# 所有服务节点：管理平台、API，说明为什么这样部署，能避免什么问题

## API服务

API服务面向终端SDK，接受终端SDK调用并返回数据，包括：机构、菜单、卡片、JS规则等。

考虑的点：

* 高并发：
  + 终端数量多，且未来还会持续增长
  + 终端有本地缓存、本地数据库，可降低对同一数据的访问频次
* 高可用：为避免单点故障，建议部署大于1台服务器

部署方案：

* 部署4台API服务器，通过Nginx做负载均衡，采用轮询算法，由4台API服务器共同承担终端SDK的访问。随着时间推移，如果发现4台不够，可以很容易地增加服务器分担负载，保证服务稳定
* 为避免Nginx成为瓶颈，设计部署2台Nginx服务器，通过DNS（或LVS）进行负载均衡

## 打点服务

打点服务面向终端SDK，接收终端SDK上报的打点数据，包括：卡片解析、菜单点击等。

考虑的点：

* 高并发：
  + 终端数量多，且未来还会持续增长
  + 终端先将打点数据记录到本地数据库，定时调用打点服务
* 高可用：为避免单点故障，建议部署大于1台服务器

部署方案：

* 部署2台打点服务器，通过Nginx做负载均衡
* 与API服务用同一套Nginx

## 管理平台

管理平台分为客户门户和运营门户。

客户门户面向来自领航平台的产品客户，根据实际需要自定义Logo、菜单、卡片等。

运营门户面向管理员，负责审核客户门户提交的数据。

考虑的点：

* 高并发：
  + 用户数量不多
  + 用户操作有“思考时间”，并发程度较低
* 高可用：为避免单点故障，建议部署大于1台服务器

部署方案：

* 部署2台客户门户服务器、2台运营门户服务器，通过Nginx做负载均衡
* 与API服务用同一套Nginx

# 测试方案

## 启停测试

//TODO

## 负载均衡测试

//TODO