为该多合一算法添加新的版本，添加方式与创建时相同。

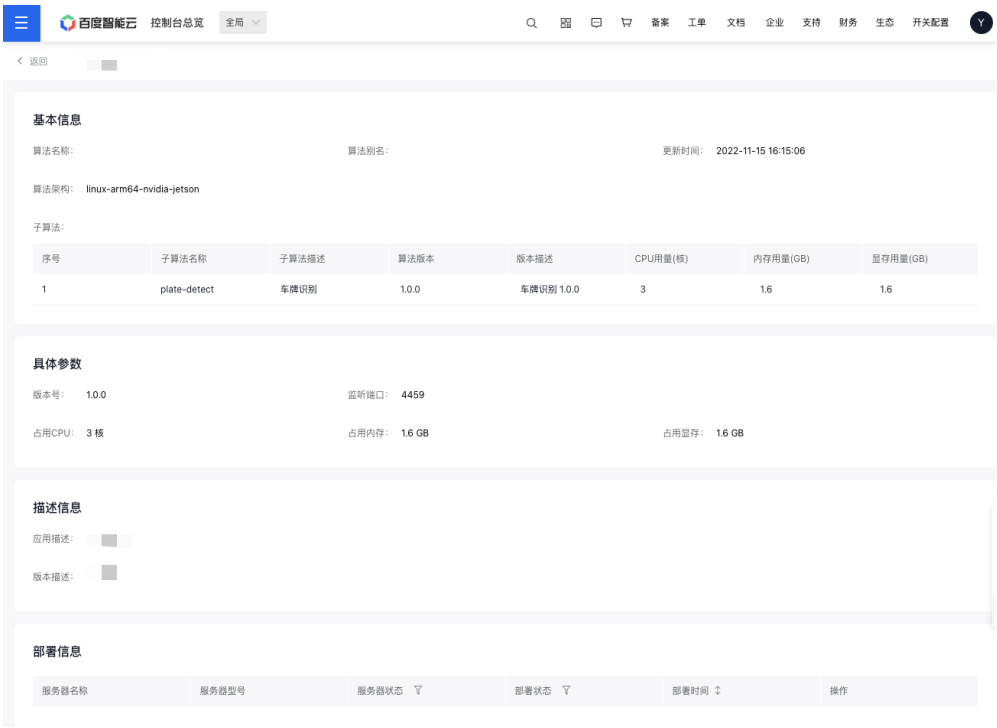
注：同一个多合一算法的各个版本，仅支持使用相同的模式进行创建。

同时，展示该算法所包含的全部版本，包括版本名称、版本描述、创建时间等，并支持"查看详情"、"复制"、"删除"操作。



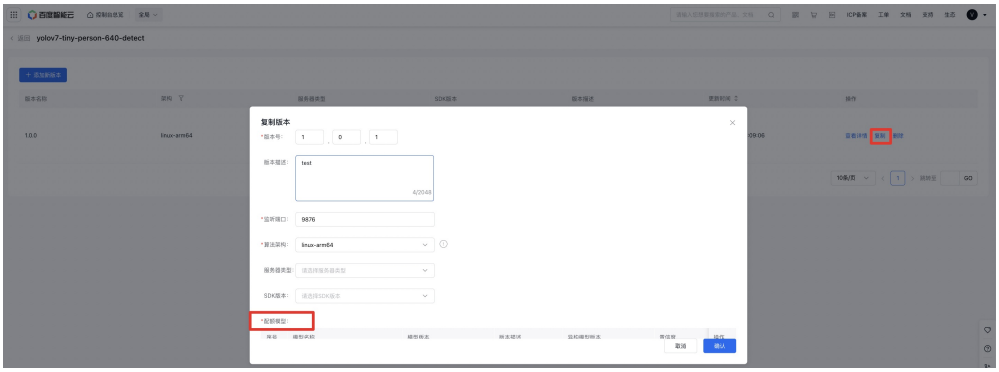
- 查看详情

在"查看详情"页面，展示该应用、该版本的详细信息，包含应用介绍、版本信息、部署信息等。



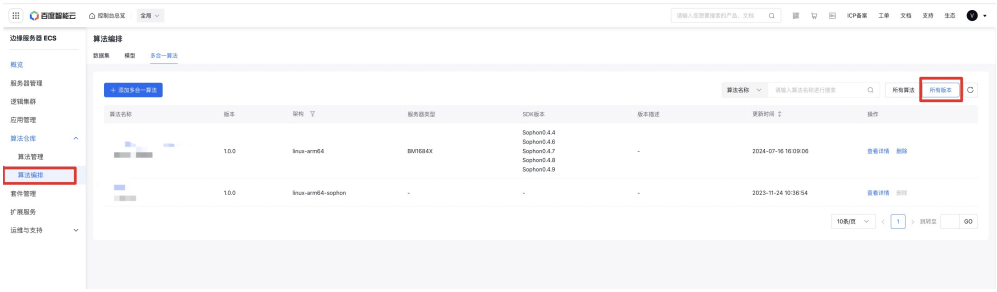
- 复制

点击"复制"按钮，可以复制该算法版本的参数信息，帮助用户添加、编辑新版本，进行版本管理，还可以添加其他模型从而生成新的多合一算法。



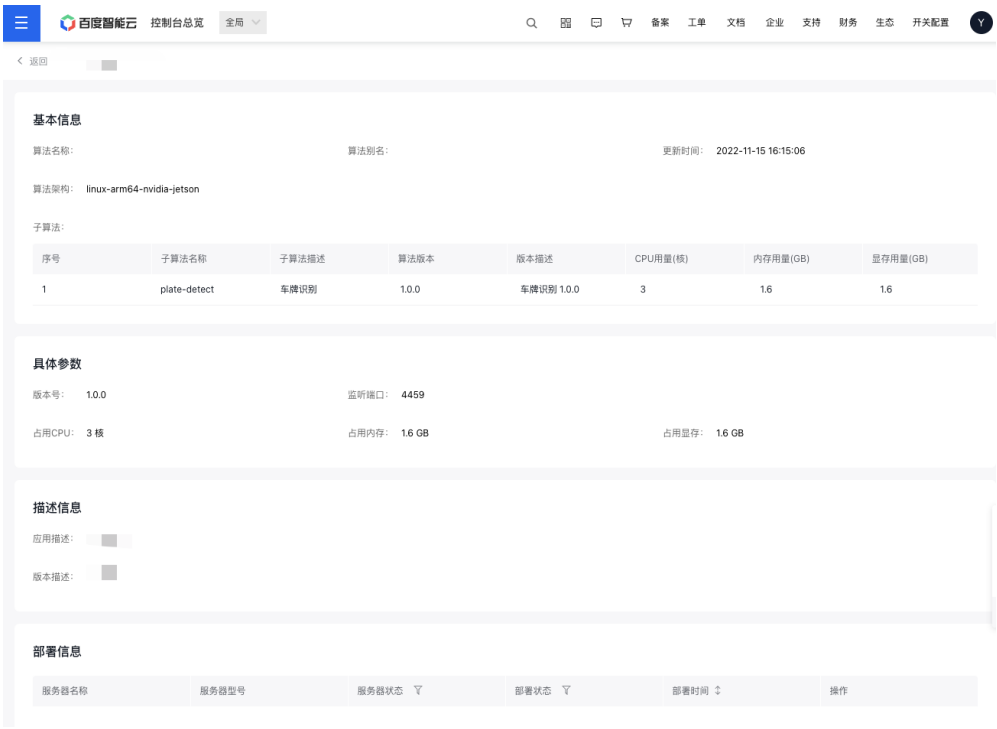
所有版本

在"所有版本"模式下，列表展示账号已创建的所有多合一算法的所有版本，同时展示算法名称、版本号、架构、服务器类型、SDK版本、版本描述、更新时间，并支持"查看详情"、"删除"操作。



- 查看详情

在"查看详情"页面，展示该算法、该版本的详细信息，包含算法介绍、版本信息、部署信息等。



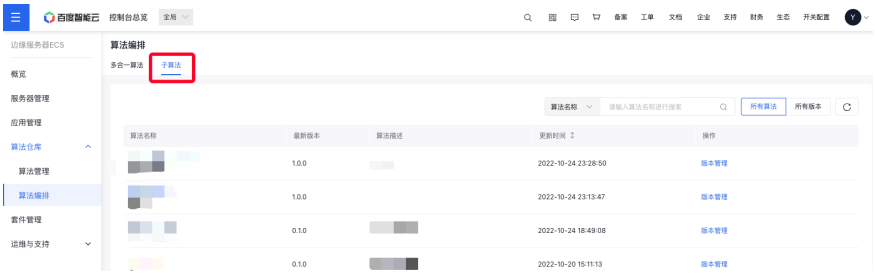
子算法

边缘服务器ECS为用户提供算法编排服务，用户可以依照需求，自定义添加、编辑、管理多合一算法。

子算法是多合一算法的素材，用于合成多合一算法。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“算法仓库”，选择"算法管理"。
3. 点击"子算法"，查看、管理子算法。

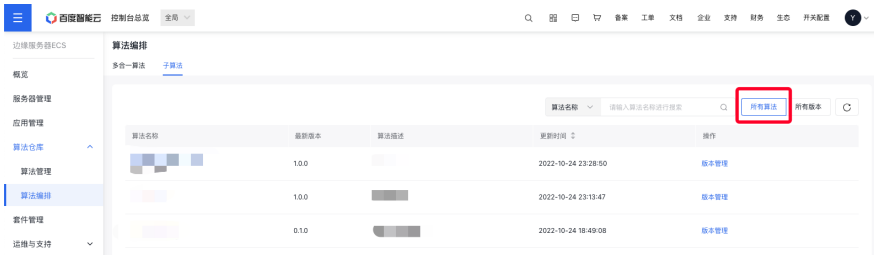


子算法列表

展示全部子算法，展示方式包括：所有算法、所有版本。

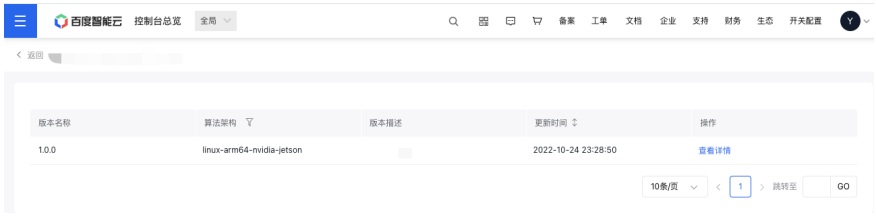
所有算法

在"所有算法"模式下，列表按照应用名称展示所有子算法，每款算法展示一次，同时展示算法的最新版本、算法描述、更新时间，并支持"版本管理"操作。



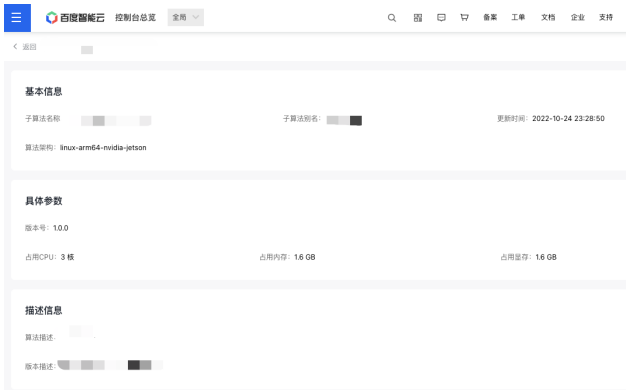
- 版本管理

在"版本管理"页面，展示该算法所包含的全部版本，包括版本名称、版本描述、更新时间等，并支持"查看详情"操作。



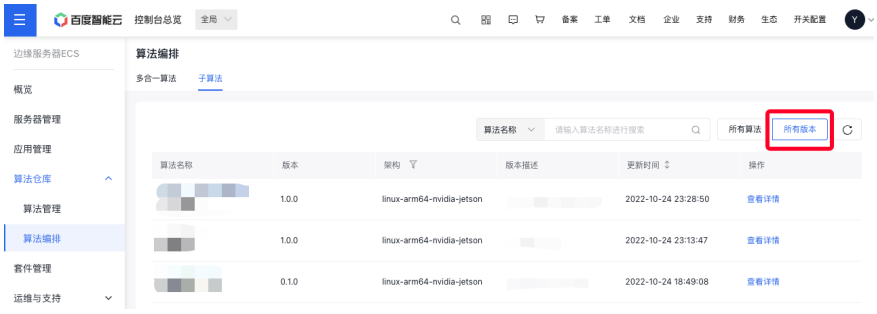
- 查看详情

在"查看详情"页面，展示该算法、该版本的详细信息，包含算法介绍、版本信息等。



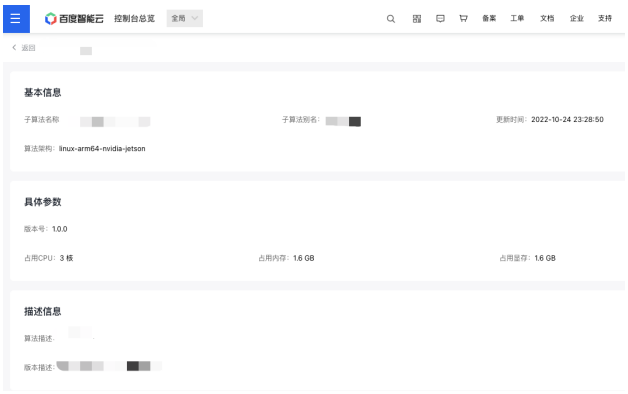
所有版本

在"所有版本"模式下，列表展示所有子算法的所有版本，同时展示算法名称、版本号、版本描述、更新时间，并支持"查看详情"操作。



- 查看详情

在"查看详情"页面，展示该算法、该版本的详细信息，包含算法介绍、版本信息等。



模型转换参数一览表

框架	网络	ModelToOnnx 参数	盒子类型	SDK	OnnxToModel 参数	备注
Paddle Paddle	ppyoloe	--model_dir ./ --model_filename ruixing_phase2_trash_bin_detection.pdmodel --params_filename ruixing_phase2_trash_bin_detection.pdiparams --save_file model.onnx --enable_dev_version True --opset_version 13 --enable_onnx_checker True	天数	igie_1.0.0	--input ["image","scale_factor"] --shape [[1,3,640,640],[1,2]] --precision fp16 --algorithm_name ppyoloe	ModelToOnnx参数：--model_filename与--params_filename要和上传的模型文件解压出来的模型结构文件和模型参数文件名字一致
Paddle Paddle	ppyoloe	--model_dir ./ --model_filename ruixing_phase2_trash_bin_detection.pdmodel --params_filename ruixing_phase2_trash_bin_detection.pdiparams --save_file model.onnx --enable_dev_version True --opset_version 13 --enable_onnx_checker True	天数	ixrt_4.0.0	带nms、静态batch：--algorithm_name ppyoloe --precision fp16 --input_nodenames image --output_nodenames p2o.Mul.224 concat_14.tmp_0 --iou_threshold 0.5 --score_threshold 0.3 --max_output_boxes 1000 --with_nms 带nms、动态batch:--algorithm_name ppyoloe --precision fp16 --input_nodenames image --output_nodenames p2o.Mul.224 concat_14.tmp_0 --iou_threshold 0.5 --score_threshold 0.3 --max_output_boxes 1000 --with_nms --min_shape image:1x3x640x640 --opt_shape image:2x3x640x640 --max_shape image:4x3x640x640 --dynamic_batch	ModelToOnnx参数：--model_filename与--params_filename要和上传的模型文件解压出来的模型结构文件和模型参数文件名字一致
Paddle	ppyoloe	--model_dir ./ --model_filename ruixing_phase2_trash_bin_detection.pdmodel --params_filename		igie_1.0		ModelToOnnx参数：--model_filename与--params_filen

Paddle Paddle	ppyoloe -tiny	ruixing_phase2_trash_bin_detection.pdiparams --save_file model.onnx --enable_dev_version True --opset_version 13 --enable_onnx_checker True	天数	igie_1.0.0	--	ame要和上传的模型文件解压出来的模型结构文件和模型参数文件名字一致
Paddle Paddle	ppyoloe -tiny	--model_dir ./ --model_filename ruixing_phase2_trash_bin_detection.pdmodel --params_filename ruixing_phase2_trash_bin_detection.pdiparams --save_file model.onnx --enable_dev_version True --opset_version 13 --enable_onnx_checker True	天数	ixrt_4.0.0	带nms、动态batch: --algorithm_name ppyoloe_tiny --precision fp16 --input_nodenames p2o.transpose_0.tmp_0.0 --output_nodenames p2o.Mul.224 concat_16.tmp_0 --iou_threshold 0.5 --score_threshold 0.3 --max_output_boxes 1000 --min_shape p2o.transpose_0.tmp_0.0:1x3x640x640 --opt_shape p2o.transpose_0.tmp_0.0:2x3x640x640 --max_shape p2o.transpose_0.tmp_0.0:4x3x640x640 --with_nms --dynamic_batch	ModelToOnnx参数： --model_filename与--params_filename要和上传的模型文件解压出来的模型结构文件和模型参数文件名字一致
Paddle Paddle	pplcnet	--model_dir ./ --model_filename ruixing_phase2_trash_bin_detection.pdmodel --params_filename ruixing_phase2_trash_bin_detection.pdiparams --save_file model.onnx --enable_dev_version True --opset_version 13 --enable_onnx_checker True	天数	igie_1.0.0	--input ["x"] --shape [[1,3,256,192]] --precision fp16 --algorithm_name pplcnet	ModelToOnnx参数： --model_filename与--params_filename要和上传的模型文件解压出来的模型结构文件和模型参数文件名字一致
Paddle Paddle	pplcnet	--model_dir ./ --model_filename ruixing_phase2_trash_bin_detection.pdmodel --params_filename ruixing_phase2_trash_bin_detection.pdiparams --save_file model.onnx --enable_dev_version True --opset_version 13 --enable_onnx_checker True	天数	ixrt_4.0.0	--algorithm_name pplcnet --batch_size -1 --input_height 256 --input_width 192 --precision fp16 --min_shape x:1x3x256x192 --opt_shape x:8x3x256x192 --max_shape x:32x3x256x192	ModelToOnnx参数： --model_filename与--params_filename要和上传的模型文件解压出来的模型结构文件和模型参数文件名字一致
		--grid --simplify --topk-all 100 --	BM168	sophon_3.0.0	--outdir=output --target=BM1684 --shapes=[[1,3,640,640]] --	

Pytorch	yolov7	iou-thres 0.65 --conf-thres 0.35 --img-size 640 640	4	、sophon_0.4	net_name=yolov7_fp32_person_640 --opt=2 --dyn=False --cmp=True --enable_profile=True	--
Pytorch	yolov7	--grid --simplify --topk-all 100 --iou-thres 0.65 --conf-thres 0.35 --img-size 640 640	BM1684X	sophon_3.0.0、sophon_0.4	--outdir=output --target=BM1684X --shapes=[[1,3,640,640]] --net_name=yolov7_fp32_person_640 --opt=2 --dyn=False --cmp=True --enable_profile=True	--
Pytorch	yolov7	--grid --simplify --topk-all 100 --iou-thres 0.65 --conf-thres 0.35 --img-size 640 640	Rockchip (瑞芯微通用)	--	--model_arch=yolov7 --precision=fp --platform=rk3576	OnnxToModel 参数：*--precision 表示精度，可选fp或int* --platform 表示目标盒子类型，支持范围包括 ["rk3576", "rk3568", "rk3566", "rk3588", "rk3562", "rv1103", "rk2118", "rv1106"]
Pytorch	yolov7	--grid --simplify --topk-all 100 --iou-thres 0.65 --conf-thres 0.35 --img-size 640 640	Xavier NX	Jetpack 5	--	--

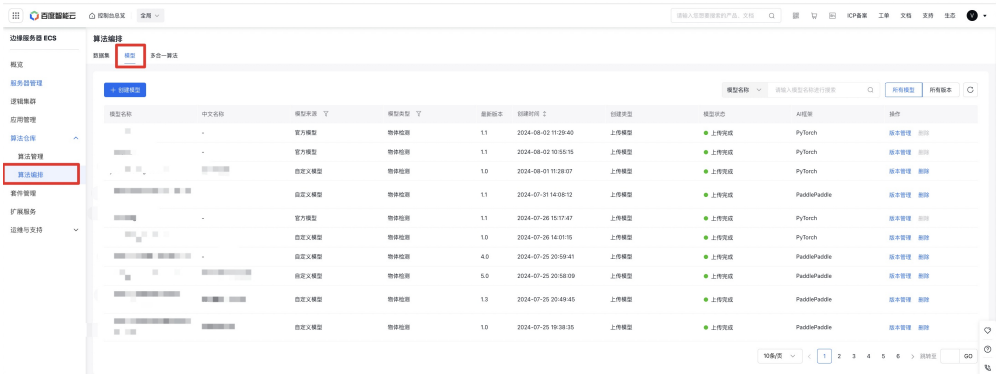
模型

边缘服务器ECS为用户提供算法编排服务，用户可以依照需求，自定义添加、编辑、管理模型和多合一算法。

模型是多合一算法的素材，用于合成多合一算法。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“算法仓库”，选择"算法编排"。
3. 点击"模型"，查看、管理全部模型。

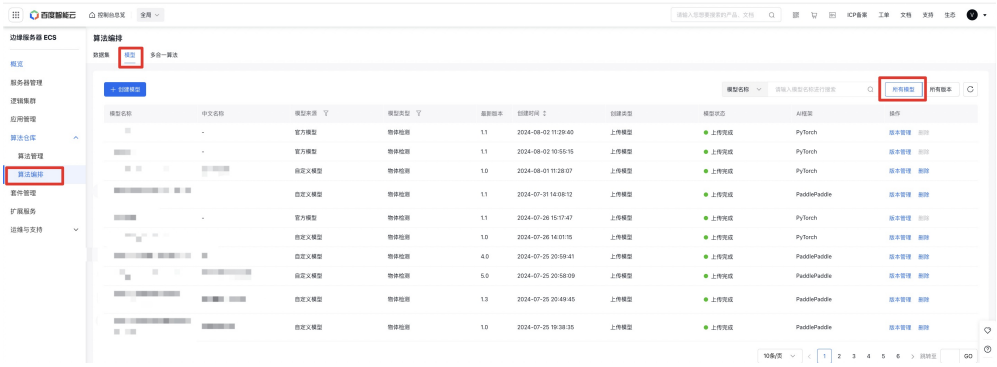


模型

模型部分内容主要是三个大模块组成，分别是“创建模型”、“所有模型”、“所有版本”，其中模型还拥有“模型检索”和对模型的一些管理操作，操作包括“版本管理”或“删除”等。

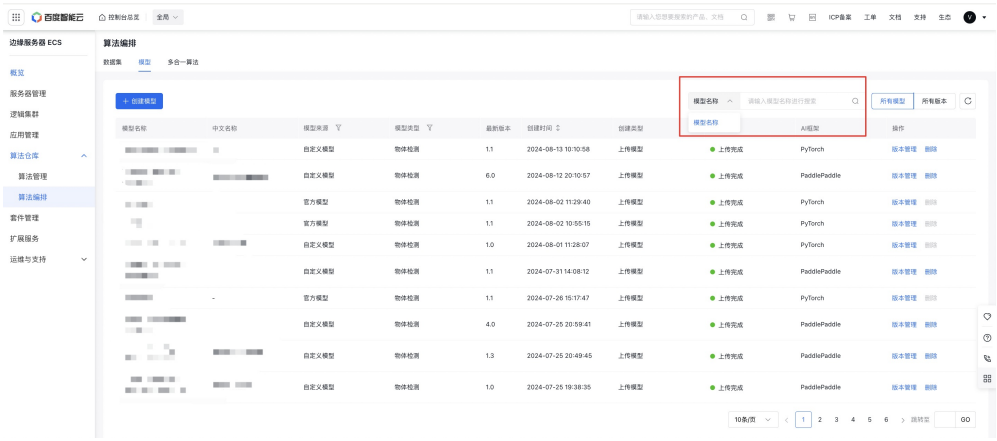
所有模型

在"所有模型"模式下，列表按照应用名称展示所有模型，每款算法模型展示一次，同时展示模型的名称、中文名称、模型来源、模型类型、最新版本、创建时间、创建类型、模型状态、AI框架，并支持"版本管理"和"删除"操作。



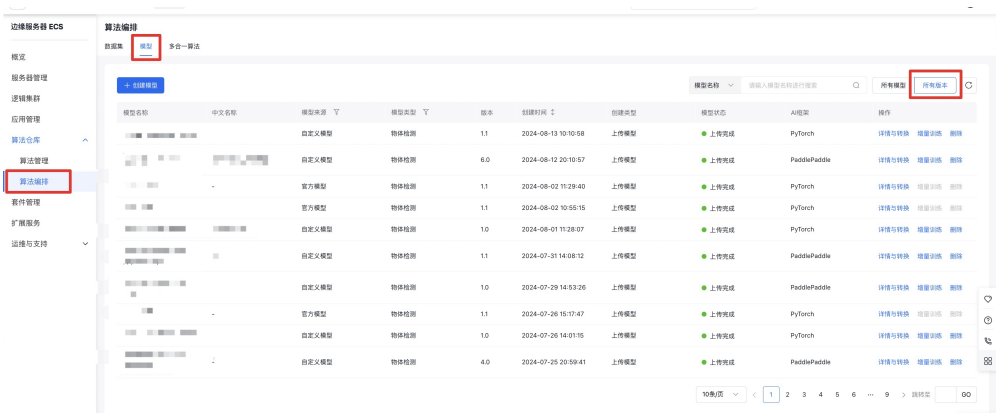
- 模型检索

在"所有模型"模式下，可以根据模型名称进行详细搜索从而更快速找到模型并进行操作。



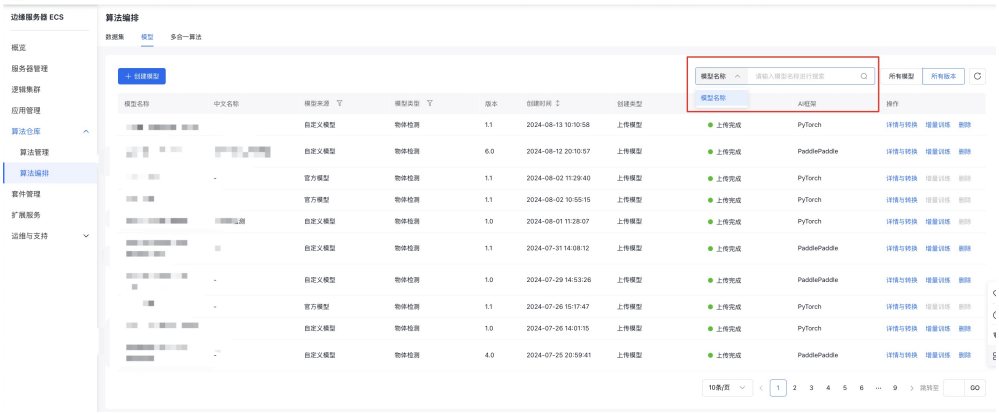
所有版本

在"所有版本"模式下，列表按照应用名称展示所有模型，所有算法模型的版本均同事展示，同时展示模型的名称、中文名称、模型来源、模型类型、最新版本、创建时间、创建类型、模型状态、AI框架，并支持"详情与转换"和"删除"操作。



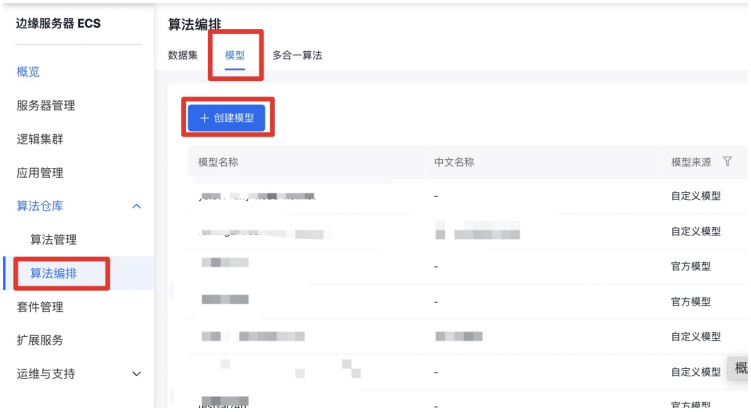
- 模型检索

在"所有版本"模式下，可以根据模型名称进行详细搜索从而更快速找到模型并进行操作。



创建模型 支持用户自主创建模型，创建方法如下：

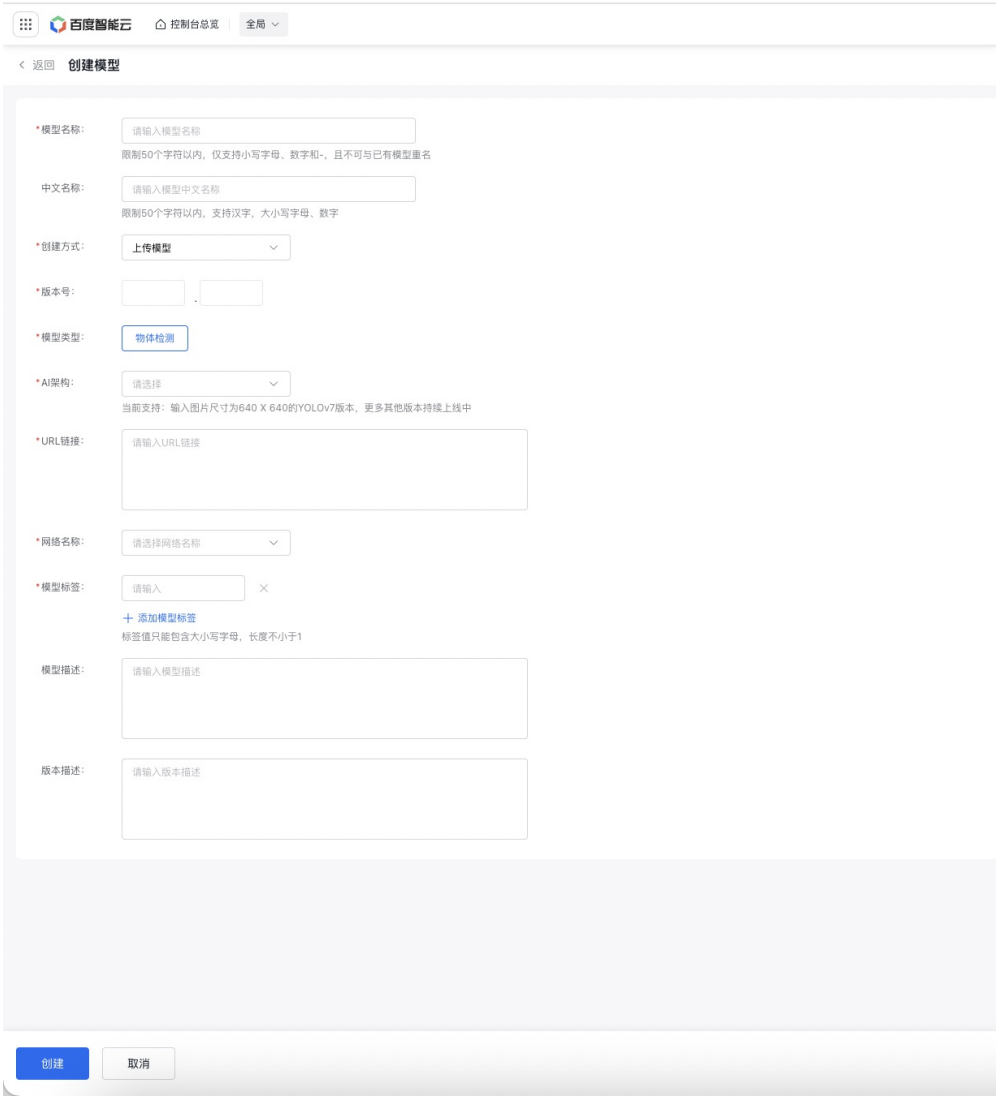
1.进入模型管理界面点击"创建模型"按钮，进入创建模型页面。



2.在添加多合一算法页面，填入各项参数；

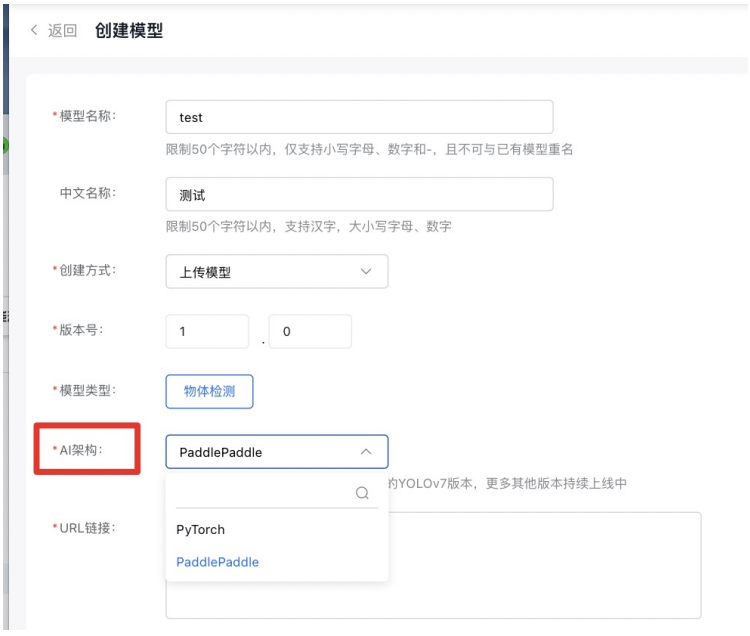
必填项：模型名称、创建方式、版本号、模型类型、AI架构、URL链接、网络名称、模型标签等；

选填项：中文名称、模型描述、版本描述等；



3.根据需求填写完成名称、选择好创建方式为上传模型，确认版本号和模型类型后，选择AI架构，目前仅支持两种AI架构，根据算法具体情况选择即可。

注：当前支持输入图片尺寸为640 X 640的PyTorch或PaddlePaddle和YOLOv7组合版本，更多其他版本持续上线中



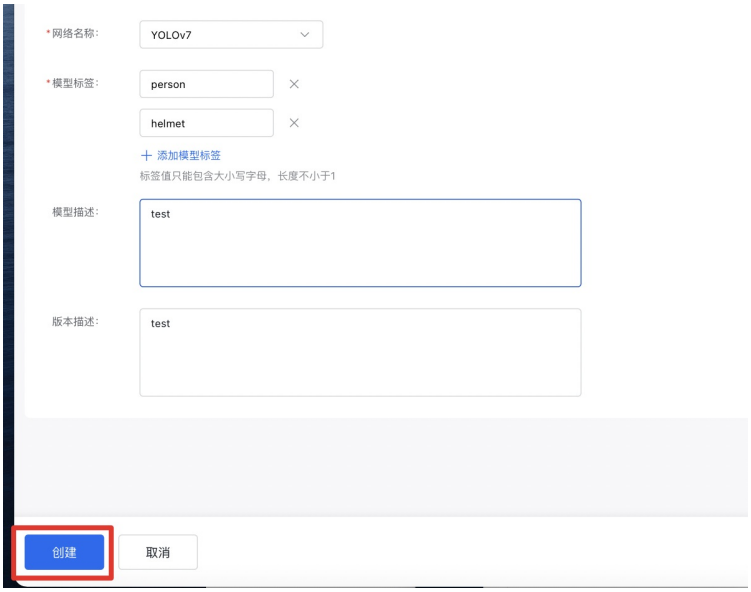
4.URL链接则需要填写算法模型的链接地址。

5.网络名称根据自身模型具体情况选择即可。

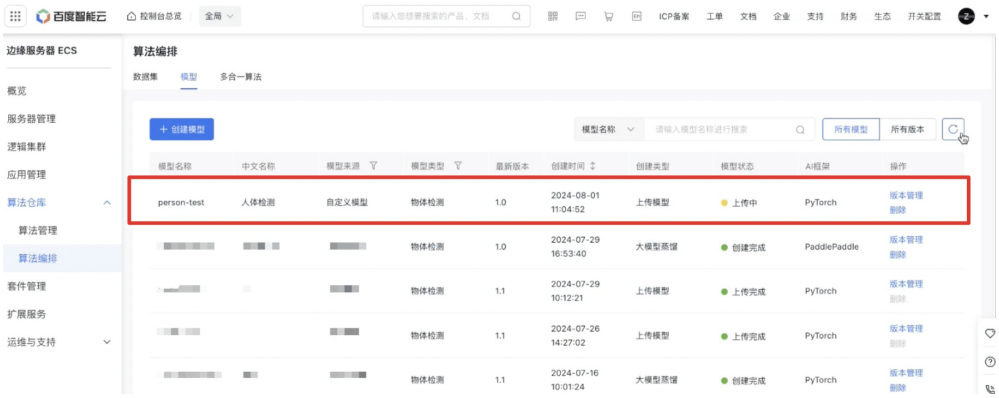
6.模型标签这里的填写规则是自定义的模型输出的结果，根据自身情况填写即可。

注: 标签值只能包含大小写字母，长度不小于1

7.接下来填写好模型描述和版本描述后点击创建即可完成模型创建。



8.模型创建完毕后会返回管理界面，可以在管理界面中查看到刚才创建的模型，等待模型状态从上传中更新至上传完成即模型上传完毕。



模型操作

- 版本管理

版本管理中可以对创建好的模型进行算法转换和删除个别版本等操作。

找到需要进行操作的模型进入版本管理界面，在"版本管理"页面，展示该算法所包含的全部版本，包括名称、中文名称、模型来源、模型类型、所有版本、创建时间、创建类型、模型状态、A框架，并支持"查看详情与转换"和"删除"操作。

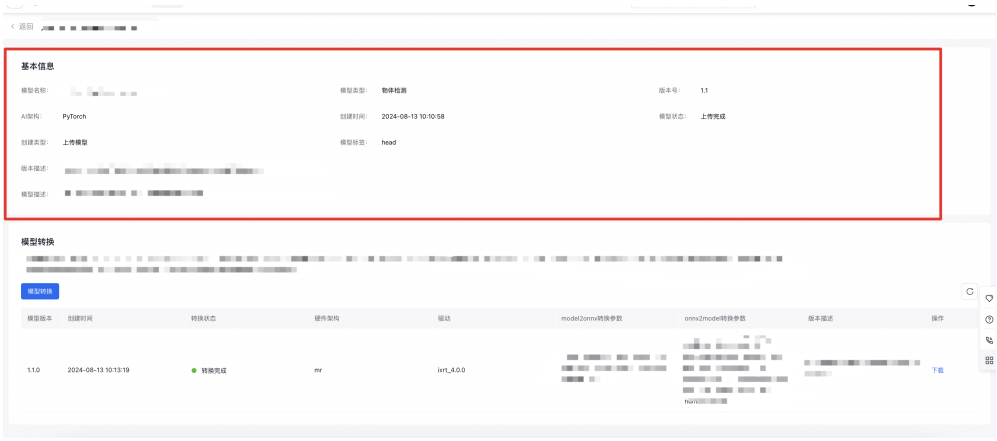
操作

- 详情与转换

进入"详情与转换"界面可以查看到第一部分为模型的一些基本信息，第二部分为"模型转换算法操作"部分，模型转换完成后才可以在多合一算法中的配额模型中找到转换好的算法模型，从而实现多合一的操作。

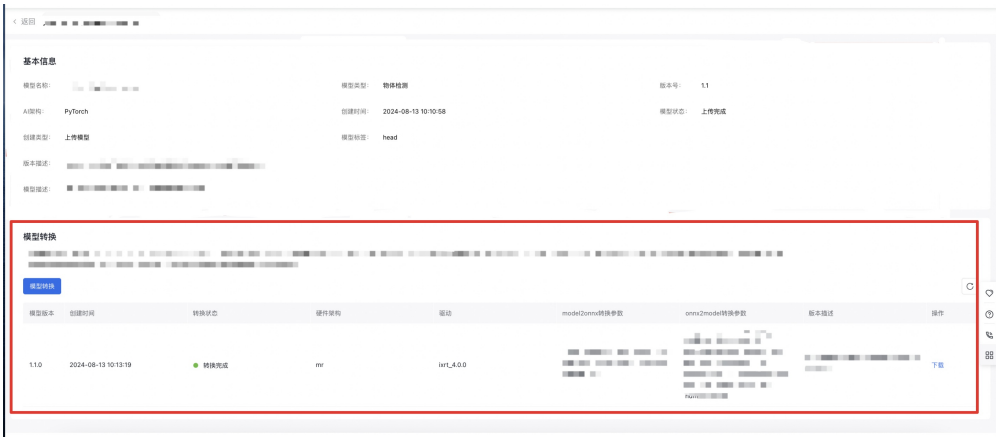
- 基本信息

基本信息中可以查看到模型的一系列详细信息，如："模型名称"、"模型类型"、"版本号"、"AI架构"、"创建时间"、"模型状态"、"创建类型"、"模型标签"、"版本描述"、"模型描述"，等一系列信息。



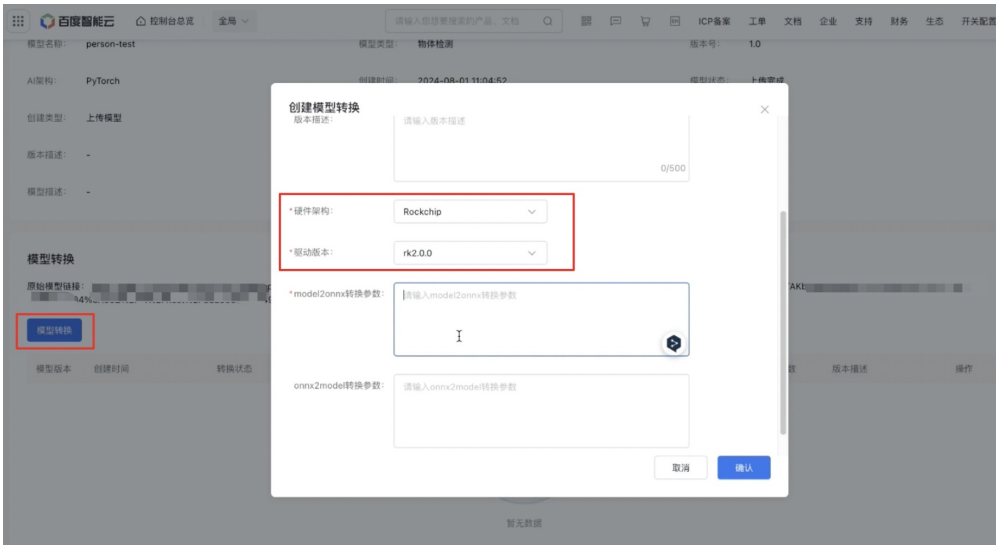
- 模型转换

模型转换部分主要包括原始模型链接和转换完成后的一些模型算法详情，原始模型链接即是创建模型时填写的URL算法链接。

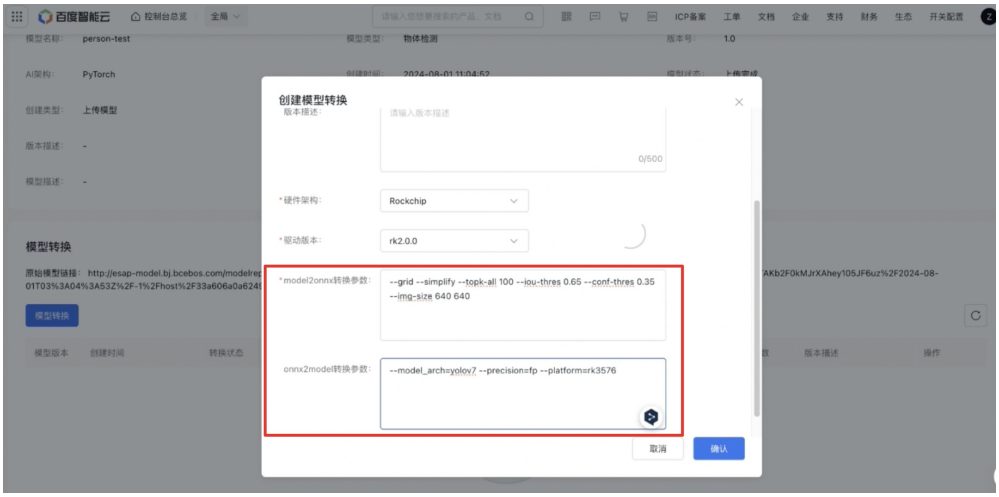


- 创建模型转换

单击模型转换进入转换操作界面，根据自身服务器架构情况和驱动选择对应版本。



模型转换主要分为两部分，第一部分为model2onnx转换，第二部分为onnx2model转换。



model2onnx参数详解：

"--grid": 将检测头映射anchor的操作放到模型内部 （强制加入）

"--simplify": 对产出的onnx模型进行结构性简化（强制加入）

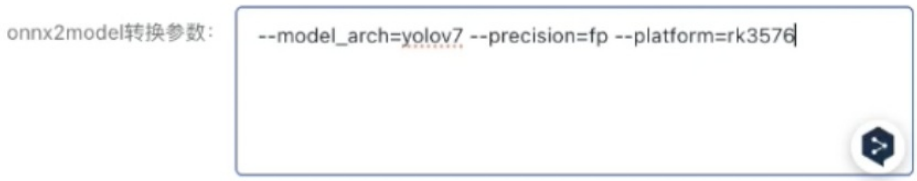
"--topl-all": 导出带NMS后处理模型时的最大目标框保留数（建议100～1000之间）

"--iou-thres": 导出带NMS后处理模型的IOU阈值 （取值范围0～1， 需要根据实际场景下的遮挡情况进行填写，遮挡情况约验证，值越大最大建议不超过0.75）

"--conf-thres": 导出带NMS后处理模型的置信度阈值（取值范围0～1）

"--img-size": 推理的目标尺寸（建议320～640 ，且为32的倍数，数值越大，其精度越高相应的推理延时越大）

onnx2model参数详解：



"--model_arch": 参数要与选择的网络类型对应。如YOLOv7网络，则填入yolov7。

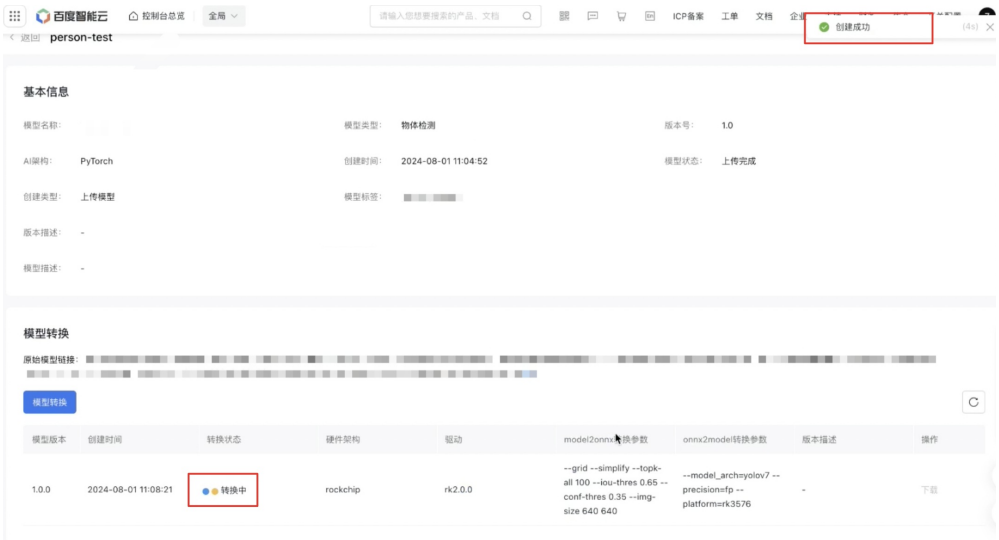
"--precision": 参数为精度参数，是用浮点还是int8，这里使用的浮点 所以填写的为fp，如果是int8则填写int。

"--platform": 参数要与选择的盒子类型对应。如RK3576类型，则填入rk3576。

> onnx2model参数除此之外还有其他类型服务器的参数可填写，可以跳转到模型转换参数一览表文档查看其他服务器的onnx2model参数填写规则。

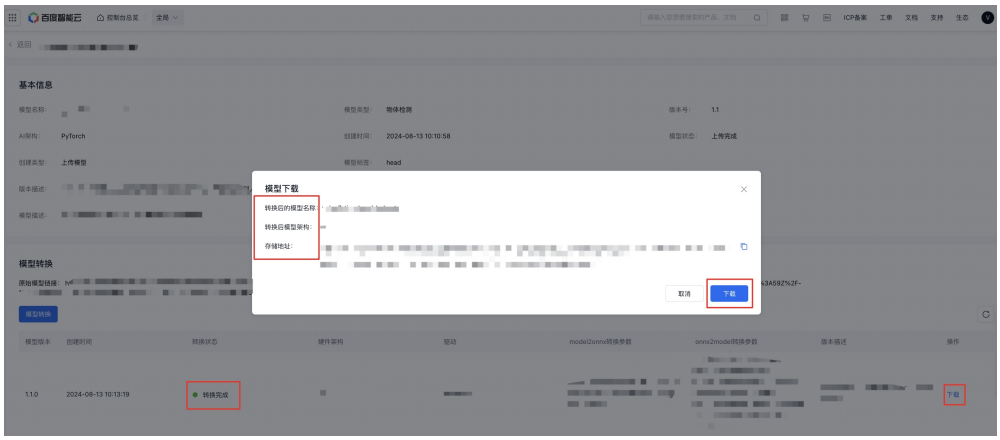
- 查看状态

创建模型转换完成后会跳转到"详情与转换"界面，在这个界面可以看到刚才转换的转换状态，等待状态转换完毕后，就可以在“添加多合一算法”中的“配额模型”中查找到这个模型了。



- 模型下载

转换完成后在右边的操作一栏会出现“下载”按钮，单击进入下载界面可以看到一些转换完成后的一些基本信息，如：“转换后的模型名称”，“转换后模型架构”，“镜像存储地址”一系列信息，可以直接复制复制镜像地址进行操作，也可以直接下载转换并加固好的模型，单击下载即可。



套件管理

套件概览

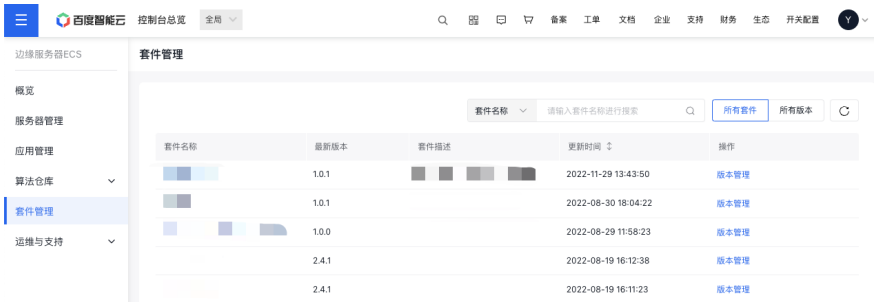
边缘服务器ECS将多款应用、算法等组件进行整合包装，成为套件，帮助用户一站式满足场景化需求。

在边缘服务器上部署套件，可以快速使用、管理该方案所需的应用和算法。多款应用与算法相互配合，借助各自能力，为用户提供各个场景下的AI、运维等解决方案能力。

边缘服务器ECS为用户提供多款套件，涵盖智慧零售、智慧工地、智慧高速等多种应用场景。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“套件管理”，查看、管理用户拥有的全部套件。

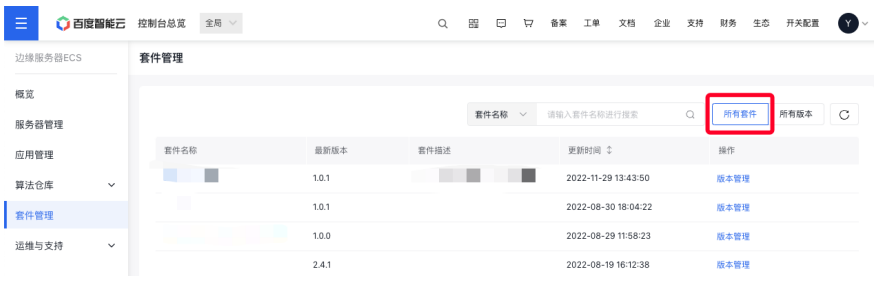


套件列表

展示该账号已购买的全部套件，展示方式包括：所有套件、所有版本。

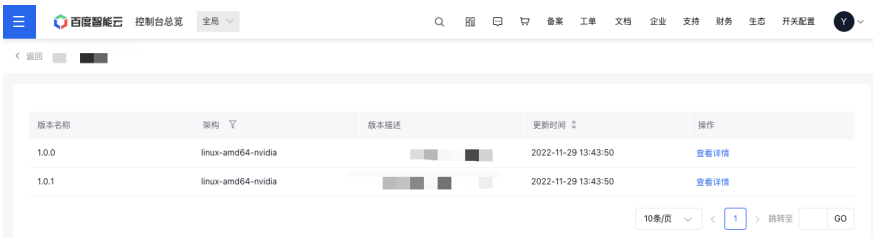
所有套件

在"所有套件"模式下，列表按照名称展示账号的所有套件，每款算法展示一次，同时展示套件的最新版本、套件描述、更新时间，并支持"版本管理"操作。



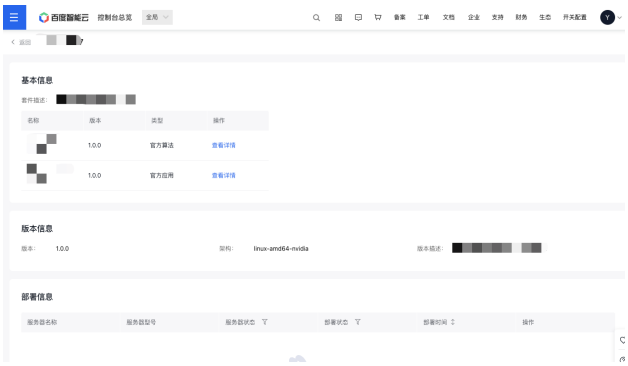
- 版本管理

在"版本管理"页面，展示该套件所包含的全部版本，包括版本名称、版本描述、更新时间等，并支持"查看详情"操作。



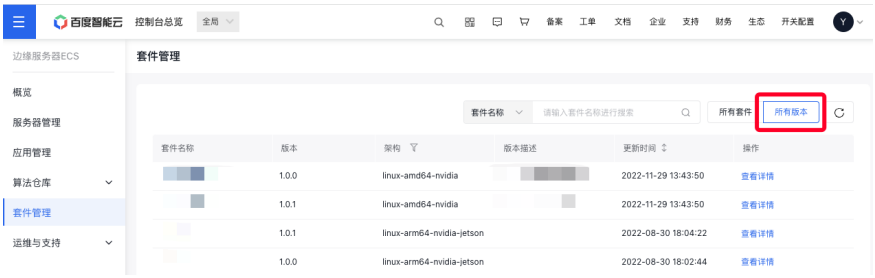
- 查看详情

在"查看详情"页面，展示该套件的基本信息、套件所包含的应用和算法详情，以及套件介绍、版本信息等。



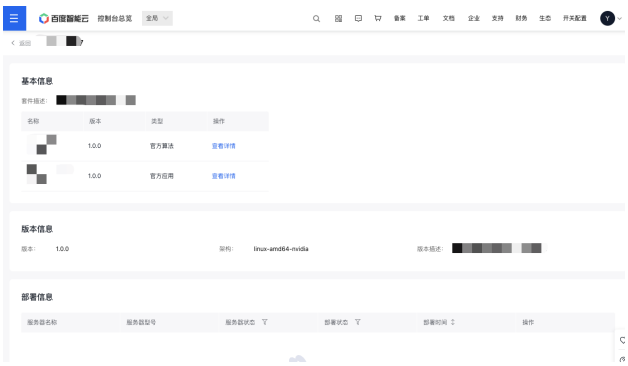
所有版本

在"所有版本"模式下，列表展示账号的所有套件的所有版本，同时展示套件名称、版本号、版本描述、更新时间，并支持"查看详情"操作。



- 查看详情

在"查看详情"页面，展示该套件的基本信息、套件所包含的应用和算法详情，以及套件介绍、版本信息等。



运维与支持

告警管理

ECS平台具备资源监控告警能力，平台根据用户所设置的警戒线，提供边缘服务器各类资源的实时告警，并记录告警历史。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“服务器管理”，进入服务器管理页面。

3. 点击"告警管理"按钮，进入告警管理页面。

当前告警

展示用户所选服务器当前的告警情况。

展示内容包括：服务器名称、告警级别、资源类型、触发时间、当前数值、触发时警戒线等。

在概览页点击告警情况的"立即处理"按钮，跳转至本页面。

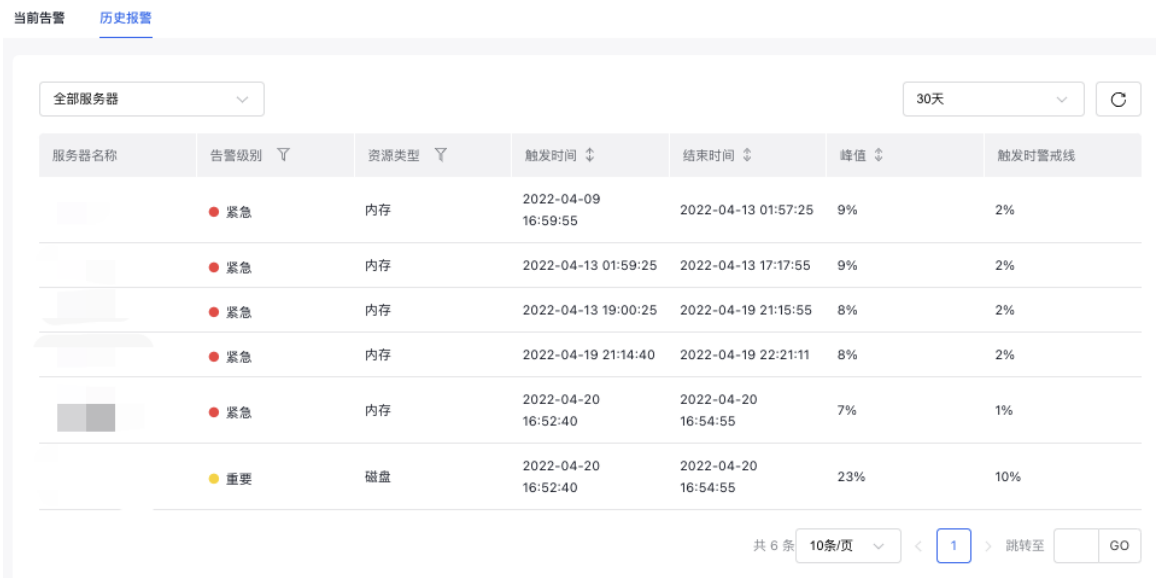
在服务器详情页点击告警情况的"立即处理"按钮，跳转至本页面，并自动选择对应的服务器名称。

历史告警

平台自动记录所有服务器的历史告警情况，并自动保存30天。

用户选定服务器名称、时间范围后，自动查询相应的历史告警信息。

展示内容包括：服务器名称、告警级别、资源类型、触发时间、结束时间、峰值、触发时警戒线等。



应用日志

ECS平台对于所有已激活服务器上运行的全部应用、全部容器，自动记录其应用日志，并保留30天，支持用户进行查看。

路径

1. 登录ECS控制台。

2. 点击左侧导航栏“服务器管理”，进入服务器管理页面。

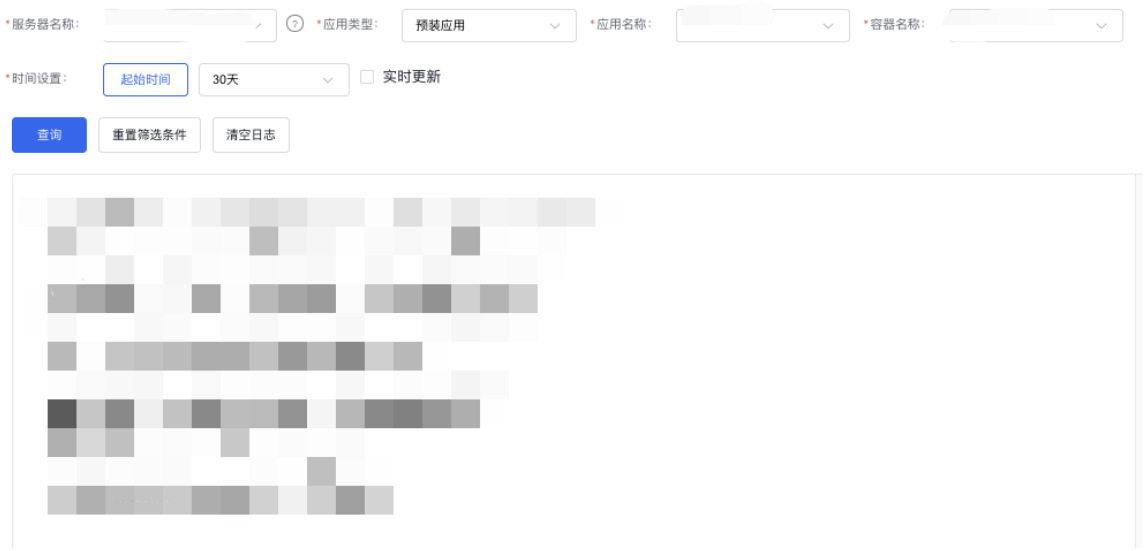
3. 点击"应用日志"按钮，进入应用日志查询页面。

日志查询

用户在查询日志时，需要选择：服务器名称（仅限在线服务器）、应用类型、应用名称、容器名称，并设置查询的时间范围。

条件设置完成后，点击查询按钮，即可进行日志查询。

同时，也支持"重置筛选条件"、"清空日志"等操作。



实时更新

如果勾选"实时更新"，点击查询按钮，则从当前时间开始，自动滚动更新显示每一条新增日志。

售后服务

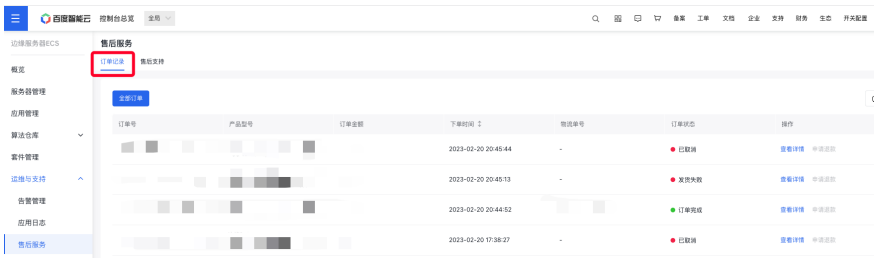
边缘服务器ECS平台为用户提供便捷的售后服务。

通过"**订单记录**"功能，用户可以管理历史订单。通过"**售后支持**"功能，用户可以反馈硬件、软件等各种问题，并快速与ECS团队取得联系。

订单记录

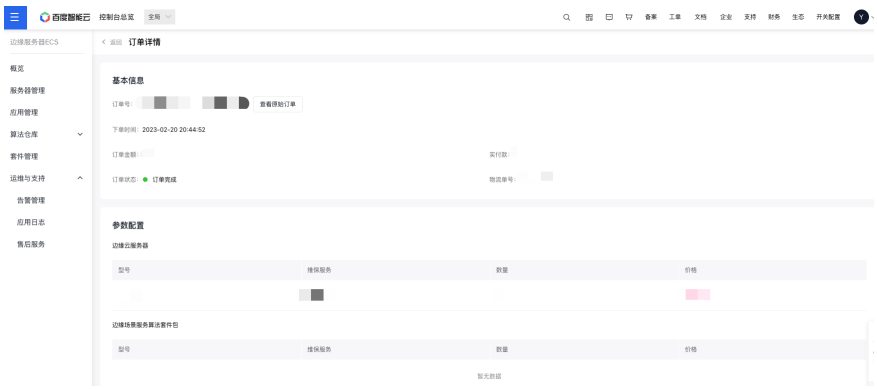
路径

1. 登录ECS控制台。
2. 点击左侧导航栏“运维与支持”，选择"售后服务"。
3. 点击"订单记录"，查看、管理本账户的历史订单。



订单详情

点击"查看详情"，可以查看订单的各项基本信息，以及所购边缘服务器的参数配置详情。



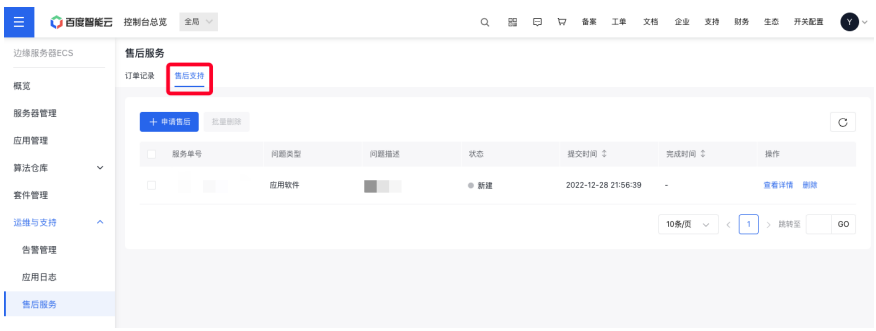
全部订单

点击"全部订单"按钮，可以查看、管理全部原始订单，包括暂未付款和已作废的订单。

售后支持

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“运维与支持”，选择"售后服务"。
3. 点击"售后支持"，查看售后支持的历史记录，或申请新的售后支持。



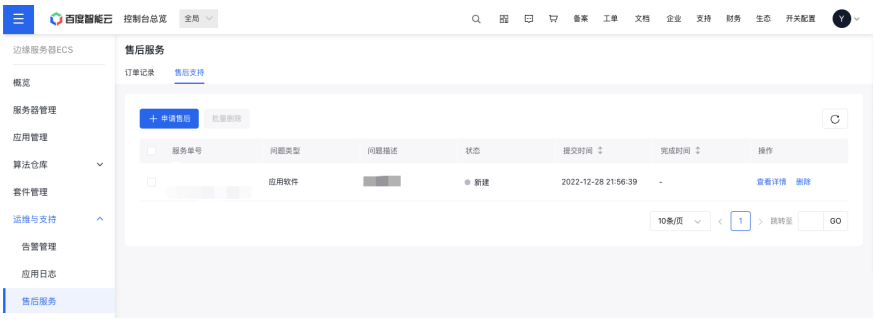
申请售后

点击"申请售后"按钮，进入售后申请页面，用户可以选择问题类型，并填入相应的问题描述、图片信息等，将遇到的问题反馈

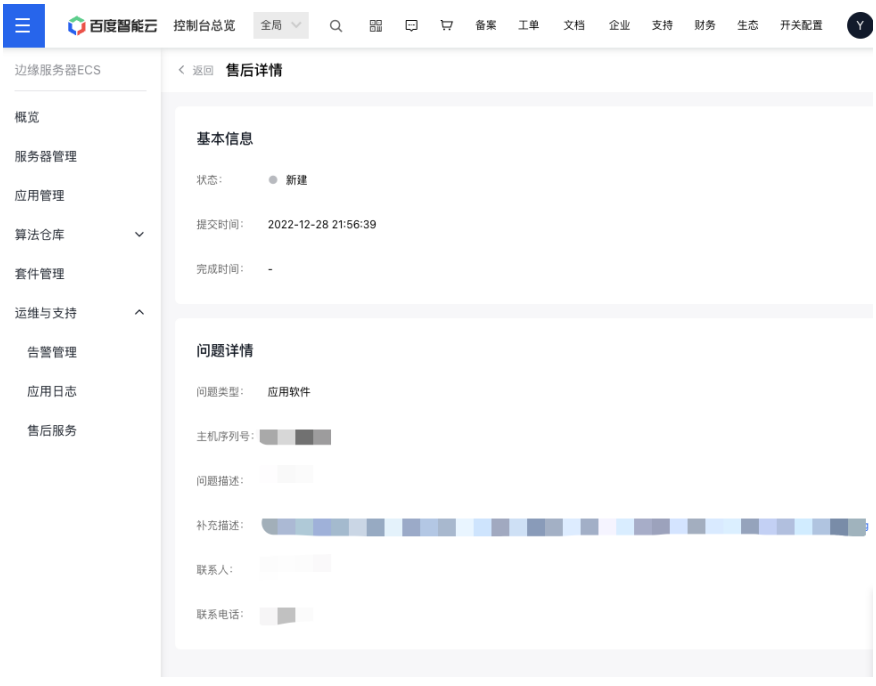
给ECS团队。ECS团队将第一时间与用户取得联系，提供售后服务，帮助用户解决问题。

售后记录

在售后支持页面，用户可以查看、管理各项售后支持的历史记录。



点击"查看详情"按钮，可以查看该条售后服务信息的问题详情、处理进度等。



边缘控制台

接入边缘控制台

在边缘网络下，用户可以通过接入边缘控制台，对边缘服务器单机进行管理和操作。

同时，在一些功能上边缘控制台也与云端管理平台保持交互关系。

接入方法

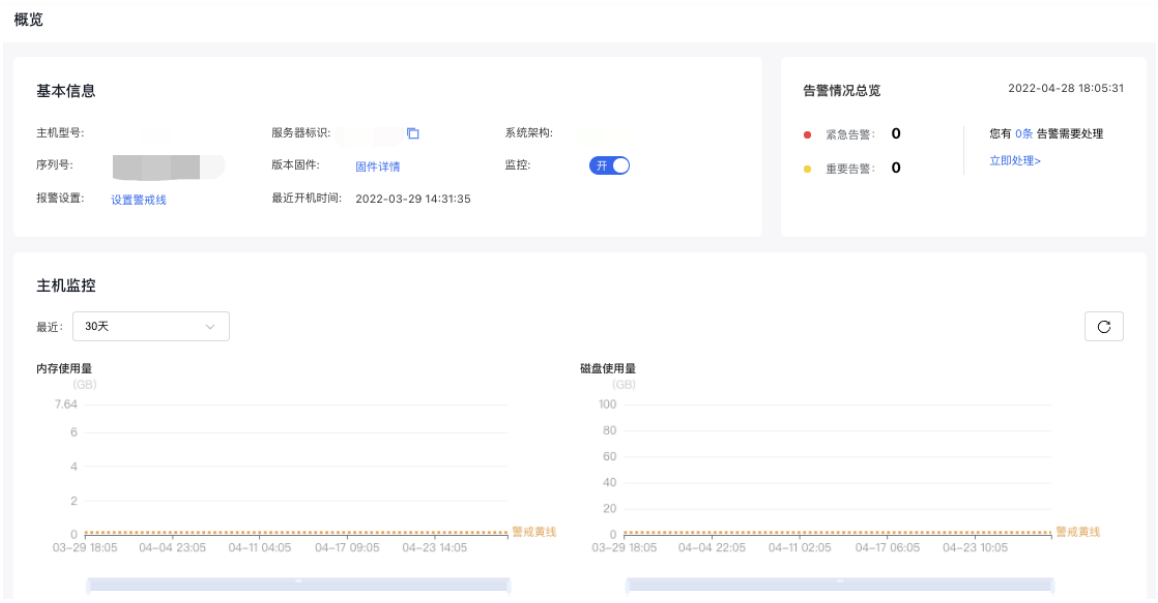
接入方式参考快速入门。

使用边缘控制台

ECS边缘控制台，是边缘服务器的本地管理平台，支持离线管理边缘服务器单机。 边缘控制台的操作方式与云端管理平台基本保持一致。

概览

在概览页面，可以查看该边缘服务器的基本信息、资源使用监控、告警情况等。



网络配置

在网络配置页面，自动识别并展示该服务器拥有的全部网络，用户可以对该服务器的网络配置进行修改。

此处展示网络名称、网络类型、IP地址、子网掩码、网关地址、默认路由，并支持"修改配置"操作。

修改配置

控制台自动识别网络名称，用户可以通过自动获取或手动输入的方式对网络配置进行修改。

- 自动获取

该网络自动DHCP获取相关配置参数。

- 手动输入

该网络的配置参数由用户手动输入。

需要输入的参数包括：

必填项：IP地址、子网掩码；

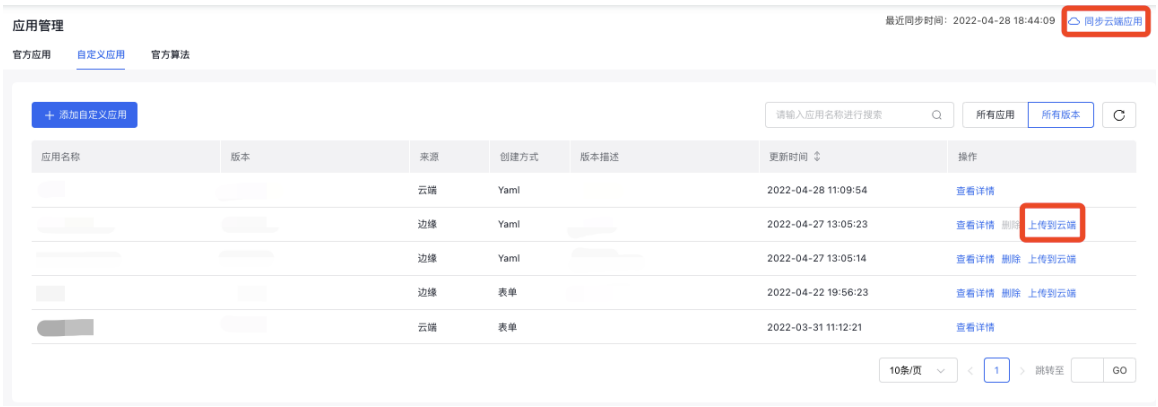
选填项：首选DNS服务器、备选DNS服务器、网关地址；

同时需要选定本网络是否被设置为默认路由（输入网关地址后可设置默认路由）。

应用管理

在应用管理页面，展示本服务器上已购买的全部官方应用、已创建的全部自定义应用、已购买的全部官方算法。

应用管理的查看方式、操作方式等与云端管理平台相同。



- 同步云端应用

更新自定义应用列表，将从云端创建的自定义应用同步到边缘控制台。

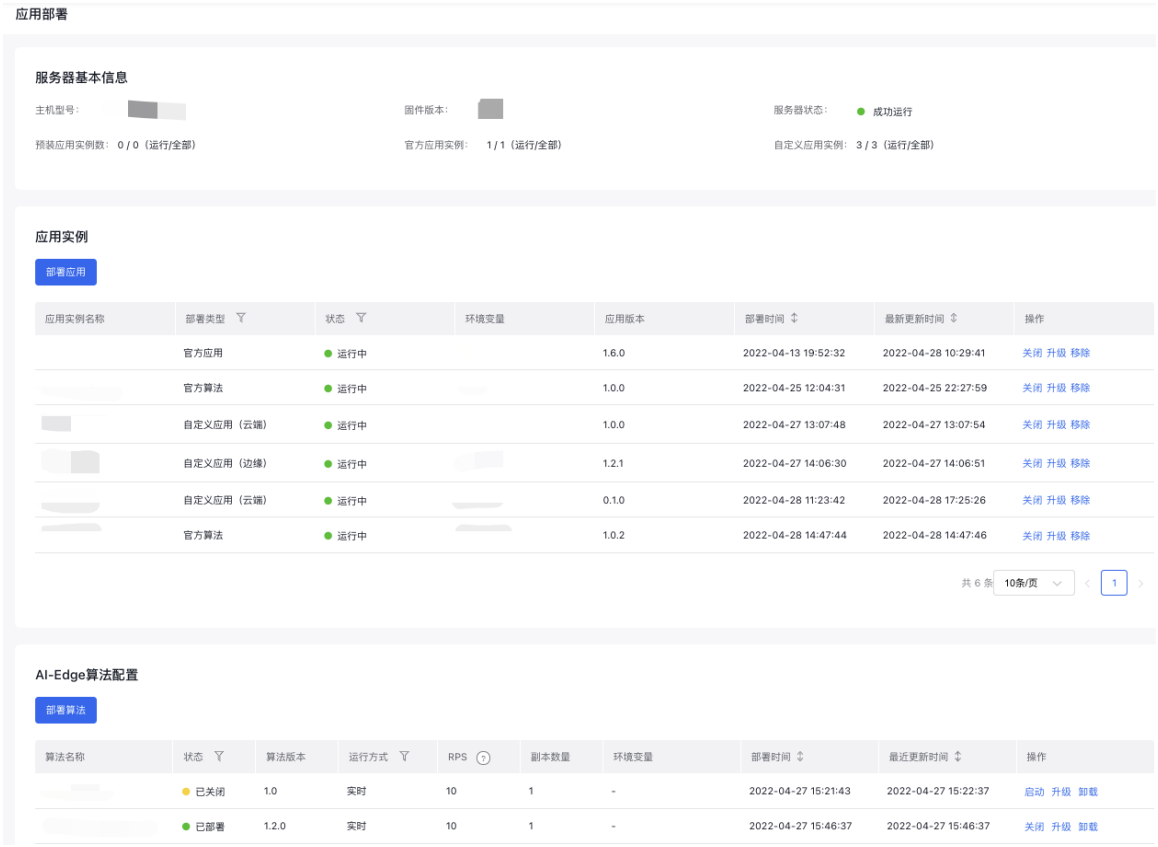
- 上传到云端

将从边缘控制台创建的自定义应用上传到云端管理平台，并更新云端管理平台的自定义应用列表。

应用部署

在边缘控制台应用部署页面，用户可以查看本服务器上已部署的套件、应用、算法。

同时，也支持用户部署套件、应用、算法，查看方式、操作方式等与云端管理平台相同。



运维与支持

告警管理

- 当前告警

展示本服务器当前的告警情况。

展示内容包括：告警级别、资源类型、触发时间、当前数值、触发时警戒线等。

在概览页点击告警情况的"立即处理"按钮，跳转至本页面。

- 历史告警

边缘控制台自动记录本服务器的历史告警情况，并自动保存7天。

用户选定时间范围后，自动查询相应的历史告警信息。

展示内容包括：告警级别、资源类型、触发时间、结束时间、峰值、触发时警戒线等。

应用日志

展示本服务器的应用日志。边缘控制台自动保留本服务器的应用日志，并存储7天。

用户在查询日志时，需要选择：应用类型、应用名称、容器名称，并设置查询的时间范围。

条件设置完成后，点击查询按钮，即可进行日志查询。

同时，也支持"重置筛选条件"、"清空日志"等操作。

配置多用户访问控制

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。

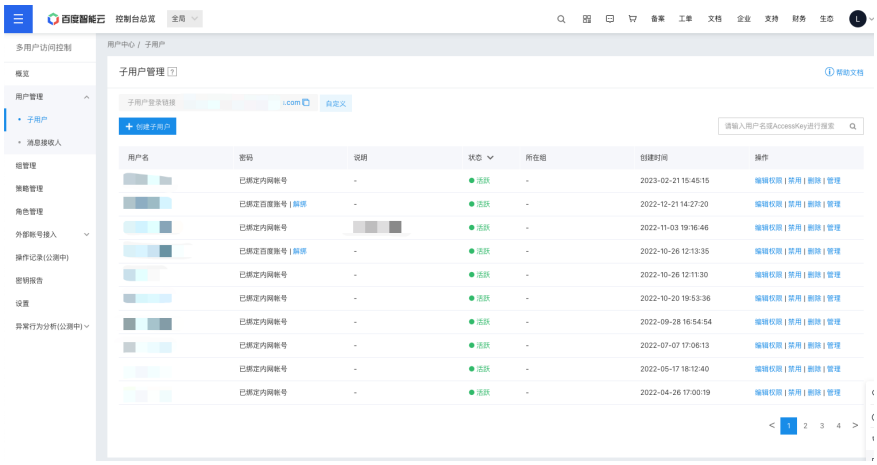
使用场景

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

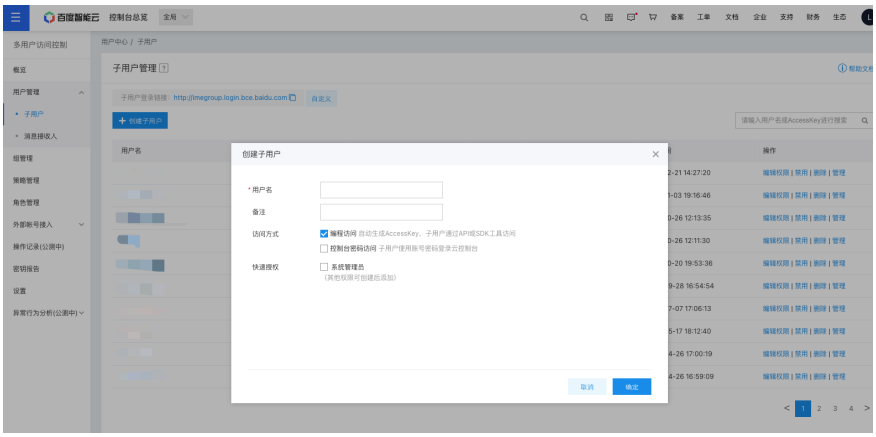
创建子用户

1. 主账号用户登录后在控制台选择“多用户访问控制”进入用户管理页面。

2. 在左侧导航栏点击“用户管理”，在“用户管理 > 子用户”列表页，点击“创建子用户”。



3. 在弹出的“创建子用户”对话框中，完成填写“用户名”和确认，返回“子用户管理列表”区可以查看到刚刚创建的子用户。(如果子账号需要上传文件，建议授权“编程访问”，否则上传文件会报错“鉴权失败”)



配置策略

策略类型

ECS 支持系统策略和用户自定义策略两种。

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

系统策略

系统策略包含 ESReadAccessPolicy、ESOperateAccessPolicy、ESFullControlAccessPolicy，权限范围详细如下：

策略名称	权限说明	权限范围
ESReadAccessPolicy	ECS只读权限	应用、算法、套件以及其部署查看 服务器查看 监控、告警查看 扩展件查看 容器查看 订单查看 远程服务、webshell分享查看
ESOperateAccessPolicy	ECS运维权限	应用、算法、套件以及其部署更新维护 服务器更新维护 告警规则更新维护 扩展件更新维护 远程服务更新维护
ESFullControlAccessPolicy	ECS管理权限	应用、算法以及部署创建删除 服务器创建删除 扩展件创建删除 订单创建、退款以及删除等 远程服务、webshell分享创建以及删除

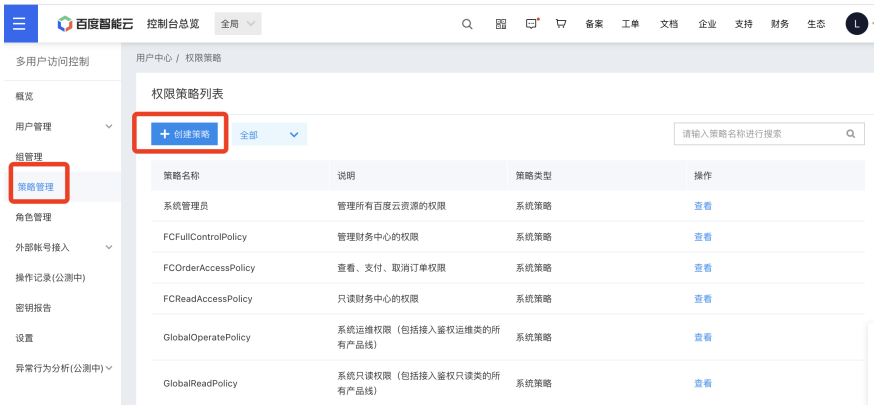
自定义策略

主用户可以通过点击“策略管理>创建策略”添加自定义策略来进行实例级权限控制，自定义策略的添加有"按策略生成器创建"、"按标签创建"和"按策略语法创建"三种方式，用户可以根据具体的权限设置修改策略内容。具体配置方法可以参见创建自定义策略。

策略创建流程

1. 点击创建策略

页面左侧选择"策略管理"，在新的页面中点击"创建策略"。

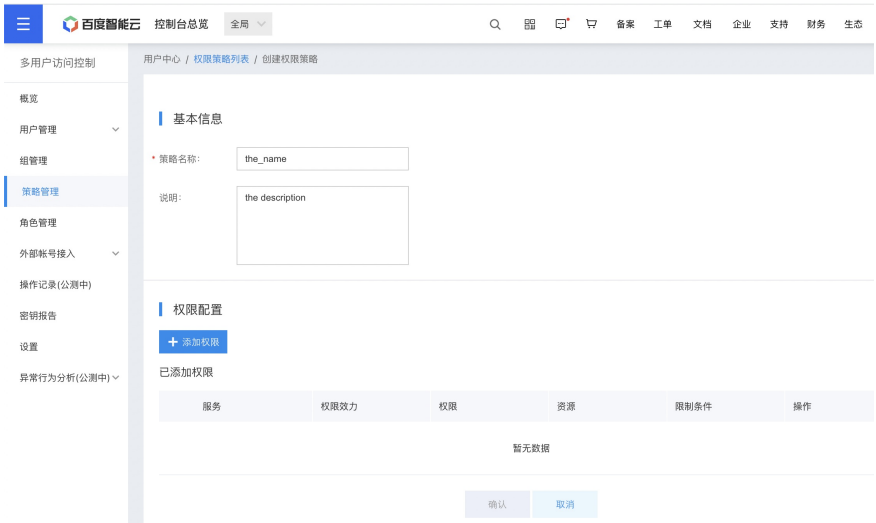


2. 选择创建策略的方式

您可以选择"按策略生成器创建"或者"按标签创建"。若您需要对某个标签下的实例创建策略，请选择按标签创建。若您不需要为标签创建策略，则选择按策略生成方式创建。

3. 填写策略基本信息

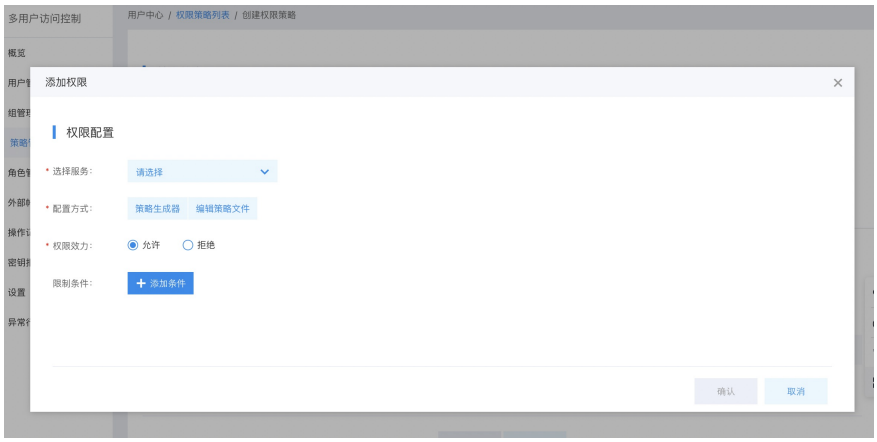
在创建策略的页面中，填写需要创建的策略的名称，并在说明中填写该策略的备注信息，如该策略应用的实例，希望赋予的子用户或者策略的使用场景，以免将该策略和其他策略混淆。



4. 填写策略权限配置

填写完策略的基本信息后，您需要填写该策略中包含的使用权限。

- 首先，您需要填写该策略生效的产品服务。若您希望将该策略生效于 ECS 服务，则在服务中选择"边缘服务器ECS"；
- 选择策略的配置方式。其中，策略生成器方式允许您将系统中已生成的权限配置到该策略中；而策略文件需要您编辑 ACL 权限策略，详细的语法可参考文档策略语法；
- 若您选择策略生成器方式，您需要进一步该权限对应的效力为允许还是拒绝。若您选择允许，则该策略中选择的权限将允许被授权的用户操作；若您选择拒绝，则该策略中选择的权限将禁止被授权用户操作；
- 选择权限，您需要选择该策略中包含的权限为只读权限还是管理权限。关于两种权限的区别和说明，请查看本页面上方"系统策略"中的说明；
- 您还可以选择该策略针对的资源。在 ECS 服务中，您可以指定该策略能够生效的具体 Bucket。若您不选择，那么将对全部 Bucket 生效；
- 最后，您需要选择该策略在生效时间上的限制条件。您可以指定该策略在某一时刻后或者某一时刻前生效。

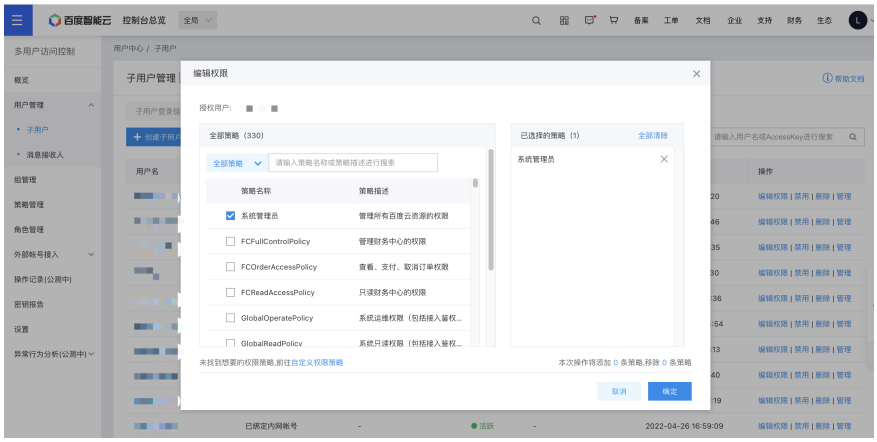


用户授权

当您创建完子用户并生成策略后，即可以为该子用户授予该策略。授予后，子用户在访问主用户资源时，将受到授予策略中已配置权限的限制。

在“用户管理->子用户”列表页中希望授权的子用户后“操作”栏中选择“添加权限”，为用户授予已生成的策略。

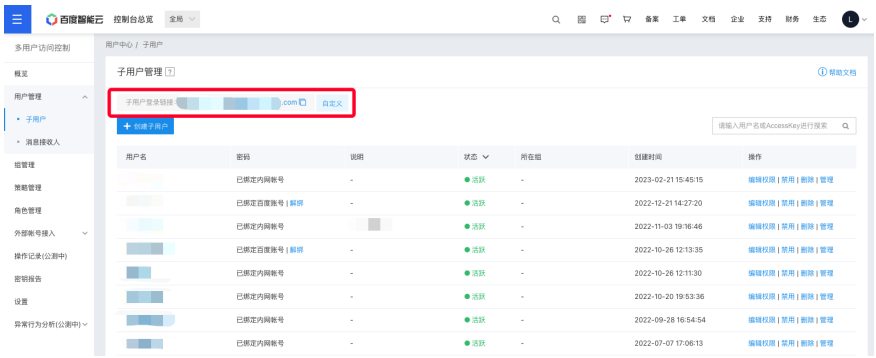
在授权页面中，选择希望授予该子用户的权限对子用户进行访问权限授权。左侧的策略列表中包括全部系统策略和您已经创建的自定义策略。您可以在搜索框中搜索相关的策略名称后，勾选该策略后点击确认完成授权。



子用户登录

添加完子用户权限后，子用户可以使用以下两种方式使用 ECS 服务。

- 点击子用户名可以查看 IAM 用户详细信息，包括 AKSK 及子用户的权限等。子用户可以使用 AKSK 通过 ECS 的周边工具、SDK、API 来使用 ECS 服务。
- 将控制台上方的子用户登录链接发送给子用户，子用户可以通过该链接登录控制台。登录后，该子用户将根据被授权的策略对主账户资源进行操作和查看。



相关文档

其他操作请参考多用户访问控制。

逻辑集群

概览

逻辑集群

逻辑集群主要分为三个功能模块，分别为：**“概览”**、**“新建逻辑集群”**和**“操作”**。

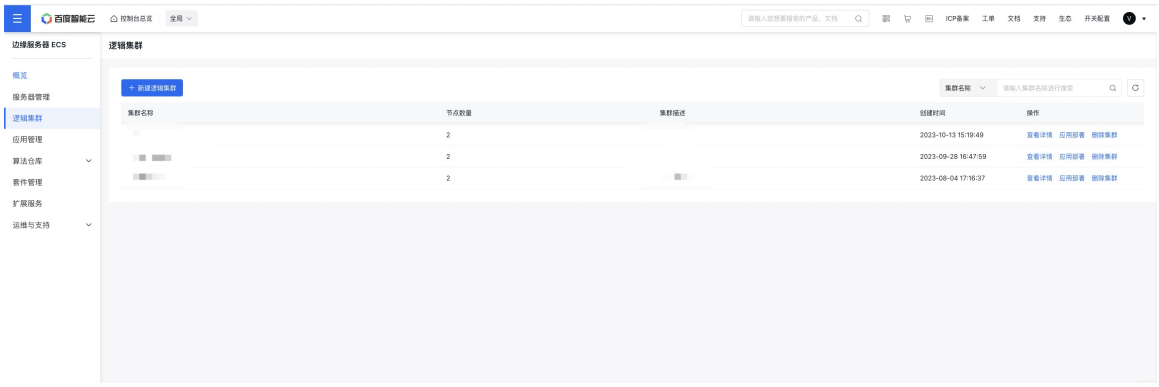
“概览”展示用户创建的全部集群名称、节点数量、集群描述和创建时间等信息；

“新建逻辑集群”用于将机器进行集中纳管创建新的逻辑集群；

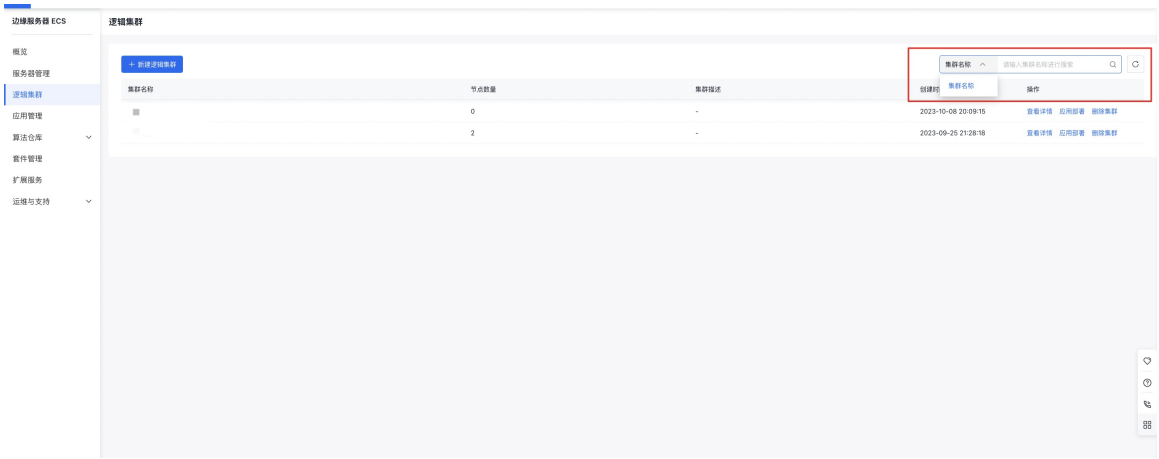
“操作”包括查看详情、应用部署、删除集群多种操作。

概览

通过逻辑集群列表，展示用户创建的全部集群名称、节点数量、集群描述和创建时间等信息。



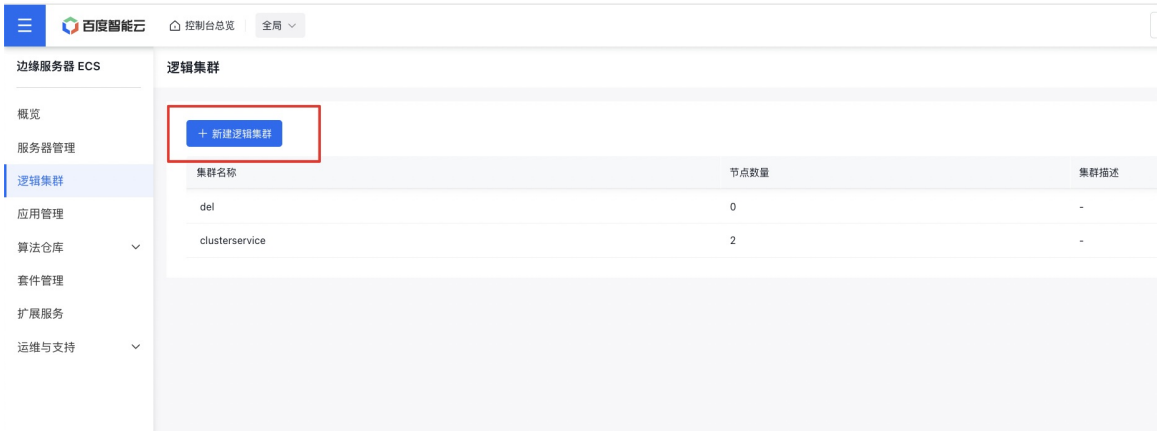
平台支持根据服务器名称、服务器序列号进行搜索。选定条件、输入关键字后，可以进行对应搜索。



通过逻辑集群列表，展示用户创建的全部集群名称、节点数量、集群描述和创建时间等信息。

新建逻辑集群

平台支持客户根据自身需求新建逻辑集群管理服务器。



操作

平台支持多种对创建的集群操作，包括查看详情、应用部署、删除集群多种操作。

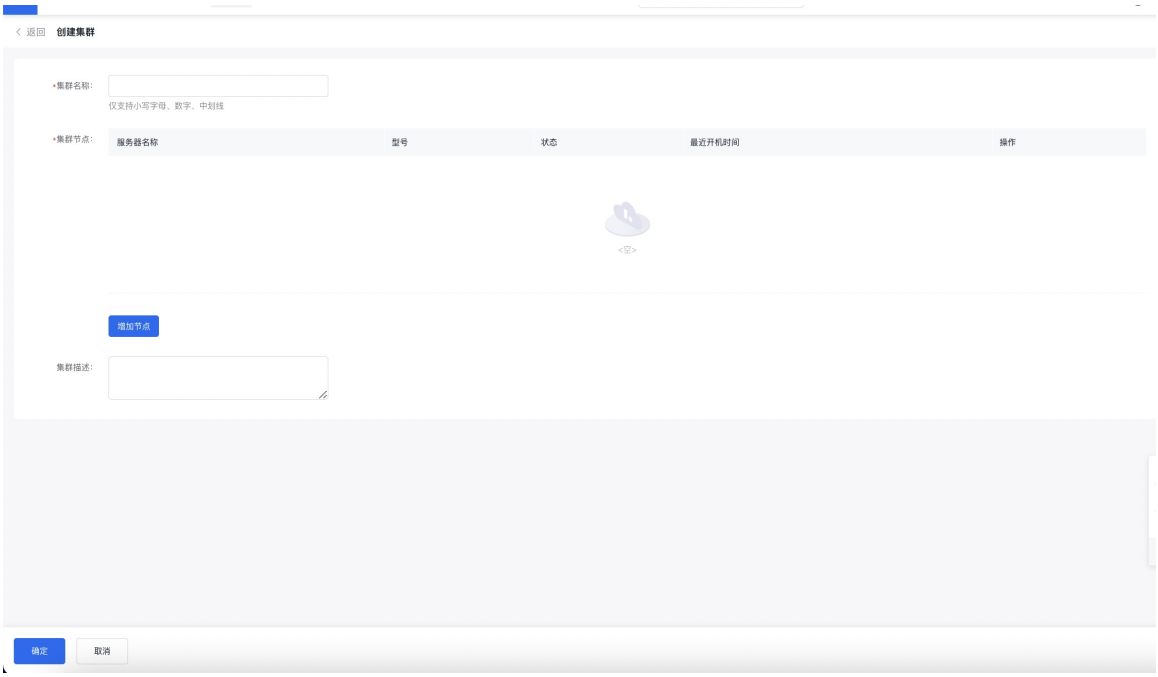
新建逻辑集群

新建逻辑集群

边缘服务器ECS为用户提供逻辑集群服务，用户可以按照自身需求，创建和管理自己的集群。

路径

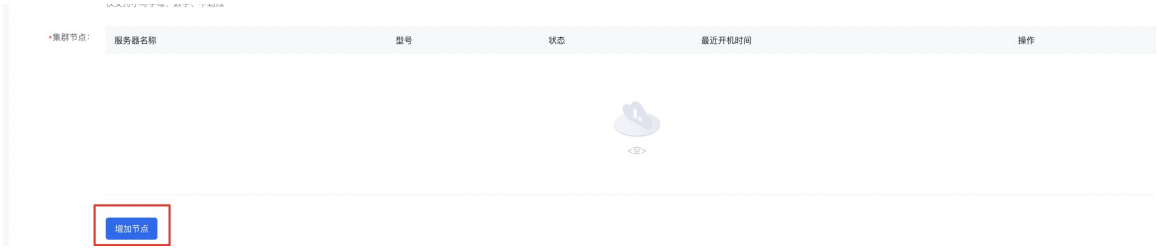
1. 登录ECS控制台。
2. 点击左侧导航栏“逻辑集群”，进入集群管理页面。
3. 点击"新建逻辑集群"按钮，进入创建集群界面。
4. 创建集群分为必填项和选填项，**必填项包括集群名称、增加节点**，选填项为集群描述，根据自身需求填写内容即可。



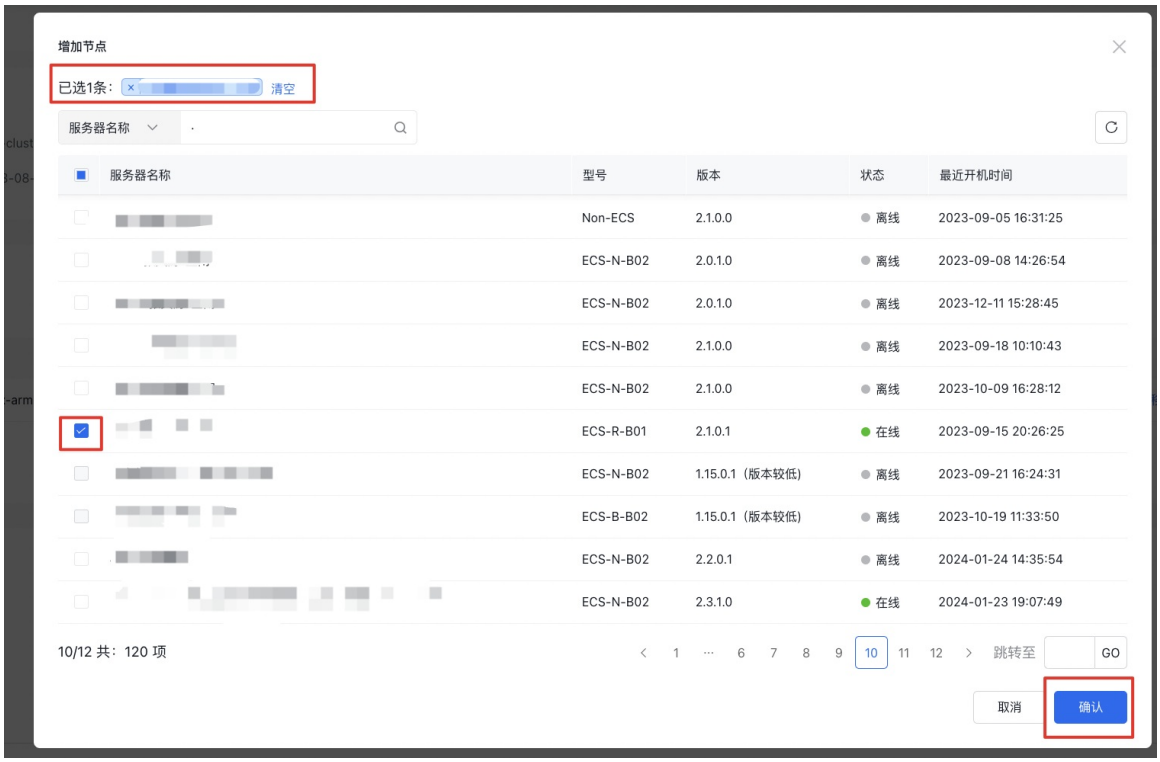
创建集群

1.根据自身需求填写好**必填项集群名称**，填写内容仅支持小写字母、数字、中划线。

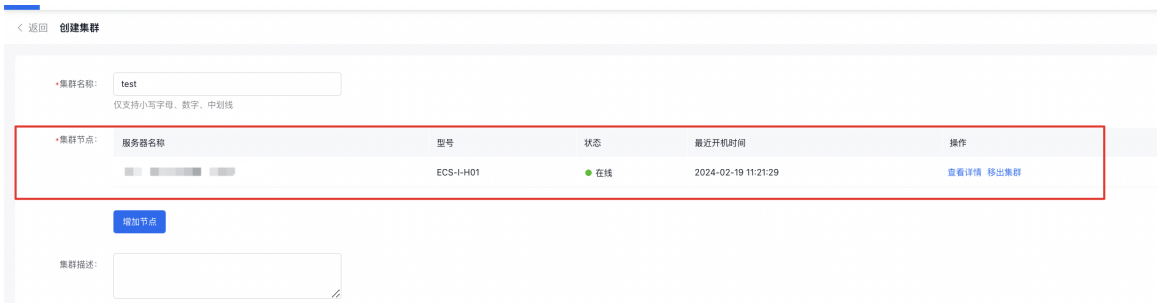
2.单击**增加节点**选择需要纳管到集群的服务器。



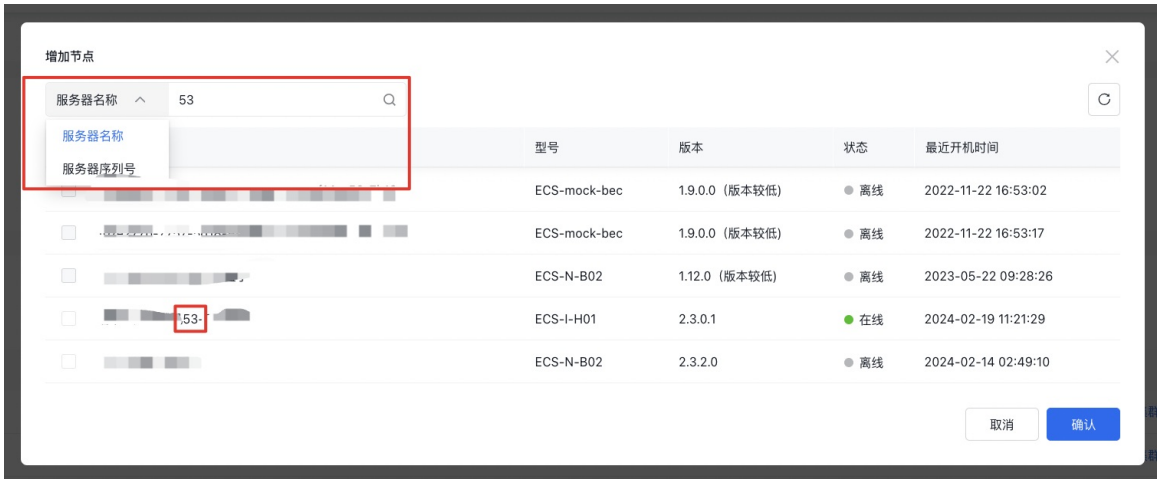
3.进入**“增加节点”**界面，根据个人需求勾选服务器即可，勾选完成后左上角会显示已经勾选了的机器，操作确认无误后点击**“确认”**即可将勾选机器添加至本次操作集群中。



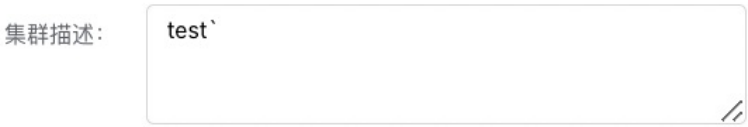
4.添加完成后在集群节点查看操作即可。



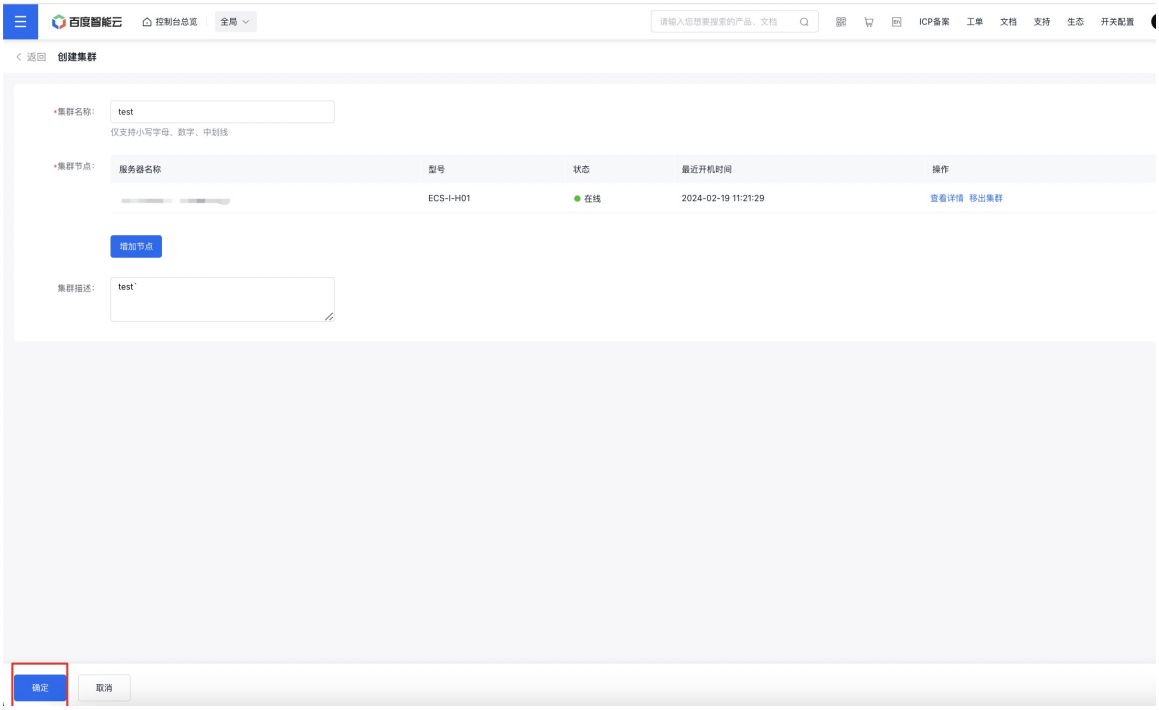
5.增加节点界面支持**检索功能**，检索功能分为**“服务器名称”**，**“服务器序列号”**两种检索方式，另外检索功能支持关键词检索。



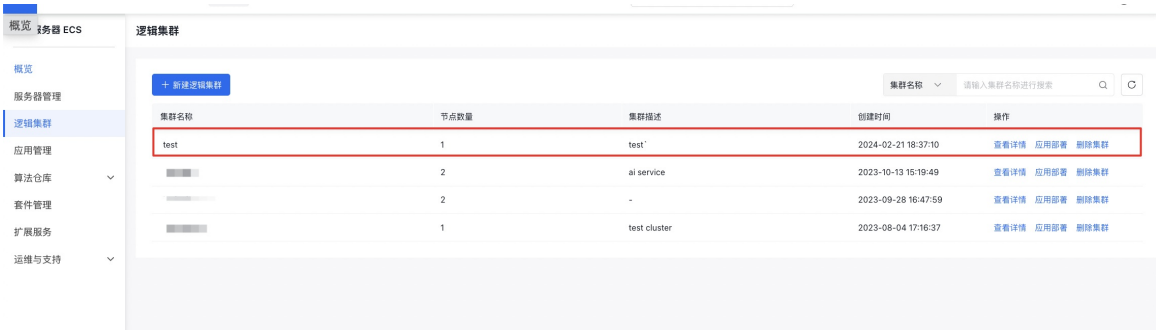
6.集群描述可以根据自身需求填写，不是必填项。



7.根据需求填写完成**“集群名称”**，在**“增加节点”**中选择好服务器，根据自身需求填写好**“集群描述”**后，一切确认无误点击左下角确定即可创建集群。



8.创建好后会自动返回逻辑集群界面，我们在逻辑集群界面即可查看到我们刚才创建好的逻辑集群。



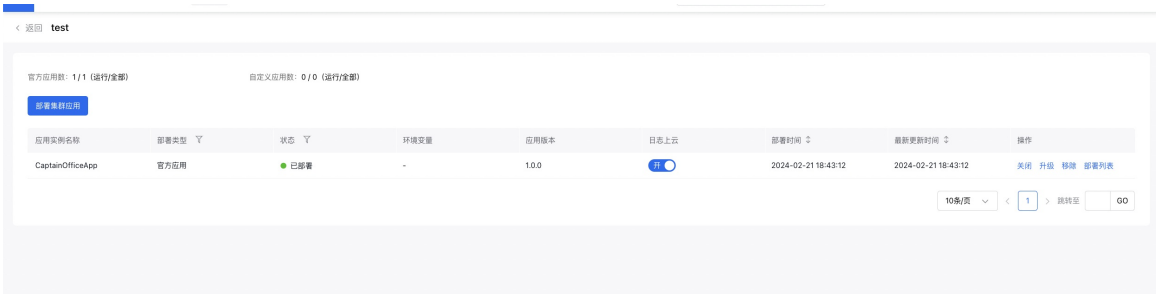
应用部署

应用部署

通过ECS管理平台，用户可以在集群内的服务器上部署应用。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“逻辑集群”，进入集群管理页面。
3. 根据需求选择创建完成的集群，点击"应用部署"按钮，进入应用部署界面详情。



部署集群应用

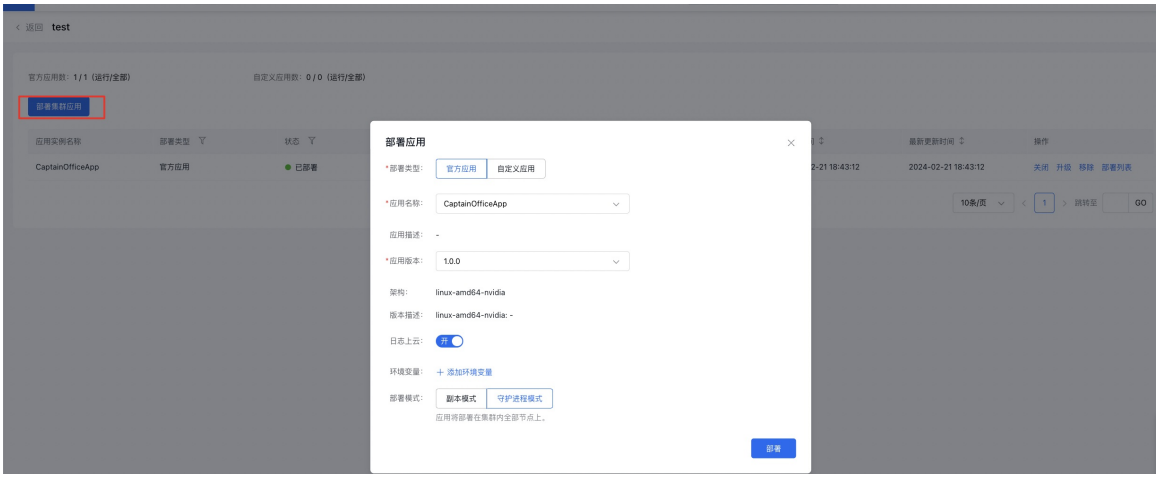
点击"部署集群应用"按钮，开始在该集群服务器上进行应用部署。

选择想要部署的应用类型、应用名称、应用版本、是否开启日志上云、部署模式，全部选择无误后单击部署即可完成部署。

注意：

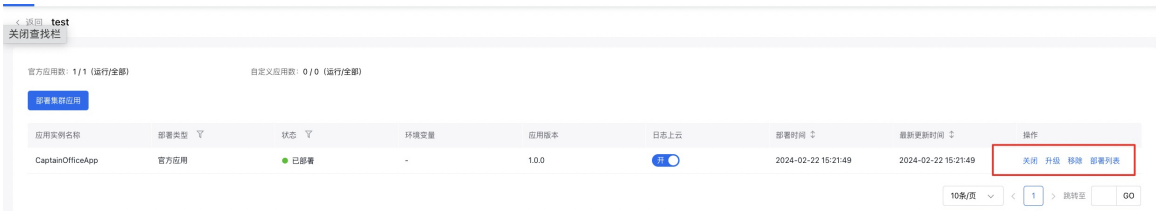
1.只有在线服务器，才能够进行应用部署。

2.部署应用时要注意应用架构和服务器架构是否符合，如果添加应用时和服务器不是相同架构则应用不会部署在机器上，应用只会匹配集群中合适的服务器架构进行部署。



管理集群应用操作

平台支持多种对集群应用的操作，包括关闭应用、升级应用、移除应用，部署列表查看等多种操作。

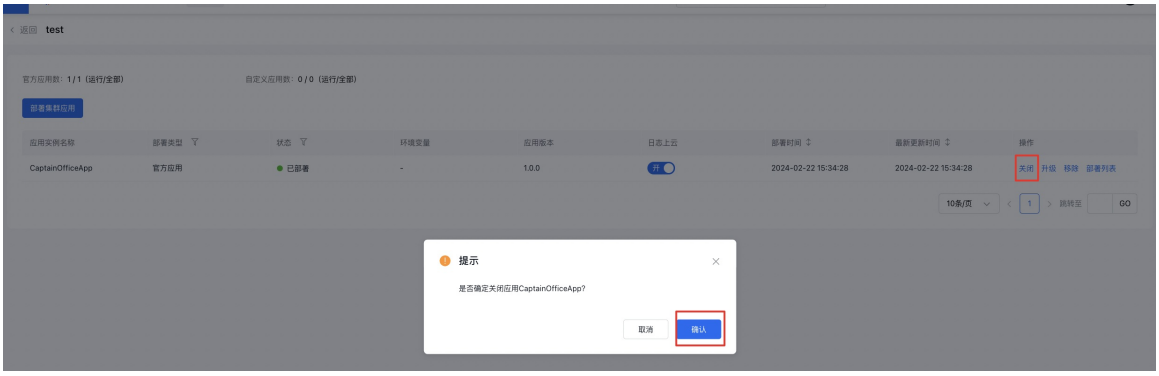


使用方法

1.关闭应用

点击**"关闭"**，进入关闭应用操作页面；

点击确认后应用将会关闭。



2.升级应用

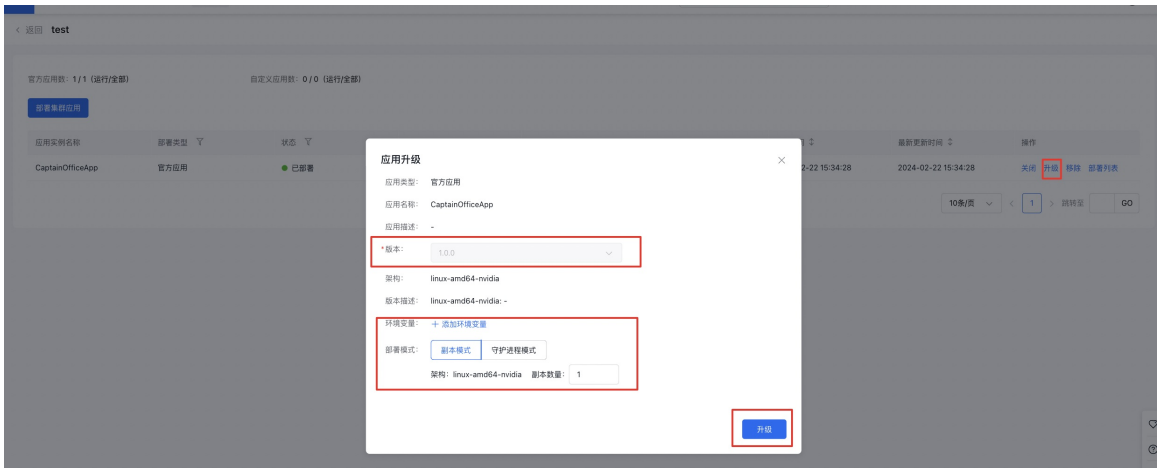
点击**"升级"**，进入关闭应用操作页面；

可以根据自身需求修改应用版本、添加环境变量、更改应用模式，选择完成后点击升级即可完成升级应用操作。

应用模式分为两种；

副本模式：为可以在此集群节点中控制实际部署的应用数量。

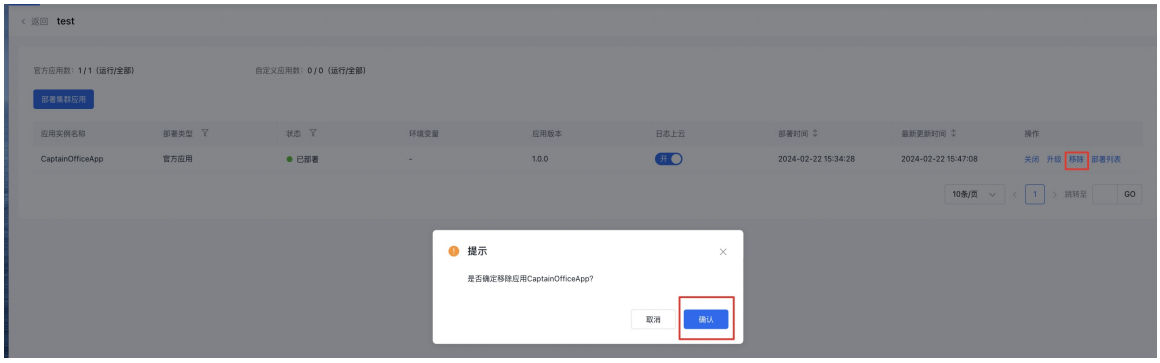
守护进程模式：为应用将部署在集群内全部节点上。



3.移除应用

点击**"移除"**，进入移除应用操作页面；

点击确认后应用将会被移除。



4.部署列表

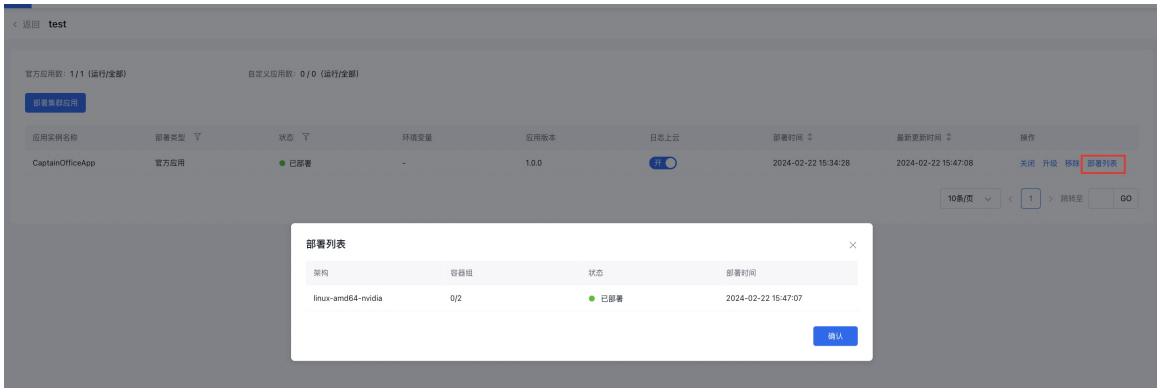
点击**"部署列表"**，进入部署应用的列表详情；

架构为此次应用的架构信息。

容器组为应用实际在此次选择的逻辑集群中部署数量和需求数量，次展示图为实际部署数量为0，需求数量为10。

状态为此应用部署的实际状态，是否部署成功。

部署时间为此应用部署的实际时间。



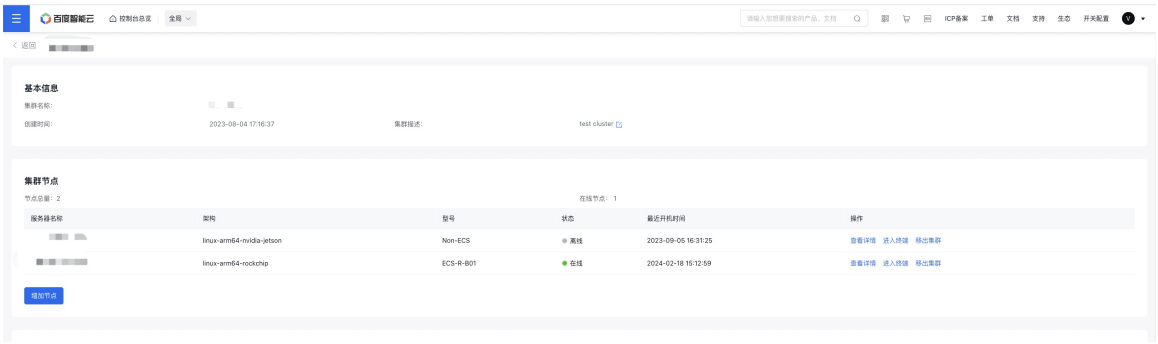
查看详情

查看详情

进入集群组的详情页，可以查看该集群的基本信息、资源使用情况等。

路径

1. 登录ECS控制台。
2. 点击左侧导航栏“逻辑集群”，进入集群管理页面。
3. 根据需求选择创建完成的集群，点击"查看详情"按钮，查看该集群详情。



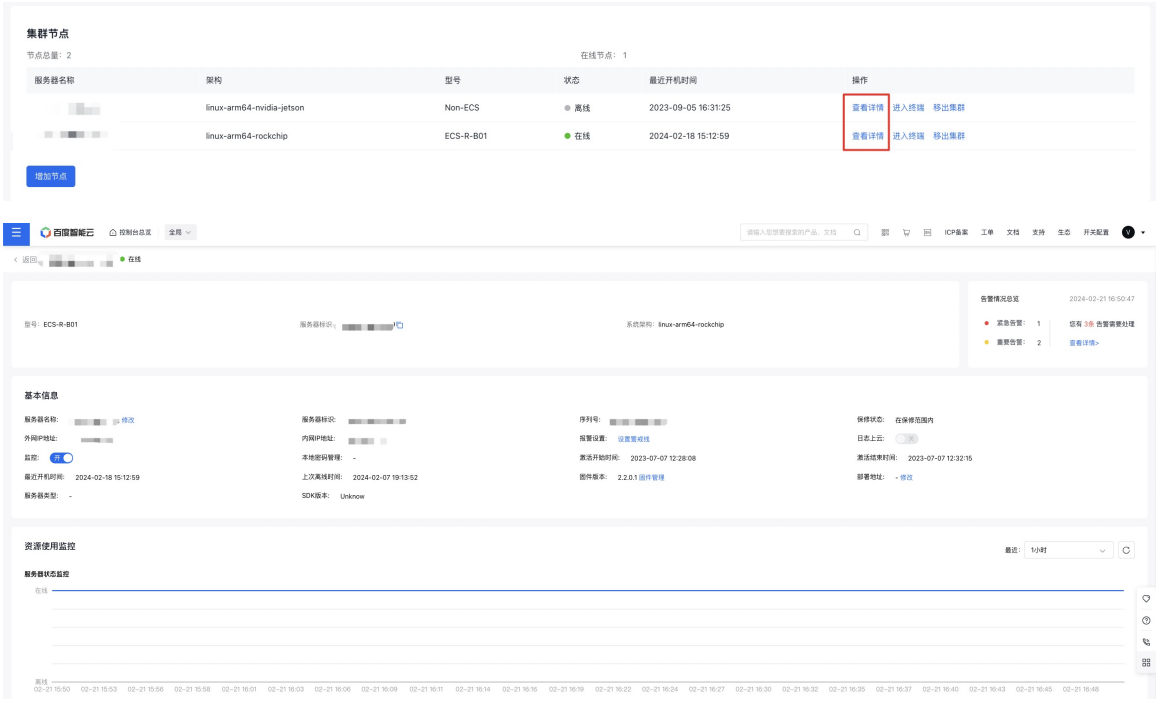
基本信息

基本信息专栏展示该集群的名称、创建时间、集群描述、等基本信息，部分信息支持用户自定义修改。

集群节点

集群节点专栏展示该集群的节点总量、服务器名称、架构、型号、状态、最近开机器时间、操作、增加节点等基本信息。

查看详情 进入查看详情界面会跳转到本次操作的服务器的**基本信息**专栏，进入单台服务器的详情页，可以查看该服务器的基本信息、资源使用情况等。



进入终端 进入终端会跳转到本次操作的服务器的**终端操作**专栏，通过"**进入终端**"功能，用户可以从云端平台，直接访问服务器

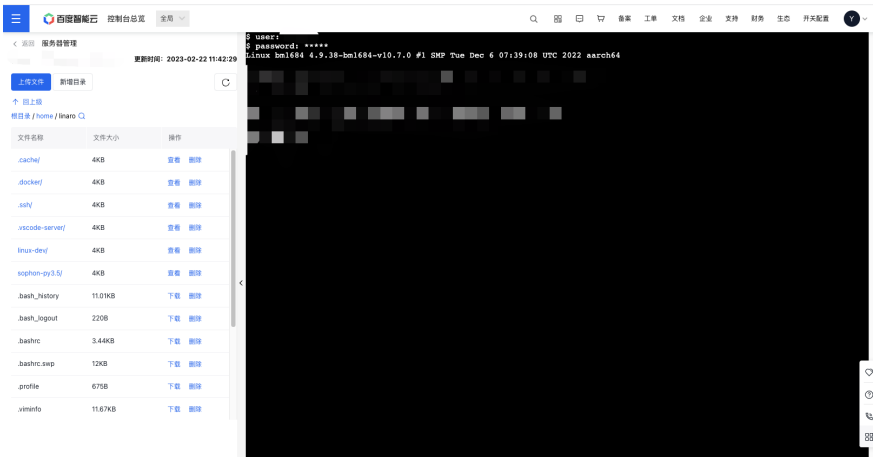
主机，对服务器进行各项操作，并支持远程文件管理。

终端登录

1. 进入终端登录页面后，在右侧操作区，输入服务器对应的SSH账号密码进行登录；

2. 先输入账号，按回车键确认，再输入密码，再次按回车键确认，验证正确后，将自动进入服务器终端操作页面。

如果不知道服务器SSH账号密码，可以联系ECS团队或提交工单，以获取账号密码。

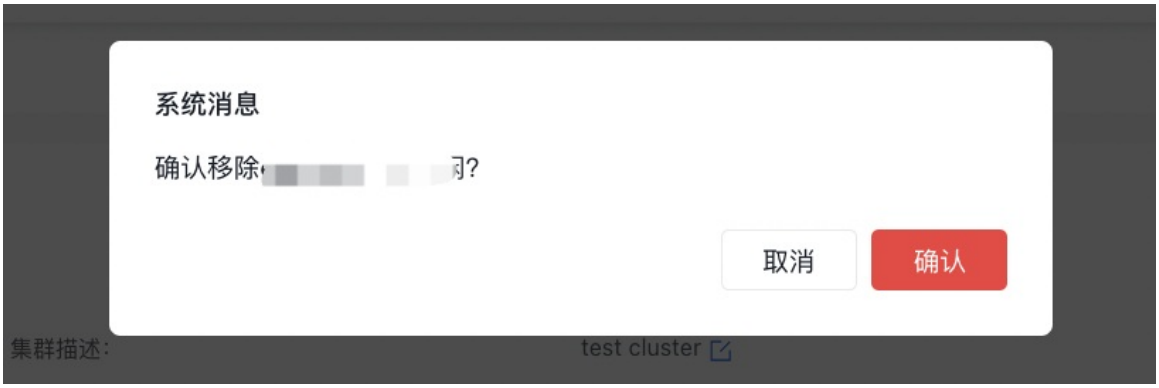


补充 进入终端中包含很多其他操作，如需要的话可以跳转到“进入终端”专栏进行查看。

移除集群

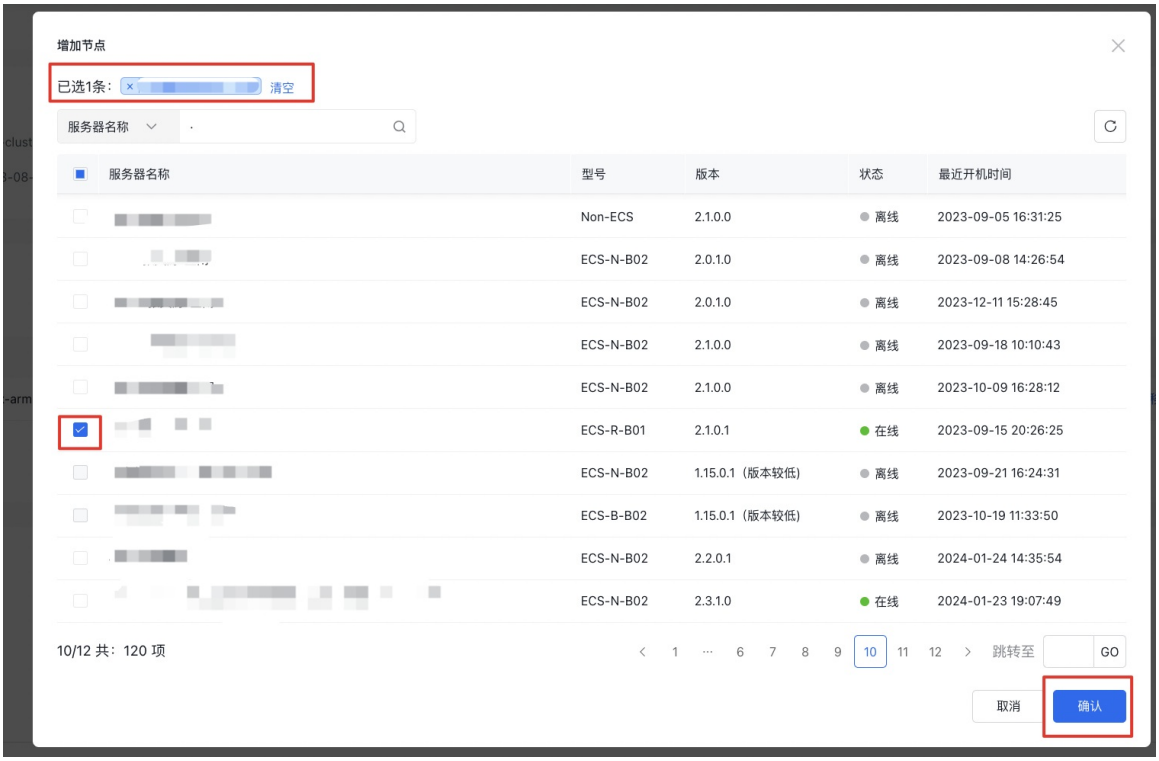
移除集群操作会将本次操作的服务器移除此集群，不再此集群中展示。

进行移除操作会弹出是否确认的信息提示，选择确认后即操作成功。



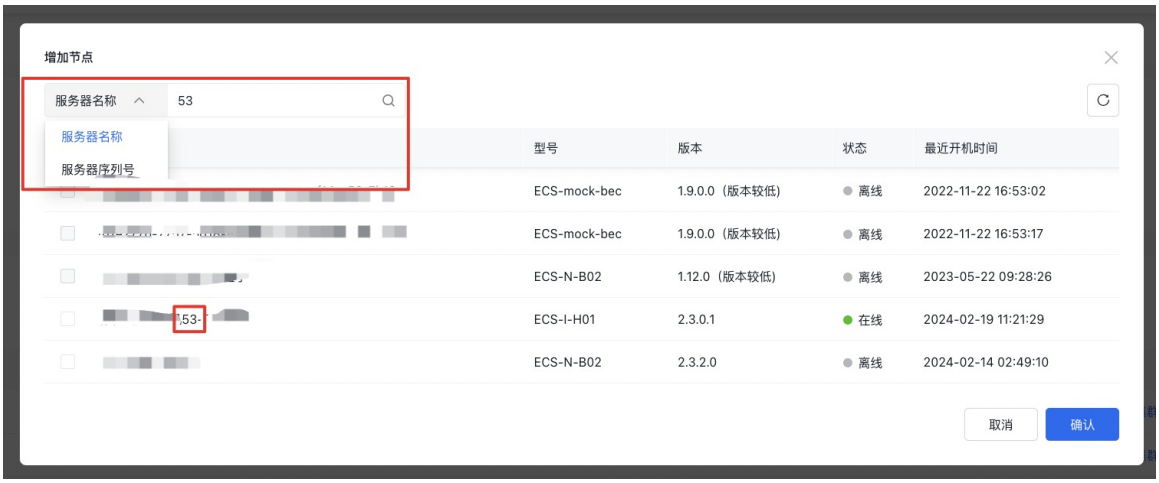
增加节点 **增加节点**操作可以将不属于本集群中的服务器重新添加至当前操作的集群中。

单击**“增加节点”**进入界面，根据个人需求勾选服务器即可，勾选完成后左上角会显示已经勾选了的机器，操作确认无误后点击**“确认”**即可将勾选机器添加至本次操作集群中。



添加完成后在集群节点查看操作即可。

增加节点界面支持**检索功能**，检索功能分为**“服务器名称”，“服务器序列号”**两种检索方式，另外检索功能支持关键词检索。



资源使用监控

平台对该集群服务器的资源使用情况进行实时记录，并根据用户设置的时间条件进行展示。

展示内容包括：服务器在离线状态、内存使用量（使用率）、磁盘使用量（使用率）、CPU使用量（使用率）、带宽使用量

（使用率）等，此专栏还可以根据自身需求选择需要查看的**时间段**和**查看当前集群所有服务器的资源监控或单独服务器。**

扩展服务

Lucky平台

Lucky平台介绍

为了满足用户的多样性需求，百度智能云ECS为用户提供更多的扩展服务。本文将介绍Lukcy平台的基本情况和功能等。

平台简介

边缘智能视频管理平台（简称Lucky平台），通过提供设备统一纳管、信息统一采集、模型统一管理的AI视觉集成化服务，帮助客户进行异构视频源管理、监控信息管理和分立的平台管理需求。

在智慧视频类场景下，用户往往面临着设备数量多、视频监控难、算法调度复杂的难题。Lucky平台为用户提供了设备、视频、算法的一体化管理平台，通过一站式的服务，帮助用户解决了视频类场景下的管理问题。

功能

- **设备管理**
  - 增加设备：支持添加新的摄像头设备，对摄像头完成基本配置后可以纳入平台进行管理；
  - 设备操作：支持对设备进行开启、关闭等操作；
  - 设备信息：支持查看、设置、修改设备的名称、厂商、位置等基本信息。
- **算子管理**
  - 算子配置：支持对设备上所运行的算子进行选择、配置、修改；
  - 识别间隔：支持设置、修改设备上算子每次识别的时间间隔；
  - 视频存证：识别结果的相关视频片段可以自动进行留存。
- **视频管理**
  - 视频预览：支持查看设备原始流、合成流视频的预览；
  - 算法识别区域绘制：支持在视频画面上自由绘制AI算法的识别区域，不同算法可以设置不同的识别区域，并支持修改；
  - 视频列表：根据筛选条件，展示相关的设备视频列表，包括设备名称、原始流画面、合成流画面、全屏查看、位置信息等。
- **监控大屏**
  - 视频展示：监控大屏上可以同时展示多个设备的监控视频；

- 识别结果展示：大屏显示出现的AI算法识别结果；

- **事件中心**
  - 事件触发条件设置：支持用户编辑设置AI事件的触发条件，平台根据用户所设置的条件进行AI事件告警和通知等；
  - 事件的原始信息：用户可以查看触发AI事件时对应的原始视频、对应图片等；
  - 报警查询：支持用户查看一定时间范围内，所出现的AI事件，以及事件的持续时间、发生次数等。

特性

- **一站式视频服务**
  - 服务升级：基于领先的音视频AI技术，无需终端硬件改造投入，完成从简单监测到AI智慧服务的升级；
  - 集成式方案：提供集成式的视频+AI解决方案，实现多品牌监控设备的管理、大屏仪表盘、视频与AI渲染联动。
- **全生命周期管理**
  - 统一视觉设备管理框架：支持多类型、多品牌IP视觉设备统一管理和视觉流信息处理；
  - 视觉设备与模型渲染：支持视觉设备与多种AI模型的灵活绑定，实现AI模型与设备场景化匹配、按需使用；
- **任务动态调度**
  - 抽帧识别：抽帧识别实例将抽出的帧发送给指定的算子列表进行识别，并获取识别结果；
  - 按需合成：拉流解码的过程中，从事件中心订阅指定流的结果，并将结果按需合成到合成流中；
  - 任务调度中心：任务调度器将原来分散的持久化内容统一管理起来，使分流、抽帧识别、合成模块无状态化，进而可以横向扩展。

操作指南

登录平台

前置任务 边缘服务器已经预装了软件服务。且已通电、插入网线，可以正常使用使用。

用一台与此边缘服务器在相同局域网的PC电脑访问此地址：https:// 边缘节点ip地址:81（地址形式参考：https:// ip:81）

获取和修改边缘服务器IP地址

服务器支持有线网卡和无线网卡访问方式：发货的边缘服务器带有默认的有线网卡DHCP连接，需要从路由器界面或其他方式获取DHCP分配的IP。

注：目前只支持chrome浏览器。

进入Lucky平台登录界面：

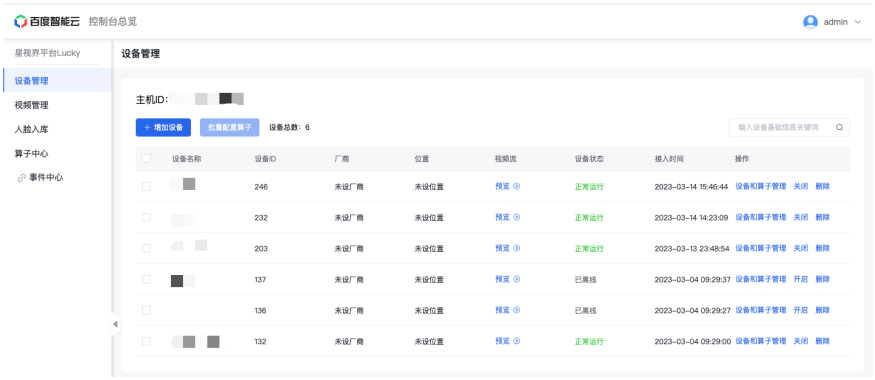
初始登录账号

用户名：admin

密码：admin@ecs1235

（为提高安全性，初次登录后请尽快修改密码）

登录后即可进入Lucky平台主页，可以开始使用。

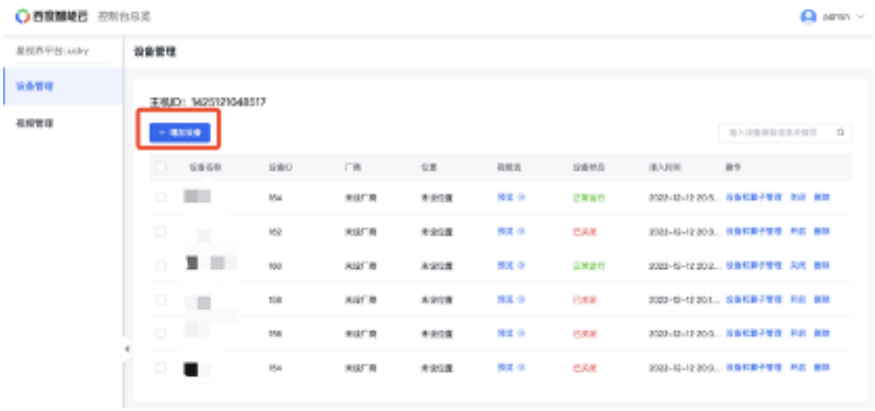


设备与算法管理

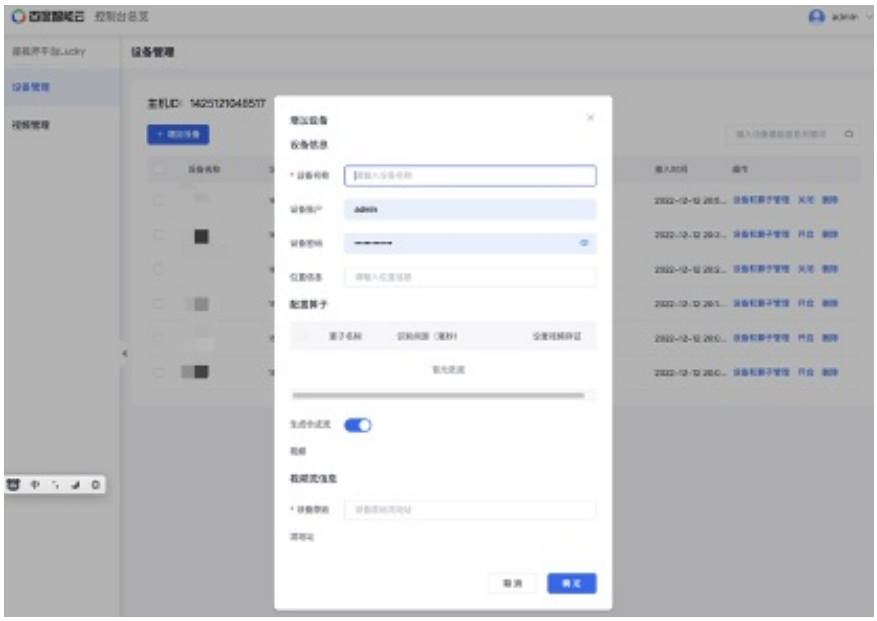
添加摄像头设备

添加摄像头设备的操作步骤：

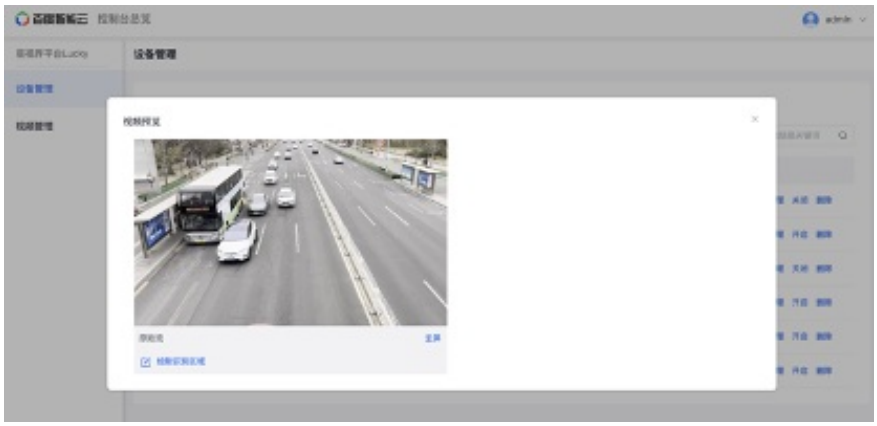
1、点击【增加设备】按钮，输入摄像头设备名称、RTSP流访问的用户名和密码、RTSP视频流地址(RTSP://IP:port/xxxxx格式的地址，海康威视一般地址为RTSP://192.168.1.64:554/)。



2、选择初始绑定的算子（同1个摄像头设备支持绑定多个算子）。



3、添加完成后，点击【预览】，可以看到原始视频，默认不创建带有识别结果的算子合成视频流。如果要同时生成合成流，在增加设备时需要选择【生成合成流】。

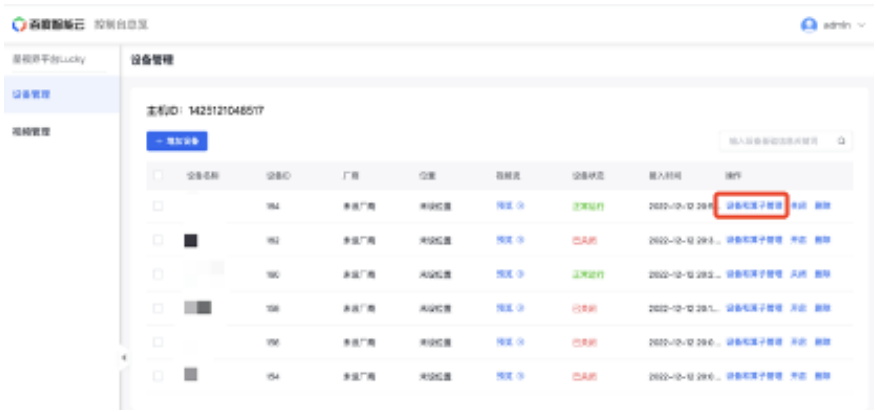


如果要展示合成流视频，点击【查看合成流】按钮即可。

原始流、合成流视频可以同屏查看。

更新绑定AI算子 ##

1、点击【设备和算子管理】，进入【设备详情】，在【配置算子】模块可以绑定、解绑AI算子，同时可以对于算法识别间距、设置视频存证、是否生成合成流，以及设备视频流地址等进行查看、编辑。



2、点击首页【预览】按钮可查看原始流和新算子的合成流。

注：更新绑定新算子时，会自动删除以前的算子。

修改算子识别间隔 ##

默认算子识别间隔是指算子从摄像头视频中的抽帧并识别的频率，默认是1秒1帧，可以按照现场的需求，通过修改识别间隔来改变这一数值。

对于想要操作的设备，点击【设备和算子管理】按钮，在"配置算子"模块，可以对"识别间距"参数进行修改。

可以批量选择若干视频设备一次性修改。如果选中的设备当前没有绑定该算法，那么本次配置会在下次该视频设备绑定该算法时生效。设备绑定的识别任务的识别间隔优先采用修改过的识别间隔，如果未设置过则采用相应算法的默认识别间隔。

配置视频存证 ##

视频存证功能可以在检测到异常行为时，自动保存事件发生时刻前后总计20~30秒左右的视频内容，便于过后查看。

对于想要操作的设备，点击【设备和算子管理】按钮，在"配置算子"模块，可以设置开启视频存证。

绘制识别区域 ##

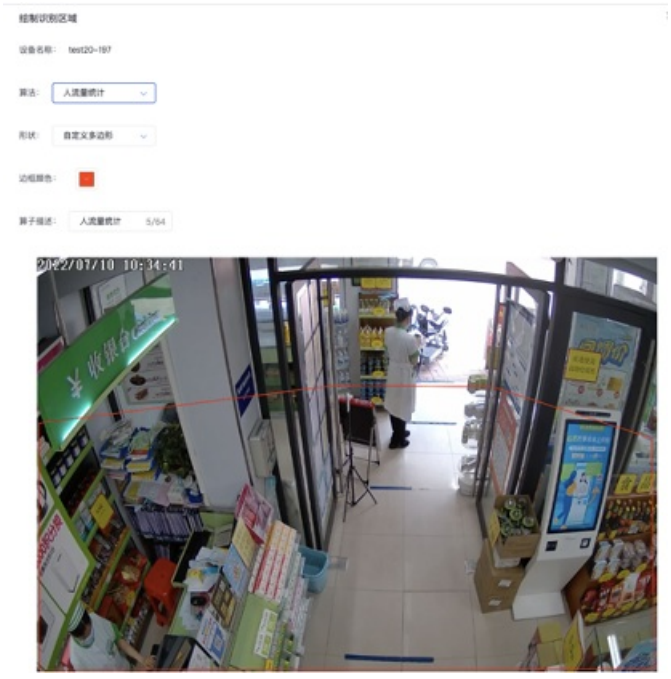
在默认情况下，算法是在整个视频画面进行识别。如果希望只在指定画面的指定区域识别，可以进行多边形画框设置周界区

域。

1、进入指定摄像头设备的预览页面，点击绘制区域；

2、选择算法、边框颜色、区域名称，即可开始绘制识别区域；

注：绘制时依次点击若干点形成多边形的顶点，最后回到起始点进行闭合（最少是3个点）。对于同一设备、同一算法，支持绘制多个区域，并共享同一个区域名称。如果该设备还绑定了其他算法，可以在界面左上角继续设置其他算法、颜色、名称。点击清除可以清空当前算法设置。



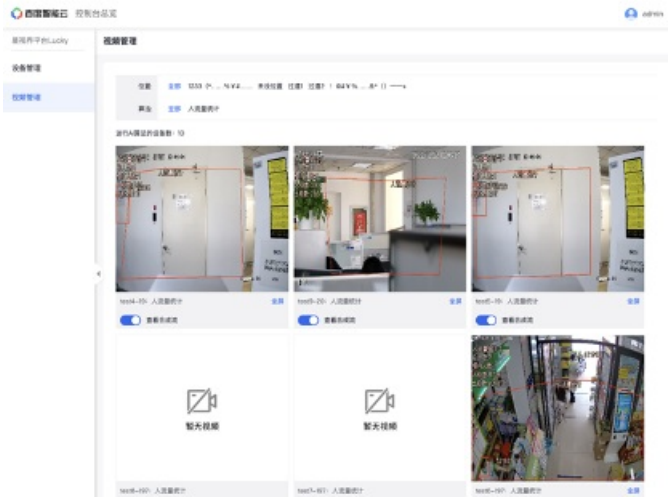
3、绘制完毕后点击确定，之后可以在预览或视频管理界面看到指定区域的识别效果。

识别区域设置完成后，算法识别只在中间多边形区域进行识别，区域外即便有相应目标也不识别。

视频管理 ##

Lucky平台支持基于厂商、位置、智能场景（AI识别类型）筛选合成流视频集中展示识别结果。用户可以在"视频管理"页面，对合成流视频进行查看。

点击左侧边栏【视频管理】按钮，即可进入视频管理功能页面，查看视频合成流。



API参考

概述

样机算子部署列表

序号	中文名字	英文名字	算子说明	备注
1	人脸识别	face-detect	返回人脸识别的基本信息，陌生人等	已部署
2	头部识别	head-detect	监控是否有人员头部	已部署
3	打电话识别	calling-detect	监控是否有人员打电话	已部署

API功能简要说明

接口整体说明

无特殊说明，如下2个HTTP请求头域为必填：

```
Content-Type: application/json  //无特殊说明，都是json类型。
Content-Length: xxxx // 内容长度
```

下文将讲述的接口列表总体分为这几类：

设备管理接口：用于添加视频设备、设置抽帧频率、查询视频设备自身和绑定技能相关的详细信息、以及进行设备的删除。

技能管理接口：用于查询技能列表、将技能与视频设备进行绑定、以及删除绑定关系。

事件管理接口：用于查询视频智能分析的事件列表和订阅异常事件。

人脸库管理接口：1：N识别时，用于向人脸库中添加、删除人脸。

AI识别接口：用于AI识别。

人脸库接口相关解释如下：

人脸库管理要完成1：N识别，首先需要构建一个人脸库，用于存放所有人脸特征，人脸库结构如下：

人脸库、用户、用户下的人脸层级关系如下所示：

```
 |- 用户组（group_id）default
    |- 用户01（uid）
       |- 人脸1（faceid-1）
|- 人脸2（faceid-2）
    |- 用户02（uid）
       |- 人脸（faceid）
  ....
```

关于人脸库的设置限制

①每个人脸库下，有一个默认用户组default；

②用户组（group）default下，可添加多个user_id；

③每个用户（user_id）现阶段支持注册多张人脸。

API列表

序号	API 类别	API 名称	功能说明
1	设备管理接口	添加视频设备	添加视频设备，包括原始流地址，抽帧频率等信息，建议最多添加 4 路视频设备
2		查询视频设备列表	查询已经添加的视频设备信息
3		查询视频设备详情	查询视频设备的视频流的信息、设备配置的抽帧频率、设备绑定的技能信息等
4		删除视频设备	删除接入的视频设备及对应的视频分析流
5	技能管理接口	查询技能列表	查询服务器上已经部署的技能列表名称（建议最多 4 个，样机预先部署）
6		绑定技能新建视频分析流	给指定视频设备绑定指定的技能（单个视频设备最多绑定 4 个技能），获取视频分析流信息
7		删除绑定技能和视频分析流	给指定视频设备删除绑定的技能和创建的视频分析流
8	事件管理接口	订阅异常事件	设定事件的推送地址，可实现异常事件自动推送
9		查询视频分析事件	获取指定设备指定技能的识别事件列表（所有识别结果）
10	人脸库管理接口	新增用户	在用户组中新增用户信息，并保存用户人脸图片
11		人脸查询	获取某个用户基本信息
12		人脸详情	获取用户的人脸图片信息
13		人脸图片修改	修改用户人脸
14		人脸删除	删除用户的人脸图片信息
15		人脸图片下载	下载人脸图片

API功能使用步骤

设备管理、技能管理、时间管理三类API的使用步骤整体上可分为三步：

添加摄像头、绑定AI技能、查询/获取识别事件。

具体参考如下：

1）调用“添加视频设备”接口添加设备，建议等待10s再执行后续操作。

2）后续分场景操作如下（注：（）内的API为按需可选API）：

场景1：绑定技能，获取视频智能分析的异常事件（现阶段异常事件规则提前预置）：（查询视频设备列表）——>（查询技能列表）——>技能绑定到设备新建视频分析流——>订阅异常事件，推送异常事件。

场景2：绑定技能，按需绑定和删除多个技能，并获取异常事件（现阶段异常事件规则提前预置）：（查询视频设备列表）——>(查询技能列表)——>技能1绑定到设备新建视频分析流——>技能2绑定到设备新建视频分析流——>技能3绑定到设备新建视频分析——>技能4绑定到设备新建视频分析流——>订阅异常事件，推送异常事件。

注：在视频设备和技能绑定后，可按需使用查询视频设备详情获取视频设备的视频流的信息、设备配置的抽帧频率、设备绑定的技能信息等。

3）通过“删除绑定技能和视频分析流”删除特定的技能分析流。

4）通过“删除视频设备”删除特定设备。

人脸库管理相关接口使用步骤

新增用户（可添加多张人脸，建议人脸总数量为3，1张正脸，2张侧脸）-->**查询用户人脸列表**-->**删除用户人脸**（可选）

API字段格式说明

所有POST请求都是JSON格式的body。

如果上一个字段是JSON对象，下一个字段出现+xxx这样的参数名称，表示xxx是JSON的key。如果上一个字段是JSON数组，下一个字段是+xxx，表示xxx是JSON数组的每一项的key。如果出现++xxx，表示这个xxx是其前面第一个+yyy字段的子字段。+的个数表明层级，+个数相同表明从json字段深度上看，字段是相同深度的。

API调用方式说明

该章节的每个接口的完整调用地址由IP：端口+接口URL拼接而成。

默认协议为https。视频识别模块接口都是81，且API路径以/console/开头。事件中心默认为81端口，且API路径以/msg/开头。
如果事件中心的访问返回404页面不存在，可以尝试82端口（这可能意味着事件中心与视频管理是独立部署的）。

AI识别接口的端口为7999，协议为http。

API调用时权限认证说明

目前，只有"设备管理和技能管理API接口"需要进行认证，其他接口可直接调用。权限认证需要首先调用用户登录接口，获取登录token（token获取方式参考用户登录API接口），然后在后续调用"设备管理和技能管理API接口"时在请求头中携带该请求头(Authorization)：

```
 例如：
curl -X GET --header "Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY1NTA3MDUwfQ.-RTto3RhQnY2yjafd-uruXvd9jQlzKV62TFA9kseJ7E" -L https://192.168.1.3:81/console/api/v1/ai/query
```

用户登录接口

用户登录和认证API接口

接口描述 该接口用于用户登录或校验用户是否已经登录。用户登录请求的响应消息中会有用于后续身份校验的token，登录后token有效期默认是36000秒，过期后需要重新登录。同一个用户同一时间段内进行多次登录，会获得不同的token，每个token都生效，拥有各自有效期。比如前端admin登录web后，客户端依然可以调用登录接口获得新的token，不会导致web前端登录失效。

请求结构

POST /console/user/auth

请求头域 无特殊请求头。

请求参数

参数	类型	是否必选	描述	示例
userName	String	否	登录用户名。如果是登录请求必填，否则忽略该字段	admin
password	String	否	登录密码。如果是登录请求必填，否则忽略该字段	pwd1
token	String	否	仅当用户登录后，发起后续的其他API请求时需要，登录请求时该字段必须为空	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY1NTA3MDUwfQ.-RTto3RhQnY2yjafd-uruXvd9jQIzKV62TFA9kseJ7E

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败。失败时请忽略data字段。
message	String	成功或错误提示
data	**Map**	**设备信息**
role	String	该用户角色，当前固定为super
userName	String	当前登录用户的用户名
token	String	请求token为空时，会返回该字段作为已登录的凭证。用户在调用需要登录认证的API时需要携带该token作为请求头

请求示例 1、登录请求示例

```
 POST /console/user/auth
Content-Type: application/json
Content-Length: 1254

{
  "userName": "admin",
  "password": "pwd1"
}
```

2、认证请求示例

```
 POST /console/user/auth
Content-Type: application/json
Content-Length: 1254

{
   "token": "xxxx"
}
```

响应示例 1、登录响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status":0,
  "message": "success",
  "data" : {
   "role": "super",
    "userName": "admin",
    "token": "xxxx"
  }
}
```

2、认证响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status":0,
  "message": "success",
  "data" : {
   "role": "super",
    "userName": "admin"
  }
}
```

用户退出登录接口

接口描述 用户登出。

请求结构

POST /console/user/logout

请求头域 无特殊请求头。

请求参数

参数	类型	是否必选	描述	示例
token	String	是	已登录用户收到的认证token，用于标识本次登录会话	eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY1NTA3MDUwfQ.-RTto3RhQnY2yjafd-uruXvd9jQlzKV62TFA9kseJ7E

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**忽略该字段**

请求示例

```
 POST /console/user/logout
Content-Type: application/json
Content-Length: 1254

{
  "token": "xxx"
}
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status": 0,
  "message": "success",
  "data" : null
}
```

设备管理和技能管理接口

查询技能列表

接口描述 本接口用于查询当前支持的全部技能列表。查询到的技能可以用于构建视频分析流。

请求结构

GET /console/api/v1/ai/query

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

无

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**忽略该字段**
value	String	技能英文名称
label	String	技能中文名称
elapse	Int32	默认的算法识别间隔，单位毫秒
param	Array of JSON	算法交互参数配置。数组每一项代表一组特定用途的参数，每组参数可以包括若干相关项（比如逆行区域多边形和逆行方向线可以为一组，不同组表示不同逆行区域，例如左侧车道和右侧车道各为一组）。如果没有配置，则为空。
purpose	String	该组参数的用途, 算法侧会基于该字段解析特定用途的参数值。
text	String	参数组名称，有默认值，调用者也可以修改。
repeatable	Boolean	是否允许出现多个purpose相同的组。比如多个逆行区配置，多个周界区域等。
items	Array of JSON	每组内的相关项配置集合
+purpose	String	该参数项用途。算法侧会基于该字段进行特定用途的解析。
+text	String	该参数项调用者配置的文本描述。调用者可以动态修改该值。
+type	String	该参数的前端组件类型。目前可用的包括： polygon(多边形), arrow(有向线段)，segment(线段），integer(整数), float(浮点数)。
+attr	JSON	参数项属性(比如取值范围，约束条件等，便于前端限制），目前暂未用到

请求示例

```
GET /console/api/v1/ai/query
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930k
```

响应示例

```
HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
    "status": 0,
    "message": "success",
    "data": [
        {
            "value": "face",
            "label": "人脸识别",
            "param": [
                {
                    "purpose": "filterRegion",
                    "text": "过滤区域",
                    "repeatable": true,
                    "items": [
                        {
                            "purpose": "filterPolygon",
                            "text": "filter_corridor",
                            "type": "polygon"
                        }
                    ]
                }
            ]
        }
    ]
}
```

添加视频设备

接口描述 添加新的RTSP流设备。

请求结构

POST /console/api/v1/device/add

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
username	String	是	设备RTSP用户名，如果设备RTSP不需要用户名和密码认证，可以随便写一个，可以为空，长度<=30个中英文字符	testme
password	String	是	设备RTSP密码，如果设备RTSP不需要用户名和密码认证，可以随便写一个，可以为空，长度<=30个中英文字符	pwd1
deviceName	String	是	设备名称，不允许为空，长度<=32个中英文字符	device001
deviceRtsp	String	是	设备RTSP流地址，长度<=256个中英文字符	rtsp://192.168.1.64:554/
aiList	Two-dimensional Array	否	要绑定的AI技能列表。二维数组的每一个元素必须是一个长度为1的一维数组。每个一维数组是一个字符串，表示一个AI技能	比如添加人脸识别和安全帽识别： [["face"], ["hat"]]
position	String	否	位置描述，长度<=30个中英文字符	大门口
manufacturer	String	否	厂商名称，默认为"未设厂商"，长度<=96个中英文字符	海康威视
noRealMerge	boolean	否	是否合成AI视频分析流，仅当aiList不为空时有效，默认为false。如果为true则仅用于AI识别，不提供AI视频分析流供效果展示	false

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	int32	接口返回状态，0表示成功，其他表示失败
message	字符串	成功或错误提示
data	**JSON**	**设备信息**
deviceId	int64	设备ID
dispatchAddr	字符串	生成的设备原始RTSP分发流地址

请求示例

```
 POST /console/api/v1/device/add HTTP/1.1
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930


{
  "username": "testme",
  "password": "pwd1",
  "deviceName": "device001",
  "deviceRtsp": "rtsp://192.168.1.64:554/",
  "aiList": [["face"], ["hat"]],
  "noRealMerge": false
}
```

响应示例

```
 {
  "status":0,
  "message": “success”,
  "data" : {
   "deviceId": 101,
   "dispatchAddr": “rtsp://192.168.1.64:554/dispatch/101”
 }
}
```

查询视频设备列表

接口描述 查询RTSP流设备列表，支持关键字过滤。

请求结构

POST /console/api/v1/device/list

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
status	Array	是	状态过滤的字符串数组。设备状态包括：正常运行、已离线、已关闭。	["正常运行", "已离线"]
filter	String	否	关键字过滤，如果为空则忽略关键字匹配。如果设备的名称、RTSP地址、厂商、位置中的任何一个字段包含filter关键字，则满足匹配。 status和filter二者是AND的关系	192.168.1.64
pageSize	Int32	是	分页查询的每页记录数	20
pageNum	Int32	是	分页查询的页号，最小为1	5
total	Boolean	否	是否返回符合条件的记录总数	true

响应头域 无特殊响应头。

响应参数

注意：返回结果首先按照createAt从大到小排序，createAt相同时按照deviceName字典序从小到大排序。

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**设备信息**
list	Array	设备信息数组
+id	Int64	视频设备的唯一标识ID
+deviceName	String	设备名称
+position	String	设备位置
+manufacturer	String	设备厂商
+createAt	String	设备创建时间，日期字符串，精确到秒
+status	String	设备状态。例如： “正常运行”
pages	Map	页数信息
+total	Int32	符合条件的记录总数，仅当请求的total为true是有效

请求示例

```
POST /console/api/v1/device/list
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930

{
"status": ["正常运行", "离线"],
"filter": "192.168.3.4",
"pageSize": 20,
"pageNum": 3,
"total": true
}
```

响应示例

```
{
  "status": 0,
  "message": "success",
  "data": {
    "list": [
      {
        "id": 70,
        "status": "正常运行",
        "deviceName": "22",
        "createAt": "2022-04-29 17:40:38",
        "position": "大门口",
        "manufacturer": "海康威视"
      },
      {
        "id": 62,
        "status": "已禁用",
        "deviceName": "18",
        "createAt": "2022-04-29 17:36:06",
        "position": "保安亭",
        "manufacturer": "大华"
      },
      {
        "id": 24,
        "status": "离线",
        "deviceName": "a",
        "createAt": "2022-04-29 10:29:34",
        "position": "超市门口",
        "manufacturer": "海康威视"
      }
    ],
    "pages": {
      "total": 3
    }
  }
}
```

绑定技能新建视频分析流

接口描述 绑定AI技能。仅当添加RTSP视频流设备时没有绑定AI技能，才可以通过该接口绑定AI技能。一旦视频流设备已经绑定AI技能，不能再调用该接口。

请求结构

POST /console/api/v1/device/bind_ai

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
deviceIdList	Array	是	视频设备ID数组	[34, 78]
aiList	Two-dimensional Array	是	AI英文技能列表，不允许为空。每个元素是只包含AI技能名称的长度为1的一维数组。	[[“face2”],[“smoke-detect”], [“fire-detect”]]
noRealMerge	Boolean	否	默认为false。 如果为true表示只绑定AI技能，不合成AI视频分析流	false

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**Array**	**成功时表示绑定的AI视频分析流信息**
+deviceId	Int64	视频流设备ID
+mergeId	Int64	AI视频分析流ID
+url	String	AI视频分析流RTSP地址

请求示例

```
 POST /console/api/v1/device/bind_ai
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930


{
    "deviceId": [123, 100],
    "aiList":[["face2"],["smoke-detect"], ["fire-detect"] ]
}
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
   "status":0,
   "message": "success",
   "data" : [
     {
        "deviceId": 100,
       "mergeId": 1000,
      "url": "rtsp://192.168.1.64:554/merge/132323423"
      },
      {
        "deviceId": 120,
       "mergeId": 1300,
      "url": "rtsp://192.168.1.64:554/merge/132324523"
      }
   ]
}
```

批量修改设备绑定的识别任务的识别间隔

接口描述 批量地设置设备绑定的识别任务的识别间隔。

请求结构

PUT /console/api/v1/device/ai/elapse

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
deviceId	Array	是	批量修改的视频设备ID	[23,27]
ai	Map	是	算法对应的识别间隔. key是算法英文唯一标识，value是毫秒级识别间隔	{“hat”: 1100, “face”: 1500}

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**忽略该字段**

请求示例

```
PUT /console/api/v1/device/ai/elapse
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930


{
  "deviceId": [100, 123],
  "ai": {
    "hat": 1120,
    "face": 1500
  }
}
```

响应示例

```
HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status": 0,
  "message": "success",
  "data": null
}
```

设置周界识别区域

接口描述 设置周界识别区域。设置后仅对区域内的目标进行识别。该接口会删除设备上当前全部AI的区域设置，然后用本次请求重新设置。

请求结构

PUT /console/api/v1/device/region

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述
deviceId	Int64	是	添加设备时返回的唯一标识ID，例如：23
regions	Map	是	区域绘制设置。key是算法英文名称，value是该算法绘制设置。如果regions为空，比如"regions":{}表示清空全部周界区域。
+ai	String	是	固定为与key相同，表示算法英文名称
+text	String	是	周界区域名称
+bcolor	String	是	RGB颜色的十六进制形式，例如： #FF0099
+polygons	Array	是	多边形顶点的顺序表示。支持绘制多个多边形，数组每一项描述一个多边形区域。
++polygon	Array	是	多边形顶点描述的整数数组。例如[3, 2, 5, 6, 4, 9]表示多边形由(x1=3, y1=2), (x2=5, y2=6), (x3=4, y3=9)这3个点构成

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**忽略该字段**

请求示例

```
PUT /console/api/v1/device/region HTTP/1.1

{
  "deviceId": 18,
  "regions": {
    "jiandao": {
      "ai": "jiandao",
      "text": "剪刀区域",
      "bcolor": "#ff0099",
      "polygons": [
        {
          "polygon": [55, 66, 655, 666, 150, 550]
        },
        {
          "polygon": [155, 166, 1655, 1666, 1150, 1550]
        }
      ]
    }
  }
}
```

响应示例

```
{
 "status":0,
 "message": "success",
 "data": null
}
```

查询周界识别区域设置

接口描述 查询已设置的周界识别区域信息。

请求结构

GET /console/api/v1/device/region?deviceId=35

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

GET的路径参数。

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**Map**	**忽略该字段**
regions	Array	每个周界区域设置信息
+ai	String	算法英文名称
+aiComment	String	算法中文名称
+text	String	周界区域名称
+bcolor	String	十六进制RGB的周界边框颜色
+polygons	Array	对象数组，该算法的多个多边形顶点信息
++polygon	Array	浮点数数组，单个多边形顶点的表示。参考周界设置中该字段的解释

请求示例

```
GET /console/api/v1/device/region?deviceId=35
Content-Type: application/json
Content-Length: 0
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930k
```

响应示例

```
HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status": 0,
  "message": "success",
  "data": {
    "regions": [
      {
        "ai": "jiandao",
        "aiComment": "剪刀",
        "bcolor": "#ff0099",
        "text": "区域设置1",
        "polygons": [
          {
            "polygon": [ 1,2, 100, 200, 50, 150 ]
          }
        ]
      }
    ]
  }
}
```

更新视频设备绑定的AI技能

接口描述 本接口用于更新或者清空视频流设备上绑定的AI技能。

请求结构

POST /console/api/v1/device/update_ai

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
deviceId	Int64	是	视频设备ID	23
mergeList	Array	是	该视频设备下待更新AI分析视频信息， 目前数组长度只能固定为1。如果aiList长度为0，则清空绑定的AI技能及其对应的AI视频分析流	{"mergeId": 3, "aiList": [["face"], ["hat"]]}

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**忽略该字段**

请求示例

```
 POST /console/api/v1/device/update_ai
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930


{
  "deviceId":10,
  "mergeList": [ {"mergeId": 34, "aiList": [["face"], ["fire"]]
}
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status": 0,
  "message": "success",
  "data": null
}
```

查询视频设备详情

接口描述 查询视频设备详情。

请求结构

POST /console/api/v1/device/query_device_detail

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
deviceId	Int64	是	添加设备时返回的唯一标识ID	23

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**Map**	**设备信息**
id	Int64	设备ID
deviceName	String	设备名称
status	String	设备状态
username	String	设备RTSP访问用户名
password	String	设备RTSP访问密码
dispatchAddr	String	设备原始RTSP视频流地址，该RTSP地址可以直接访问，不需要密码
deviceRtsp	String	添加设备时输入的设备RTSP流地址
position	String	设备位置
manufacturer	String	设备厂商
createAt	String	创建时间
aiElapse	Map	key是AI技能英文唯一标识，value是毫秒级整数的识别任务识别间隔设置
videoRecord	Map	key是AI技能英文唯一标识，value是布尔类型，表示是否开启视频存证
mergeList	Array	经过视频分析的合成流数组。目前数组最多只有1项。
+mergeId	Int64	合成流数据库记录主键ID
+url	String	合成流RTSP地址
+urlRtmp	String	合成流RTMP地址
+urlFlv	String	flv播放地址的http path部分。 比如/merge/32.flv
+aiList	Array	该合成流包含的所有技能对象构成的数组
++value	String	技能英文名称
++label	String	技能中文名称
aiParam	Array of JSON	算法任务的参数配置信息。
+name	String	算法英文名称
+param	JSON	算法任务参数配置
++purpose	String	参数组用途
++display	String	参数组前端展示名称
++items	Array of JSON	参数组包含的具体配置项
+++purpose	String	参数项用途
+++display	String	参数项前端展示名称
+++value	Any	参数项的值。不同前端组件的值类型不同。比如integer类型的项，值为整数;polygon类型的项，值为浮点数数组。

请求示例

```
 POST /console/api/v1/device/query_device_detail
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930


{
  "deviceId":100
}
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status":0,
  "message": "success",
  "data"{
   "id":100,
 "deviceName": "device001",
  "username": "testme",
  "password": "pwd1",
 "dispatchAddr": "rtsp://192.168.1.64:554/dispatch_1",
 "deviceRtsp": "rtsp://192.168.1.64:554/channels/101",
 "createAt": "2021-02-20 11:05:06",
    "position": "大门口",
    "manufacturer": "海康威视",
    "aiElapse": {
        "face2": 1500
    },
    "videoRecord": {
        "face2": true
    },
 "mergeList": [
  {
  "mergeId":1000,
  "url": "rtsp://192.168.1.64:8194/merge/1345",
        "urlRtmp": "rtmp://192.168.1.64:8195/merge/1345",
        "urlFlv": "/merge/1345.flv",
  "aiList": [
   {
   "value": "face2",
   "label": "人脸检测"
  }
    ],
    "aiParam": [ // 返回已经设置的算法参数配置,如果某个算法没有设置，会默认设置一个空的aiParam.param，以便前端方便处理。
      {
        "name": "wash-hand-detect",
        "param": [
```

```
        {
          "purpose": "filterRegion",
          "display": "客户自己设置的描述1",
          "items": [
            {
              "purpose": "filterPolygon",
              "display": "客户自己设置的描述1.1",
              "value": {
                "polygon": [
                  0.13,
                  0.14,
                  0.85,
                  0.17,
                  0.89,
                  0.802
                ],
                "text": "区域1"
              }
            }
          ]
        },
        {
          "purpose": "confidence",
          "display": "客户自己设置的描述2",
          "items": [
            {
              "purpose": "confidence",
              "display": "客户自己设置的描述2.1",
              "value": 85
            }
          ]
        }
      ]
    }
  ]
 }
}
```


删除视频设备

接口描述 删除视频设备。

请求结构

POST /console/api/v1/device/permanentDisable

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
deviceId	Int64	是	视频设备的ID的唯一标识	23

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**忽略该字段**

请求示例

```
 POST /console/api/v1/device/permanentDisable
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930


{
  "deviceId": 100
}
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status":0,
  "message": "success",
  "data": null
}
```

设置AI识别任务的运行时间段

接口描述 为视频流绑定的AI识别任务，设置运行时间段，确保AI识别任务只在匹配的时间段运行，非运行时间段的AI识别任务会暂停。如果一个视频流上的所有AI识别任务都暂停，则对应的合成流也会暂停,当符合运行时间段时，AI识别任务和对应合成流（如果已经暂停）也会恢复运行。

请求结构

PUT /console/api/v1/device/set_cycle_detect_job

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
deviceId	Array of Int64	是	批量操作的设备ID列表	[3, 7]
aiRunTime	Json	是	AI识别任务的运行时间段配置	{"aiRunTime": { "aiCycleRunTime": {"face": [{"month": [1,2,12], "weekday": [0, 1, 6], "day": [1,2,31], "hour": [8,9,10,11,14,15,16,17,18], "hourMinuteRange": [30,31,...59] }] } } }
+aiCycleTime	Json	是	不同AI的运行时间段配置。key是AI算法英文唯一名称，value是一组时间段配置。如果key不包含某个算法，则表示该算法对应的识别任务一直运行。value数组中只要有一项匹配某个时间段（不必数组中每一项都匹配），那么在对应时间段上该AI识别任务就会运行。	
++month	Array of Int32	否	匹配运行的月份，可选值为1~12。如果为空或不设置，则表示不匹配任何月份	[1,2, 12]
++day	Array of Int32	否	匹配运行日期的数组，可选值为1~31。如果为空或不设置，则表示不匹配任何日期。	[3, 9, 31]
++weekday	Array of Int32	否	匹配运行星期的数组，可选值为0~6。0表示星期日。如果为空或不设置，则表示不匹配任何星期。	[0, 1, 5, 6]
++hour	Array of Int32	否	匹配运行小时的数组，可选值为0~23。如果为空或不设置，则表示不匹配任何小时。	[0, 1, 13, 23]
++hourMinuteRange	Json	否	指定起始分钟的运行时间段配置。如果未设置（为空属于已设置)则忽略该字段。如果该字段已设置，则忽略hour字段。	9:30 ~ 18:30
+++startHour	Int32	否	开始运行的小时。默认为0	23
+++startMinute	Int32	否	开始运行的分钟。默认为0	59
+++endHour	Int32	否	结束运行的小时。 默认为0	14
+++endMinute	Int32	否	结束运行的分钟。默认为0	29

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	int32	接口返回状态, 0表示成功，其他表示失败
message	字符串	成功或错误提示
data	**JSON**	**占位符，目前固定为null**

请求示例

```
 POST /console/api/v1/device/add HTTP/1.1
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930


{
  "deviceId": [1, 3],
  "aiRunTime": {
    "aiCycleRunTime": {
      "face": [
        {
          "month": [1,2,...,12],
          "weekday": [0, 6],
          "day": [1,2,3,...,31],
          "hour": [8, 9,10,11]
        },
        {
          "month": [1,2],
          "weekday": [1,2,...,5],
          "day": [1,2,3,...,31],
          "hourMinuteRange": {
            "startHour": 8,
            "startMinute": 30,
            "endHour": 18,
            "endMinute": 30
          }
        }
      ],
      "fire": [
        {
          "month": [1,2,...,12],
          "weekday": [0,1,..., 6],
          "day": [28，29， 30， 31],
          "hour": [8,9,...,17]
        }
      ]
    }
  }
}
```

如上示例，配置了设备1和3的AI任务运行时间段。表示人脸算法运行时间段为：1~12月每个周末的8:00~11:59分 和 1,2月每个工作日的8:30~18:30；火焰识别算法运行时间段为: 每个月的28~31号的8:00~17:59分。该设备的其他算法为全时间段运行。 **响应示例**

```
{
 "status": 0,
 "message": "success",
 "data" : null
}
```

配置视频设备算法参数

接口描述 为设备上的算法任务添加参数，比如周界过滤区域。

请求结构

POST /console/api/v1/device/ai/param

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	Authorization: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY3NDc4OTc4fQ.JL9R7CqIhPFdQWZry4e9ONZntUTmNzBp0sDE05rMxI

请求参数

参数	类型	是否必选	描述	示例
deviceId	Int64	是	设备ID	142
aiList	Array of JSON	是	全部算法任务的参数配置信息。数组每一项是一个算法的配置。这里不支持部分修改，每次需要指明全部任务的配置，不指定则默认为空配置。	
+name	String	是	算法英文名称	face
+param	Array of JSON	否	算法参数组配置，默认为空。	
++purpose	String	是	参数组的用途	filterRegion
++display	String	是	参数组前端展示名称	过滤区域
++items	Array of JSON	是	参数组内每个参数项配置	
+++purpose	String	是	参数项的用途	filter_corridor
+++display	String	是	参数项的前端展示名称。比如多边形类型，则前端视频画面会在多边形框旁边展示该名称。	过滤区域多边形
+++value	Any	是	参数项的值。不同组件的值类型不同，比如integer类型只是一个整数，polygon类型是一个浮点数数组。	比如多边形的顶点信息：`[0,0，0.6,0,0.6,0.6，0,0.6]`

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	int32	接口返回状态, 0表示成功，其他表示失败
message	字符串	成功或错误提示
data	**JSON**	**设备信息**
deviceId	int64	设备ID
dispatchAddr	字符串	生成的设备原始RTSP分发流地址

请求示例

```
 POST /console/api/v1/device/add HTTP/1.1
Content-Type: application/json
Content-Length: 1254
Authorization:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyTmFtZSI6ImFkbWluIiwiZXhwIjoxNjY4NjgxODg5fQ.NkjiNOeMnzP9To-CuV4F930


{
  "deviceId": 74,
  "aiList": [
    {
      "name": "face",
      "param": [
        { // 第一组参数
          "purpose": "filterRegion",
          "display": "过滤区域1",
          "items":
            [
              {
                "purpose": "filterPolygon",
                "display": "多边形1",
                "value": {
                  "text": "区域1",
                  "polygon": [0.33,0.44, 0.55,0.77, 0.99,0.102]
                }
              }
            ]
        },
        { // 第二组参数
          "purpose": "carReverseRegion",
          "display": "车辆逆行区域2",
          "items":
            [
              { // 组内第一项参数
                "purpose": "carReversePolygon",
                "display": "车辆逆行多边形3",
                "value": {
                  "text": "区域2",
                  "polygon": [0.133,0.44, 0.155,0.77, 0.199,0.102]
                }
              },
              { // 组内第二项参数
                "purpose": " "carReverseLine",
```

```
                    "display": "车辆逆行方向3",
                    "value": {
                        "line": [0.133,0.22, 0.144,0.66] ,
                        "text": "线名称xxx"
                    }
                }
            ]
        }
    },
    { // 第3组参数
        "purpose": "confidence",
        "display": "置信度1",
        "items": [
            { // 组内唯一项参数
                "purpose": "confidence",
                "display": "置信度配置1",
                "value":  0.85
            }
        ]
    }
  ]
 }
 ...
}
```

响应示例

```
{
 "status":0,
 "message": “success”,
 "data" : null
}
```

批量更新不同设备的绑定算法任务

接口描述 能够批量更新不同设备的绑定算法任务

请求结构

POST /console/api/v1/device/batch/update

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	如果不想走登录认证获取token，也可以联系值班人员获取Authorization内容，内容格式为字母编码：xxxxx

请求参数

参数	类型	是否必选	描述	示例
aiList	Array	是	绑定算法列表，如果为空则相当于清空识别任务，只保留RTSP原始流。	["ai1", "ai2"]
noRealMerge	Boolean	否	是否开启合成流。true表示不开启合成流，false表示开启合成流	true

aiElapse	Map	是	算法识别间隔设置，单位毫秒,key是算法英文名称，value是识别间隔.	{ "ai1": 1300, "ai2": 1000}
videoRecord	Map	否	key是配置任务运行时间段的算法英文名称，value是Boolean，表示是否开启视频存证 ，开启视频存证,如果需要边缘保留视频，必须对指定算法设置为true，否则事件中心前端展示匹配异常的原始消息时，将不展示视频。	{"ai1": true, "ai2": true}
aiRunTime	Map	否	指定绑定的算法任务将在哪些时间段运行。	
+aiCycleRunTime	Map	否	时间段配置。key是配置任务运行时间段的算法英文名称，value是该算法可以运行的若干时间段。如果key不包含某个算法，则该算法全天运行。不在算法检测时间段时，算法任务将会被暂时关闭（设备算法任务列表依然会返回该算法任务），待满足运行时间段后再恢复。	
++day	Array	否	取值范围为0,1,...31 所有的数组value,如果为空数组，则当前项会失配。如果算法下全部失配，则该算法任务全时段关闭。	20
++weekday	Array	否	哪几个星期可以运行。这里表示工作日。可取值：0,1,...,5,6 代表周日、周一、周二。。。,周五、周六。	[1,2,3,4,5]
++month	Array	否	哪几个月份运行,取值范围为[1,2,...,12]分别代表1~12月份，示例取值为12月	[12]
++hourMinuteRange	Map	是	一天的哪几个时间段运行,取值范围为[00:00~23:59]分,示例取值为12月08:00:00~12:59:59秒	[08:00:00~12:59:59]
+++startHour	Int32	是	指定开始运行时间的小时部分。可取值为: 0, 1, ..., 23，分别表示0点，1点，..., 23点	8
+++endHour	Int32	是	指定结束运行时间的小时部分。可取值为: 0, 1, ..., 23，分别表示0点，1点，..., 23点	12
+++startMinute	Int32	是	指定开始运行时间的分钟部分。可取值为: 0, 1, ..., 59，分别表示0分钟，1分钟，..., 59分钟	0
+++endMinute	Int32	是	指定结束运行时间的分钟部分。可取值为: 0, 1, ..., 59，分别表示0分钟，1分钟，..., 59分钟	59
devices	Array	是	指明哪些设备需要更新.	
+deviceId	Int64	是	rtsp视频流设备ID，可以理解为摄像头的ID	13
+name	String	是	设备名称	设备01
+position	String	否	表示地址信息,填写摄像头设备位置,可以为空，为空则默认设置为"未设位置"	位置1
+manufacturer	String	否	表示厂商的详细信息,可以填写摄像头厂商的详细信息，可以为空，为空则默认设置为"未设厂商"	厂商A

响应头域 无特殊响应头。

响应参数

注意：返回结果首先按照createAt从大到小排序，createAt相同时按照deviceName字典序从小到大排序。

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	nil	数据返回设置字段，本接口固定为nil。

请求示例


```
{
"aiList": ["ai1", "ai2"],
"noRealMerge": true,
"aiElapse": {
 "ai1": 1300,
 "ai2": 1000
},
"videoRecord": {
 "ai1": true,
 "ai2": true
},
"aiRunTime": {
 "aiCycleRunTime": {
  "ai1": [{
   "day": [1, 2],
   "weekday": [1, 2, 3, 4, 5],
   "month": [1, 2, 12],
   "hourMinuteRange": {
    "startHour": 8,
    "endHour": 17,
    "startMinute": 30,
    "endMinute": 45
   }
  },
  {
   "day": [28, 29, 30, 31],
   "weekday": [6, 0],
   "month": [12],
   "hourMinuteRange": {
    "startHour": 8,
    "endHour": 12,
    "startMinute": 0,
    "endMinute": 59
   }
  }
 ],
  "ai2": [{
   "day": [1, 2, 4, 28, 31],
   "weekday": [0, 1, 2, 6],
   "month": [1, 2, 12],
   "hourMinuteRange": {
    "startHour": 8,
    "endHour": 17,
    "startMinute": 30,
    "endMinute": 45
   }
  }]
 }
},
"devices": [{
```

```
    "deviceId": 13,
    "name": "设备01",
    "position": "位置1",
    "manufacturer": "厂商A"
   },
   {
    "deviceId": 15,
    "name": "设备02",
    "position": "位置2",
    "manufacturer": "厂商B"
   }
  ]
}
```

响应示例

```
{
  "status": 0,
  "message": "success",
  "data": nil
}
```

注册摄像头信息变化通知接口

接口描述 添加接收摄像头相关信息的URL地址。

说明：允许重复注册，比如调用者每次重启时调用一次，这样就可以重新接受数据推送。目前只支持http://格式的url，尚不支持https。 在以下情况会主动推送：

1、接收方调用该接口注册后，会主动推送。

2、如果某次推送失败，会在30秒后重试。

3、后端进程重启时会主动推送一次。

4、摄像头信息变更时（比如摄像头离线、更新摄像头绑定算法任务等）会主动推送。

请求结构 POST /console/api/v1/set-notify-url

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	如果不想走登录认证获取token，也可以联系值班人员获取Authorization内容，内容格式为字母编码：xxxxx

请求参数

参数	类型	是否必选	描述	示例
url	String	是	接收摄像头设备配置相关数据的http://ip:port/xxx形式的URL地址。未来可能支持https形式。	url是类似http://a.b.c.d:4423/recv这种http接收地址
type	String	是	接受何种信息的变更通知。当前可选值只有"device"，包括设备名称、流地址、绑定算法信息	type目前固定为device

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	success， 或者失败原因
data	JSON	暂无数据，可以忽略该字段

响应错误码参数

错误码	含义
10000	请求参数无效；比如非JSON请求，或者url无效等。
10004	更新数据库失败。

请求示例

```
{
  "type": "device",
  "url": "http://a.b.c.d:1234/recv/test"
}
```

响应示例

```
{
  "status": 0,
  "message": "success",
  "data": null
}
```

查询注册列表

接口描述

查询已经注册的接收摄像头信息的URL列表。

请求结构

GET /console/api/v1/list-notify-url

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	如果不想走登录认证获取token，也可以联系值班人员获取Authorization内容，内容格式为字母编码：xxxxx

请求参数

无请求参数。

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	success， 或者失败原因
data	JSON	注册URL信息列表
+url	String	注册时的URL
+type	String	接受何种信息的通知

请求示例

```
无请求示例
```

响应示例

```
{
  "status": 0,
  "message": "success",
  "data": [
    {
      "url": "http://a.b.c.d:454/xxx",
      "type": "device"
    }
  ]
}
```

删除摄像头信息变化通知接口

接口描述 删除已注册的接收摄像头相关信息的URL地址。

请求结构

POST /console/api/v1/delete-notify-url

请求头域

请求头名称	描述	示例
Authorization	认证头。登录响应返回的token内容，如果不填或者填错会拒绝请求	如果不想走登录认证获取token，也可以联系值班人员获取Authorization内容，内容格式为字母编码：xxxxx

请求参数 说明：允许重复调用，删除不存在的数据不报错。

参数	类型	是否必选	描述	示例
url	String	是	接收摄像头设备配置相关数据的http://ip:port/xxx形式的URL地址。未来可能支持https形式。	url是类似http://a.b.c.d:4423/recv这种http接收地址
type	String	是	接受何种信息的变更通知。当前可选值只有"device"，包括设备名称、流地址、绑定算法信息	type目前固定为device

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态，0表示成功，其他表示失败
message	String	success， 或者失败原因
data	JSON	暂无数据，可以忽略该字段

响应错误码参数

错误码	含义
10000	请求参数无效；比如非JSON请求，或者url无效等。
10004	更新数据库失败。

请求示例

```
{
  "type": "device",
  "url": "http://a.b.c.d:1234/recv/test"
}
```

响应示例

```
{
  "status": 0,
  "message": "success",
  "data": null
}
```

事件管理接口

新增事件异常检测配置

接口描述 添加原始消息的异常检测规则。默认情况，与异常检测规则匹配的原始消息，会保存其对应的图片和存证视频。原始消息、匹配异常检测配置的消息对应的图片和视频，默认保存1小时，过期会清理，也可以通过部署事件中心时修改环境变量，调整保存时长（单位小时）。

请求结构

POST /msg/config/add

请求头域

无特殊请求头。

请求参数

参数	类型	是否必选	描述	示例
name	String	是	异常检测配置的名称，要求全局唯一。长度1-64位的英文字母、数字或下划线。	ab_3sd
desc	String	否	异常检测配置的自定义描述。最长输入128个汉字或字符。	abcd配置xxx
topic	String	是	消息大类，通常官网的AI识别结果，都是使用lucky-ai	lucky-ai
schema	String	是	topic下的具体消息类型。当前官网的AI识别结果，对应的schema就是算法英文唯一名称	face
exceptionDetect	Map	是	异常检测规则配置信息	
+rawJson	String	是	原始AI识别消息格式说明。通常为json字符串形式。具体需要参考AI识别结果的文档	{"objects":[{"text": "未戴安全帽", "w": 100, "h": 100, "x": 33, "y": 44}, {"text": "未戴安全帽", "w": 123, "h": 134, "x": 300, "y": 440}], "objects_count ":2}
+logicAnd	Boolean	否	多条规则之间的逻辑关系，如果仅有一条则可以忽略该项。true: 多条规则必须都匹配才算匹配；false: 多条规则有一条匹配即可	true
+algs	Array	是	多条规则的详细配置信息。每条规则是一个Map结构	
++algName	String	是	规则类型。可选值：arithmetic和regexp， 分别表示算术表达式和正则表达式	arithmetic
++param	Map	是	规则的具体设置，不同algName的要求的字段不同	
+++expr	String	是	检测表达式的内容。对于arithmetic，是由原始消息中部分字段的别名构成的四则运算。对于regexp，是原始消息中指定字段的正则匹配表达式。算术表达式内部必须以空格为间隔符，参考请求示例。	
+++alias	String	是	对于regexp类型，该字段必填(对于arithmetic则无须设置该字段），指明expr是哪个别名字段对应的正则匹配。正则表达式规则可以参考：https://github.com/google/re2/wiki/Syntax	a
alarmDetect	Map	是	将相同异常检测配置匹配的多条连续原始消息聚合为单条异常的配置。	
+mergeInterval	int32	是	聚合原始消息的起止时间间隔，单位秒	30
+isPostMerge	int32	否	是否为后聚合。true: 后聚合， false: 预聚合。	30
+errorTimes	int32	是	如果为后聚合，该字段必填，否则该字段被忽略。表示mergeInterval秒内发生errorTimes次异常匹配时，开始生成后聚合的记录信息。	5
+receivers	Array	是	目前需要固定设置为[{}]，暂无特殊用途	[{}]

> 说明
>
> 聚合报警是指对指定时间窗口内的来着相同设备的多条异常消息进行聚合。新的异常消息距离当前报警记录首条异常超过阈值时，则该消息会生成新的聚合报警记录。 预聚合是指：时间窗口内的异常消息无论多少条都聚合为同一个报警记录，并且该窗口内首条异常消息即生成报警记录，后续消息会更新该报记录的结束时间。 后聚合是指：时间窗口内的异常消息达到指定条数时生成报警记录，该时间窗口内此后的消息会更新报警记录的结束时间。后聚合主要是避免误报的情况，比如5秒内3次发现火焰才生成报警记录，那么偶尔一次的火警误识别不会生成报警记录。

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**忽略该字段**

请求示例

```
POST /msg/config/add HTTP/1.1
Content-Type: application/json
Content-Length: 1254

{
    "name": "your_config_name2",
    "desc": "your_description2",
    "topic": "lucky-ai",
    "schema": "calling_detect",
    "exceptionDetect": {
        "rawJson": "{\"objects\":[{\"text\": \"未戴安全帽\", \"w\": 100, \"h\": 100, \"x\": 33, \"y\": 44}, {\"text\": \"未戴安全帽\", \"w\": 123, \"h\": 134, \"x\": 300, \"y\": 440}], \"objects_count\":2}",
        "logicAnd": true,
        "algs": [
            {
                "algName": "arithmetic",
                "param": {
                    "expr": "f * 2 + 1 > 5"
                }
            },
            {
                "algName": "arithmetic",
                "param": {
                    "expr": "f * 2 + 1 > 5"
                }
            },
            {
                "algName": "regexp",
                "param": {
                    "alias": "b",
                    "expr": "未戴.全"
                }
            }
        ]
    },
    "alarmDetect": {
        "mergeInterval":30,
        "isPostMerge":true,
        "errorTimes":2,
        "receivers": [{}]
    }
}
```

响应示例

```
HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status":0,
  "message": "success",
  "data": null
}
```

获取原始消息别名字段接口

请求结构

GET /msg/config/list-fields

请求头域

无特殊请求头。 **请求参数**

参数	类型	是否必选	描述	示例
rawJson	String	是	原始消息中message字段格式，也就是AI算法识别返回的识别结构，具体需要参考对应的AI算法描述。	{"rawJson":"{"objects":[{"text":"未戴安全帽","w":100,"h":100,"x":33,"y":44},{"text":"未戴安全帽","w":123,"h":134,"x":300,"y":440}],"objects_count":2}"

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	Map	key是别名名称，value是对应原始消息字段名称和类型的描述
+keyName	String	别名对应的原始消息字段名称
+type	String	原始消息该字段从rawJson解析出来的类型。目前有string和number两种。number表示整数或浮点数。

请求示例

```
 POST /msg/config/add HTTP/1.1
Content-Type: application/json
Content-Length: 1254

{
   "rawJson": "{\"objects\":[{\"text\":\"未戴安全帽\",\"w\":100,\"h\":100,\"x\":33,\"y\":44},{\"text\":\"未戴安全帽\",\"w\":123,\"h\":134,\"x\":300,\"y\":440}],\"objects_count\":2}"
}
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
 "data": {
  "a": {
   "keyName": "message.objects.h",
   "type": "number"
  },
  "b": {
   "keyName": "message.objects.text",
   "type": "string"
  },
  "c": {
   "keyName": "message.objects.w",
   "type": "number"
  },
  "d": {
   "keyName": "message.objects.x",
   "type": "number"
  },
  "e": {
   "keyName": "message.objects.y",
   "type": "number"
  },
  "f": {
   "keyName": "message.objects_count",
   "type": "number"
  }
 },
 "message": "success",
 "status": 0
}
```

查询异常检测配置列表

接口描述 分页查询异常检测配置列表。

请求结构

POST /msg/config/list

请求头域

无特殊请求头。

请求参数

参数	类型	是否必选	描述	示例
pageNum	Int32	是	分页查询中的页号，从1开始	2
pageSize	Int32	是	每页返回的记录数量	20

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态，0表示成功，其他表示失败。status==0是可能是部分删除成功，部分删除失败，具体参考data字段。
message	String	成功或错误提示
data	Map	分页查询结果
+total	Int32	异常检测配置总条数
+list	Array	查询结果
++id	String	异常检测配置的数据库记录ID
++name	String	异常检测配置名称
++desc	String	异常检测配置描述
++topic	String	异常检测配置的topic
++schema	String	异常检测配置的schema
++name	String	异常检测配置名称
++createAt	Int64	距离格林威治时间1970年01月01日的毫秒级时间戳，表示配置添加的时间
++updateAt	Int64	距离格林威治时间1970年01月01日的毫秒级时间戳，表示配置更新的时间

请求示例

```
 POST /msg/config/list HTTP/1.1
Content-Type: application/json
Content-Length: 1254

{
    "pageNum":1,
    "pageSize":20
}
```

响应示例

```
HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
    "status": 0,
    "message": "success",
    "data": {
        "list": [
            {
                "id": "64914e7020e2cd00015cb6c8",
                "name": "your_config_name2",
                "desc": "your_description2",
                "topic": "lucky-ai",
                "schema": "calling_detect",
                "createAt": 1687244400530,
                "updateAt": 1687244400530
            },
            {
                "id": "64915f0a20e2cd0001f80727",
                "name": "your_config_name2",
                "desc": "your_description2",
                "topic": "t1",
                "schema": "s1",
                "createAt": 1687248650157,
                "updateAt": 1687248650157
            }
        ],
        "total": 2
    }
}
```

删除事件异常检测配置

接口描述 删除异常检测配置。

请求结构

POST /msg/operation/delete

请求头域

无特殊请求头。

请求参数

参数	类型	是否必选	描述	示例
collection	String	是	删除的对象类型。目前固定为walrus_task_config	walrus_task_config
ids	Array	是	异常检测配置的数据库记录ID。可以配置多个ID，实现批量删除	["abcd11345", "134343234ad"]

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败。status==0是可能是部分删除成功，部分删除失败，具体参考data字段。
message	String	成功或错误提示
data	Map	删除结果， 仅当status为0时有意义。
+failIds	Array	删除失败的ID列表
+succIds	Array	删除成功的ID列表

请求示例

```
POST /msg/config/add HTTP/1.1
Content-Type: application/json
Content-Length: 1254

{
    "collection":"walrus_task_config",
    "ids":["64916a2520e2cd0001f80728", "64916a2520e2cd0001f80899"]
}
```

响应示例

```
HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
    "data": {
        "failIds": ["64916a2520e2cd0001f80899"],
        "succIds": [
            "64916a2520e2cd0001f80728"
        ]
    },
    "message": "success",
    "status": 0
}
```

事件订阅

接口描述 注册接收时间中心消息的URL地址。现阶段异常事件规则事先预置或者通过web前端配置，后续可按需开放API接口配置，比如：人脸识别技能的异常规则为有人脸即推送异常事件。

请求结构

POST /msg/config/set-notify-url

请求头域

无特殊请求头。

请求参数

参数	类型	是否必选	描述	示例
receivers	Array	是	URL全路径数组，使用post方式向URL推送异常事件。当前只支持http，不支持https等。如果数组长度为0且reset为true，表示删除所有接收者。 注意不要遗漏http://前缀。	["http://1.2.3.4:8980/r1",“http://a.b.c.d:9977/r2"]
msgType	String	是	消息类型，当前固定为raw，表示原始消息通知。	"msgType": "raw"
reset	Boolean	否	true: 清理之前该msgType的所有receivers为本次请求receivers。 false: 将当前的设置合并到之前的设置中。如果之前已经有相同receiver，则覆盖旧的receiver相关设置。默认为false。	"reset": true
attr	Map	否	key是要进行属性设置的receiver，value是具体属性。如果reset为false，则旧的receiver的属性也可以在本次更新。	"attr": {"withImage": "never" }
+withImage	String	否	如何接收本消息对应的抽帧图片。可选值包括：never(从不接收） exception（仅当事件中心判别消息为异常消息时） normal（仅当事件中心判别消息非异常时） any（总是接收）。默认为any	"withImage": "exception"
+filter	Map	指定只对符合过滤条件的消息进行推送。topic,schema,device_id,filter_func之间是逻辑AND的关系，数组字段本身是逻辑或的关系。		
++topic	Array	String数组，指定哪些topic需要推送，只要满足数组中一项即匹配该字段。如果为空则忽略该过滤条件	["lucky-ai", "topic2"]	
++schema	Array	String数组，指定哪些schema需要推送，只要满足数组中一项即匹配该字段。如果为空则忽略该过滤条件		
++device_id	Array	String数组，指定哪些设备ID需要推送，只要满足数组中一项即匹配该字段。如果为空则忽略该过滤条件		
++filter_func	String	javascript配置的过滤函数，输入是原始消息，输出是bool，返回true表示匹配，false表示不匹配。如果为空则忽略该过滤条件		

> 过滤说明 假设rawMsg为原始消息的json结构体，则rawMsg满足rawMsg.topic in filter.topic AND rawMsg.schema in filter.schema AND rawMsg.device_id in filter.device_id AND eval(rawMsg.filter_func) == true 时，才会推送。

> **示例说明**
>
> 只推送检测到目标中未戴安全帽的原始消息，可以使用如下filter_func:
>
> `function filter_func(msg){  var obj =JSON.parse(msg);if(obj.message.objects.length>0){ var sub_obj =`

> obj.message.objects;for(var i=0; i<sub_obj.length; i++){ if (sub_obj[i].text.indexOf('未戴安全帽') != -1){ return true;}}}return false;};

> 只推送匹配了异常检测的filter_func:
>
> "function filter_func(msg){ var obj =JSON.parse(msg);if(obj.tag) { if(obj.tag.length){return true};}; return false}"
>
> 如果原始消息匹配了异常检测配置，原始消息会有tag字段（字符串数组，记录异常检测配置ID），表示匹配了哪些异常检测配置，所以这里检测了tag字段是否为空。

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	**忽略该字段**

请求示例

```
 POST /msg/config/set-notify-url HTTP/1.1
Content-Type: application/json
Content-Length: 1254

{
  "receivers": ["http://1.2.3.4:8980/new-recv1", "http://2.3.4.5:9977/new-recv2"]
  "msgType": "raw",
  "reset": false,
  "attr": {
    "http://1.2.3.4:8980/new-recv1": {
      "withImage": "exception",
      "filter": {
        "topic": ["lucky-ai", "t2"],
        "schema": ["face", "smoke"],
        "device_id": ["132", "432"],
        "filter_func":"function filter_func(msg){ var obj =JSON.parse(msg);if(obj.message.objects.length>0){ var sub_obj = obj.message.objects;for(var i=0; i<sub_obj.length; i++){ if (sub_obj[i].text.indexOf('无上报') != -1){ return true;}}}return false;};"
      },
      "http://2.3.4.5:9977/new-recv2": {
        "withImage": "normal"
      },
      "http://a.b.c.d/old-existed/receiver": {
        "withImage": "any"
      }
    }
}
```

注：如果希望全部URL不再接收通知，可将receivers置空并设置reset为true，请求示例如下：

```
{
 "receivers": [],
 "msgType": "raw",
 "reset": true
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
 "status":0,
 "message": "success",
 "data": null
}
```

异常事件推送数据格式

推送方式

向订阅URL以POST方式推送异常事件

推送参数

推送消息为JSON格式。收到响应http 200表示成功。目前无论是否推送成功，不会重试，只会记录失败日志。

参数	类型	描述
id	String	消息存储ID，只在单个边缘盒子唯一，不同边缘盒子之间可能重复
tag	Array	String数组，是否异常的标识。该字段不存在或者数组为空表示非异常，否则表示异常。非空时，每一项表示对应异常检测配置在DB中的主键ID。例如：["625df16f82cdb900013a5039", "625df16f82cdb900013a5041"]
topic	String	固定为lucky-ai
schema	String	技能英文名称
msg_id	String	消息唯一标识
local_time	Int64	毫秒级消息时间戳
device_id	String	单个边缘盒子上的摄像头设备的字符串唯一标识（只在该边缘盒子上唯一，有可能与其他边缘盒子的某个摄像头ID相同），与添加设备接口返回的字符串id唯一对应
machine_id	String	边缘盒子的唯一标识，不同边缘盒子该字段值不同
message	Map	具体识别结果
+image	String	BASE64编码的JPEG图片数据，解码BASE64后可以直接作为JPEG打开
+objects	Array	多个识别结果的描述数组
++text	String	识别结果文本
++bcol	String	对象绘图边框颜色。RGB格式。例如”#ff0000”表示红色

or		
++x	Int32	识别区域的左上角x坐标
++y	Int32	识别区域左上角y坐标
++h	Int32	识别区域高度
++w	Int32	识别区域宽度
++content	Map	识别结果的规格化输出，目前只有人流量算子有该字段
+++current	Int32	画面当前人数，目前只有人流量算子有该字段
region	Map	周界识别区域。如果没有设置，该字段内容为空
+local_time	整数	毫秒级时间戳，表示周界区域设置时间
+ai	String	算法英文名称，同schema
+id	String	该边缘盒子上该算法对应的周期区域唯一标识
+deviceId	String	边缘盒子上摄像头ID，同上述device_id字段
+text	String	周界区域名称
+bcolor	String	边框颜色。指定的十六进制RGB颜色。例如 #ff00ff表示全红+全蓝组合的颜色
+regions	Array	每个字典对象代表一个多边形
++polygon	Array	Int32类型的数组。以(x1,y1), (x2,y2),…,(xn,yn)多边形顶点坐标表示的多边形。例如polygon[0]代表x1, polygon[1]代表y1, polygon[2]代表x2

推送事件格式示例

数据格式与“查询视频分析事件”API的响应参数data部分对应。

```
 POST /your-receiver-URL/xxxx HTTP/1.1
Content-Type: application/json
Content-Length: 1254

{
    "id": "5349b4ddd2781d08c09890f4",
    "topic": "lucky-ai",
    "schema": "face",
    "msg_id": "D5-1608186865880-3473",
    "local_time": 16081865880,
     "machine_id": "边缘盒子唯一标识XXX",
    "device_id": "135",
    "region": {
        "id": "5349b4ddd2781d08c0989077",
        "ai": "face",
        "local_time": 1653878385345,
        "device_id": "135",
        "text": "区域1",
        "bcolor": "0xff00bb",
        "regions": [
            {
                "polygon": [34, 55, 340, 547, 42, 250]
            },
            {
                "polygon": [24, 25, 340, 447, 52, 250]
            }
        ]
    },
    "message": {
        "image": "JPEG文件的BASE64编码数据",
        "objects": [
            {
                "x": 3, "y": 4, "w": 100, "h": 200, "bcolor": "#ff00bb",
                "text": "张三"
            },
            {
                "x": 13, "y": 14, "w": 200, "h": 100, "bcolor": "#ff00bb",
                "text": "李四"
            }
        ]
    }
}
```

查询已注册事件订阅地址

接口描述 查询已经注册的事件接收URL地址。

请求结构

GET /msg/config/get-notify-url

请求头域 无特殊请求头。

请求参数 无

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**Array**	消息列表
attr	Map	订阅时一些特殊需求设置。key是URL地址，value是Map。目前attr可以为空
+withImage	String	指明向该URL发送消息时是否需要携带对应图片。可选值包括：exception、normal、any、never，分别表示：仅当异常检测匹配为异常时、仅当异常检测未匹配时、总是携带图片、从不携带图片
id	String	数据库记录ID，在相同盒子上唯一，多个盒子之间不保证唯一
msgType	String	调用订阅接口时设置的消息类型，目前固定为raw
receivers	Array	接收消息的URL地址列表
createTime	Int64	毫秒级时间戳, 首次调用订阅接口的时间
updateTime	Int64	毫秒级时间戳， 最近一次调用订阅接口的时间
reset	Boolean	最近一次调用订阅接口时设置的reset值

请求示例

```
GET /msg/config/get-notify-url
Content-Type: application/json
Content-Length: 0
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
    "data": [
        {
            "attr": {
                "http://192.168.1.87:9999/a/b": {
                    "withImage": "exception"
                }
            },
            "id": "63635cb42008200001cac9ab",
            "msgType": "raw",
            "receivers": [
                "http://192.168.1.87:9999/a/b"
            ],
            "reset": false,
            "createTime": 1667456180600,
            "updateTime": 1667456180788
        }
    ],
    "message": "success",
    "status": 0
}
```

查询视频分析事件

接口描述 由于嵌入式设备性能较低和SD卡读取较慢，查询全量消息时，建议每次查询时间间隔不要太大，不超过10分钟。

请求结构

POST /msg/message/raw/query

请求头域 无特殊请求头。

请求参数

参数	类型	是否必选	描述	示例
topic	String	是	当前固定为lucky-ai, 表示AI识别消息	lucky-ai
schema	String	是	要查询的AI技能英文名称	face
startTime	String	是	查询开始时间，精确到秒	2020-01-12 23:22:11
endTime	String	是	查询结束时间，精确到秒。查询 startTime <= time <= endTime的消息，建议时间间隔小于10分钟	2020-01-12 23:32:25
pageSize	Int32	是	pageSize<=1000	100
pageNum	Int32	是	该时间段第一次查询时建议pageNum为1，此时会返回第1页数据和该时间段总共的数据量total。后续该时间段可以每次按照加1递增pageNum进行查询， 当pageNum>1时total会固定为0	2

响应头域 无特殊响应头。

响应参数

参数	类型	描述
status	Int32	接口返回状态, 0表示成功，其他表示失败
message	String	成功或错误提示
data	**JSON**	消息列表
total	Int32	事件总量，pageSize=1时返回总量total的数值，pageNum>1时total会固定为0
list	Array	事件数组
+id	String	消息存储数据库ID
+topic	String	固定为lucky-ai
+schema	String	技能英文名称
+msg_id	String	消息唯一标识
+local_time	Int64	毫秒级消息时间戳
+device_id	String	摄像头设备ID。在同一个边缘盒子上唯一，在不同边缘盒子之间可能有重复
+region	Map	周界区域的信息，如果没有设置，该字段内容为空
++local_time	Int64	毫秒级时间戳，表示周界区域设置时间
++text	String	周界区域名称
++bcolor	String	指定的十六进制RGB颜色。例如 #ff00ff表示全红+全蓝组合的颜色
++regions	Array	多边形数组，每个数组元素代表一个多边形区域
+++polygon	Array	Int32数组，以(x1,y1), (x2,y2),…,(xn,yn)多边形顶点坐标表示的多边形。例如polygon[0]代表x1, polygon[1]代表y1, polygon[2]代表x2
+message	Map	具体识别结果
++image	String	图片BASE64编码。仅当该消息为异常消息(tag字段对应数组长度不为空)时才非空
++objects	Array	多个识别结果的描述数组
+++text	String	识别结果文本
+++x	Int32	识别区域的左上角x坐标
+++y	Int32	识别区域左上角y坐标
+++h	Int32	识别区域高度
+++w	Int32	识别区域宽度

请求示例

```
 POST /msg/message/raw/query HTTP/1.1
Content-Type: application/json
Content-Length: 1254

{
  "topic": "lucky-ai",
  "schema": "face2",
  "startTime": "2021-02-20 11:05:06",
  "endTime": "2021-02-20 11:15:06" ,
  "pageSize": 10,
  "pageNum": 1
}
```

响应示例

```
 HTTP/1.1 200 OK
cache-control: no-cache
content-length: 484
content-type: application/json; charset=utf-8
date: Thu, 17 Nov 2022 00:45:02 GMT
server: nginx/1.21.6

{
  "status": 0,
  "message": "success",
  "data": {
    "total": 2,
    "list": [
      {
        "id": "",
        "topic": "lucky-ai",
        "schema": "hat",
        "msg_id": "D16-1638734232-4232",
        "local_time": 1653878385123,
        "message": {
          "image": "jpeg图片BASE64编码",
          "objects": [
            {
              "x": 3,"y": 4,"w": 150, "h": 350, bcolor": "0xff00ff",
              "text": "未戴安全帽:0.874"
            },
            ...
          ]
        }
      }
    ]
  }
}
```

人脸管理接口

人脸识别

接口描述 本接口主要用于人脸识别

请求方式

POST

请求URL

/baidu/5gai/v1/face-api/detect

请求参数

参数	类型	是否必填	描述	示例
image	字符串	是	图像数据	base64编码 {"image": "base64(xxxx)"}

请求示例

```
=
"}
```

响应参数 | 参数 | 类型 | 是否可缺省|说明 | | --- | --- | --- |---| | objects | 列表 |否| 检测到的目标数组 | | message | 字符串 |否| 返回信息 |

响应参数objects说明

参数	类型	是否可缺省	说明
x	整数	否	横坐标
y	整数	否	纵坐标
w	整数	否	框宽
h	整数	否	框高
text	字符串	是	"查无:0.522" 或者 "组=GroupA,用户=UserA11,匹配度=0.931"
bcolor	字符串	是	框颜色，以8bit的RGB表示,默认"0xFFFF00"
font	整数	是	字体大小(像素)，默认24

响应示例

```
 {
  "objects":
  [
    {"x":1, "y":1, "w":100, "h":100, "text": "查无:0.521", "bcolor": "0xFFFF00", "font": 24 },
    {"x":22, "y":22, "w":120, "h":120, "text": "组=GroupA,用户=UserA11,匹配度=0.931", "bcolor": "0xFFFF00", "font":
24 }
  ],
  "message": "ok"
}
```

新增用户

接口描述

添加新的用户，添加时必须同时携带一张用户人脸图片作为初始化的人脸。本接口会固定使用default作为用户组名称（如果没有，内部会自动创建该组），目前不支持多组的配置。

请求结构

POST /console/api/v1/face/user/add

请求头域

Content-Type: application/json

请求参数

参数	类型	是否必填	说明
name	string	是	用户名
data	string	是	图片的BASE64编码的内容

响应参数

参数	类型	说明	
message	String	成功或错误提示	
status	Int32	返回码，成功为0，失败为http错误码。	
data	Json	新增用户和图片的ID	
+id	Int64	用户ID	
+picIDs	Array of Int64	与请求图片按顺序对应的图片ID列表	

> 说明 存在接口返回失败但是人脸用户已经添加成功（可以通过查询用户接口可以看到新增的用户），此时通常说明调用人脸算法失败，但是一个『没有任何人脸的用户信息』已经添加到数据库。只需要待人脸算法正常时，继续调用更新人脸库的接口新增人脸图片即可。

响应头域

Content-Type：application/json

请求示例

```
 {
   "name":"小猪1号",
   "data": ["/9j/4AAQSkZJRgABAQAAAQABAADs...", "/9j/4AAQSkZJRgABAQAAAQABAADs..."]
}
```

响应示例

```
 {
   "message":"success",
   "status": 200,
   "data": {
      "id": 3,
      "picIDs": [7, 9]
   }
}
```

新增单张人脸图片

接口说明 人脸库对指定用户新增1张人脸图片。 **请求方式** POST /console/api/v1/face/pic/add

请求头域 无特殊请求头。

请求参数

参数	类型	是否必填	说明
id	Int64	是	用户ID
data	String	是	新增图片base64字符串

响应头域 Content-Type: application/json

响应参数

参数	类型	说明
message	String	成功时success，失败 failed
status	Int	返回码，成功为0，失败为http错误码
data	Int64	新增图片的ID

请求示例

```
 {
   "id":1,
   "data": "/9j/4AAQSkZJRgABAQAAAQABAADs..."
}
```

响应示例

```
 {
   "message": "success",
   "status": 0,
   "data": 3
}
```

人脸查询

接口描述

查询人脸库的用户列表

请求结构 POST /console/api/v1/face/user/list

请求头域

无特殊请求头

请求参数

参数	类型	是否必填	说明
filter	string	是	搜索用户名包含字符串的用户
pageNum	int	否	页码（默认为1）
pageSize	int	否	每页记录数（默认为20）

响应头域

Content-Type: application/json

响应参数

参数	类型	说明
message	String	成功时success，失败 failed
status	Int32	返回码，成功为0，失败时非0
data	Json	失败时返回失败原因(字符串), 成功时返回数据list（json格式）
+total	Int32	总记录数
+list	Array	用户组列表
++id	Int64	用户ID
++name	String	用户名称
++createAt	String	创建时间
++picNum	Int32	图片数量
++infoNum	Int32	目前固定为0

请求示例

```
 {
   "filter": "user01",
   "pageNum": 1,
   "pageSize": 20
}
```

响应示例

```
 {
"message":"success",
"status": 0,
"data:{
       "list":[
          {
             "id": 2,
             "name": "小猪1号",
             "createAt": "2020-08-12 02:02:02",
             "picNum": 2,
             "infoNum": 0
          },
          {
             ...
          }
       ],
       "total":34
   }
}
```

人脸详情

接口描述 查询人脸库指定用户的详细信息。

请求结构 GET /console/api/v1/face/user/detail

请求头域 无特殊请求头。

请求参数 | 参数 | 类型| 是否必填 | 说明 | | --- | --- | --- | --- | | id | Int64 | 是 | 人脸库用户的ID |

响应参数

参数	类型	说明
message	String	成功时success，失败 failed
status	Int32	返回码，成功为0，失败时非0
data	Json	用户信息
+id	Int64	用户ID
+name	String	用户名称
+createAt	String	创建时间,格式为类似"2005-12-30 13:14:15"
+picNum	Int32	图片数量
+picId	Array	图片ID数组

请求示例 GET /console/api/v1/face/user/detail?id=2

响应示例

```
{
  "data: {
            "id": 3,
            "name": "小猪1号",
            "createAt":" 2020-08-12 02:02:02",
            "picNum": 2,
            "picId":[
             36,
             37
        ],
  },
  "message":"success",
  "status": 0
}
```

人脸图片修改

接口说明 人脸库更新指定用户的人脸图片。 **请求方式** POST /console/api/v1/face/pic/replace

请求头域 无特殊请求头。

请求参数

参数	类型	是否必填	说明
id	Int64	是	用户ID
reserve	Array	是	要保留的图片ID，为空表示旧图片全部清除
data	Array	是	新增图片base64数组

响应头域 Content-Type: application/json

响应参数

参数	类型	说明
message	String	成功时success，失败 failed
status	Int	返回码，成功为0，失败为http错误码
data	Json	目前固定为null

请求示例

```
 {
   "id":1,
   "reserve":[
    36,
    37
   ],
   "data":[ /9j/4AAQSkZJRgABAQAAAQABAADs..." ,"/9j/4AAQSkZJRgABAQAAAQA..."]
}
```

响应示例

```
 {
   "message": "success",
   "status": 0,
   "data": null
}
```

人脸图片下载

接口描述 下载某个人脸对应的图片。

请求结构 GET /console/api/v1/face/pic/download

请求头域 无特殊请求头。

请求参数 | 参数 | 类型| 是否必选 | 描述 | 示例 | | --- | --- | --- | --- | --- | | id | Int64 | 是 | 人脸图片的ID | 3 |

响应头域 Content-Type: application/octet-stream Content-Transfer-Encoding: binary

响应参数 响应内容为原始jpeg图片二进制内容。响应body的内容可以保存为xxx.jpeg，这样可以直接打开图片查看。

请求示例

```
GET /console/api/v1/face/pic/download?id=2
```

响应示例

```
jpeg图片文件的二进制内容
```

人脸删除

接口说明 人脸库删除指定用户。会连同该用户下所有图片一并清理。 **请求结构** POST /console/api/v1/face/user/delete **请求头域** 无特殊请求头。

请求参数

参数名称	类型	是否必填	说明
id	Array	是	用户ID列表

响应头域 Content-Type: application/json

响应参数

参数名称	类型	说明
data	String	返回数据，成功为空字符串,失败为失败原因
message	String	成功时success，失败时返回原因
status	Int32	返回码，成功为0，失败为非0

请求示例

```
 {
   "id": [23, 24] ,
}
```

响应示例

```
 {
   "data:"",
   "message": "success",
   "status": 0
}
```

删除单张人脸图片

接口说明 人脸库删除指定用户的1张人脸图片。该接口删除指定用户最后一张人脸图片后，会同时删除该用户，如果希望重新添加该用户人脸图片，必须先调用添加用户接口。 **请求方式** POST /console/api/v1/face/pic/delete

请求头域 无特殊请求头。

请求参数

参数	类型	是否必填	说明
id	Int64	是	人脸用户ID
picId	Int64	是	请求删除图片的ID

响应头域 Content-Type: application/json

响应参数

参数	类型	说明
message	String	成功时success，失败 failed
status	Int	返回码，成功为0，失败为http错误码
data	Json	目前固定为null

请求示例

```
 {
   "id": 1,
   "picId": 3
}
```

响应示例

```
{
  "message": "success",
  "status": 0,
  "data": null
}
```

EventHub事件中心

事件中心操作指南

智能事件中心是基于原始消息进行事件配置，实现异常行为的报警及报警信息查询功能，包括原始消息、事件配置和报警查询功能。事件中心地址为：https:// 服务器ip:81/msg 。

原始消息

1. 功能简介

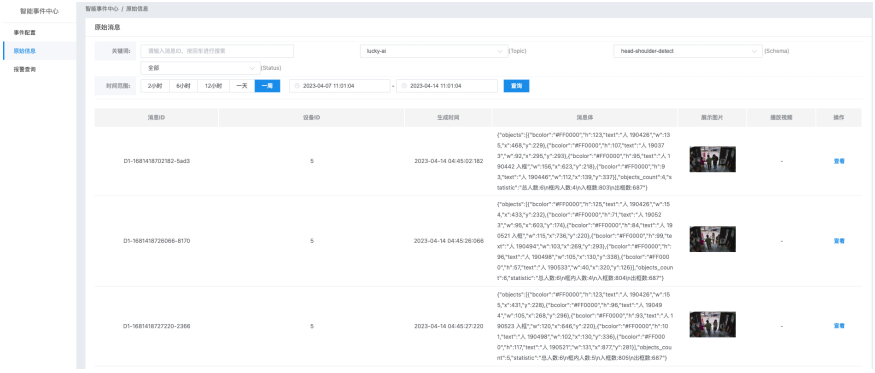
原始消息是指算子上报的最初原始消息，基于原始消息体可用于事件配置，实现异常行为或事件上报。

2. 操作说明

- 原始消息查询

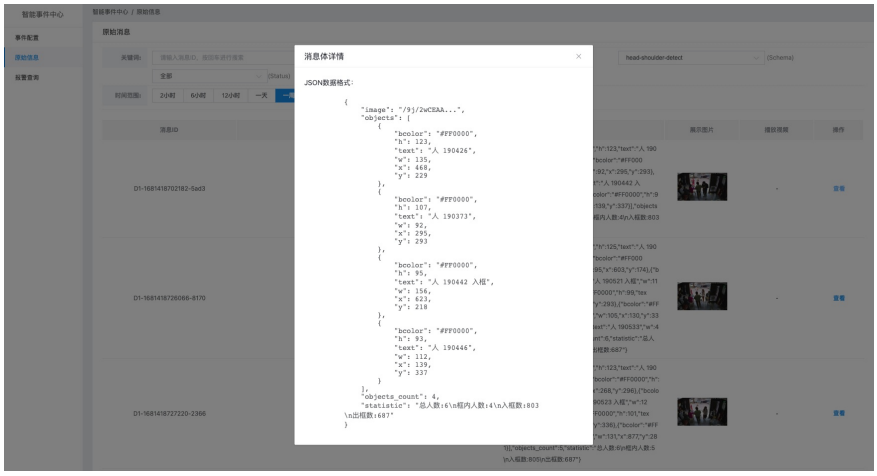
原始消息可以通过输入消息ID、选择主题、选择Schema及时间快捷键或自定义时间来查询。时间默认展示最近2小时。

原始消息列表包括消息ID、生成时间和消息体，操作支持查看消息体的JSON。



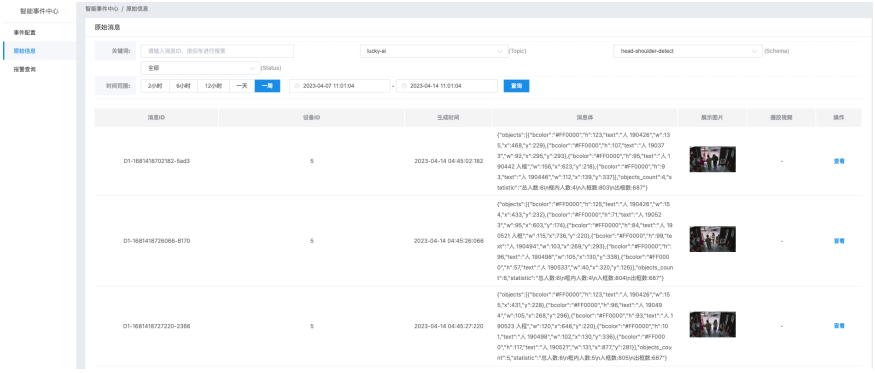
- 消息体JSON查看

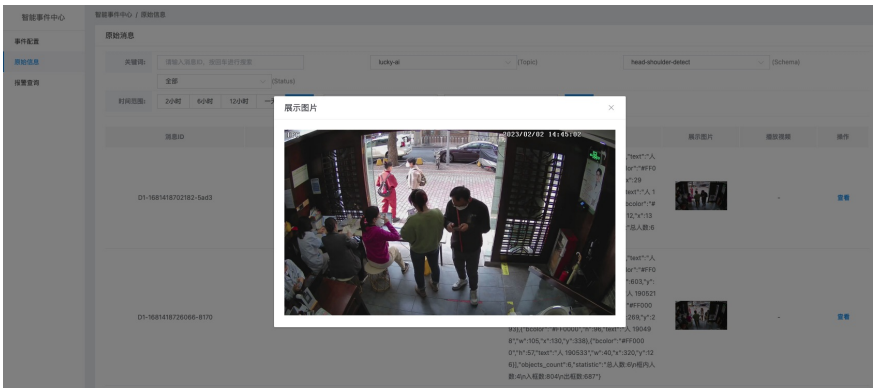
点击表格操作栏的【查看JSON】按钮，会弹出消息体的JSON内容。



- 消息图片查看

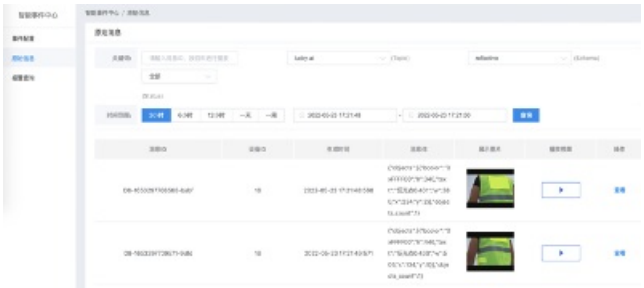
点击【展示图片】列的任意图片，会展示被检测到的符合条件的留存视频图片，支持放大查看。





- 视频播放

右侧异常消息列可以调整到报警对应的原始消息，可以查看图片和视频（视频需要等待15秒左右待完全生成后可以播放）。



事件配置

1. 功能简介

事件配置是指基于基本信息、筛选数据结构、异常检测配置、事件聚合配置和报警配置生成的执行任务，包括事件配置列表展示和配置项详情及编辑，详情如下：

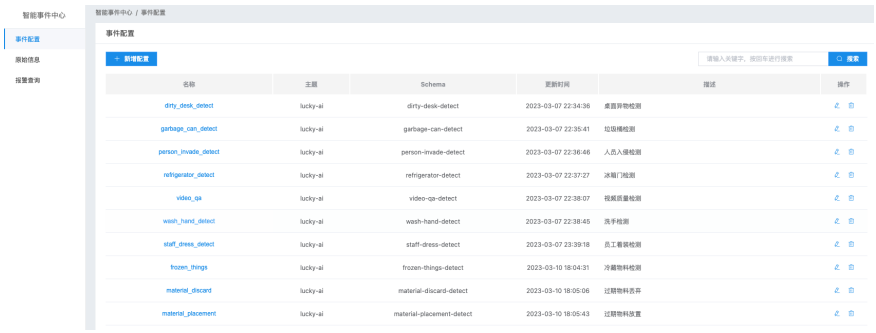
事件配置列表：展示事件配置项名称、主题、Schema、更新时间和描述信息，支持异常检测的修改。

事件配置项详情/编辑：可以查看事件配置的基本信息、数据结构、异常检测配置、事件聚合配置和报警配置。

2. 操作说明

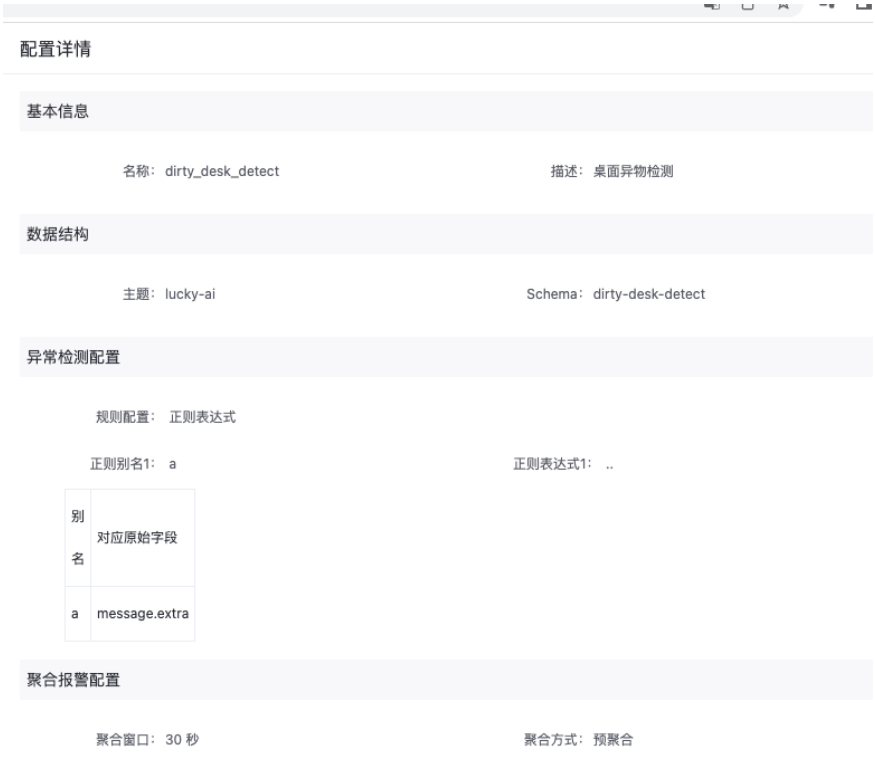
- 事件配置列表

事件配置列表会展示当前平台涉及报警的事件配置项，支持搜索，如图所示：



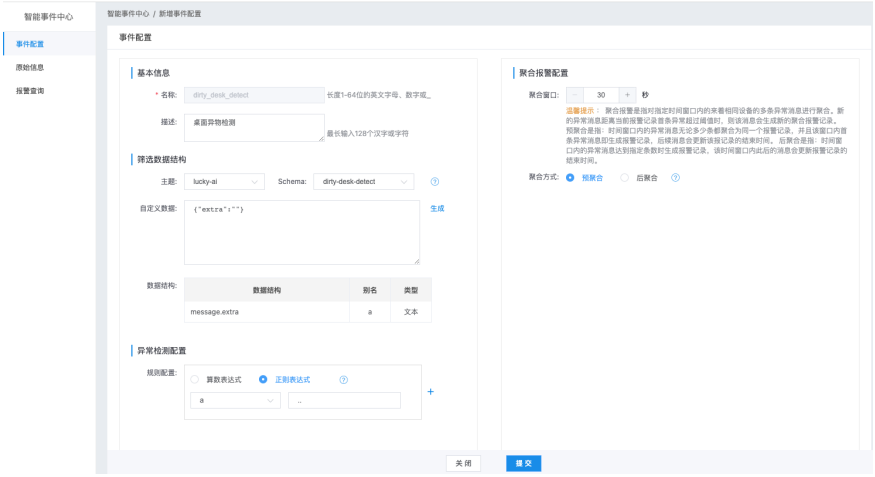
- 事件配置详情

点击【名称】栏下的事件名称，会滑出详情页面：



- 事件配置编辑

点击操作栏的【编辑】按钮，会跳转至编辑页面，可对异常检测配置模块进行修改：



报警查询

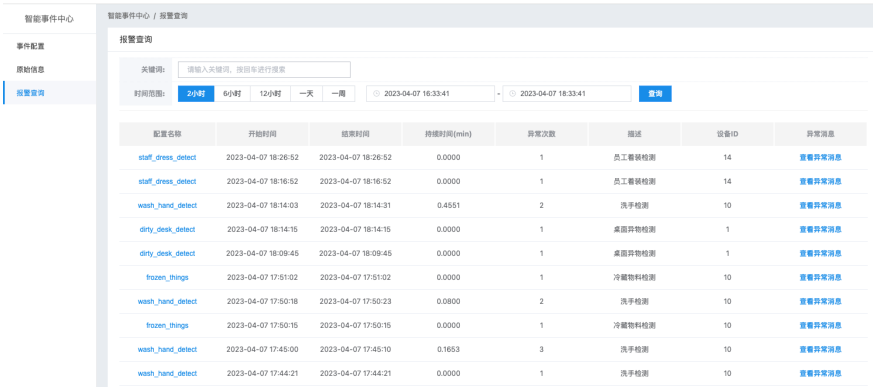
1. 功能简介

报警查询是基于事件配置中的异常检测模块，若检测出异常，会发出报警。通过搜索事件配置项名称或报警模板标题可以查询报警信息。

2. 操作说明

- 报警查询

默认展示全部报警信息，展示字段有配置名称、创建时间、标题、开始时间和持续时间（单位为分支）：



可以通过关键词（基于配置项名称及配置模板的标题搜索）、时间范围或自定义时间来查询报警。系统默认展示最近2小时的报警信息。

- 报警详情

点击【配置名称】栏任意报警配置，会弹出报警详情页面，展示报警基础信息和通告信息。其中报警基础信息有事件配置项名称、创建时间、报警结束、上次报警时间。通告信息包括标题、描述、内容、开始时间、结束时间和持续时间。

EdgeInfer边缘视频推理引擎

API参考

接口概览

基本概念

变量名	意义	格式	示例
deviceId	识备标识	任意字符串	dev1
bcolor	框的颜色	0xRRGGBB 格式的字符串	0xFF1000 表示R=255,G=16,B=0的颜色
frameId	抽帧任务号	正整数	1
polygon	多边形	数字列表，以 [x1,y1, x2,y2, ... xn,yn] 表示	[0,0, 1,0, 0,1] 表示 以(0,0),(1,0),(0,1)为顶点的多边形

使用流程

1. 先创建抽帧任务(POST /v1/frame)，创建成功后会返回frameId，可以通过(GET /v1/frame/status/$id)查看任务状态；
2. 【可选】创建合成任务(POST /v1/merge), 可以通过(GET /v1/merge/status/$id)查看任务状态, 或者使用ffmpeg或者vlc确定合成视频是否可以播放；
3. 添加AI算法(POST /v1/ai)，可以通过(GET /v1/ai)查询算子是否已经存在，已经存在的就不需要添加；
4. 创建识别任务(POST /v1/detect)，创建时需要填入第1步的frameId和第3步的算法名。如果识别成功，则会在合成视频上显示识别结果；
5. 查看状态，可以查看内部处理时间(GET /v1/metric)。

算法功能接口

注册算法

接口描述 本接口用于注册算法。

接口内容 POST /v1/ai

请求参数

参数名称	类型	是否必填	说明
name	string	是	算法名称
comment	string	是	备注
url	string	是	请求地址

请求示例

```
 POST /v1/ai
{
"name": "carStay",
"comment": "车辆逗留",
"url": "http://1.2.3.4:8888/baidu/5gai/v1/car-stay/detect"
}

Response:
{
"message": "Success"
}
```

修改算法

接口描述 本接口用于修改算法。

接口内容 PUT /v1/ai/$name

请求参数

参数名称	类型	是否必填	说明
comment	string	是	备注
url	string	是	请求地址

请求示例

```
 PUT /v1/ai/carStay
{
"comment": "车辆逗留",
"url": "http://1.2.3.4:8888/baidu/5gai/v1/car-stay/detect"
}

Response:
{
"message": "Success"
}
```

查询算法列表

接口描述 本接口用于查询算法列表。

接口内容 GET /v1/ai

响应参数

参数名称	类型	是否必填	说明
name	string	是	算法名称
comment	string	是	备注
url	string	是	请求地址

请求示例

```
 GET /v1/ai

Response:
{
"data": [
{"name":"face", "comment": "人脸识别", "url": "http://1.2.3.4:8888/baidu/5gai/v1/car-stay/detect"} ,
{"name":"xxxx", "comment":"xxxx", "url": "http://a/b/c"}
],
"message": "Success"
}
```

抽帧功能接口

抽帧任务创建

接口描述 本接口用于抽帧任务的创建。

接口内容 POST /v1/frame

请求参数

参数名称	类型	是否必填	说明
input	string	是	输入地址
deviceId	string	是	设备号（唯一）

支持协议

输入地址支持以下协议：

协议	示例
rtsp	rtsp://1.2.3.4:554/raw/stream1
rtmp	rtmp://1.2.3.4:1935/raw/stream1
tcp	tcp://1.2.3.4:1234

响应参数

参数名称	类型	是否必填	说明
id	int	是	任务id

请求示例

```
 Request:
POST /v1/frame
{
"input": "rtsp://1.2.3.4:554/test",
"deviceId": "id1"
}


Response:
{
"data": {"id": 1},
"message": "Success"
}
```

抽帧图片获取

接口描述 本接口用于获取抽帧图片。

接口内容 GET /v1/frame/image/$frameId

请求参数

参数名称	类型	是否必填	说明
frameId	int	是	抽帧任务号

正常返回值：jpg格式的图片，可以用浏览器直接显示。

异常返回值：400 任务号不存在，500 内部错误。

特别说明：仅返回最近生成的一张jpeg图片。如果不开启识别任务，则不会生成jpeg图片，也会导致获取不到图片。

抽帧任务查询

接口描述 本接口用于查询抽帧任务。

接口内容 GET /v1/frame

请求参数

参数名称	类型	是否必填	说明
id	int	否	任务号
deviceId	string	否	设备号

响应参数

参数名称	类型	是否必填	说明
message	string	是	执行结果

请求示例

```
 Request:
GET /v1/frame?id=1&deviceId=id1

Response:
{
"data": [
{
"id": 1,
"input": "rtsp://1.2.3.4:554/test1",
"deviceId": "id1"
},{
"id": 2,
"input": "rtsp://1.2.3.4:554/test2",
"deviceId": "id2"
}
],
"message": "Success"
}
```

抽帧任务删除

接口描述 本接口用于删除抽帧任务。

接口内容 DELETE /v1/frame/id/$id

DELETE /v1/frame/$deviceId

以上两个接口是相同的功能，区别在于以下请求参数。

请求参数

参数名称	类型	是否必填	说明
id	int	是	任务id
deviceId	string	是	设备号

响应参数

参数名称	类型	是否必填	说明
message	int	是	执行结果

请求示例

```
 Request:
DELETE /v1/frame/id/1
DELETE /v1/frame/device1

Response:
{
"message": "Success"
}
```

返回成功时，表示该任务已经停止，并且会删除所有关联的合成、识别任务。

识别功能接口

创建识别任务

接口描述 本接口用于创建识别任务。

接口内容 POST /v1/detect

请求参数

参数名称	类型	是否必填	说明
deviceId	string	是	设备标识
frameId	int	是	抽帧号
ai	string	是	算法名称
elapse	int	否	单位ms， 识别间隔，默认1000ms

请求示例

```
 POST  /v1/detect
{
"deviceId": "ABC",
"frameId": 2,
"ai": "face",
"elapse": 1000
}


Response:
{
"data": {"id": 1},
"message": "Success"
}
```

识别任务查询

接口描述 本接口用于查询识别任务。

接口内容 GET /v1/detect

请求参数

参数名称	类型	是否必填	说明
id	int	否	任务号
deviceId	string	否	设备号

注意：

1. 不支持分页；

2. id和deviceId为搜索条件，可以不填，不填表示不匹配该项数据。

响应参数

参数名称	类型	是否必填	说明
message	string	是	执行结果

请求示例

```
 Request:
GET /v1/detect?id=1&deviceId=id1

Response:
{
"data": [
{
"id": 1,
"deviceId": "ABC",
"frameId": 2,
"ai": "face",
"elapse": 10
},{
"id": 2,
"deviceId": "ABC",
"frameId": 2,
"ai": "face",
"elapse": 10
}
],
"message": "Success"
}
```

删除识别流

接口描述 本接口用于删除识别流。

接口内容 DELETE /v1/detect/id/$taskid

DELETE /v1/detect/$deviceId

请求参数

参数名称	类型	是否必填	说明
deviceid	string	是	设备标识
taskid	string	是	识别流任务id

请求示例

```
 DELETE /v1/detect/id/1
DELETE /v1/detect/ABC


Response:
{
"message": "Success"
}
```

合成模块

合成任务创建

接口描述 本接口用于创建合成任务。

接口内容

POST /v1/merge

请求参数

参数名称	类型	是否必填	说明
input	string	否	输入地址
deviceId	string	是	设备号
output	string	是	输出地址

注意：

1. 输出地址支持rtsp和rtmp协议；

2. 如果deviceId不存在对应的抽帧任务，则会根据input创建抽帧任务。

响应参数

参数名称	类型	是否必填	说明
id	int	是	任务号

请求示例

```
 Request:
POST /v1/merge
{
"input": "rtsp://1.2.3.4:554/test",
"deviceId": "id1",
"output": "rtsp://1.2.3.4:554/test2"
}

Response:
{
"data": {"id": 1},
"message": "Success"
}
```

合成任务查询

接口描述 本接口用于查询合成任务。

接口内容 GET /v1/merge

请求参数

参数名称	类型	是否必填	说明
id	int	否	抽帧任务号
deviceId	string	否	设备号

注意：

1. 不支持分页；

2. id和deviceId为搜索条件，可以不填，不填表示不匹配该项数据。

响应参数

参数名称	类型	是否必填	说明
message	string	是	执行结果

请求示例

```
 Request:
GET /v1/merge?deviceId=id1


Response:
{
"data": [
{
"id": 1,
"input": "rtsp://1.2.3.4:554/test1",
"deviceId": "id1",
"output": "rtsp://1.2.3.4:554/test2"
},
{
"id": 2,
"input": "rtsp://1.2.3.4:554/test1",
"deviceId": "id1",
"output": "rtsp://1.2.3.4:554/test3",
}
],
"message": "Success"
}
```

合成任务删除

接口描述 本接口用于删除合成任务。

接口内容 DELETE /v1/merge/id/[id]

DELETE /v1/merge/[deviceId]

以上两个接口是相同的功能，区别在于以下请求参数。

请求参数

参数名称	类型	是否必填	说明
id	int	是	任务号
deviceId	string	是	设备号

响应参数

参数名称	类型	是否必填	说明
message	string	是	执行结果

请求示例

```
 Request:
DELETE /v1/frame/id/1
DELETE /v1/frame/device1


Response:
{
"message": "Success"
}
```

AI算法

头部识别

接口描述

本接口主要用于头部检测识别

请求方式

POST

请求端口

7999

请求URL

/baidu/5gai/v1/head-detect/detect

header为Content-Type:application/json

Host: head-detect

请求参数

参数名称	类型	是否必填	说明
image	string	是	图像数据，base64编码例如 {"image": "base64(xxxx)"}

请求示例

curl -X POST -d @data_head.json IP:7999/baidu/5gai/v1/head-detect/detect -H "Host:head-detect"

其中**data_head.json**是图像存储为json格式文件

正常返回如下json结构，"message"为 "ok"。失败时"message"显示错误信息

参数	类型	说明
objects	list	检测到的目标数组
message	string	返回信息

objects 数据结构如下

	类型	是否可缺省	意义
x	int	否	横坐标
y	int	否	纵坐标
w	int	否	框宽
h	int	否	框高
text	string	是	"发现人员:confidence" 备注confidence是置信度
bcolor	string	是	框颜色，以8bit的RGB表示,默认"0xFFFF00"
font	int	是	字体大小(像素)，默认24

例如：

```
{
 "objects": [
  {"x":1, "y":1, "w":100, "h":100, "text": "发现人员:0.91", "bcolor": "0xFFFF00", "font": 24 },
  {"x":1, "y":1, "w":100, "h":100, "text": "发现人员:0.89", "bcolor": "0xFFFF00", "font": 24 }
 ],
 "message": "ok"
}
```

打电话检测

请求描述

本接口主要用于打电话动作的检测

请求方式

POST

请求端口

7999

请求URL

/baidu/5gai/v1/calling-detect/detect

header为Content-Type:application/json

Host: calling-detect

请求参数

参数名称	类型	是否必填	说明
image	string	是	图像数据，base64编码例如 {"image": "base64(xxxx)"}

请求示例

curl -X POST -d @data_calling.json IP:7999/baidu/5gai/v1/calling-detect/detect -H "Host:calling-detect"

其中**data_head.json**是图像存储为json格式文件

正常返回如下json结构，"message"为 "ok"。失败时"message"显示错误信息

参数	类型	说明
objects	list	检测到的目标数组
message	string	返回信息

objects 数据结构如下

	类型	是否可缺省	意义
x	int	否	横坐标
y	int	否	纵坐标
w	int	否	框宽
h	int	否	框高
text	string	是	"打电话:confidence" 备注confidence是置信度
bcolor	string	是	框颜色，以8bit的RGB表示,默认"0xFFFF00"
font	int	是	字体大小(像素)，默认24

例如：

```
 {
  "objects": [
   {"x":1, "y":1, "w":100, "h":100, "text": "打电话:0.91", "bcolor": "0xFFFF00", "font": 24 },
   {"x":1, "y":1, "w":100, "h":100, "text": "打电话:0.89", "bcolor": "0xFFFF00", "font": 24 }
  ],
  "message": "ok"
}
```

人脸算法

查询用户组列表

接口说明

查询用户组列表（可分页查询），按照创建时间倒序排列

请求URL

/baidu/5gai/v1/face-api/face-set/group/getlist [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
pageNo	int	否	页码（默认为1）
pageSize	int	否	每页记录数（默认为20，最大为200）

请求示例

```
 {
   "pageNo":1,
   "pageSize":20
}
```

返回说明

参数名称	类型	是否必填	说明
Message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码
Data	String/JsonObjects	是	失败时返回失败原因，成功时返回数据list
+totalCount	int	是	总记录数
+totalPage	int	是	总页数
+pageNo	int	是	页码
+pageSize	int	是	每页记录数
+group_list	jsonArray	是	用户组列表
++groupId	String	是	用户组ID
++groupName	String	是	用户组名称

返回示例

```
 {
   "message":"success",
"status": 200,
"data":{
       "totalCount":2,
       "totalPage":1,
       "pageNo":1,
       "pageSize":20,
       "group_list":[
          {
             "groupId":"1",
             "groupName":"TEST"
          },
          {
             "groupId":"pig_face",
             "groupName":"小猪脸组"
          }
       ]
   }
}
```

创建用户组

接口说明

用于创建用户组

请求URL

/baidu/5gai/v1/face-api/face-set/group/add [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
groupId	String	是	用户组ID，标识一组用户组（由数字、字母、下划线组成，长度：1~48（数据库长度为1024））
groupName	jsonArray	是	用户组名称（长度：1~32（数据库长度为1024））

请求示例

```
 {
  "groupId":"pig_user",
   "groupName":"小猪组"
}
```

返回说明

参数名称	类型	是否必填	说明
Message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码
Data	String	是	返回数据，成功为空字符串,失败为失败原因

返回示例

```
 {
   "data:"",
   "message":"success",
   "status": 200
}
```

删除用户人脸

接口说明

删除用户人脸图片（删除用户的人脸）

请求URL

/baidu/5gai/v1/face-api/face-set/face/del [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
groupId	String	是	用户组ID，标识一组用户组（由数字、字母、下划线组成，长度：1~48（数据库长度为1024））
userId	String	是	用户ID（由数字、字母、下划线组成，长度1~20（数据库长度为1024））
faceId	Int	是	人脸id

请求示例

```
{
  "groupId":"pig_face",
  "userId":"pig_2",
  "faceId":1
}
```

返回说明

参数名称	类型	是否必填	说明
Data	String	是	返回数据，成功为空字符串,失败为失败原因
message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码

返回示例

```
{
  "data:"",
  "message":"success",
  "status": 200
}
```

添加用户人脸

接口说明

增加用户的人脸图片信息

请求URL

/baidu/5gai/v1/face-api/face-set/face/add [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
image	String	是	用户图片信息（总数据大小应小于10M），图片的base64值，base64编码后的图片数据（编码后的图片大小不超过2M）
groupId	String	是	用户组ID，标识一组用户组（由数字、字母、下划线组成，长度：1~48（数据库长度为1024））
userId	String	是	用户ID（由数字、字母、下划线组成，长度1~20（数据库长度为1024））

请求示例

```
{
  "image":"/9j/4AAQSkZJRgABAQAAAQABAADs...",
  "groupId":"pig_face",
  "userId":"pig_1"
}
```

返回说明

参数名称	类型	是否必填	说明
data	String	是	返回数据，成功为空字符串,失败为失败原因
message	String	是	成功时success，失败 failed
status	Int	是	返回码，成功为200，失败为http错误码

返回示例

```
 {
   "data:"",
   "message":"success",
   "status": 200
}
```

查询用户列表

接口说明

查询用户列表（可分页查询）

请求URL

/baidu/5gai/v1/face-api/face-set/group/getuserlist [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
groupId	String	是	用户组ID，标识一组用户组（由数字、字母、下划线组成，长度：1~48（数据库长度为1024））
pageNo	int	否	页码（默认为1）
pageSize	int	否	每页记录数（默认为20，最大为200）

请求示例

```
 {
   "groupId":"pig_face",
   "pageNo":1,
   "pageSize":20
}
```

返回说明

参数名称	类型	是否必填	说明
Message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码
Data	String/JsonObjects	是	失败时返回失败原因, 成功时返回数据list
+totalCount	int	是	总记录数
+totalPage	int	是	总页数
+pageNo	int	是	页码
+pageSize	int	是	每页记录数
+user_list	jsonArray	是	用户列表
++userId	String	是	用户ID
++userName	String	是	用户名称

返回示例

```
 {
   "message":"success",
   "status": 200,
   "data:{
      "totalCount":2,
      "totalPage":1,
      "pageNo":1,
      "pageSize":20,
      "user_list":[
         {
            "userId":"pig_1",
            "userName":"小猪1号"
         },
         {
            "userId":"pig_2",
            "userName":"小猪2号"
         }
      ]
   }
}
```

查询用户人脸列表

接口说明

查询用户人脸列表

请求URL

/baidu/5gai/v1/face-api/face-set/face/getlist [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
groupId	String	是	用户组ID，标识一组用户组（由数字、字母、下划线组成，长度：1~48（数据库长度为1024））
userId	String	是	用户ID（由数字、字母、下划线组成，长度1~20（数据库长度为1024））

请求示例

```
 {
   "groupId":"pig_face",
   "userId":"pig_1",
}
```

返回说明

参数名称	类型	是否必填	说明
Message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码
Data	String/objects	是	失败时返回失败原因，成功时返回[1,2,3 …]

返回示例

```
 {
   "message":"success",
   "status": 200，
   "data": [1,2]
}
```

查询全量用户组

接口说明

查询所有用户组

请求URL

/baidu/5gai/v1/face-api/face-set/group/getall [POST]

请求格式

header应为Content-Type:application/json

返回说明

参数名称	类型	是否必填	说明
Message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码
Data	String/JsonObjects	是	失败时返回失败原因, 成功时返回数据list
+groupId	String	是	用户组ID
+groupName	String	是	用户组名称

返回示例

```
 {
  "message":"success",
"status": 200,
"data":[
      {
         "groupId":"1",
         "groupName":"TEST"
      },
      {
         "groupId":"pig_face",
         "groupName":"小猪脸组"
      }
   ]
}
```

注册用户

接口说明

在用户组中新增用户信息，并保存用户人脸图片

请求URL

/baidu/5gai/v1/face-api/face-set/user/add [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
userId	String	是	用户ID（由数字、字母、下划线组成，长度1~20（数据库长度为1024））
userName	String	是	用户名称（长度：1~32（数据库长度为1024））
Image	String	是	用户图片信息（总数据大小应小于10M），图片的base64值，base64编码后的图片数据（编码后的图片大小不超过2M）
groupId	String	是	用户组ID，标识一组用户组（由数字、字母、下划线组成，长度：1~48（数据库长度为1024））

请求示例

```
 {
   "userId":"pig_1",
   "userName":"小猪1号",
   "image":"/9j/4AAQSkZJRgABAQAAAQABAADs...",
   "groupId":"pig_face"
}
```

返回说明

参数名称	类型	是否必填	说明
Message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码
Data	String	是	返回空字符串

返回示例

```
 {
   "message":"success",
   "status": 200,
   "data:""
}
```

删除用户

接口说明

用于删除用户

请求URL

/baidu/5gai/v1/face-api/face-set/user/del [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
groupId	String	是	用户组ID，标识一组用户组（由数字、字母、下划线组成，长度：1~48（数据库长度为1024））
userId	String	是	用户ID（由数字、字母、下划线组成，长度1~20（数据库长度为1024））

请求示例

```
 {
   "groupId":"pig_face",
   "userId":"pig_2",
}
```

返回说明

参数名称	类型	是否必填	说明
Message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码
Data	String/objects	是	返回空字符串

返回示例

```
 {
   "message":"success",
   "status": 200，
   "data:""
}
```

删除用户组

接口说明

用于删除用户组

请求URL

/baidu/5gai/v1/face-api/face-set/group/del [POST]

请求格式

header应为Content-Type:application/json

请求说明

参数名称	类型	是否必填	说明
groupId	String	是	用户组ID，标识一组用户组（由数字、字母、下划线组成，长度：1~48（数据库长度为1024））

请求示例

```
 {
   "groupId":"pig_face",
}
```

返回说明

参数名称	类型	是否必填	说明
Message	String	是	成功时success，失败 failed
Status	Int	是	返回码，成功为200，失败为http错误码
Data	String/objects	是	返回数据，成功为空字符串,失败为失败原因

返回示例

```
 {
   "message":"success",
   "status": 200,
   "data:""
}
```

人脸识别

接口说明

检测人脸位置并且在数据库匹配人脸

请求URL

/baidu/5gai/v1/face-api/detect [POST]

请求格式

header应为Content-Type:application/json

请求参数

参数名称	类型	是否必填	说明
image	String	是	图像数据，base64编码例如 {"image":"base64(xxxx)"}

正常返回如下json结构，"message"为 "ok"。失败时"message"显示错误信息

参数名称	类型	说明
objects	list	检测到的目标数组
message	string	返回信息

objects 数据结构

参数名称	类型	是否必填	说明
x	int	是	横坐标
y	int	是	纵坐标
w	int	是	框宽
h	int	是	框高
text	string	否	"text": "组={},用户名={},用户id={},匹配度={}
bcolor	string	否	框颜色，以8bit的RGB表示,默认"0x008000"
font	int	否	字体大小(像素)，默认30

请求示例：

```
 {
   "objects": [
   {"x":1, "y":1, "w":100, "h":100, "ttet": "组=男，用户名=岳云鹏，用户id=2,匹配度=0.981, "bcolor": "0x008000",
"font": 30 }
  ],
   "message": "ok"
}
```