MLMtoken 可信分控策略网关

设备管理配置说明

——简称：边缘 TokenBoss 盒子——

v 1.30

==========

目 录

CONTENTS

==========

一、系统概况.....2

二、API 网关.....5

三、网络配置.....35

四、无线配置 【*仅带无线 WIFI 的设备型号支持】.....37

五、定向加速.....38

六、智能组网.....40

七、网络诊断.....41

八、设备信息.....43

九、账号与安全.....45

一、系统概况

1.1 登录管理后台

网关设备通电联网后，电脑在浏览器地址栏输入后台地址即可进入管理后台。

配置项	默认值/说明
后台访问地址	192.168.137.1
登录账号	admin
初始密码	admin123（登录后可自行修改）

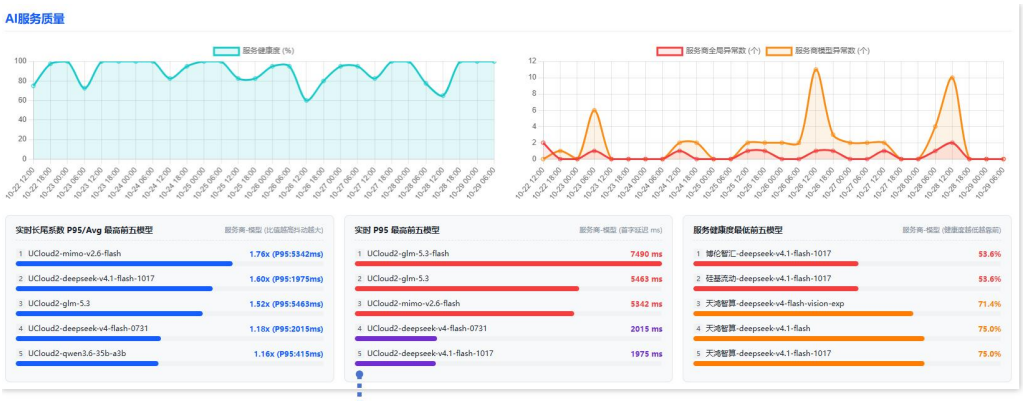
1.2 系统概况

登录后首先展示系统概况模块，用于查看 AI 大模型的运营统计以及网关的运行状态与网络参数。

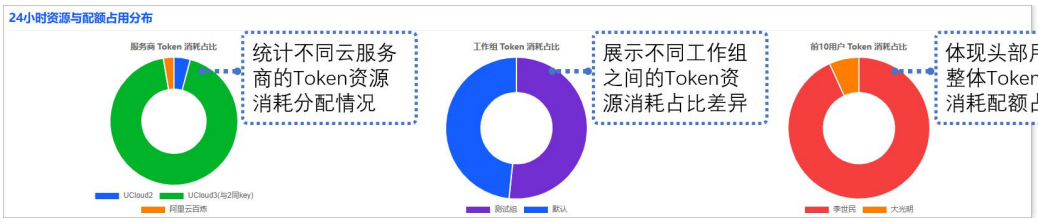
1.2.1 大模型网关运营看板



总请求次数：	过去 24 小时网关累计转发的 AI 请求总数（成功 + 被拦截）
消耗 Token 总计：	24 小时内消耗的 Token 总量，拆分输入（prompt）与输出（response）两部分
安全与准入拦截数：	敏感词过滤、模型白名单、越权调用等被网关拦截的请求数
Token 超额告警：	命中工作组 / 成员配额阈值而触发告警或阻断的次数
当前连接数：	此刻正在处理的并发连接数（实时指标）
活跃用户数：	过去 24 小时内有调用行为的坐席
服务健康度：	上游服务商整体可用率

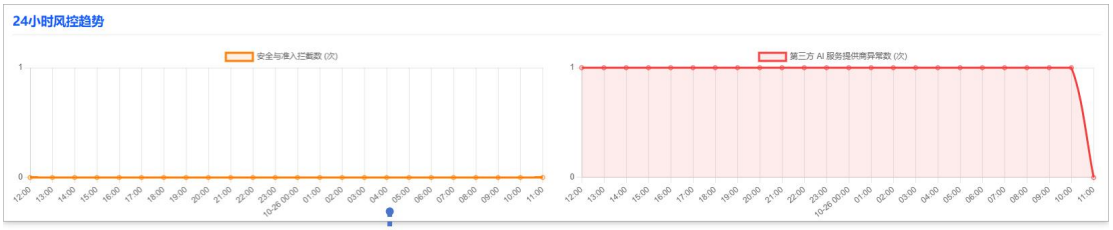


服务健康度：	健康度会更具权重将成功请求占比 / 可用上游占比，越接近 100% 越好
异常数统计：	服务商全局异常：整个服务商不可用（403、欠费、超时、宕机）的次数； 服务商模型异常：该服务商下某个具体模型报错的次数
P95/Avg 前五模型：	榜单列出抖动最严重的 5 个服务商 - 模型
P95 最高前五模型：	榜单列出延时最严重的 5 个服务商 - 模型
健康最低前五模型：	榜单列出健康度低的 5 个服务商 - 模型





面向大模型API服务的运行状态监控四宫格折线图，完整覆盖请求流量、Token消耗、请求速率、Token吞吐四大核心运行指标，直观呈现全天的资源波动规律，精准定位流量峰值时段。



安全与准入拦截数：　敏感词过滤、模型白名单、越权调用等被网关拦截的次数

第三方AI服务提供商异常：　上游服务商API调用时请求异常的频次



显示服务商、敏感词、Token配额、越狱攻击、未授权访问等报警明细

1.2.2 系统状态


系统概览
自动刷新：开启
运行时间：0天 05时 24分
CPU 温度：47.5 ℃
9.1%
CPU 使用率
17.4%
内存使用率
已用171M / 总985M
9%
存储使用率
已用183M / 总2003M
网口链路状态：
LAN
WAN
1000M
网络信息
LAN IP：192.168.137.1
WAN 协议：dhcp
2.4G WiFi：netbooster_2.4G 信道：auto
5G WiFi：netbooster_5G 信道：auto
DHCP 已分配设备
192.168.137.118 | 10:91:d1:c0:d4:d8 | CHINAMI-4RSKIFJ
192.168.137.125 | 00:e0:4c:68:3d:d6 | Thinkpad-wzX1
192.168.137.228 | 56:e8:2b:34:2e:27 | OPPO-Reno10-5G
无线接入设备
phy1-ap0 | 56:e8:2b:34:2e:27
phy1-ap0 | 10:91:d1:c0:d4:d8
可查看设备运行时间、CPU 温度、资源占用率及网口链路状态。
可查看设备 LAN、WAN、WiFi 参数。
查看加速宝 DHCP 分配的设备。
显示无线接入的终端 mac 信息。

二、API 网关

API 网关用于对接上游第三方 AI 服务厂商，将上游 AI 接口统一接入网关，再对内网下游用户分发代理 API Key，实现 AI 接口统一代理、密钥隔离、调用统计、用量管控。

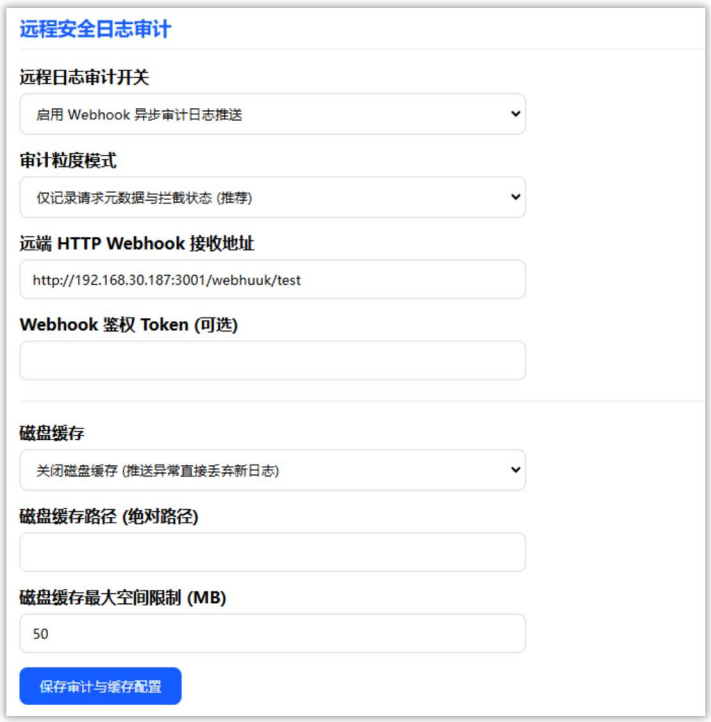
2.1 API 网关

2.1.1 API 网关服务状态



配置项	默认值/说明
服务运行状态	开启 / 关闭 API 网关转发服务。
内网发布 URL	网关本地监听地址。 内网下游客户端，访问网关机器 IP+端口 进行 AI 接口调用
数据保存间隔配置	默认 24 小时。用于控制网关数据的聚合采样频率。 【统计报表】中能够直观体现
数据最长保存时间	默认值为 1 年（365 天），网关会自动按配置周期清理超期的历史运行数据，避免存储资源被无效占用

2.1.2 远程安全日志审计



作用：把 API 调用的内容日志异步推送到远端安全审计平台。

配置项	默认值/说明
远程日志审计开关	启用 / 关闭 Webhook 异步审计日志推送
审计粒度模式	仅记录请求元数据与拦截状态（推荐） 完全记录日常所有对话提问与回答全文 ① 只记元数据（调用方、时间、模型、Token 数、是否被拦截）， 隐私友好、体积小； ② 全量记录（含请求 / 响应内容），排查问题更全，但隐私和存储成本高
远端 HTTP Webhook 接收地址	接收日志的远端服务器地址
Webhook 鉴权 Token（可选）	远端校验用的 Bearer Token
磁盘缓存开关	远端暂时不可达时的兜底策略： 关闭：推不出去就直接丢，不占本地磁盘； 开启：先落盘缓存，等远端恢复后再补推，不丢日志

配置项	默认值/说明
磁盘缓存路径（绝对路径）	缓存日志在本机的存放路径
磁盘缓存最大空间限制（MB）	缓存占用磁盘上限，写满后按策略淘汰（防撑爆磁盘）

点底部 【保存审计与缓存配置】 生效

第三方 Webhook 日志接收服务的开发指南详见 **第 2.7 章节**

2.2 AI 服务商管理

2.2.1 故障切换与负载策略

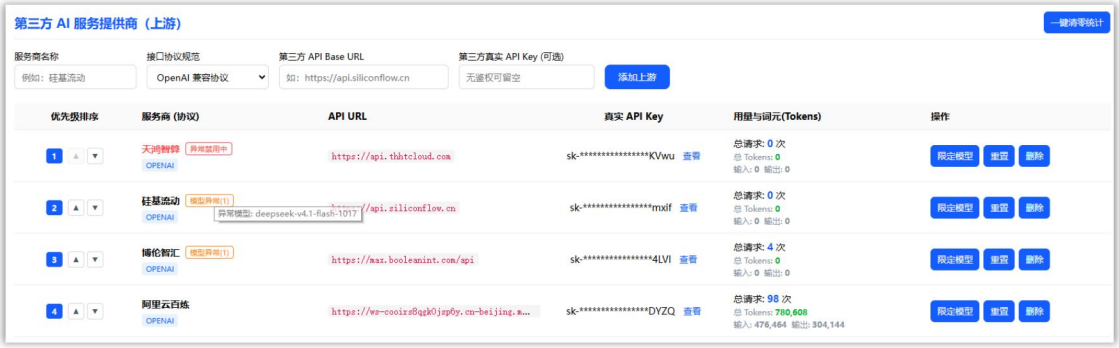


作用：决定多上游服务商时 "按优先级顺序主备切换" 还是 "按负载均衡模式分流"，以及上游故障时 "要不要临时隔离"、隔多久自动探活恢复。*探测到存在故障的上游服务商系统会用不同颜色标记。

配置项	默认值/说明
负载模式	主备模式（默认）：按服务商列表的优先级顺序（▲/▼ 调整）首选主用服务商，主用故障时自动顺延切到下一个备用服务商 负载均衡模式：每个上游服务商按负载均衡进行分流，主备不分先后
故障处置	连接异常禁用（默认）：异常服务商在设定秒数内暂停调度，流量自动漂移到其他健康服务商，期满后台自动探活、恢复服务 异常不禁用：服务商异常时只标记状态、触发报警，但不隔离，后续请求仍访问到它；按设定秒数后自动取消异常标记
探活周期	定义异常服务商被标记后，多久可再重新探测
首字超时（TTFT）与重试	请求发出后的设定时间内未收到模型反馈即判定超

配置项	默认值/说明
	时，超时后自动在池内其他健康上游重试 N 次（设定的次数），避免慢响应拖垮用户体验
QoS 劣化检测	系统自动学习该模型 / 服务商的正常 TTFT 基线，当 P95 首字延迟达到基线 2.5 倍时，判定为 "性能劣化"，触发报警并按故障策略隔离 / 切流

2.2.2 第三方 AI 服务提供商（上游）



作用：录入真实 AI 服务商接口地址与原始 API Key，网关以此作为上游转发目标。

配置项	默认值/说明
服务商名称	自定义名称，用于区分不同 AI 源
接口协议规范	备选项为 OpenAI 兼容协议 / Anthropic 原生 / Google Gemini
第三方 API	AI 服务商接口地址，兼容 OpenAI 协议接口
第三方 API Key	AI 服务商给到的原始密钥（不直接暴露给内网用户）
模型异常标记	监测到异常的模型，鼠标移动到异常标记上会直观看到哪些模型有问题

操作步骤：

1. 依次填入服务商名称、第三方 API Base URL、第三方真实 API Key、选择接口协议规范。

2. 点击【添加上游】，添加后出现在服务商列表。

列表操作：

- 优先级排序：▲/▼ 调整主备服务商的优先级顺序。

- 查看：点击查看真实 API Key，可以复制。
- 限定模型：设置允许对外提供的模型白名单，避免越权调用。



点击【限定模型】，系统会获取上游服务商 API 中提供的所有模型列表，通过选择可以设置白名单，避免下游或模型池的越权使用。全不选择定为所有模型可用。

命中过的模型系统会将最新的 P95 监测指标及质量标记在对应的模型上。方便管理员定位查看

- 重置：重置用量与词元统计。
- 删除：移除上游 AI 服务商（下游已绑定该服务商的分发 Key 将不可使用）。
- 用量与词元（Tokens）：统计上游服务商总请求次数、总消耗 Token 数量。
- 一件清零统计：重置全部服务商的请求计数、Token 消耗统计，不会删除配置，不会清除图表统计数据。

注意：支持多上游 AI 源同时配置，内网可选择不同上游服务商做转发。

2.2.3 模型池管理

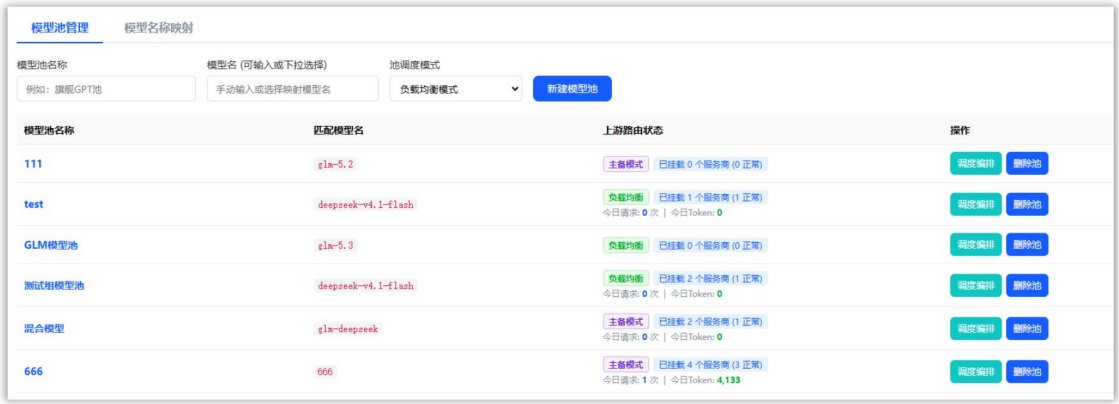
模型池是网关对上游算力进行逻辑聚合，其聚合对象分两种情况：

1. 同名直聚：多家上游提供同一模型名（如 glm-5.2、deepseek-v4.1-flash）时，直接把这些上游服务商归到同一个池；

2. 映射聚合：通过 "模型名称映射" 规则，把不同上游名称不同的多个真实模型名（如 deepseek-v4.1-flash 与 deepseek-v4-flash-0731）统一映射为一个对外别名（如 deepseek-v4.1），再把这些对应上游服务商归入同一池。

调用与调度逻辑：

- 下游客户端只需指定模型池对外暴露的统一模型名 8 即可调用，无需感知后端具体是哪家服务商；
- 网关在池内多家上游之间自动完成路由选择、加权负载均衡与故障切换；
- 探测到故障的上游，系统会用不同颜色在列表中标记，便于运维一眼识别。



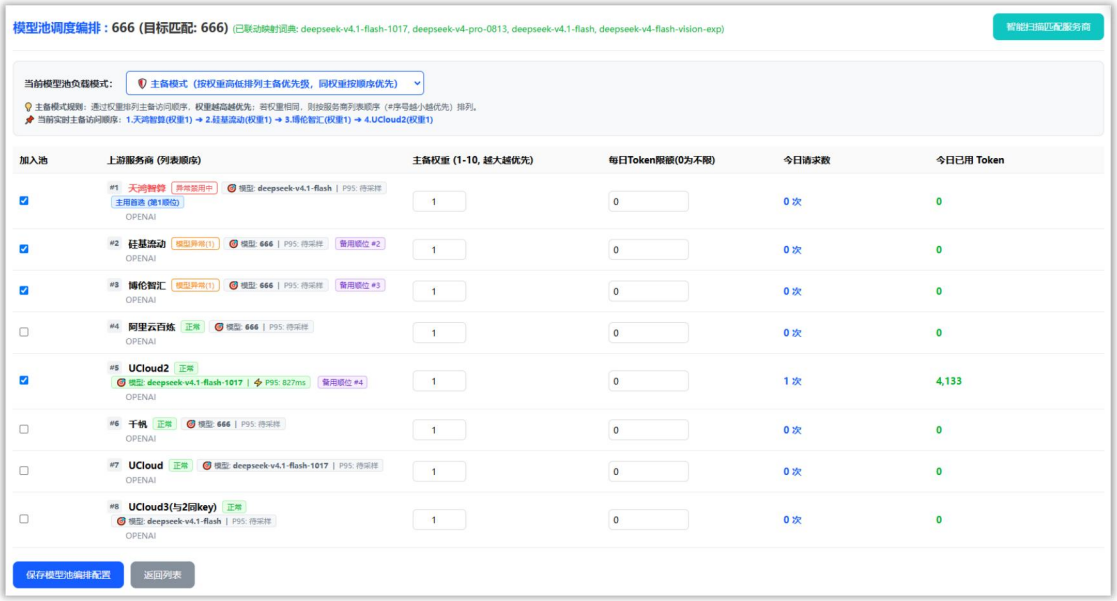
配置项	默认值/说明
模型池名称	自定义一个名称，可给下游分发引用
模型名	下游调用时使用的统一模型名 *多家上游提供同一模型名或映射聚合后的对外别名
池调度模式	主备模式（权重优先）：调度编排中的权重优先级顺序，首选主用服务商，主用故障时自动顺延切到下一个备用服务商 负载均衡模式：每个上游服务商按负载均衡进行分流，主备不分先后

操作步骤：

1. 依次填入模型池名称、模型名。
2. 点击【新建模型池】，添加后出现在列表。

列表操作：

- 调度编排：进入该池，配置池内挂载上游服务商及权重。

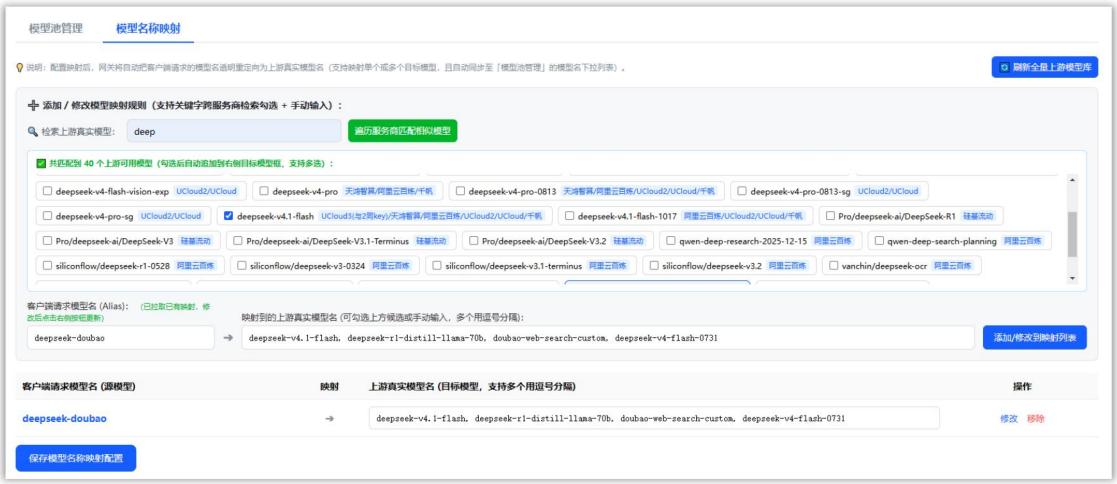


配置项	默认值/说明
智能扫描匹配服务商	自动探测并勾选上游支持当前目标模型，快速推荐可挂载的服务商
加入池	勾选：该上游参与此模型池的转发；不勾： 不参与。
并发负载策略	按权重分配流量，权重越大分到的请求越多；默认都为 1，即平均分摊。
每日 Token 限额	给该上游服务商这个模型池里设置每日 Token 消耗上限，0 为不限，次日自动恢复
今日请求数	展示下游当日通过模型池请求服务商的次数统计
今日已用 Token	展示上游当日在本模型池下已经消耗的 Token 量
保存模型池编排配置	勾选、权重、限额改完后点保存生效
上游服务商标签说明	异常禁用中（红）：该服务商当前被故障策略隔离，不接请求 模型异常（n）（橙）：该服务商池中的模型有 n 个存在异常 正常（绿）：服务商健康可用 主用首选（第 1 顺位）/ 备用顺位 #2/#3/#4：主备模式设置权重后会显示调用的顺序 P95：待采样 / ***ms：实时 P95 延迟；新节点数据不足时显示 "待采样"

- 删除池：移除该模型池。

2.2.4 模型名称映射（*需配合模型池使用）

作用：不同上游服务商对模型模型的名称定义可能不同（UCloud 叫 deepseek-v4.1-flash，百度叫 deepseek-v4-flash-0731）。名称映射就是把这些五花八门的真名，统一映射成客户端友好的别名，网关通过模型池将别名自动做透明重定向。



配置项	默认值/说明
刷新全量上游模型库	当上游服务商的限定模型出现变更后，需点击重新拉取所有上游最新的模型列表
检索上游真实模型	输入关键字（如 deepseek），可遍历服务商匹配相似模型。
匹配模型（勾选）/ 映射到的上游真实模型名	勾选匹配的模型后会自动追加到“映射到的上游真实模型名”输入框中
客户端请求模型名	给所勾选的真实模型名统一定义一个友好的别名 *用此名创建模型池后，可分发给下游给客户端使用

操作步骤：

1. 检索上游真实模型，输入关键字（如 deepseek-v4.1），点 【遍历服务商匹配相似模型】；

2. 自动匹配候选，跨所有上游搜索匹配模型，列出可用模型及所属服务商；

3. 勾选所要统一映射的模型名称，系统自动追加到“映射到的上游真实模型名”输入框中；

4. 在“客户端请求模型名”输入框中定义一个友好的别名；

5. 点 【添加 / 修改到映射列表】，自动新增至列表中，最后点左下 【保存模型名称映射配置】完成映射配置；

6. 切换到【模型池管理】，新建模型池，模型名选择自定义的映射别名。

7. 通过模型池的【调度编排】筛选服务商，定义权重后完成配置

8. 在【内网用户管理】菜单分发这个模型池后，用户就可以通过 workbuddy 或 cherrystudio 等客户端获取到这个映射别名。

9. 客户端在使用这个映射别名时，网关会自动进行调度，实现下游零感知、上游多源容灾。

2.3 内网用户管理

2.3.1 用户故障诊断/配置定位



作用：可以帮管理员快速定位分发用户的配置，以及追踪服务调用过程中出现的网关错误、服务不可用等异常情况。

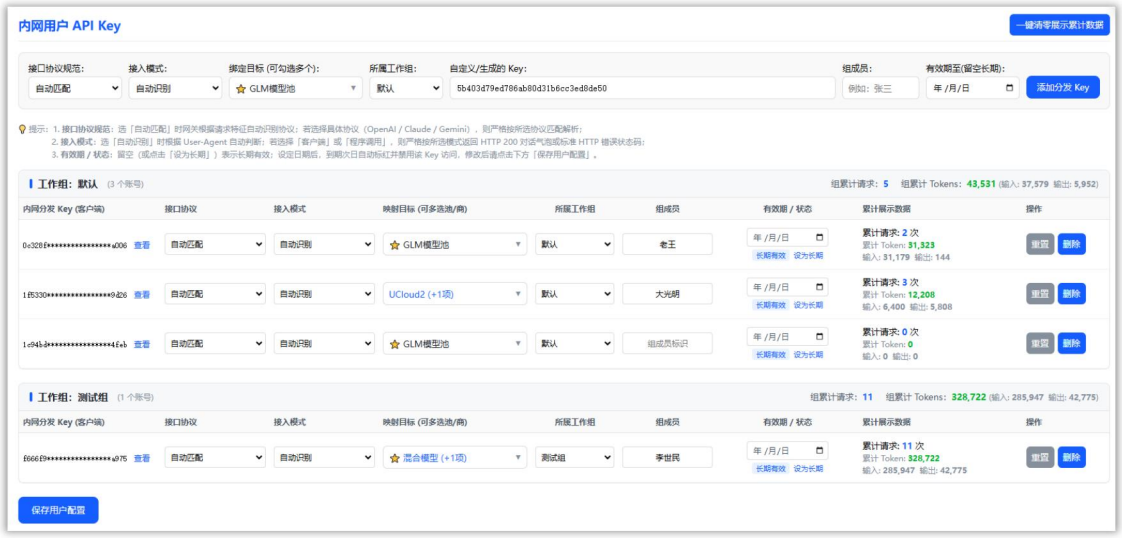
操作步骤：

1. 选择需要查询的用户。

2. 点击【快速诊断】，自动显示该用户 24 小时内发生的调用异常明细。

3. 点击【配置定位】，页面自动切到该用户的配置项。

2.3.2 内网用户 Key（下游）



作用：生成内网客户端代理密钥，内网应用无需持有上游真实密钥，仅使用网关分发 Key 访问网关，由网关代为转发请求。

配置项	默认值/说明
接口协议规范	备选项为自动匹配（默认） / OpenAI 兼容协议 / Anthropic 原生 / Google Gemini 网关根据请求特征自动识别协议；也可强制指定 OpenAI / Claude / Gemini
接入模式	备选项为自动识别（默认） / 客户端 / 程序调用 网关根据请求自动识别，也可强制定义只能接客户端 / API 调用
绑定目标（可多选）	下拉选择已经添加好的上游服务商 / 模型池，支持多选
所属工作组	选择工作组（工作组在 API 网关高级中维护），用于配额和统计
自定义/生成的 Key	内网客户端 / 程序调用使用的代理密钥，真实上游 Key 不下发
组成员	使用者备注，便于溯源
有效期	设定日期后到期日自动标红并禁用该 Key，留空代表长期

操作步骤：

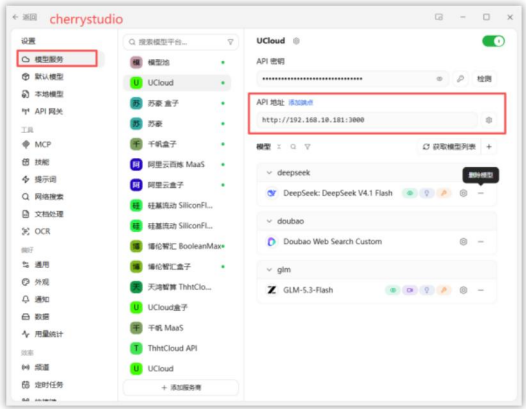
1. 定义接口协议规范及接入模式
2. 下拉选择绑定目标，上游服务商或模型池，支持多选。
3. 选择工作组
4. 设置分发 Key（自动生成或手动填写）。
5. 填写组成员。
6. 设置有效期
7. 点击【添加分发 Key】，新增下游密钥。

列表操作:

- 内网分发 Key：给内网业务、客户端使用的密钥，点击显示查看完整密钥。
- 映射目标：该 Key 绑定的上游 AI 源（服务商及模型池，可多选）。
- 组成员：记录使用人员，支持编辑保存修改。
- 累计展示数据：统计该分发 Key 的请求次数、消耗 Token。
- 重置：重置用量与词元统计。
- 删除：作废该内网分发 Key，客户端将无法继续调用。
- 一键清零所有统计：重置全部下游 Key 的请求计数、Token 消耗统计，不会删除配置，不会清除图表统计数据

注意：客户端（WorkBuddy 、 Cherry Studio 等）只填写内网分发 Key，不要填写上游真实 API Key，网关内部自动替换成上游真实密钥转发。

***WorkBuddy 、 Cherry Studio 的配置截图**

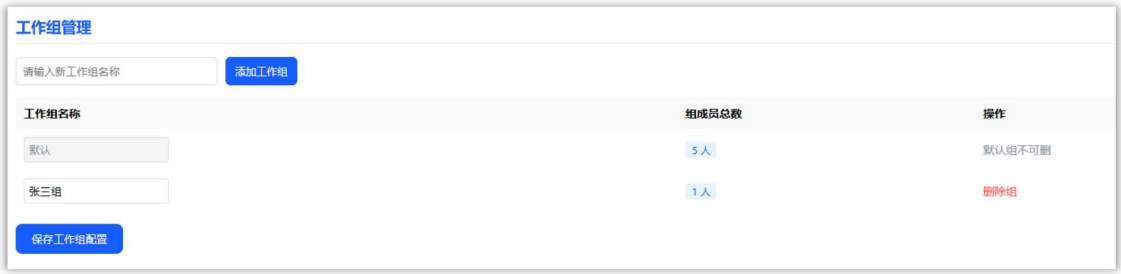




API 网关功能核心逻辑：上游配置真实服务商地址 + 密钥，下游生成内网分发密钥，内网客户端只使用分发 Key 访问网关，网关转发到真实 AI 服务商。

2.4 Token 策略

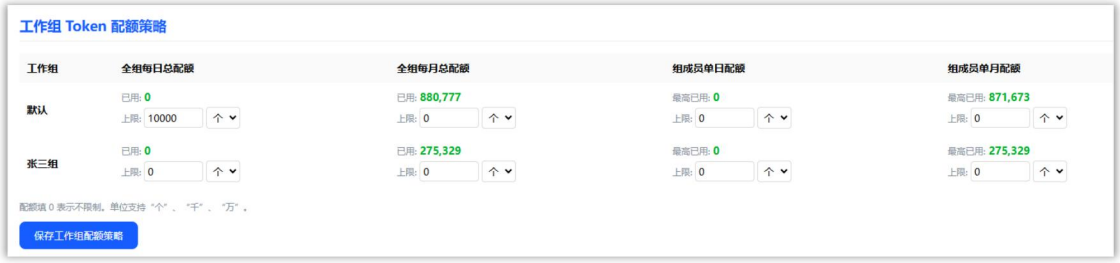
2.4.1 工作组管理



作用：将内网下游使用者（即分发 Key 的持有人）按部门 / 项目分组，便于按组设置 Token 限额、统计用量，实现精细化的团队级管控。

配置项	默认值/说明
输入框 + 【添加工作组】	输入新组名（如 "研发部"" 运营组 "），点添加即可新建工作组
工作组名称	可直接在框内修改组名，点下方 "保存工作组配置" 生效
删除组	默认组显示 "默认组不可删"（系统内置，不能删除）；自建组显示 "删除组"（红字，可删除）

2.4.2 工作组 Token 配额策略



作用：用于按组和按成员设置用量上限。

配置项	默认值/说明
全组每日总配额	整组一天内所有成员加起来最多能用多少 Token，超限当天停服

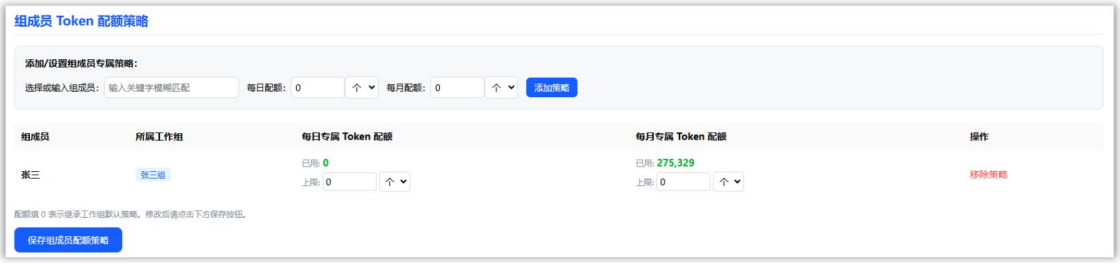
配置项	默认值/说明
全组每月总配额	整组一个月内所有成员加起来最多能用多少 Token，超限当月停服
组成员单日配额	组内单个成员（单个分发 Key）每天的上限
组成员单月配额	组内单个成员（单个分发 Key）每月的上限

列表操作:

- 上限设置：填数字，旁边单位下拉支持 个 / 千 / 万；填 0 代表不限制
- 用量展示：全组维度显示 "已用"（全组累计用量）；成员维度显示 "最高已用"（组内消耗最多的那个成员用了多少）

修改完成后点底部 【保存工作组配额策略】生效

2.4.3 组成员 Token 配额策略



作用: 给某个特定成员（分发 Key）单独设置更严 / 更宽的每日、每月 Token 上限，用于对重点人员、高消耗账号做精细化管控。

配置项	默认值/说明
选择或输入组成员	支持关键字模糊匹配，快速定位某个成员
每日配额	该成员每天上限，单位支持 个 / 千 / 万
每月配额	该成员每月上限，单位支持 个 / 千 / 万

点击【添加策略】加入下方专属策略列表

2.4.4 Token 超额告警与阻断策略



作用：Token 配额命中后的告警与处置方式配置，超额时网关根据策略响应。

配置项	默认值/说明
超额告警提示	开启 / 关闭客户端超额告警
超额阻断机制	只提示不阻断（先回复告警内容，再连接 AI 答案）； 开启超额阻断（仅回复告警，直接切断不连接 AI）
告警提示内容	自定义返回给客户端的告警文案

2.5 AI 安全

2.5.1 双向安全围栏与 DLP 动态脱敏防护



作用：输入 / 输出双向独立检测：DLP 自动识别身份证、手机号、密钥、IP 等隐私数据并打码；自定义敏感词 / 正则表分 "严格阻断" 和 "动态脱敏" 两档，命中即拦或自动掩码，保障企业数据不出域、内容合规。

配置项	默认值/说明
输入与输出检测独立管控	启用提问阻断 / 启用提问动态脱敏：用户发给大模型的内容在转发前网关先做一次安全检测 启用回答熔断 / 启用回答动态脱敏：大模型返回的内容在回给用户前网关过滤一遍，防止违规答案流出
内置隐私数据识别规则（DLP）	自动识别 12 类敏感数据： 个人身份类：二代身份证号、手机号码、电子邮箱 金融账户类：银行卡号、企业统一信用代码 凭证密钥类：API 密钥（sk-/AKIA/Alza/ghp_）、JWT 令牌、数据库连接串、PEM/RSA 私钥证书块 网络地址类：内网 IPv4、互联网 IPv4、MAC 物理地址 **处置模式** 动态脱敏模式： 自动把命中的隐私信息替换为`***`掩码后放行，不中断业务。 严格阻断模式：把命中的隐私信息直接拦截，回复自定义的拦截内容
严格阻断模式 - 敏感词与正则列表	自定义配置需要严格阻断的命中信息。支持正则表达式的编写，格式“regex:表达式”
动态脱敏模式 - 敏感词与正则列表	自定义配置需要脱敏的命中信息。支持正则表达式的编写，格式“regex:表达式” *严格阻断和脱敏中存在相同的命中信息，优先执行严格阻断模式
自定义拦截提示	严格阻断触发后，给用户返回的提示文案可自定义

2.5.2 提示词注入（Prompt Injection）与越狱攻击预警



作用：用 AC 自动机做多模式极速匹配，在请求转发给大模型之前，扫描用户提问是否包含越狱 / 注入特征词，命中即拦截并记录日志。

配置项	默认值/说明
越狱指令与提示词注入特征库	已预置。也可根据实际情况进行调整
触发「越狱攻击」阻断时的客户端拦截提示语：	攻击触发后，给用户返回的提示文案可自定义

2.5.3 组成员接入安全（IP 与设备 MAC 地址准入控制）



作用：给每个组成员（分发 Key）加一道 "设备锁"：只有从指定 IP 或指定 MAC 的设备发来的请求才放行，其他来源一律拦截。即使 Key 被泄露，别人拿 Key 也调不通。

配置项	默认值/说明

配置项	默认值/说明
选择或输入组成员	选中要绑定的成员
自动获取最近连接 IP/MAC	一键读取该成员最近连接日志，反查 ARP/DHCP 表，勾选后自动填好 IP 和 MAC，无需手动抄 *自动获取示例：IP：192.168.10.131 ｜ MAC：66:0c:78:05:30:2d（最近连接：2026-10-26 10:30:06）
绑定访问 IP	允许调用的 IP，可手填，多个用逗号分隔
绑定设备 MAC	允许调用的设备 MAC，可手填，多个用逗号分隔
接入安全验证开关	启用设备 IP/MAC 准入校验
未授权设备拦截提示	自定义非绑定设备调用时返回给客户端的提示

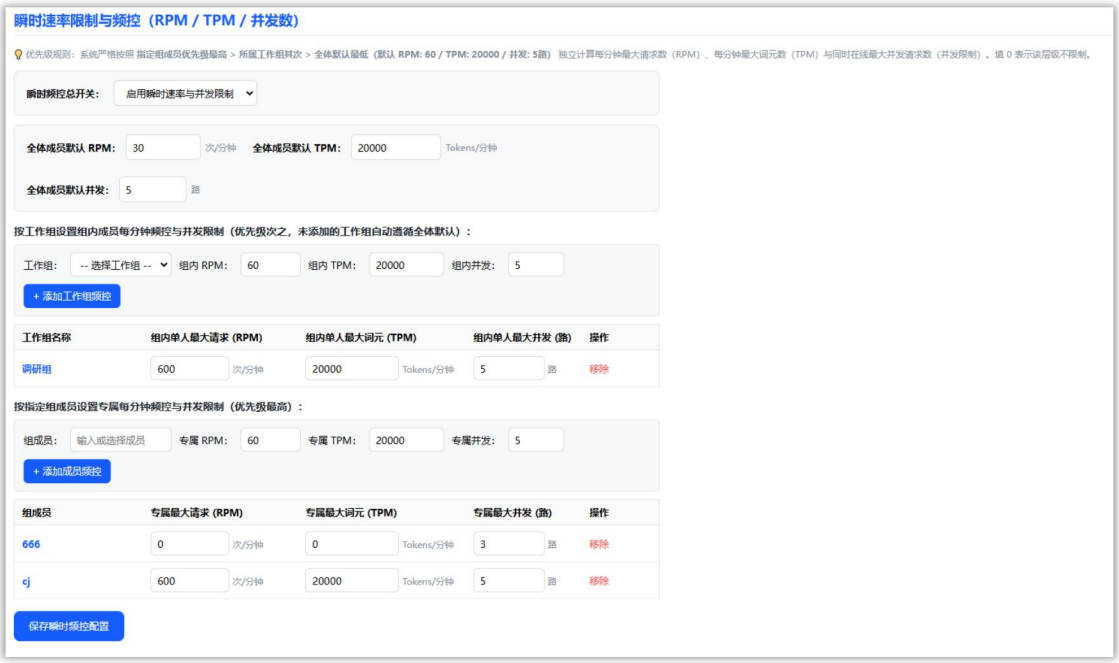
操作步骤：

1. 选择或输入组成员，点击【自动获取最近连接 IP/MAC】；

2. 系统会反查该成员最近连接时所使用的设备 IP 和 MAC；

3. 勾选后（支持多选）会自动赋值至“绑定访问 IP”和“绑定设备 MAC”输入框中，点击【添加到绑定列表】，自动新增至列表中，最有点击左下【保存组成员接入安全配置】完成配置。

2.5.4 瞬时速率限制与频控（RPM / TPM / 并发数）



作用：瞬时速率限制（ RPM / TPM / 并发数），控制每分钟请求数、 Token 数以及并发数，防止突发打满上游额度或被刷接口。

配置项	默认值/说明
RPM	每分钟最大请求次数，0 为不限制 *超过阈值的请求会被网关直接限流拒绝。
TPM	每分钟最大 Token 数（输入 + 输出合计），0 为不限制 *超过阈值的请求会被网关直接限流拒绝。
并发数	设置最大并行请求数，避免短时间内大量请求同时发起，导致服务端压力过载
三级优先级规则	① 指定组成员专属频控（优先级最高） ↓ 若该成员没设专属，则看所属组 ② 所属工作组频控（优先级次之） ↓ 若该组没设，则看全体默认 ③ 全体成员默认（最低兜底）
三层配置项	全体成员频控：没单独配置成员和工作组的使用这个策略 工作组频控：给某个组单独设更严的频控上限（该组中有成员单独配置的除外） 成员专属频控：给重点人物单独设专属限制，最高优先级

操作步骤：

- 添加工作组 / 成员频控：选 工作组 / 成员 → 填专属 RPM/TPM/并发数 → 点 【添加成员频控】。
- 移除：删掉某工作组 / 成员的专属频控，回落到工作组 / 全体成员默认。
- 总开关：可一键 【关闭瞬时速率限制】，不做 RPM/TPM/并发数 检查。

2.5.5 传输层与通信防重放攻击



作用：防止攻击者通过抓包拿到请求后，原封不动重放一次来冒充合法用户。通过【时间戳 + Nonce + HMAC 签名】三要素解决

配置项	默认值/说明
标准模式	仅校验静态 Bearer Token 兼容 Workbuddy、Cherry Studio 等第三方客户端 适用普通办公人员
高安全 HMAC 模式	强制校验 X-Timestamp、X-Nonce、X-Signature 动态签名 调用端需按约定算法对请求签名 适用高安全要求成员 / 程序调用 *签名流程：密钥对 "时间戳 + Nonce + 请求体" 做 HMAC 计算 → 网关同算法校验 → 时间戳过期或 Nonce 重复即拒绝。
防重放总开关	开启 / 关闭传输层防重放控制
时间戳与 Nonce 防重放窗口	时间戳偏差超过所设置的时间视为过期；Nonce 在窗口内不允许重复使用
当前勾选赋予模式	可以全员切换 “高安全 HMAC 签名模式” 或 “标准模式”
成员勾选区	支持单个成员或工作组的模式配置

2.5.6 请求体大小与 JSON 嵌套深度硬限制（Payload Guard）

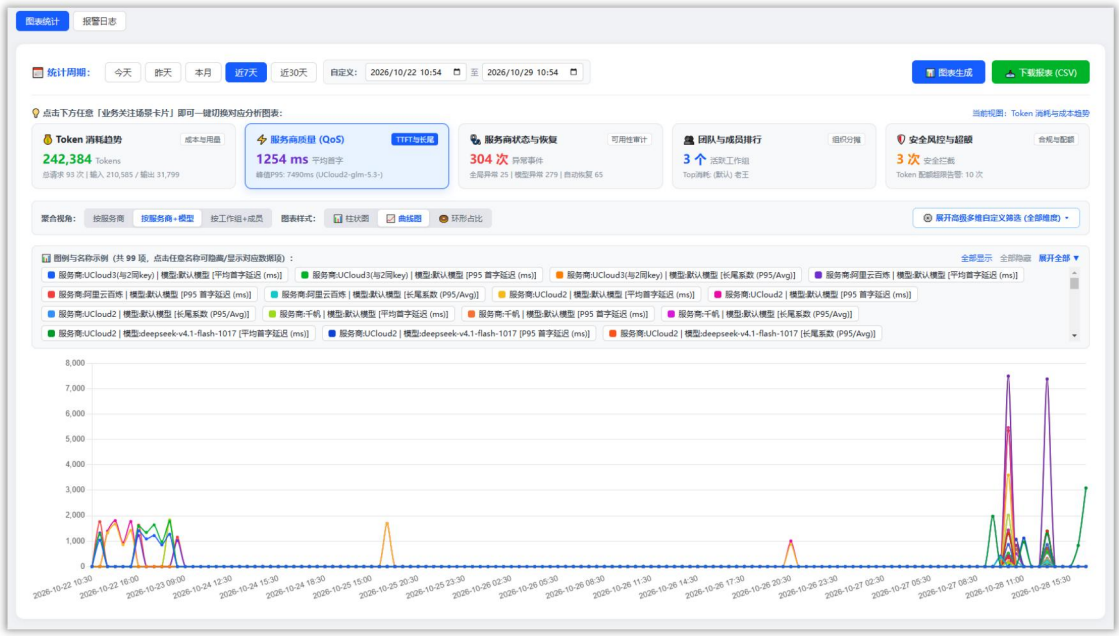


作用：对单次请求体大小（默认 10MB）和 JSON 嵌套深度（默认 32 层）做硬限制，超限即拒并记 PG-01/PG-02 报警，保护网关自身稳定性。

配置项	默认值/说明
两个硬限制	单次请求体最大限制：单次 POST 请求体超过设定值直接拒绝。 JSON 最大嵌套深度：请求 JSON 嵌套超过设定层直接拒绝（防炸弹对象）
Payload Guard 开关	启用 / 关闭硬限制防护。
设定模式	含常规业务，图文处理等，选择后系统会自动分配限制值。自定义模式支持按需调整 ● 常规业务：纯文本聊天、办公问答 ● 图文处理：带图片的多模态对话（base64 图片） ● 软件开发：长 Prompt、代码上下文、文件摘要 ● 媒体制作：音视频上传、大文件处理 ● 自定义：手动填请求体上限与 JSON 嵌套层数
超限报警	● 大小超限：报警类型 **PG-01** ● 深度超限：报警类型 **PG-02**

2.6 图表统计

2.6.1 图表统计

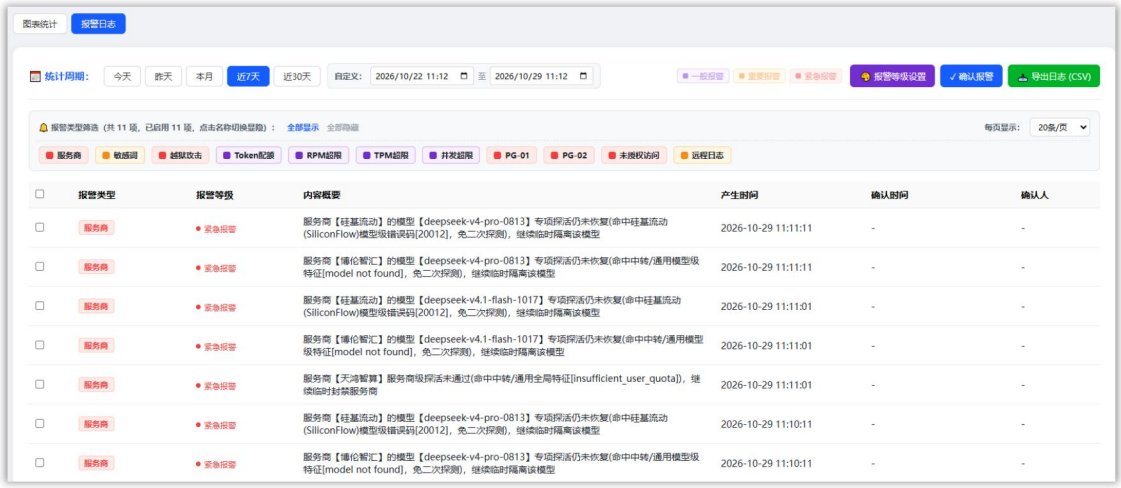


作用：一张看板覆盖成本、QoS、可用性、组织分摊、安全风控五大主题，支持时间 / 服务商 / 模型 / 团队多维聚合与三种图表样式，配合图例显隐和 CSV 导出，支撑成本核算、供应商优化与合规审计。

配置项	默认值/说明
时间与操作	● 统计周期： 今天 / 昨天 / 本月 / 近 7 天 / 近 30 天 / 自定义起止时间 ● 图表生成：按当前筛选条件刷新出图 ● 下载报表（CSV）：导出明细数据做离线分析
六大业务场景卡片	● Token 消耗趋势：成本与用量趋势分析 ● 服务商质量 QoS：TTFT 与长尾延迟分析 ● 服务商状态与恢复：可用性审计 ● 团队与成员排行：组织分摊、按人核算 ● 安全风控与超额：合规与配额审计
聚合视角与图表样式	● 聚合视角：按服务商 / 按服务商 + 模型 / 按工作组 + 成员 ● 图表样式：柱状图 / 曲线图 / 环形占比 ● 高级筛选：展开全部维度（统计指标、服务商、

配置项	默认值/说明
	模型、工作组、成员）自定义勾选统计

2.6.2 报警日志



作用：网关在对服务商 API 调用时会进行深度探测，发现异常时会生成报警记录，便于用户集中查询各类报警明细。

配置项	默认值/说明
统计周期	今天 / 昨天 / 本月 / 近 7 天 / 近 30 天 / 自定义时间
报警等级图例	筛选等级【紫色 = 一般、橙色 = 重要、红色 = 紧急】
报警等级设置	进入报警等级配置页面，详见表格下方描述
确认报警	勾选记录后批量将报警信息标记 "已处理"
导出日志（CSV）	离线导出做审计留存
报警类型筛选	服务商、敏感词、越狱攻击、Token 配额等各种类型的筛选过滤

- **报警等级设置**



作用：针对与各种不同类型的报警可以自定义设置报警等级，方便精准区分故障严重程度，避免告警风暴与消息疲劳，让高优先级异常第一时间触达对应负责人。

2.7 第三方 Webhook 服务开发指南

网关的【远程安全日志审计】模块采用**异步非阻塞**的机制运作，所有请求/拦截日志会在内部队列中缓冲，然后通过 HTTP POST 方式批量推送给接收端。如果你的接收端处理失败，网关会根据配置将日志溢写到外挂磁盘的 SQLite 缓存中。

2.7.1 通信协议规范

- 请求方法: POST
- Content-Type: application/json
- 鉴权头 (Header): 如果你在网关配置了「Webhook 鉴权 Token」，网关会在 HTTP Header 中携带 X-Audit-Token 字段。
- 响应要求: 你的接收服务端必须返回 < 400 的 HTTP 状态码(通常是 200 OK 或 201 Created)，网关才会认为推送成功。否则判定为失败并触发磁盘缓存机制。

2.7.2 JSON 数据结构 (Payload)

网关每次推送的 JSON 数据结构如下，包含了请求的完整元数据及上下文：

JSON:

```
{
  "timestamp": "2026-09-27T12:34:56.789123",  // 记录产生的时间 (ISO 8601 格式)
  "member_id": "user_abc123",                 // 实际请求的组成员标识 / Key 前缀
  "group_name": "研发组",                      // 所属工作组
  "provider_id": "provider_123456",           // 命中的上游服务商 ID
  "provider_name": "Claude 官方",             // 命中的上游服务商名称
  "model_name": "claude-3-5-sonnet-20240620", // 客户端实际请求的模型名
  "req_type": "chat",                         // 请求类型 (chat / image_generation / image_to_video 等)
  "total_tokens": 1250,                       // 本次交互消耗的总 Token 数
  "ip_address": "192.168.1.50",               // 发起请求的客户端 IP
  "status_code": 200,                         // 网关最终返回的 HTTP 状态码
  "is_blocked": false,                        // 是否触发了敏感词/配额阻断逻辑 (布尔值)
  "block_reason": "",                         // 阻断原因 (如 "Token 配额超限", "回答包含敏感词熔断")
  "duration_ms": 1500,                        // 整体请求耗时 (毫秒)
  "prompt": "用 Python 写一个快排",           // 用户提问 (仅在开启"完全记录"时有值，否则为空字符串)
  "response": "好的，以下是快排代码..."       // AI 回答 (仅在开启"完全记录"时有值，否则为空字符串)
}
```

注意： 在开启“完全记录日常所有对话提问与回答全文”模式时，为了保护内存，prompt 和 response 的最大长度会被强制截断在 15,000 个字符。

2.7.3 接收端代码示例

下面提供 Python (FastAPI/Flask) 、 Node.js 、 Java (Spring Boot)的标准接收端示例代码。

示例 A：Python (FastAPI) - 推荐

FastAPI 原生支持异步，非常适合处理这种高并发的 Webhook 推送。

Python:

```
from fastapi import FastAPI, Request, Header, HTTPException
from pydantic import BaseModel
import logging

app = FastAPI()
logging.basicConfig(level=logging.INFO)

# 必须与你在网关后台填写的 Token 保持一致
SECRET_TOKEN = "a1b2c3d4-super-secret-token"

class AuditLog(BaseModel):
    timestamp: str
    member_id: str
    group_name: str
    provider_id: str
    provider_name: str
    model_name: str
    req_type: str
    total_tokens: int
    ip_address: str
    status_code: int
    is_blocked: bool
    block_reason: str
    duration_ms: int
    prompt: str
    response: str

@app.post("/api/logs")
async def receive_audit_log(log: AuditLog, x_audit_token: str = Header(None)):
    # 1. 验证 Token
    if SECRET_TOKEN and x_audit_token != SECRET_TOKEN:
```

```
        raise HTTPException(status_code=401, detail="Invalid Audit Token")

    # 2. 业务处理 (例如存入 ElasticSearch, MongoDB 或直接写日志)
    if log.is_blocked:
        logging.warning(f"🚫 [拦截告警] 用户 {log.member_id} 触发拦截, 原因: {log.block_reason}")
    else:
        logging.info(f"✅ [正常请求] 用户 {log.member_id} 调用了 {log.model_name}, 耗时 {log.duration_ms}ms")

    # 3. 快速返回 200 OK 告诉网关接收成功
    return {"status": "success"}

# 启动命令: uvicorn main:app --host 0.0.0.0 --port 8080
```

示例 B：Node.js (Express)

JavaScript:

```
const express = require('express');
const app = express();

app.use(express.json({ limit: '10mb' })); // 如果开启全文记录，payload 可能较大

const SECRET_TOKEN = "a1b2c3d4-super-secret-token";

app.post('/api/logs', (req, res) => {
    // 1. 验证 Token
    const token = req.get('X-Audit-Token');
    if (SECRET_TOKEN && token !== SECRET_TOKEN) {
        return res.status(401).json({ error: "Unauthorized" });
    }

    const logData = req.body;

    // 2. 处理日志
    if (logData.is_blocked) {
        console.warn(`[拦截] ${logData.group_name}-${logData.member_id} 触发: ${logData.block_reason}`);
    } else {
        console.log(`[正常] ${logData.member_id} 消耗 ${logData.total_tokens} tokens`);
    }

    // 3. 返回状态码小于 400 即可
```

```
    res.status(200).json({ message: "Log received" });
});

app.listen(8080, '0.0.0.0', () => {
    console.log("Audit Webhook Server running on port 8080");
});
```

示例 C：Java (Spring Boot) - 企业级后端常用

Spring Boot 非常适合在企业现有 Java 微服务体系中快速集成该审计接口。以下提供单文件（或拆分包）即可运行的 Controller 与实体类示例。

Java 代码 (AuditWebhookController.java)：

```
package com.example.audit;

import com.fasterxml.jackson.annotation.JsonProperty;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

import java.util.Collections;
import java.util.Map;
import java.util.concurrent.CompletableFuture;

@RestController
@RequestMapping("/api")
public class AuditWebhookController {

    private static final Logger logger =
LoggerFactory.getLogger(AuditWebhookController.class);

    // 必须与你在网关后台填写的 Token 保持一致（生产环境建议写在 application.yml 中通过
@Value 注入）
    private static final String SECRET_TOKEN = "a1b2c3d4-super-secret-token";

    @PostMapping("/logs")
    public ResponseEntity<Map<String, String>> receiveAuditLog(
            @RequestHeader(value = "X-Audit-Token", required = false) String auditToken,
            @RequestBody AuditLogPayload log) {

        // 1. 验证 Token
```

```
        if (SECRET_TOKEN != null && !SECRET_TOKEN.isEmpty()
&& !SECRET_TOKEN.equals(auditToken)) {
            return ResponseEntity.status(HttpStatus.UNAUTHORIZED)
                    .body(Collections.singletonMap("error", "Invalid Audit Token"));
        }

        // 2. 异步处理日志业务（遵循最佳实践：不阻塞网关的 8 秒超时，收到后立刻丢入后台线程写
库/发告警）
        CompletableFuture.runAsync(() -> {
            if (Boolean.TRUE.equals(log.getIsBlocked())) {
                logger.warn("🚫 [拦截告警] 工作组[{}] 用户[{}] 触发拦截，原因: {}",
                        log.getGroupName(), log.getMemberId(), log.getBlockReason());
            } else {
                logger.info("✅ [正常请求] 用户[{}] 调用了 {}，消耗 {} tokens，耗时 {}ms",
                        log.getMemberId(), log.getModelName(), log.getTotalTokens(),
log.getDurationMs());
            }
            // TODO: 在此处将 log 存入 MySQL / ElasticSearch / Kafka
        });

        // 3. 立即返回 200 OK 告诉网关接收成功
        return ResponseEntity.ok(Collections.singletonMap("status", "success"));
    }

    /**
     * 审计日志 JSON 映射实体类 (Payload DTO)
     * 使用 @JsonProperty 处理网关下划线字段到 Java 驼峰命名的自动转换
     */
    public static class AuditLogPayload {
        private String timestamp;

        @JsonProperty("member_id")
        private String memberId;

        @JsonProperty("group_name")
        private String groupName;

        @JsonProperty("provider_id")
        private String providerId;

        @JsonProperty("provider_name")
        private String providerName;

        @JsonProperty("model_name")
        private String modelName;
```

```
    @JsonProperty("req_type")
    private String reqType;

    @JsonProperty("total_tokens")
    private Integer totalTokens;

    @JsonProperty("ip_address")
    private String ipAddress;

    @JsonProperty("status_code")
    private Integer statusCode;

    @JsonProperty("is_blocked")
    private Boolean isBlocked;

    @JsonProperty("block_reason")
    private String blockReason;

    @JsonProperty("duration_ms")
    private Integer durationMs;

    private String prompt;
    private String response;

    // --- Getters and Setters ---
    public String getTimestamp() { return timestamp; }
    public void setTimestamp(String timestamp) { this.timestamp = timestamp; }

    public String getMemberId() { return memberId; }
    public void setMemberId(String memberId) { this.memberId = memberId; }

    public String getGroupName() { return groupName; }
    public void setGroupName(String groupName) { this.groupName = groupName; }

    public String getProviderId() { return providerId; }
    public void setProviderId(String providerId) { this.providerId = providerId; }

    public String getProviderName() { return providerName; }
    public void setProviderName(String providerName) { this.providerName =
providerName; }

    public String getModelName() { return modelName; }
    public void setModelName(String modelName) { this.modelName = modelName; }
```

```
        public String getReqType() { return reqType; }
        public void setReqType(String reqType) { this.reqType = reqType; }

        public Integer getTotalTokens() { return totalTokens; }
        public void setTotalTokens(Integer totalTokens) { this.totalTokens = totalTokens; }

        public String getIpAddress() { return ipAddress; }
        public void setIpAddress(String ipAddress) { this.ipAddress = ipAddress; }

        public Integer getStatusCode() { return statusCode; }
        public void setStatusCode(Integer statusCode) { this.statusCode = statusCode; }

        public Boolean getIsBlocked() { return isBlocked; }
        public void setIsBlocked(Boolean isBlocked) { this.isBlocked = isBlocked; }

        public String getBlockReason() { return blockReason; }
        public void setBlockReason(String blockReason) { this.blockReason = blockReason; }

        public Integer getDurationMs() { return durationMs; }
        public void setDurationMs(Integer durationMs) { this.durationMs = durationMs; }

        public String getPrompt() { return prompt; }
        public void setPrompt(String prompt) { this.prompt = prompt; }

        public String getResponse() { return response; }
        public void setResponse(String response) { this.response = response; }
    }
}
```

Spring Boot 配置提示（application.properties / application.yml）：由于网关在开启“完全记录”时单次推送的 prompt 和 response 文本较大，建议在配置文件中放宽内嵌 Tomcat 的 POST 请求体积限制（默认 2MB 一般够用，但高并发或批量扩展时建议调大）：

```
Properties:
server.port=8080
server.tomcat.max-http-form-post-size=10MB
spring.servlet.multipart.max-File-size=10MB
spring.servlet.multipart.max-request-size=10MB
```

2.7.4 生产环境最佳实践

- 绝对的快速响应：

你的 Webhook 接收服务不应该在收到日志后进行耗时的同步操作（例如调用其他慢速 API 或执行复杂聚合计算）。网关发起的请求超时时间设置为 8 秒。正确做法是：收到日志 -> 放入你自己的 Redis 队列或后台线程 -> 立即返回 200 OK。

- Body 大小限制：

如果开启了“全文记录”，单条日志体积可能达到 30KB - 50KB。请确保你的 Nginx 反向代理或 Web 框架（如 Express、Flask）没有把 client_max_body_size 设置得过小（建议放宽到 5MB 以上以应对未来可能的批量并发投递）。

- 数据清洗与报表：

利用接收到的 is_blocked 和 total_tokens 字段，你可以轻易在自己的系统里构建出精美的 Grafana 大盘或者进行企业级的飞书/钉钉违规实时预警。

三、网络配置

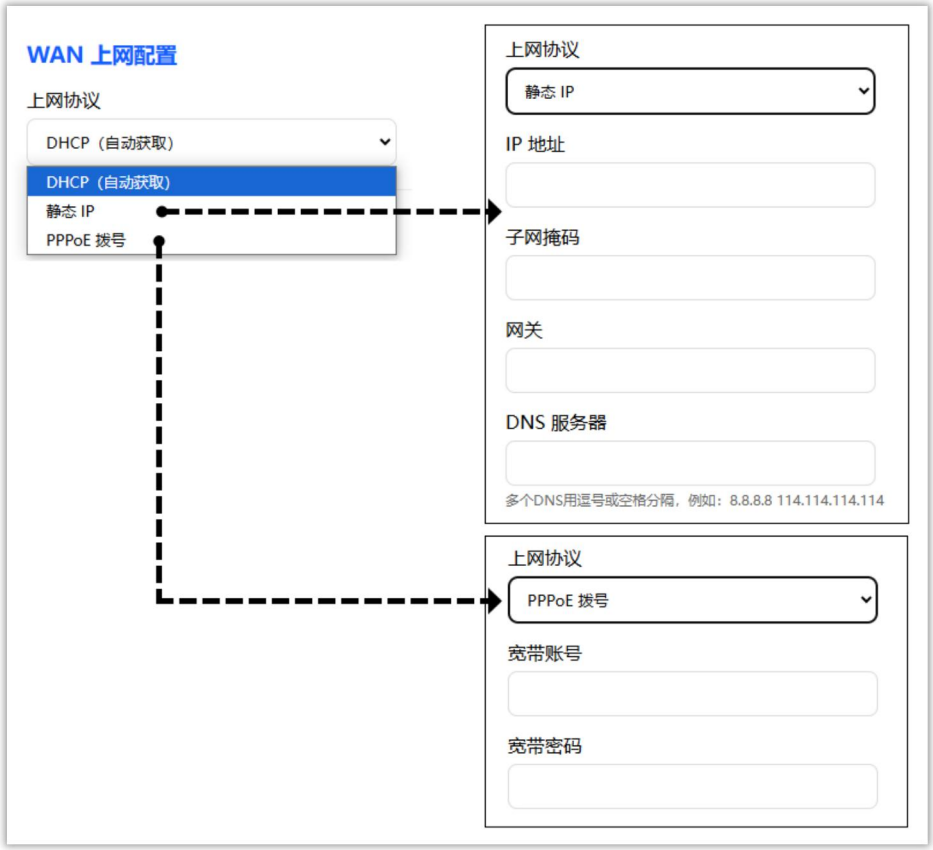
3.1 LAN 局域网配置



用于设置设备局域网 IP 地址与子网掩码参数。

注意：IP 变更后，本后台的访问地址同步变更

3.2 WAN 上网配置



配置项	默认值/说明
DHCP	设备从上游网络设备自动获取 IP 地址、子网掩码、网关、DNS 等外网参数，无需手动填写任何网络信息，即插即用，是最常用的上网方式。
静态 IP	手动填写运营商提供的固定 IP 地址、子网掩码、网关、DNS 服务器全部参数。适用于运营商分配固定公网 IP 的场景。
PPPoE 拨号	填写运营商提供的宽带账号、宽带密码，设备主动拨号连接互联网，家用光纤宽带普遍使用该模式。

3.3 路由工作模式



配置项	默认值/说明
路由模式	设备作为网关，可负责拨号、分配内网 IP，普通组网使用。
旁路模式	不充当网关，适配已有主路由的现有网络。需配置下静态 IP。

注意：旁路模式，相连路由器需做一些配置，若有需求可联系客服远程协助。联系电话：400 027 7590

四、无线配置 【*仅带无线 WIFI 的设备型号支持】

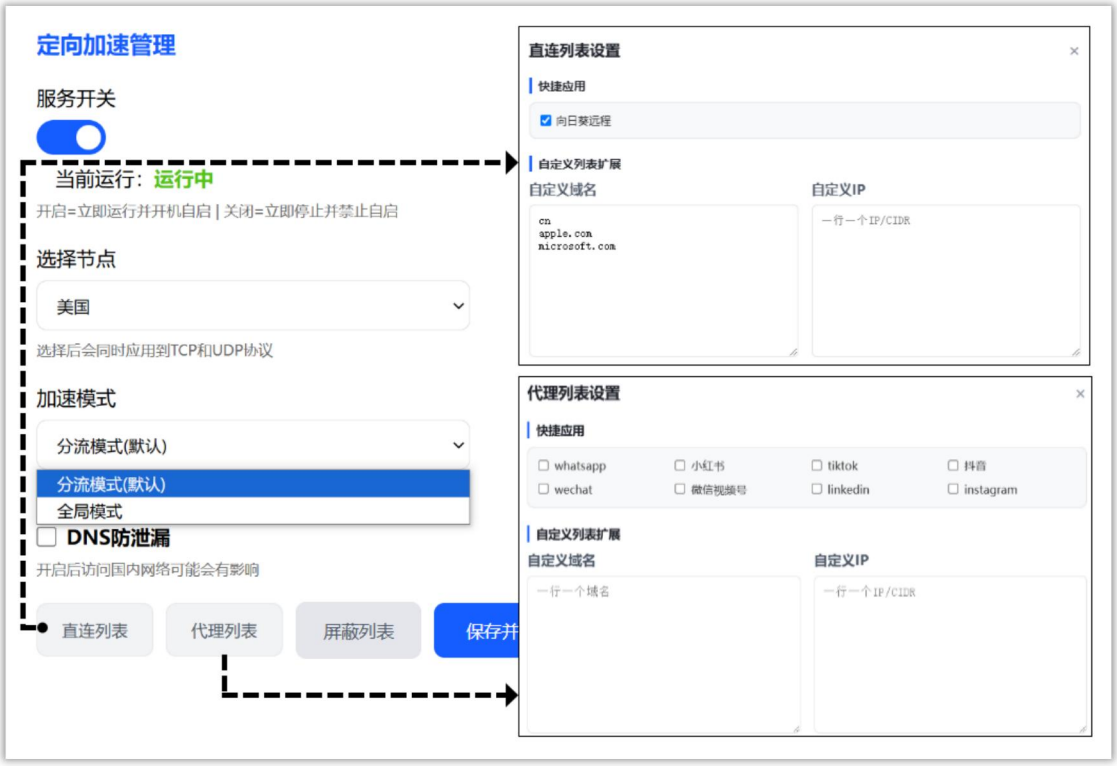


配置项	默认值/说明
WiFi 服务状态	开启 / 关闭无线信号
WiFi 名称	设置无线网络的显示名称
加密方式	WPA2/WPA3- PSK 为安全加密模式，保障无线接入安全（建议选择）
WiFi 密码	设置无线连接密码
信道	默认自动，设备自动选择干扰较小信道，可手动指定

2.4G：穿墙好、传输速率较低；5G：速度快、穿墙能力弱。

五、定向加速

5.1 定向加速管理



配置项	默认值/说明
服务开关	开启 / 关闭定向加速。
选择节点	选择加速的节点，备选项为购买节点。
分流模式	默认。仅外网代理列表中的地址走节点加速，国内地址走直连。
全局模式	访问的所有地址均经过加速节点。
DNS 防泄露	避免泄露本地网络信息，开启后有可能造成国内网站访问异常。
直连列表	指定域名 / IP 不经过加速节点，直接访问网络。 出现访问不通的情况可在此处配置。
代理列表	指定域名 / IP 走加速节点。 出现访问不通的情况可在此处配置。
屏蔽列表	配置网址黑名单。

5.2 当前定向流量出口



统计当月当前加速节点所产生的流量。

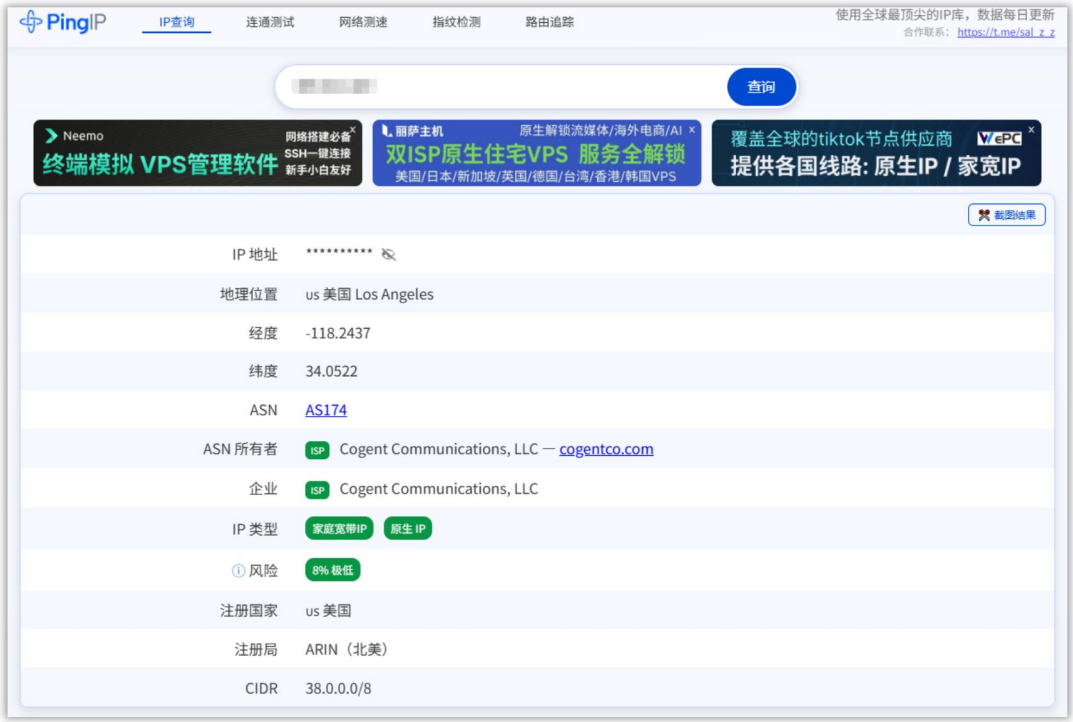
注：数据更新会有延时。

5.3 当前定向出口 IP



显示当前定向出口的 IP、所属地址及运营商信息。

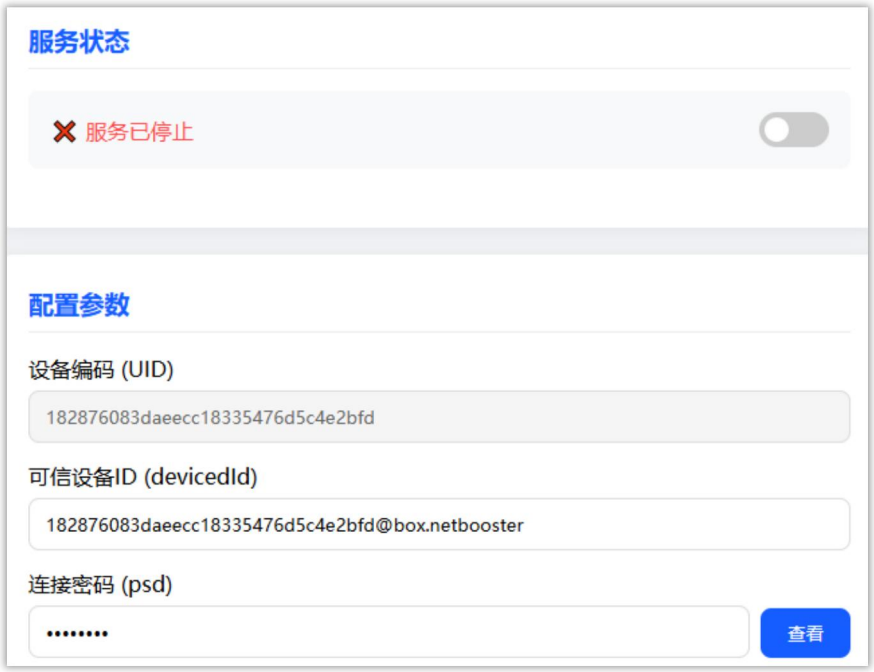
查看更多：跳转至第三方平台，可监测当前节点的参数。



六、智能组网

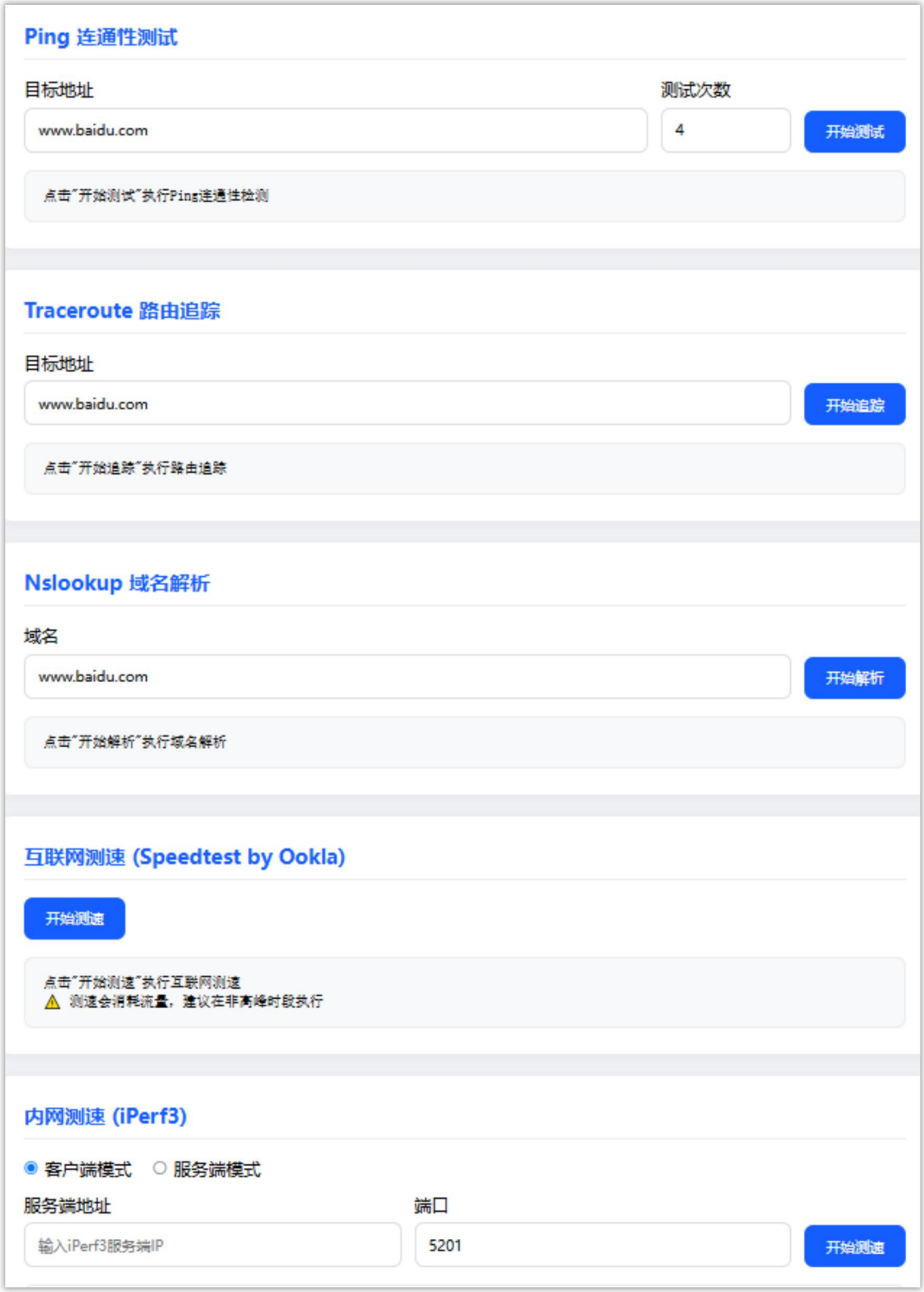
可信组网是我公司提供的一项增值服务，能够将用户名下的多台加速宝设备安全、高效地构建成专属互联网专网，助力用户实现设备间的稳定互联与数据安全传输。

如有需求，请联系开通相关业务功能。联系电话：400 027 7590



配置项	默认值/说明
服务开关	开启 / 关闭智能组网服务。 注意：开启服务前请关闭定向加速，两者不可同时使用。
可信设备 ID / 连接密码	开通业务后由相关业务人员给予分配。

七、网络诊断


Ping 连通性测试
目标地址
测试次数
www.baidu.com
4
开始测试
点击"开始测试"执行Ping连通性检测
Traceroute 路由追踪
目标地址
www.baidu.com
开始追踪
点击"开始追踪"执行路由追踪
Nslookup 域名解析
域名
www.baidu.com
开始解析
点击"开始解析"执行域名解析
互联网测速 (Speedtest by Ookla)
开始测速
点击"开始测速"执行互联网测速
⚠ 测速会消耗流量，建议在非高峰时段执行
内网测速 (iPerf3)
客户端模式
服务端模式
服务端地址
端口
输入iPerf3服务端IP
5201
开始测速

配置项	默认值/说明
Ping 连通性测试	目标地址：填写需要检测的域名或者 IP 地址。 测试次数：设置发送探测包的数量。 功能：检测目标网络是否连通，获取网络延迟、丢包情况。 用途：快速判断网络通断，查看链路时延，简单排查网络是否异常。
Traceroute 路由追踪	目标地址：输入要追踪的域名 / IP。 功能：查看数据包从本设备到目标地址，经过的每一个网络节点。 用途：定位网络卡顿、丢包故障出现在哪一段中间链路，适合排查网络访问慢、访问不通的深层问题。
Nslookup 域名解析	域名：输入需要检测的网站域名。 功能：执行 DNS 解析，返回域名对应的 IP 地址。 用途：判断 DNS 服务是否正常，排查域名解析错误类故障。
互联网测速（Speedtest by Ookla）	功能：测试外网的上传、下载实际带宽速度。 注意：测速会消耗网络流量，占用带宽，建议避开业务高峰时段。 用途：验证实际上网带宽是否符合预期。
内网测速（iPerf3）	用来测试局域网内网传输性能，有两种模式： 客户端模式：本机作为测速客户端，填写对端 iPerf3 服务端 IP 与端口（默认 5201），主动连接服务端测试内网带宽。 服务端模式：本机开启 iPerf3 服务，等待其他 iPerf 客户端来连接，本机充当测速服务器。 用途：评估局域网内网传输速度，排查内网带宽不足问题。

八、设备信息



配置项	默认值/说明
当前系统版本	当前固件 / 软件版本
在线检查并更新系统	点击自动检查新版本并升级
设备型号	硬件型号
设备序列号（SN）	设备唯一序列号，售后 / 授权绑定用
公司名称	设备厂商
联系电话	厂商客服热线
授权状态	网关是否已开通

配置项	默认值/说明
最大允许内网用户数	可接入的下游分发 Key 数量上限
网关高级功能	工作组，Token 限额配置，审计日志推送的功能授权状态
网关图表功能	AI 统计报表、报警日志等可视化能力的授权状态
授权到期日期	授权有效期，到期前需续费
激活码输入框	续费 / 变更授权时由厂商提供新激活码，提交后完成激活

九、账号与安全

9.1 账号安全与防护



配置项	默认值/说明
修改当前登录账号密码	新密码：至少 6 位，右侧眼睛图标可显示 / 隐藏 确认新密码：再输一遍，两次一致才生效
防爆破安全控制	防止黑客用脚本暴力破解 admin 密码。 ● 允许连续输错次数：设为 0 = 不锁 IP；设为 N = 连续错 N 次就锁定来源 IP ● 触发锁定后限制时长：锁定期间该 IP 完全无法登录，所设时间过后自动解封 *推荐配置：输错 5 次锁 15 分钟，能有效挡掉字典攻击，又不会误伤正常用户
HTTPS 加密传输	启用 HTTPS 加密访问：开启后后台走 https 加密通道，密码和操作不再明文跑网络；关闭则回到 http。 HTTPS 服务端口：默认端口 443，可自定义

配置项	默认值/说明
	*防止嗅探抓到 admin 密码，生产环境建议开启。

9.2 管理员账号管理



作用：支持创建多个子管理员，按功能模块划分权限。

配置项	默认值/说明
添加新管理员	依次输入管理员账号，初始登录密码，授权可访问的功能模块后添加
管理员列表	对已有的管理员账号进行修改、删除等操作

以上为本产品完整配置说明，可能会跟软件有差异，一切以软件为准。后续但凡涉及参数调整、版本升级、请记得同步升级本文档，遇到疑难问题欢迎咨询。

联系电话：万联云网 400 027 7590