有赞接入层架构演进

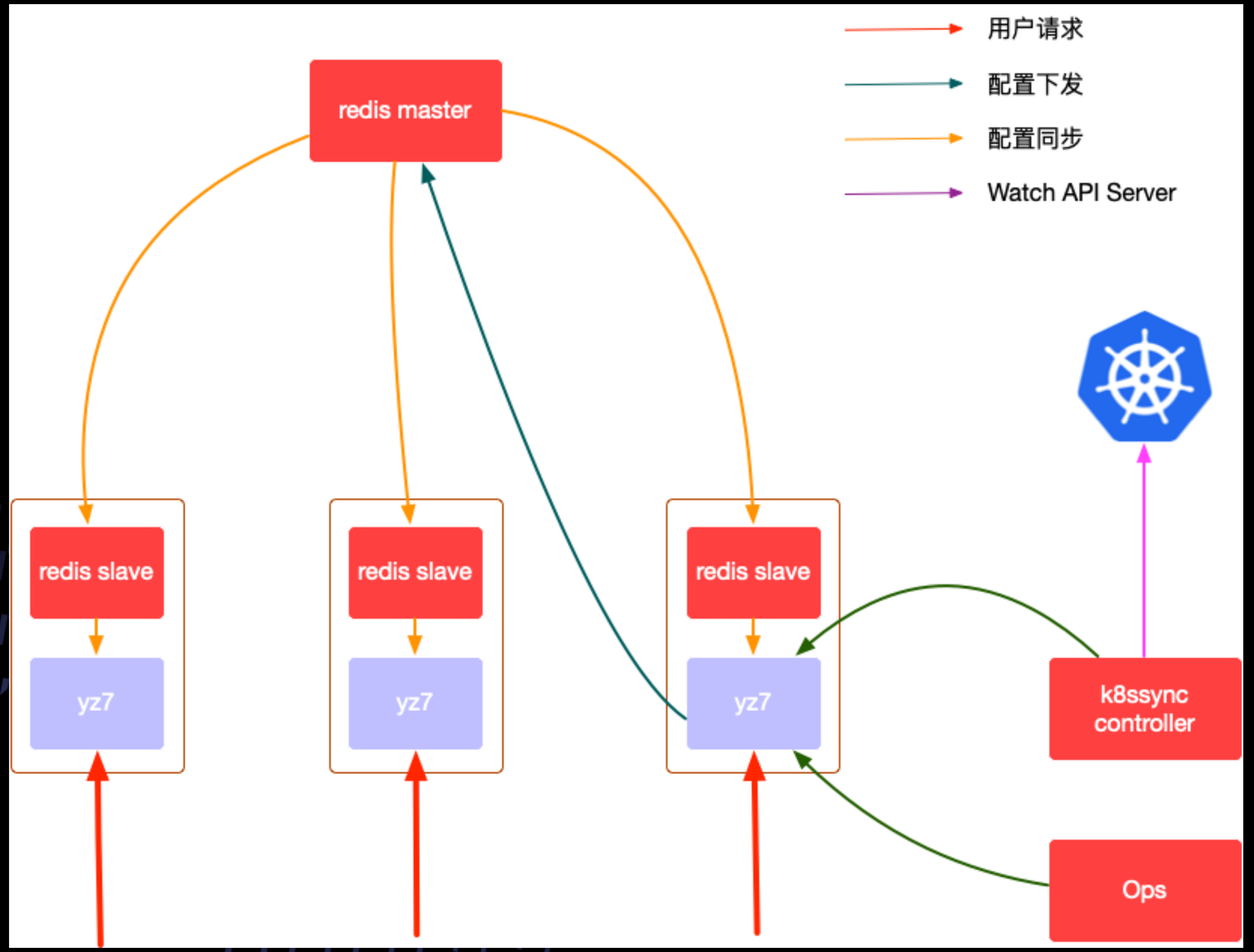
张超

有赞接入层

- 内部名为 yz7，基于 OpenResty/Nginx 实现，作为有赞业务流量的公网入口，提供 Traffic Shaping、WAF、请求路由、负载均衡等功能。

Agenda

- 旧版接入层架构痛点
- 新架构设计分析
- 新架构设计总结


用户请求
配置下发
配置同步
Watch API Server
redis master
redis slave
redis slave
redis slave
yz7
yz7
yz7
k8ssync
controller
Ops

缺陷 & 局限性

- redis master 单点问题 - 故障时配置无法下发；跨机房部署时易受机房间通信稳定性的影响

缺陷 & 局限性

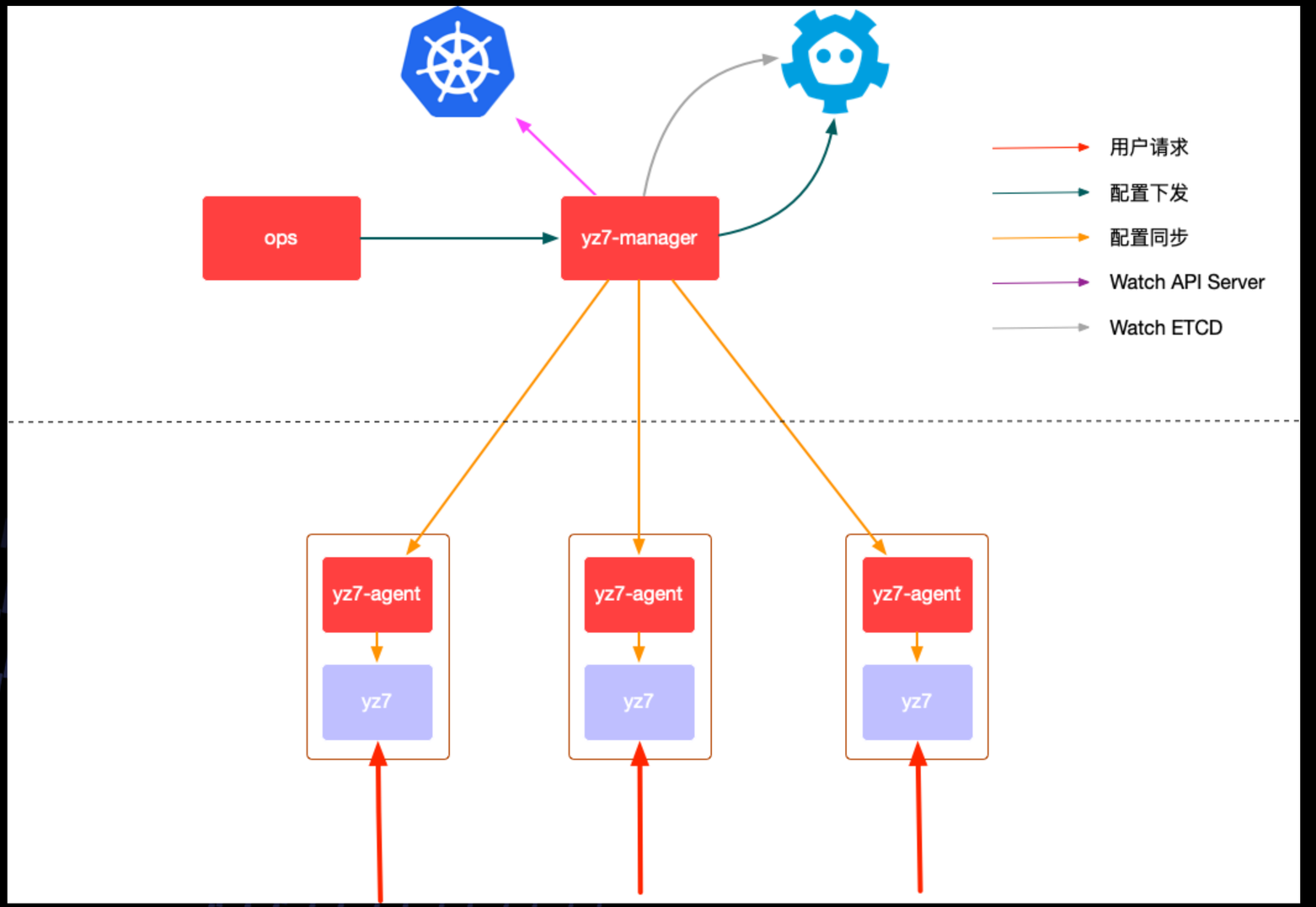
- k8ssync controller 单点问题 - 无法及时感知后端应用 Endpoints 变化，可能产生一系列服务不可用问题

缺陷 & 局限性

- 配置不具备属性特征 - 无法在配置层面进行多样性处理，例如配置的“灰度下发”

新架构设计要点

- 避免出现任何单点问题
- 组件尽设计尽可能无状态，可灰度、可回滚、可观测
- 尽量降低组件间的耦合程度，各组件职能独立，可独立测试部署


ops
yz7-manager
用户请求
配置下发
配置同步
Watch API Server
Watch ETCD
yz7-agent
yz7
yz7-agent
yz7
yz7-agent
yz7

yz7-manager

- 配置保存于 ETCD，稳定可靠
- 无状态，可水平扩容

- 接管原 k8ssync controller 的功能
- 接管原 yz7 配置 admin server
- 接管配置下发功能

Youzan.com

yz7-agent

- yz7 伴生服务，与 yz7 绑定同机部署
- 负责 yz7 配置同步
- 配置注解释义 - 如配置灰度功能
- 配置间依赖管理

yz7

- 剔除原有配置 admin server
- 预留简单 HTTP 接口接受配置推送
- 保留其他原有网关功能

Youzan.com

Confusions?

- yz7-manager <-> yz7-agent：配置下发协议如何设计?
- yz7-agent <-> yz7：配置同步 Push or Pull?
- 配置注解如何实现?
- 配置依赖如何保证?

配置下发协议

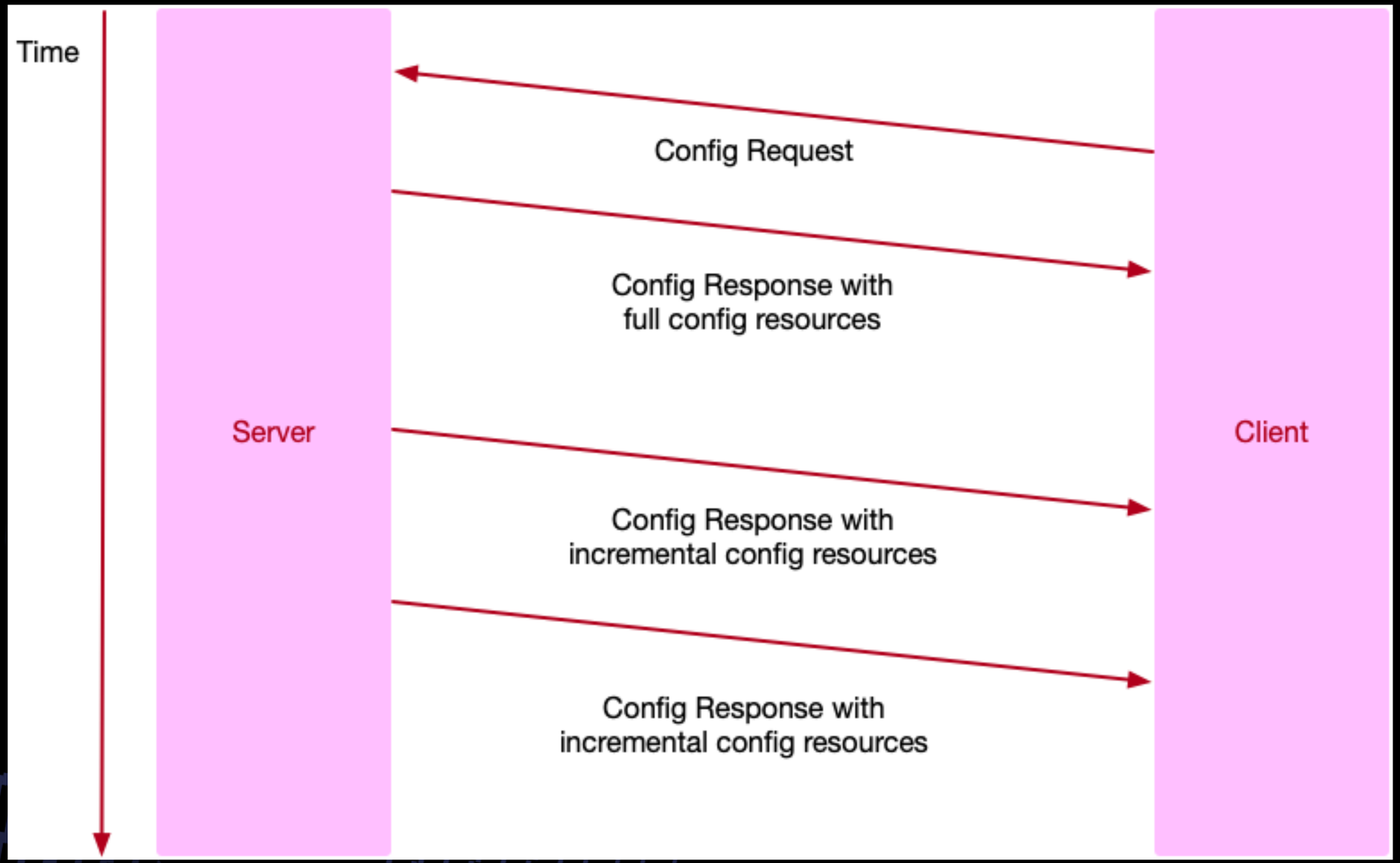
- 协议设计需要尽可能简单可靠
- 尽可能支持服务端主动推送以降低时延
- 基于 gRPC，类似于 xDS
- 初次全量；后续增量

```
service ConfigDiscoveryService {
  rpc StreamConfigResources(ConfigRequest) returns (stream ConfigResponse) {}
}

message ConfigRequest {
  Node node = 1;
  repeated ResourceCondition required_resources = 2;
}

// [#protodoc-title: ConfigResponse]

// ConfigResponse carries the data that agent wants or the error that occurred
// during the process of server side.
message ConfigResponse {
  google.rpc.Status error_detail = 1;
  repeated Resource resources = 2;
}
```


Time
Server
Client
Config Request
Config Response with
full config resources
Config Response with
incremental config resources
Config Response with
incremental config resources

Push or Pull?

- yz7 主动拉配置- 设计简单且可按需加载
- 缺点是在缺乏 watch 机制的情况下，配置缓存需要设定过期时间，导致新配置生效具有一定延迟且造成无谓的资源（CPU、带宽）浪费，而配置变更属于低频事件，因此不予采用

Push or Pull?

- yz7-agent 主动配置至 yz7 - yz7 配置无需考虑过期且组件耦合程度更低，便于交付测试
- 缺点是 yz7 刚启动时没有任何配置数据
- yz7-agent 定期会将内存中的配置缓存转储到磁盘，以供 yz7 启动/热更新时载入；同时结合定期全量推送的方式确保 yz7 能够拿到最新版本的配置

配置注解

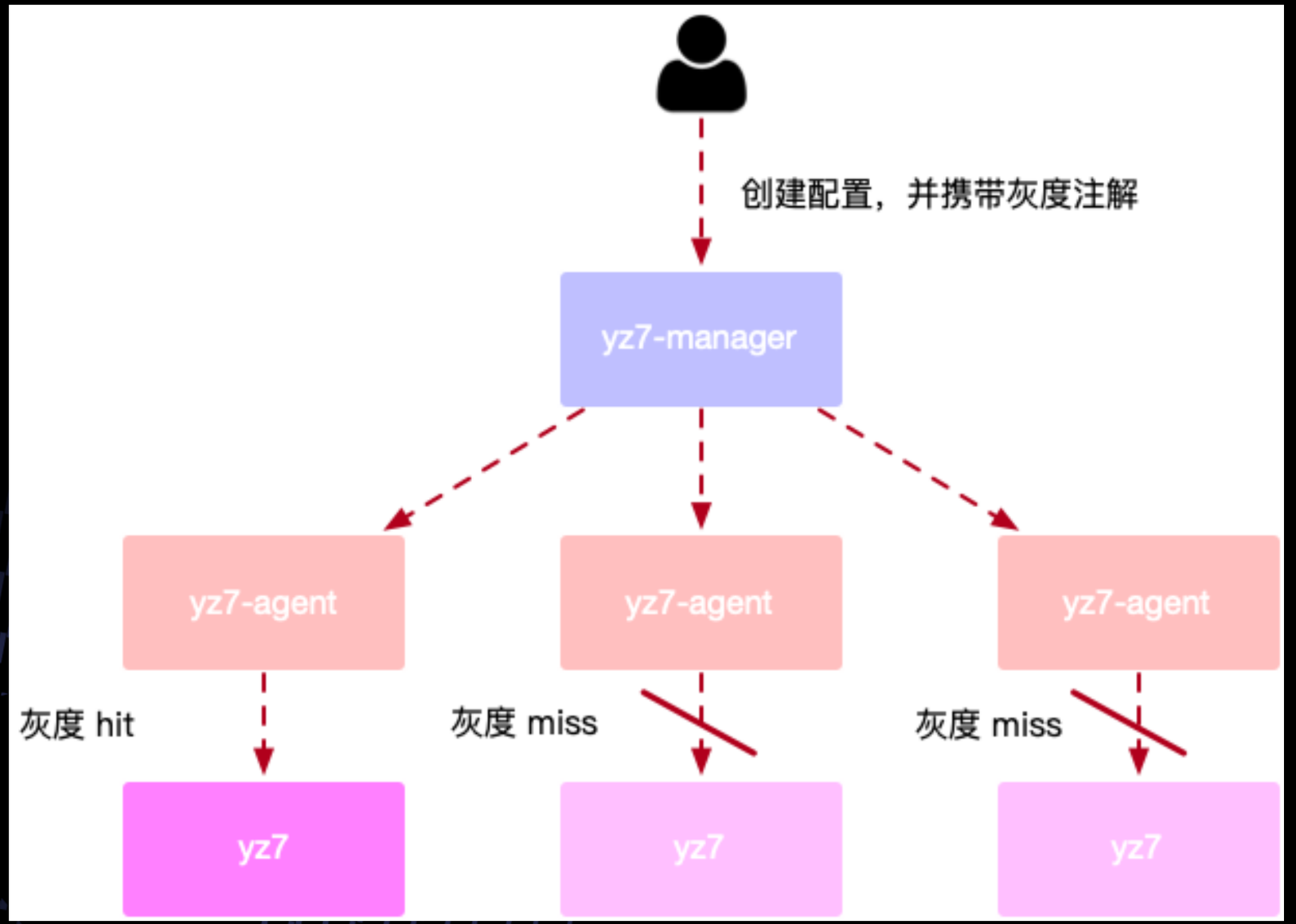
```
{
  "annotations": {
    "canary": {
      "hosts": [
        "host3",
        "host2"
      ]
    }
  },
  "id": "123e4567-e89b-12d3-a456-426614174000",
  "name": "hahaha",
  "kind": "upstream",
  "tombstone": false,
  "resource_version": 1234,
  "payload": {
      "servers": [
        { addr: "127.0.0.1:8080", "weight": 100 }
      ]
  }
}
```

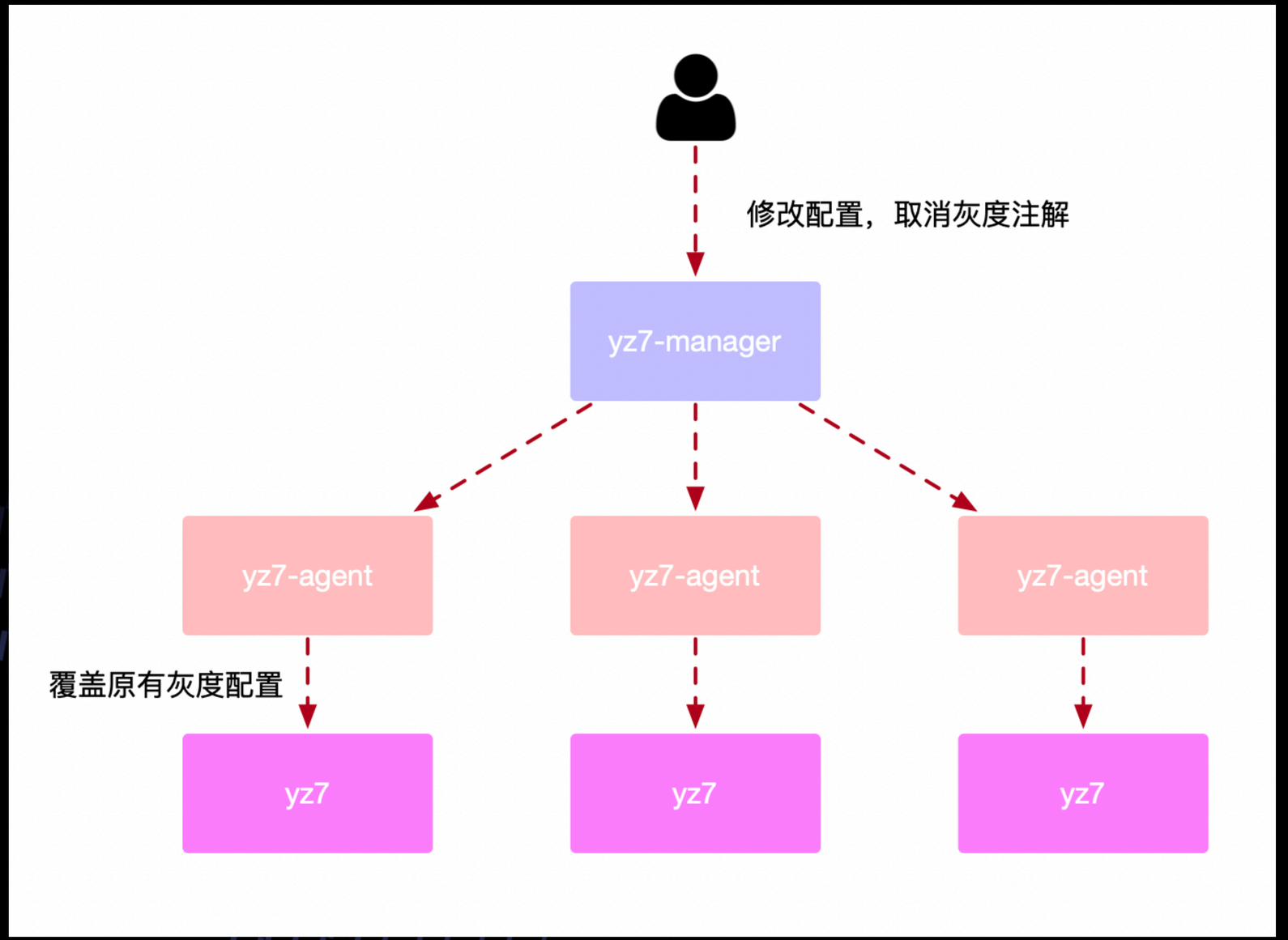
为什么需要配置灰度?

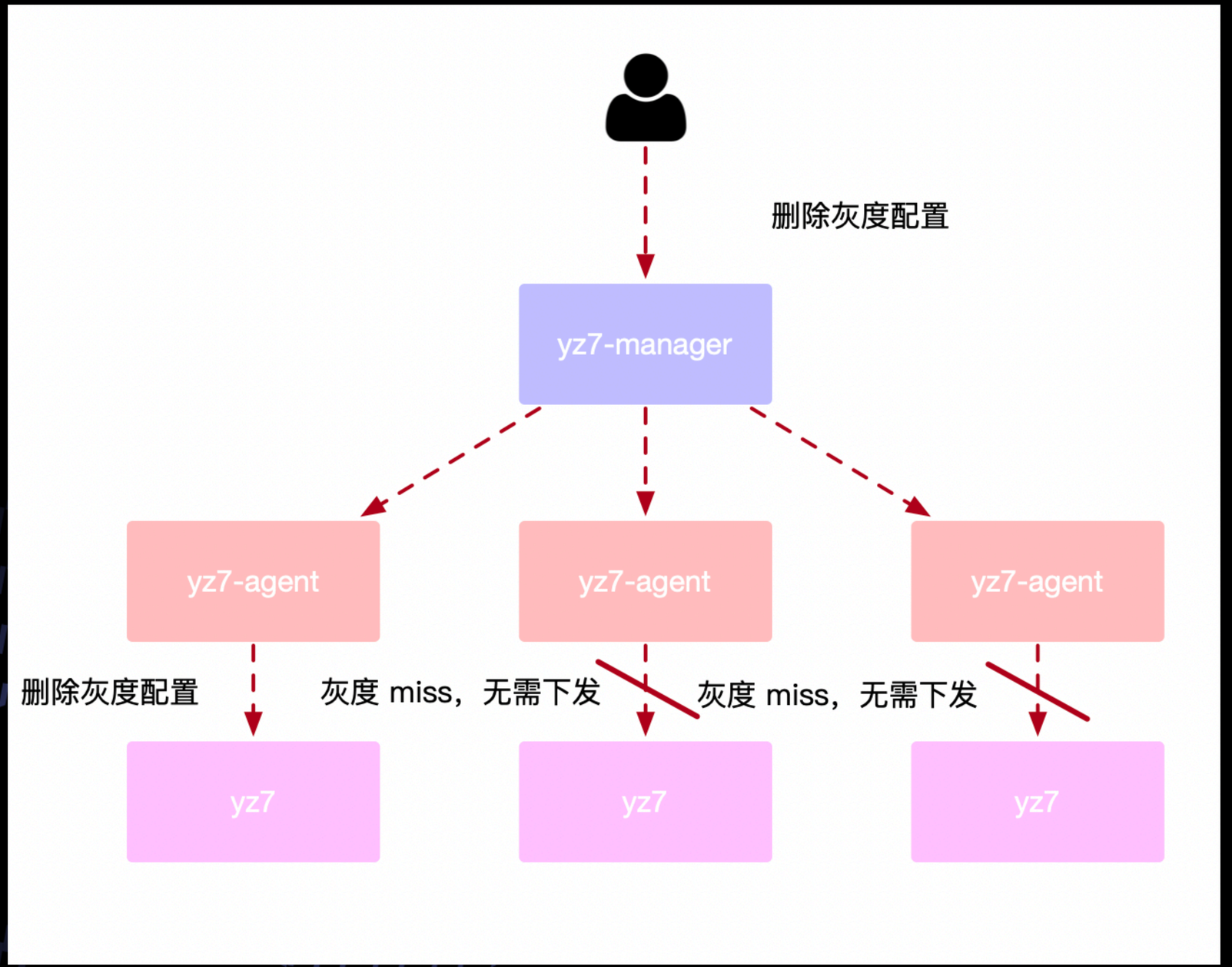
- 有赞作为 SaaS 服务提供商，域名众多，配置复杂，人为操作易出错
- 配置灰度可控制配置生效范围，有利于降低故障影响面

配置灰度操作流程

- 创建配置并标注灰度注解
- 观察此配置在对应生效实例上的表现
- 如确认表现无误，取消灰度注解使之在整个集群内生效
- 如发现有误，删除灰度配置


创建配置，并携带灰度注解
yz7-manager
yz7-agent
yz7-agent
yz7-agent
灰度 hit
灰度 miss
灰度 miss
yz7
yz7
yz7


修改配置，取消灰度注解
yz7-manager
yz7-agent
yz7-agent
yz7-agent
覆盖原有灰度配置
yz7
yz7
yz7


删除灰度配置
yz7-manager
yz7-agent
yz7-agent
yz7-agent
删除灰度配置
灰度 miss，无需下发
灰度 miss，无需下发
yz7
yz7
yz7

配置依赖管理

- 部分配置间存在引用关系，yz7-agent 推送时需要优先推送被依赖项
- 缺失依赖项的配置将被 yz7-agent 暂时搁置

设计总结

- 新架构遵循控制面和数据面分离的设计原则
- 配置下发协议参考 Envoy xDS
- 配置加入注解功能，类似于 Kubernetes 中各 Resources 的声明定义

Thanks!
Q & A

Youzan.com