蚂蚁金服
Service Mesh渐进式迁移方案

敖小剑 @ 蚂蚁金服 中间件
龙轼 @UC 基础研发部

蚂蚁金服
ANT FINANCIAL

1

1 Service Mesh演进路线

2 实现平滑迁移的关键

3 DNS寻址方案的演进

4 DNS寻址方案的后续规划

5 总结

蚂蚁金服主站落地：目标与现状

百川归海

- ✓ 对未来长期目标的认可
  - Service Mesh（带控制平面，如Istio）
  - Kubernetes
  - 微服务
- ✓ 现实中有很多挑战
  - 还有很多应用没有实现微服务化
  - 还有很多应用没有运行在kubernetes之上
  - Istio目前还不够稳定，也无法原生支持我们的规模
  - 现有系统中的众多应用不可能一夜之间全部迁移
- ✓ 最重要的：平滑迁移
  - 微服务 + Service Mesh + Kubernetes 是目标
  - 但是如何从现有体系向目标迈进，必须给出可行的实践指导
- ✓ Roadmap
  - 预计2019年初

蚂蚁金服主站落地方案的实施原则

符合远期规划

- 不走弯路，不浪费投资
- 每一步都为下一步奠定基础
- 谢绝中途推倒重来

循序渐进

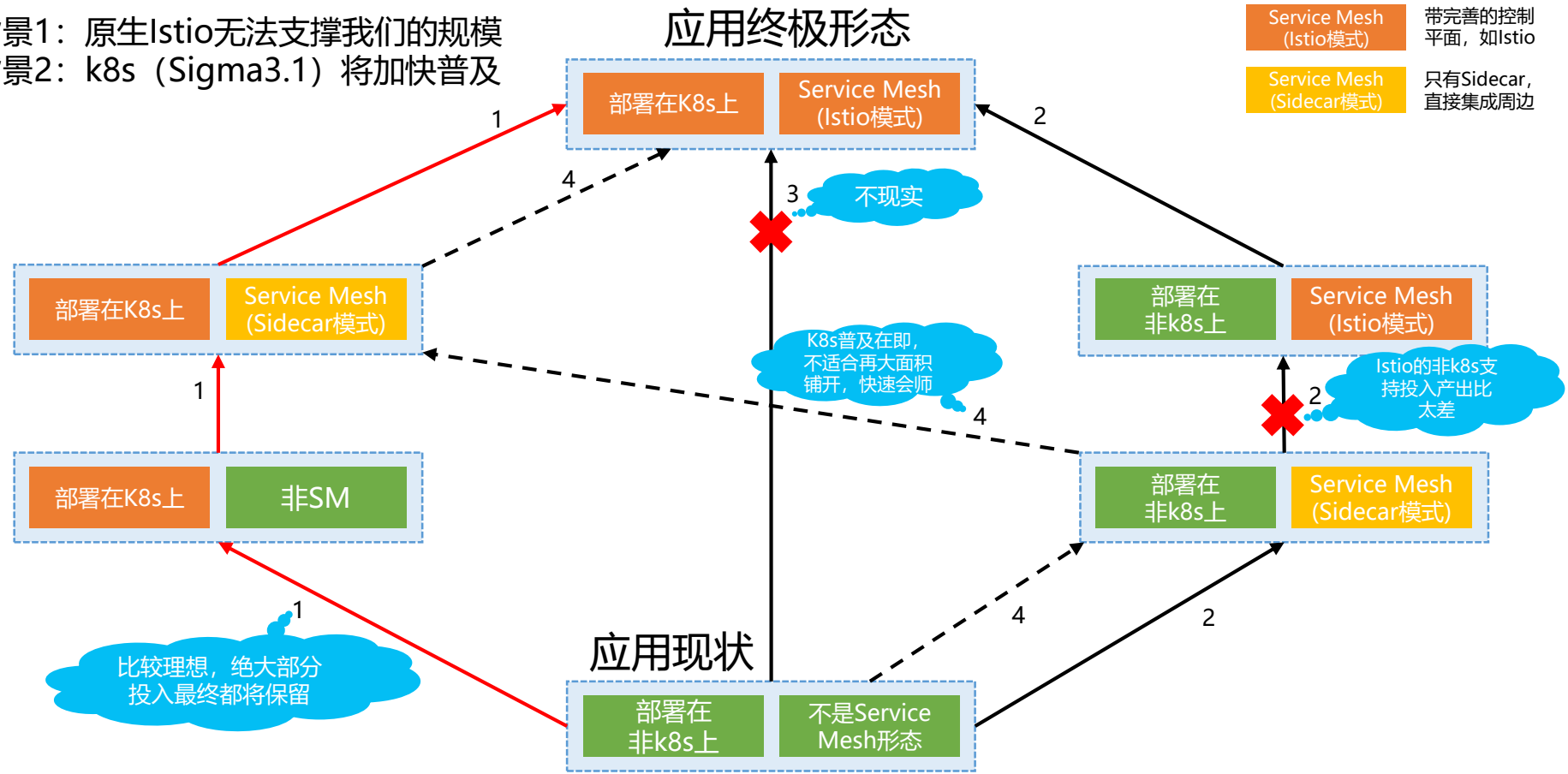
- 不要有一步登天的幻想，小步快跑
- 每一步的工作量和复杂度都控制在可接受范围内
- 每一步都简单方便，切实可行

可操纵性

- 操作层面上要有足够的弹性
- 每个步骤都是可分批进行
- 步步为营，扩大战果
- 杜绝一刀切

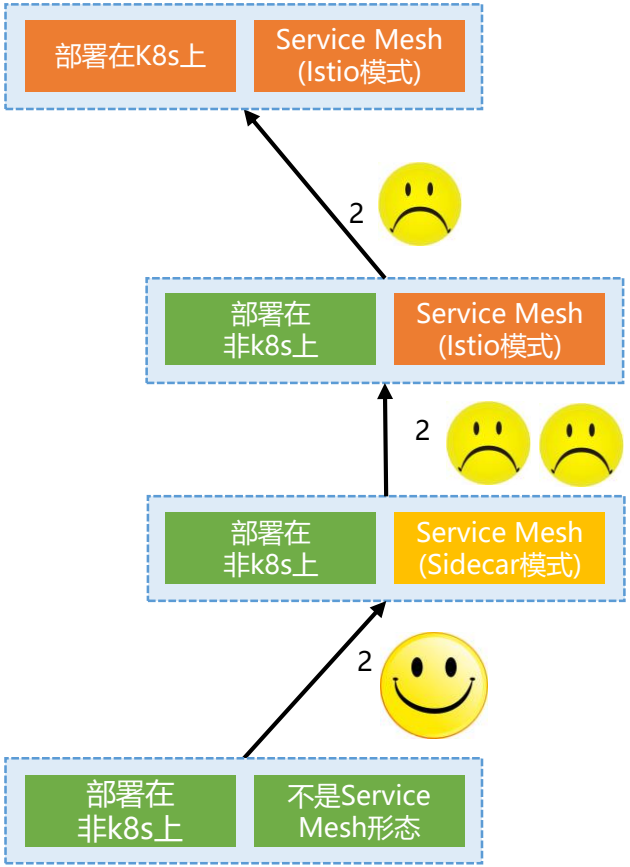
k8s和Service Mesh落地方案演进路线

背景1：原生Istio无法支撑我们的规模
背景2：k8s（Sigma3.1）将加快普及



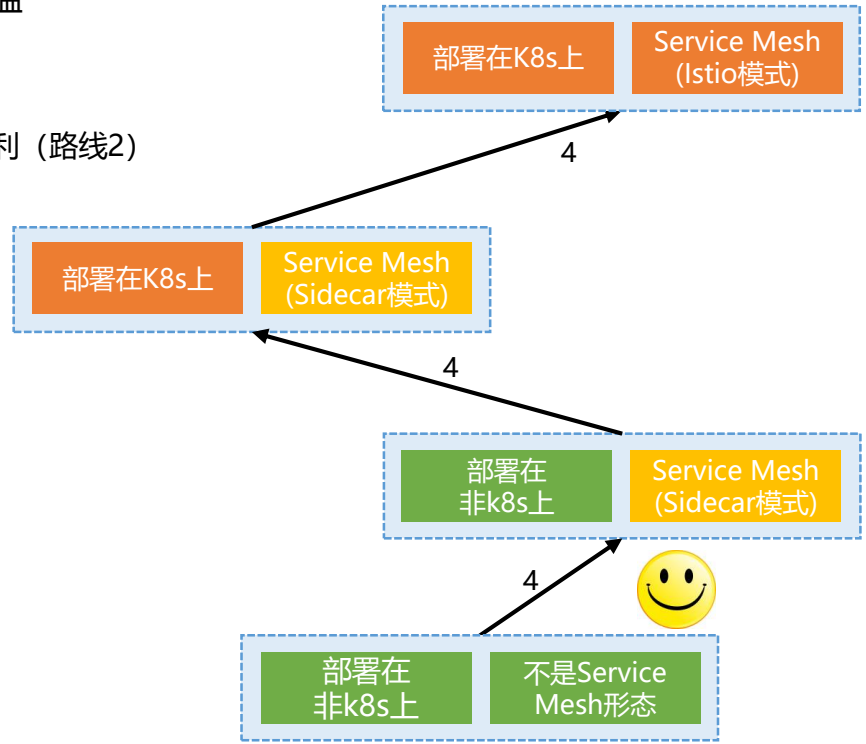
演进路线2分析

- ✓ 和路线1的核心差别
  - 是先上k8s，还是先上Service Mesh
  - 而且是终极形态的Service Mesh（意味着更偏离目标）
- ✓ 好处是第一步（非k8s上向Sidecar模式演进）非常自然
  - 容易落地
  - 快速达成短期目标
- ✓ 缺点是再往后走
  - 由于没有k8s的底层支持，就不得不做大量工作
  - 尤其istio的非k8s支持，工作量很大
  - 而这些投入，在最终迁移到k8s时，又被废弃
- ✓ 结论：
  - 不符合蚂蚁的远期规划（k8s是我们的既定目标）
  - 会造成投资浪费（k8s铺开在即）



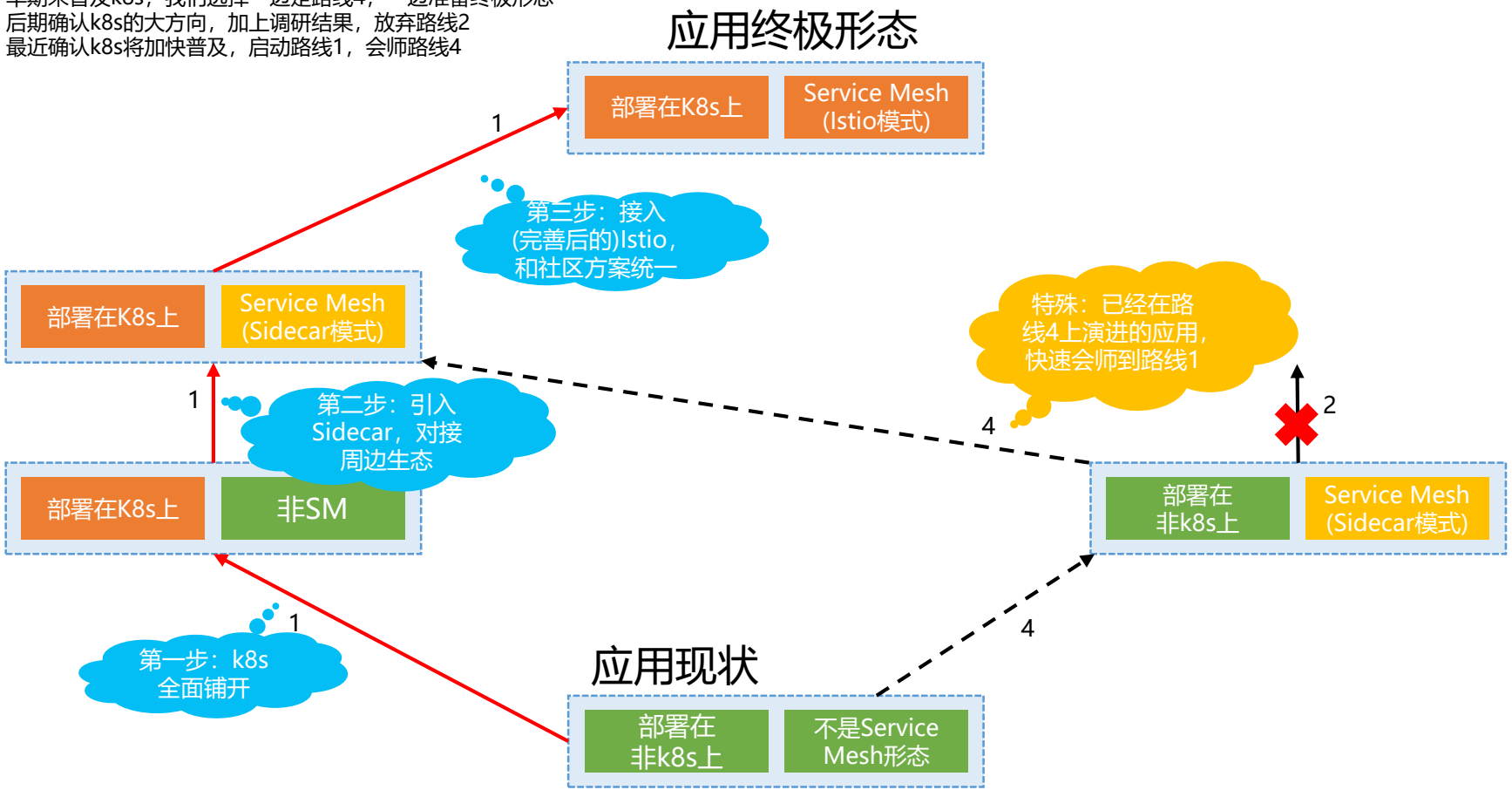
演进路线4分析

- ✓ 可以理解为路线1的折衷版本
  - 路线1的前提是要先大规模铺开k8s，这是一个很高的门槛
  - 路线2能快速拿到短期红利，但是偏离长期目标
  - 路线4的折衷方式
    - 在k8s还没有铺开前，先吃下非k8s下Sidecar模式快速落地的红利（路线2）
    - 然后避开非k8s下继续演进的大坑，回归长期目标（路线1）
- ✓ 好处（和路线2一样）
  - 在k8s未铺开前，先向前迈进一步，避免卡壳
- ✓ 缺点
  - 存在少量的投资浪费（不过和拿到的红利相比是值得的）
- ✓ 存在变数
  - 是Sidecar模式的Service Mesh普及快？还是k8s普及快
- ✓ 结论：
  - 特殊时期（k8s铺开前）的选择



蚂蚁金服选择的演进路线

背景1：早期未普及k8s，我们选择一边走路线4，一边准备终极形态
背景2：后期确认k8s的大方向，加上调研结果，放弃路线2
背景3：最近确认k8s将加快普及，启动路线1，会师路线4



2

1 Service Mesh演进路线

2 实现平滑迁移的关键

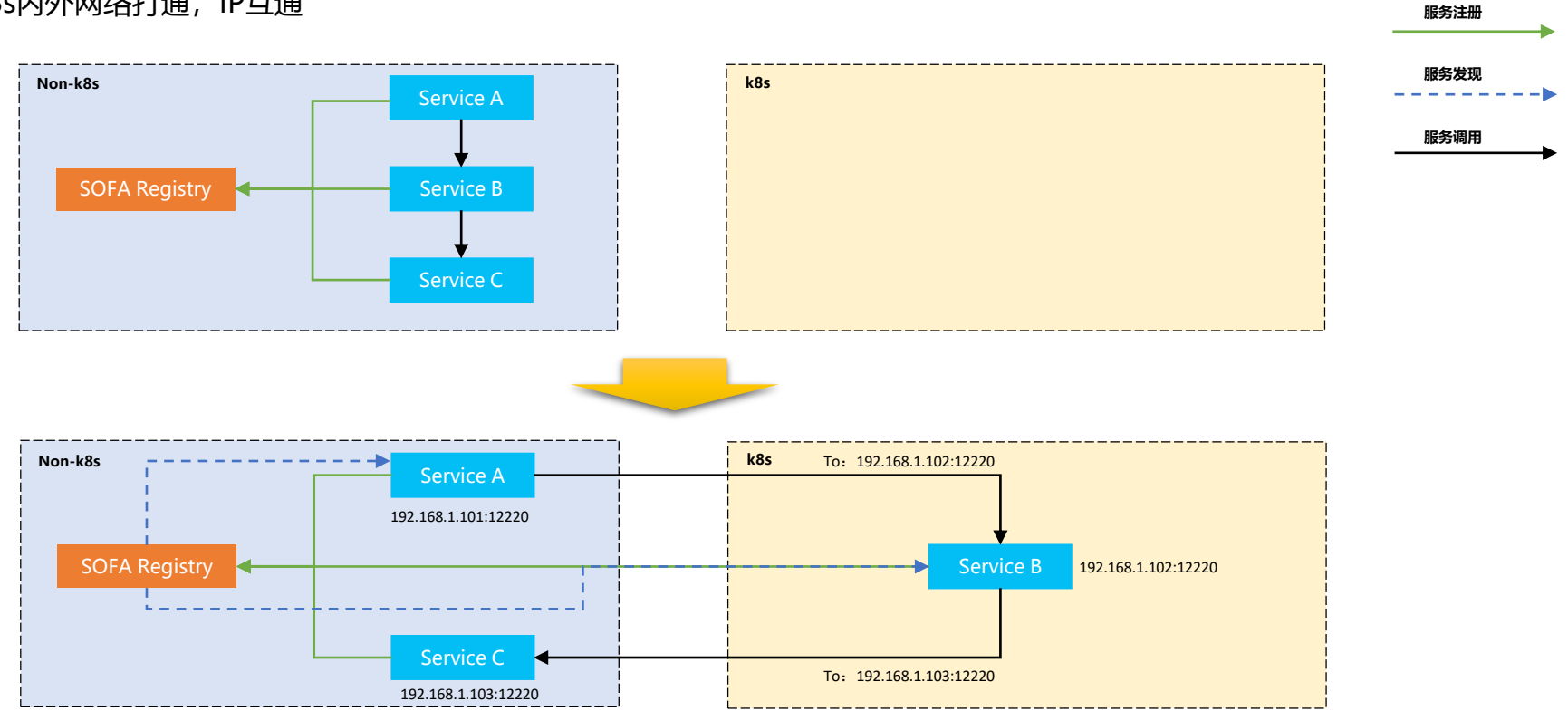
3 DNS寻址方案的演进

4 DNS寻址方案的后续规划

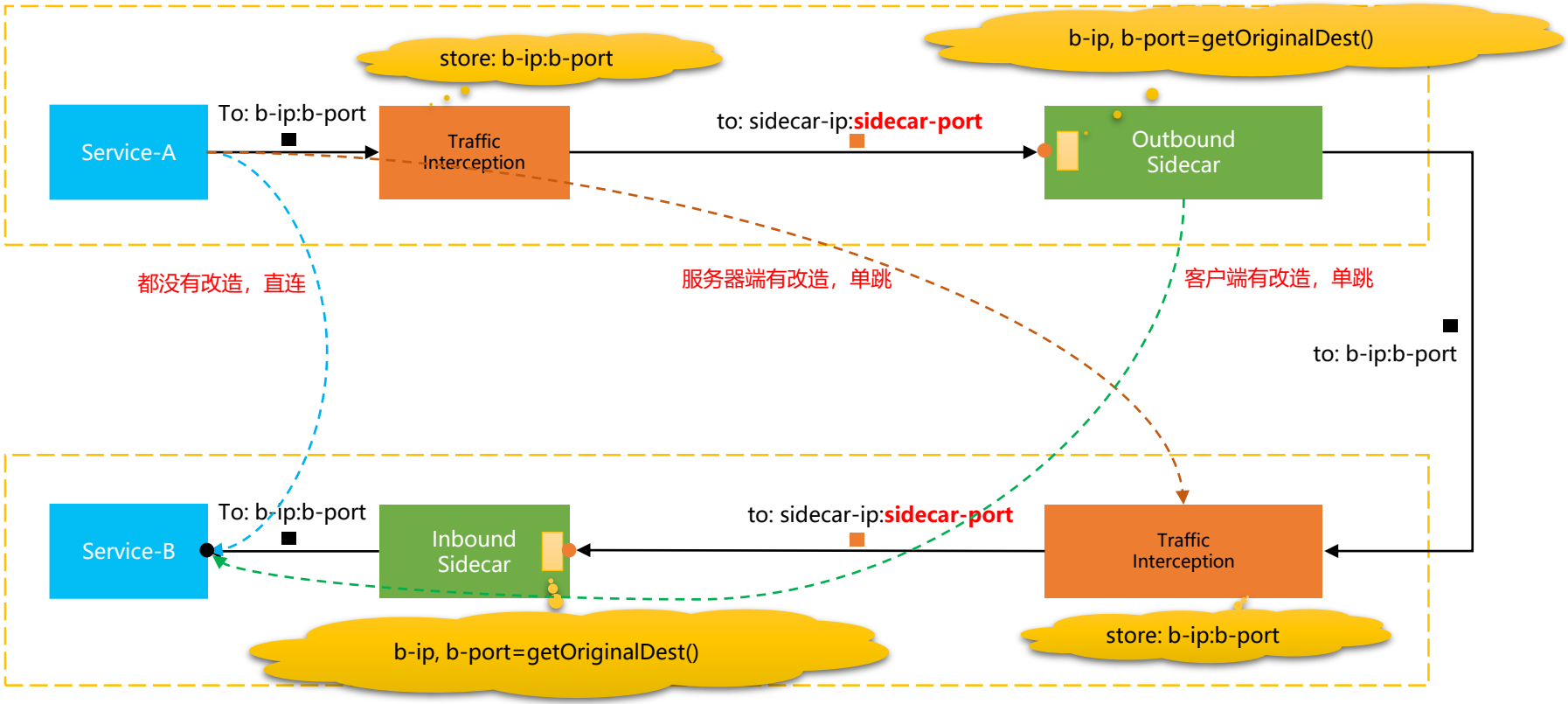
5 总结

保证迁移前后服务间网络互通

背景：k8s内外网络打通，IP互通



透明拦截带来的升级弹性：对应用透明，支持直连/单跳/双跳



Service Mesh时代的客户端和寻址方式

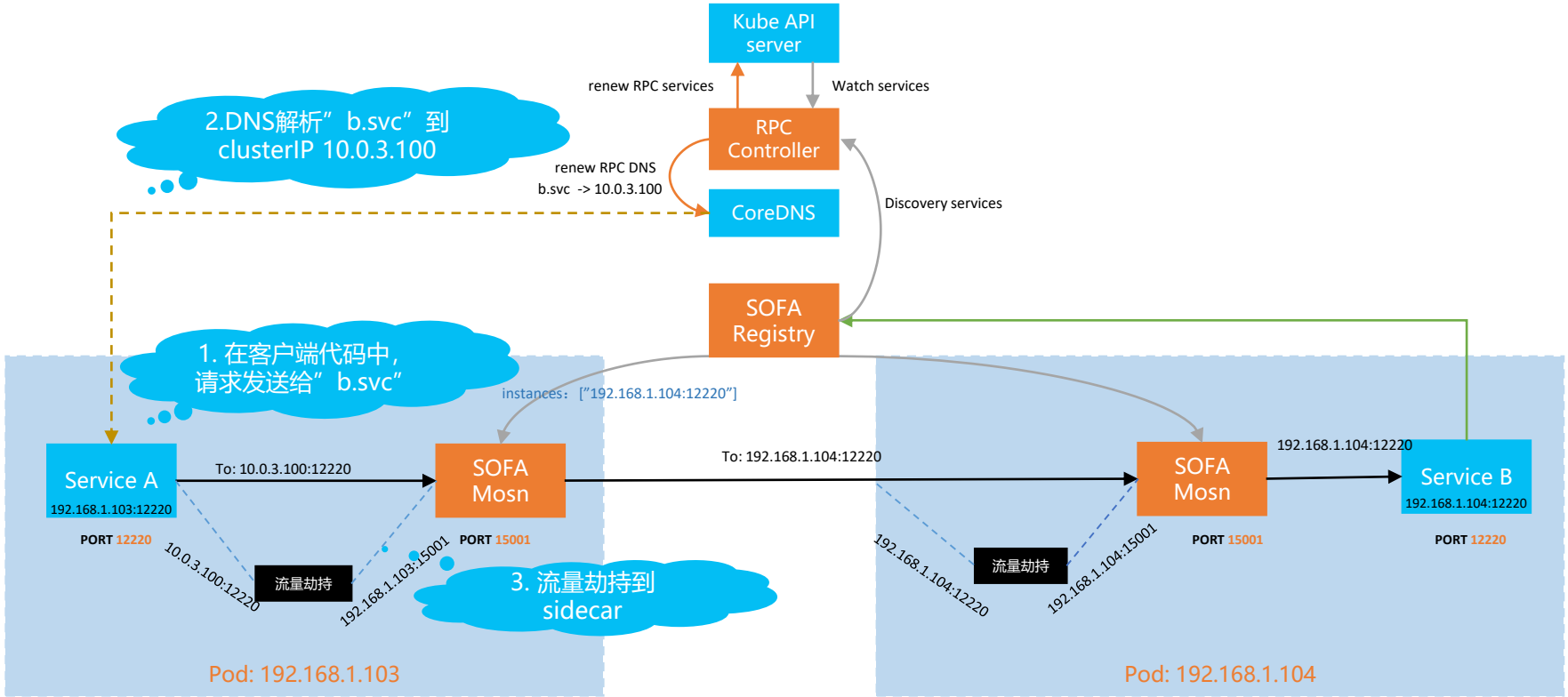
客户端应该尽可能的轻薄通用: 实现简单，方便跨语言，减少升级可能

引入DNS寻址方式(基于域名和DNS的Naming Service)

蚂蚁金服 ANT FINANCIAL

- ✓ DNS寻址
  - 支持度最好，使用最普遍
  - 所有编程语言/平台都支持的
- ✓ 产品的长期方向
  - SOFAMesh和SOFAMosn中已经基于x-protocol实现了DNS通用寻址方式
  - 为了兼容RPC应用和k8s（微服务）的服务注册模型，需要为每个RPC接口提供DNS支持
  - 未来Serverless中的Function也计划提供DNS寻址支持
  - 可能会有更广泛的使用场景
- ✓ 演进思路
  - 简化原有SDK （短期需求）
  - 同时引入域名和DNS，实现通用寻址 （长期目标）

客户端通过域名来对服务进行访问



3

1 Service Mesh演进路线

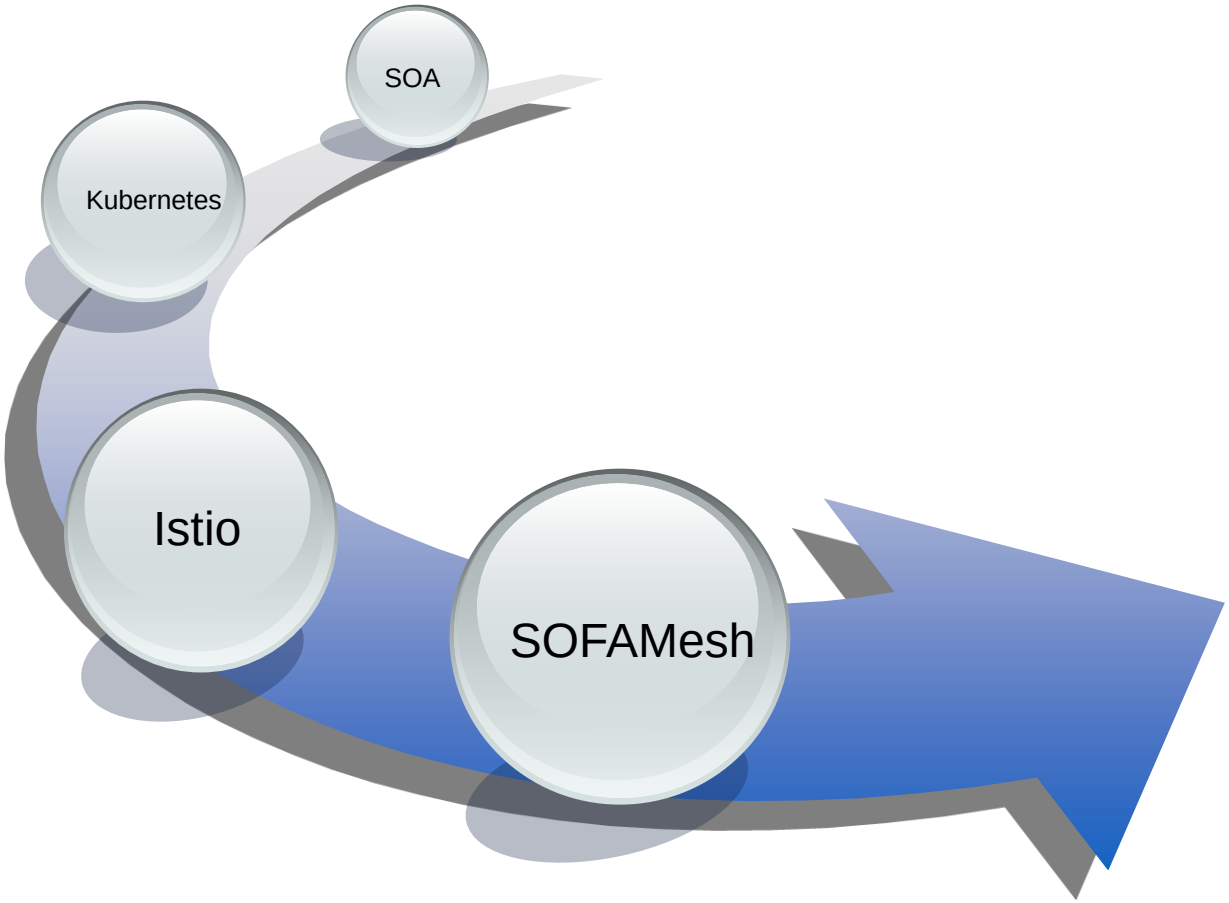
2 实现平滑迁移的关键

3 DNS寻址方案的演进

4 DNS寻址方案的后续规划

5 总结

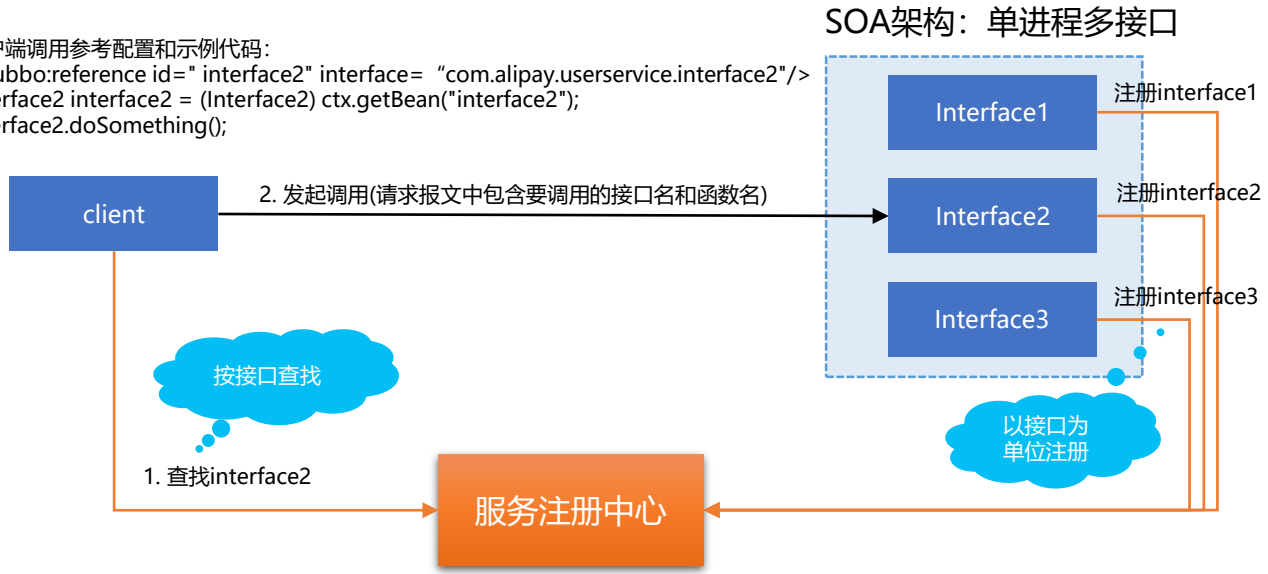
DNS寻址方案的演进路线



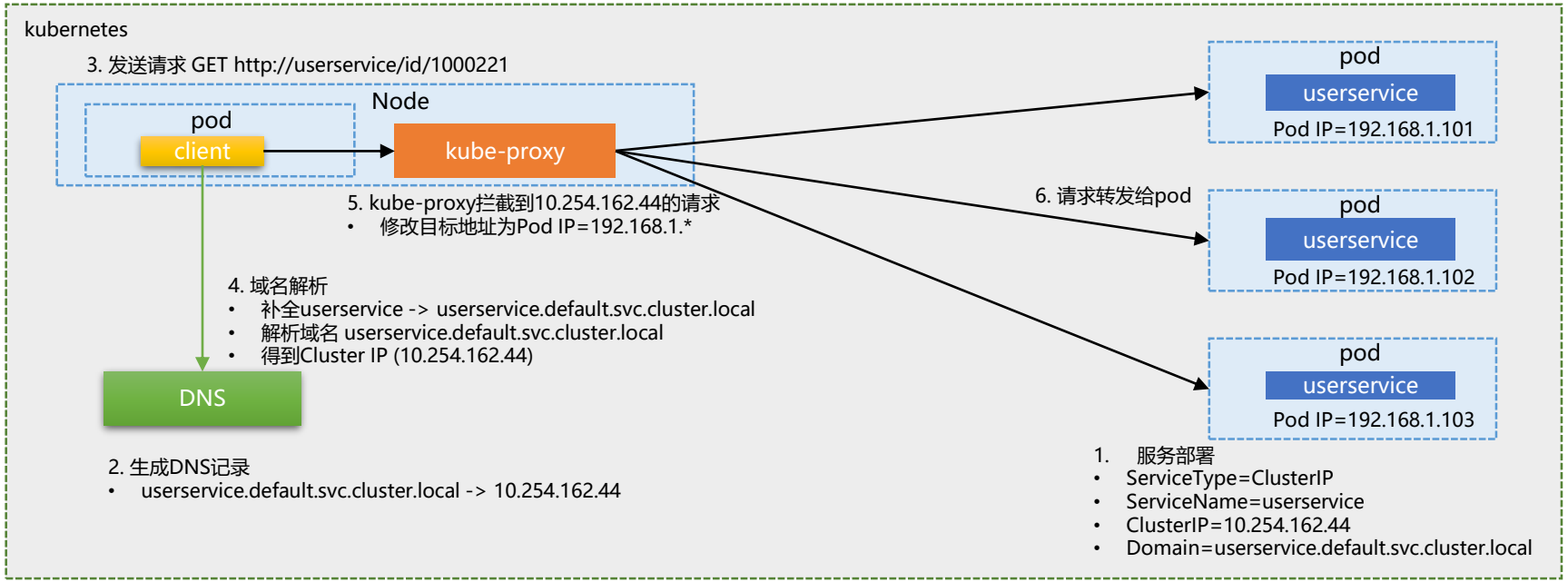
SOA时代的寻址：服务发现和服务注册

客户端调用参考配置和示例代码：

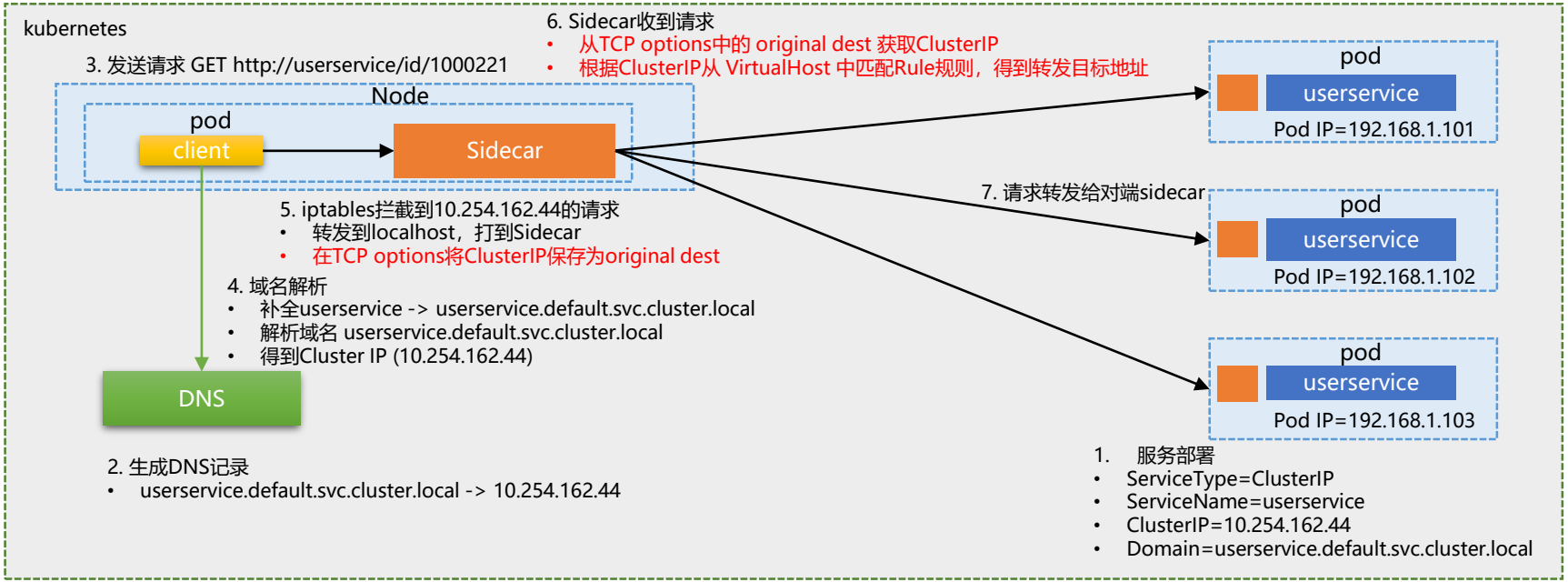
```
<dubbo:reference id=" interface2" interface= “com.alipay.userservice.interface2"/>
Interface2 interface2 = (Interface2) ctx.getBean("interface2");
interface2.doSomething();
```



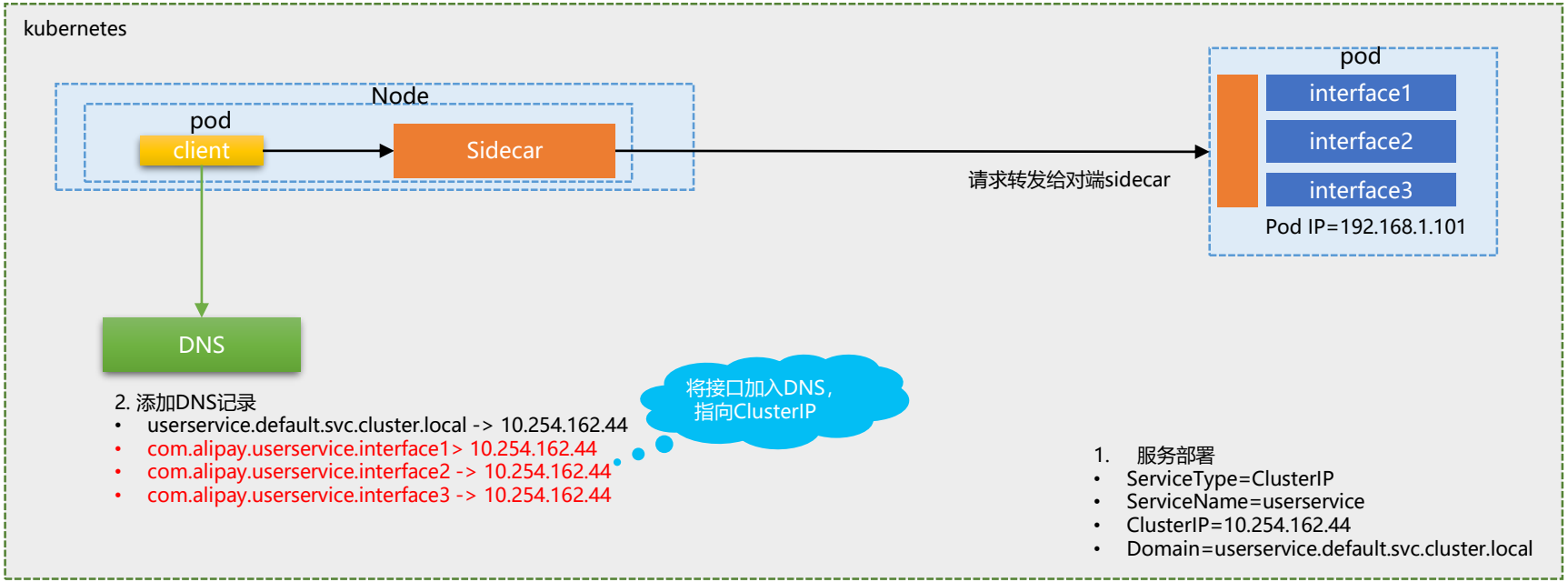
Kubernetes的DNS寻址



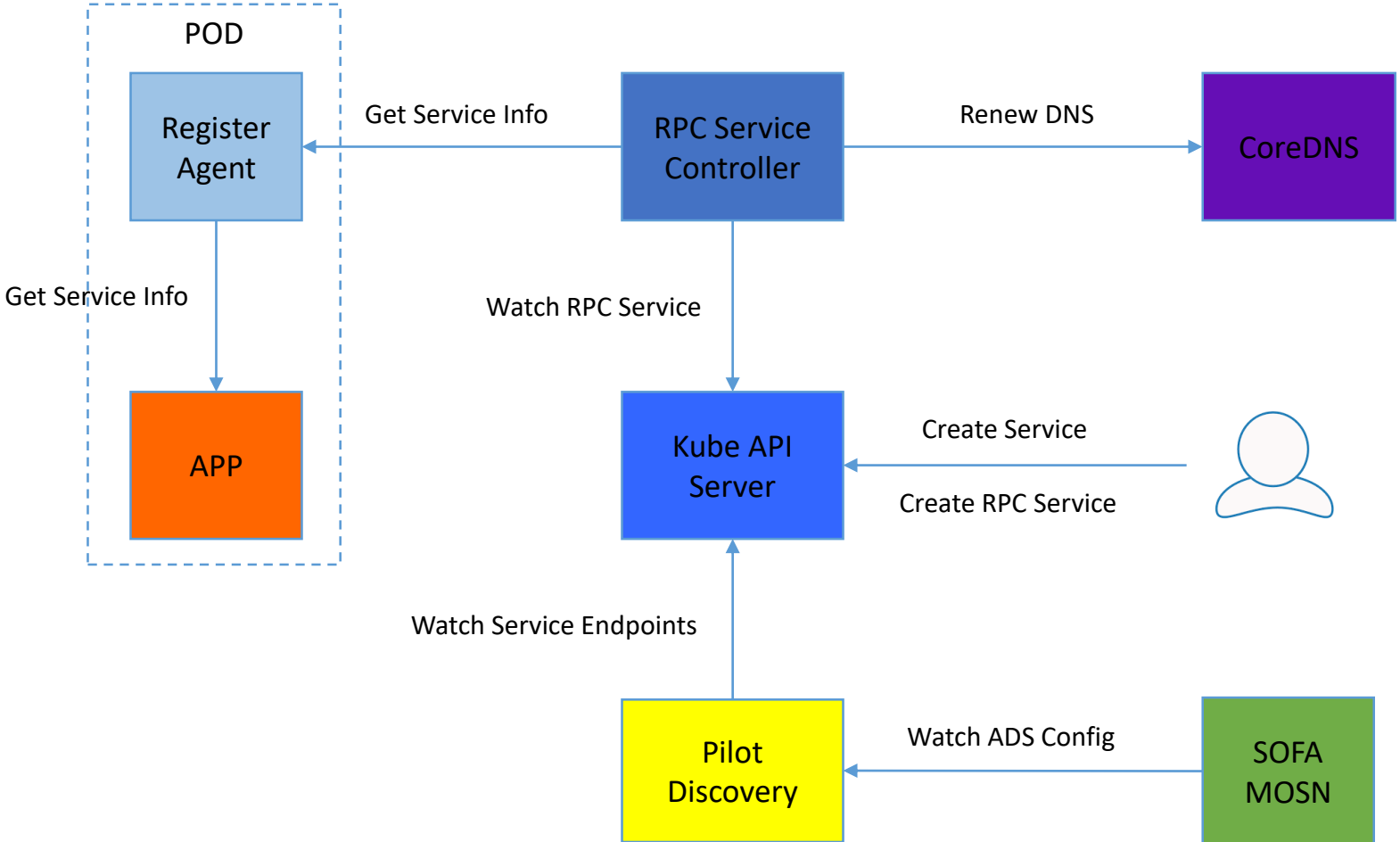
Istio 的DNS寻址



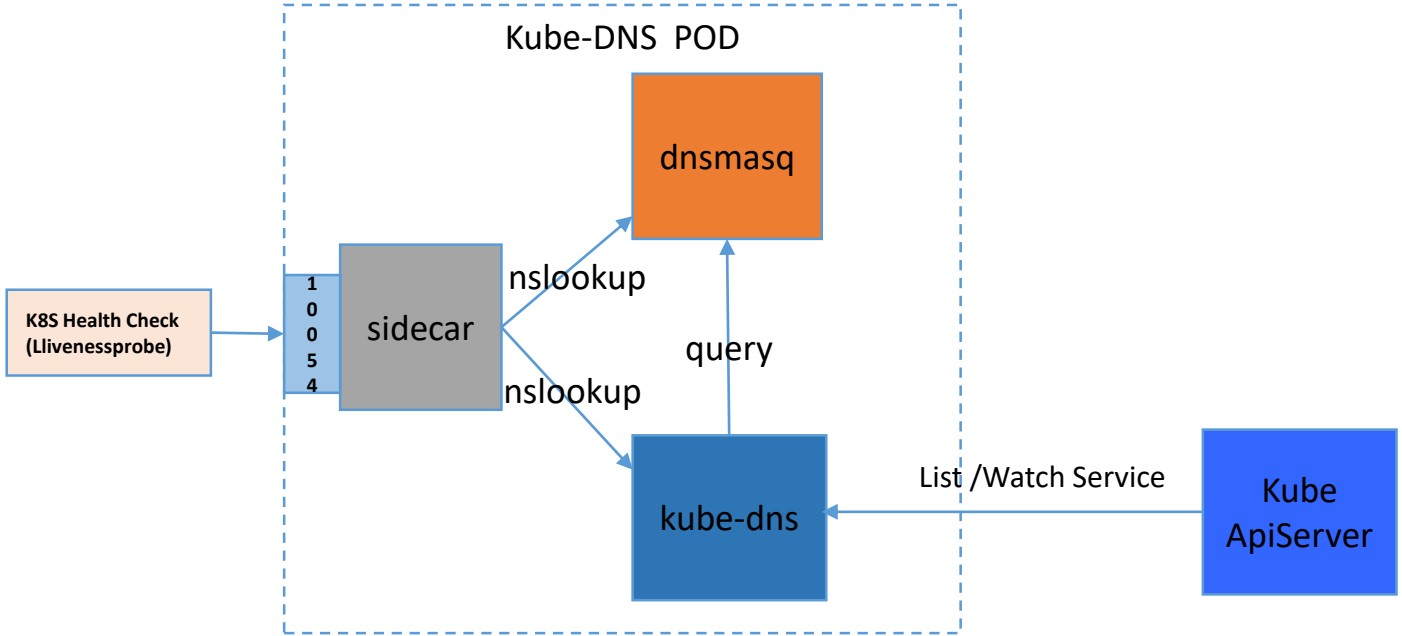
SOFAMesh 的DNS寻址



SOFAMesh 的DNS寻址方案的实现细节

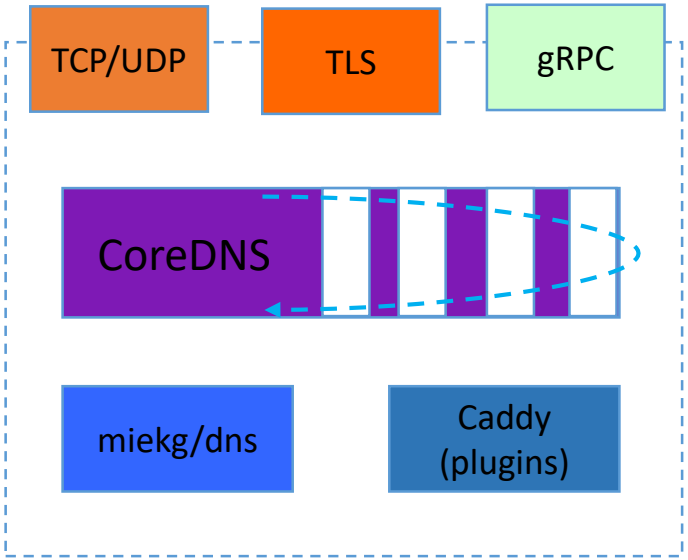


Kube-DNS 的困境



ReNew Me ? Sorry , You Can’t

CoreDNS 的力量




```
// Ordering is VERY important. Every plugin will
// feel the effects of all other plugin below
// (after) them during a request, but they must not
// care what plugin above them are doing.
var Directives = []string{
    "metadata",
    "tls",
    "reload",
    "nsid",
    "root",
    "bind",
    "debug",
    "trace",
    "health",
    "pprof",
    "prometheus",
    "errors",
    "log",
    "dnstap",
    "chaos",
    "loadbalance",
    "cache",
    "rewrite",
```

ReNew Me ？ Yes, I’m power by plugins

https://ahmet.im/blog/coredns-grpc-backends

CoreDNS Plugins

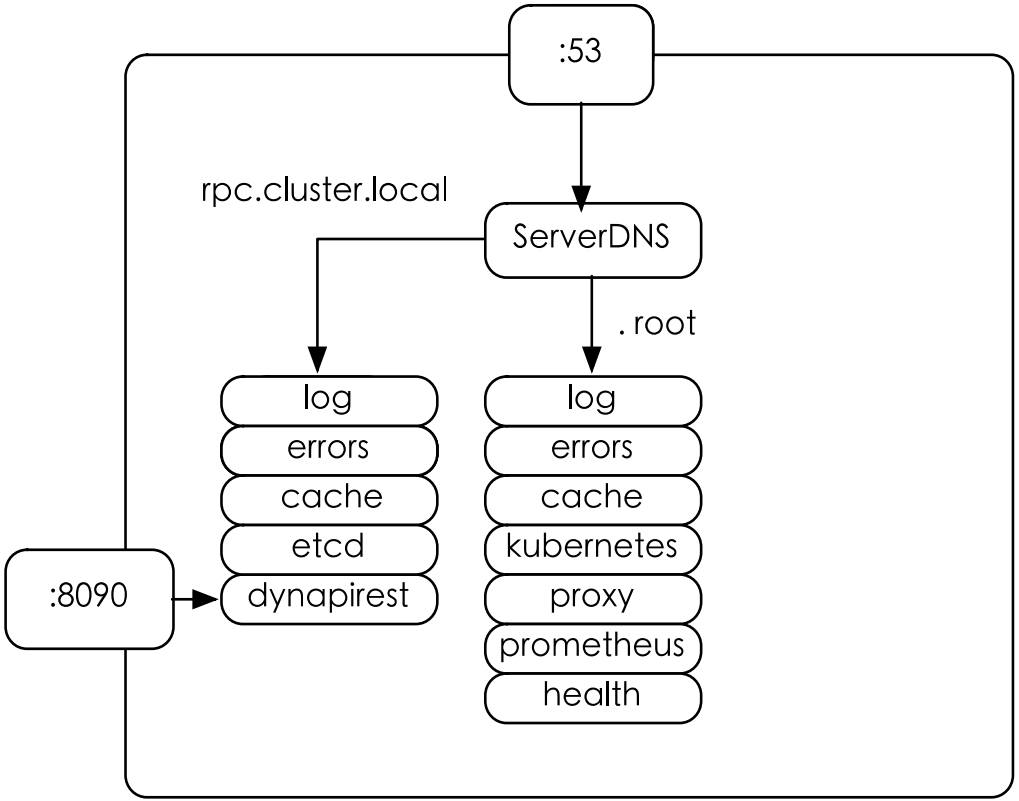
```
// If writing a response after calling another ServeDNS method, the
// returned rcode SHOULD be used when writing the response.
//
// If handling errors after calling another ServeDNS method, the
// returned error value SHOULD be logged or handled accordingly.
//
// Otherwise, return values should be propagated down the plugin
// chain by returning them unchanged.
Handler interface {
    ServeDNS(context.Context, dns.ResponseWriter, *dns.Msg) (int, error)
    Name() string
}
```

https://godoc.org/github.com/coredns/coredns/plugin

https://coredns.io/2017/03/01/how-to-add-plugins-to-coredns/

https://github.com/coredns/presentations/blob/master/A-Deep-Dive-into-CoreDNS-2018.pdf

CoreDNS 的记录更新



https://tools.ietf.org/html/rfc2136

https://cloud.google.com/sdk/gcloud/refere/dns/record-sets/transaction

https://docs.aws.amazon.com/zh_cn/Route53/latest/APIReference/API_ChangeResourceRecordSets.html

https://github.com/coredns/coredns/pull/1822

https://github.com/coredns/dynapi/pull/1

CoreDNS 的记录更新

蚂蚁金服 ANT FINANCIAL

```go
package dynapi

// Writable is an interface providing
// functionality for other plugins that wish to
// implement dynamic DNS API updates.
// The `dynapirest` plugin will scan during setup
// to find plugins implementing the interface and
// provide dynamic API REST interface.
type Writable interface {
    // GetZones returns all zones handled by the implementer of the interface.
    GetZones() []string
    // Create attempts to create a dns resource record in the zone specified in `request`.
    // Returns a nil error if successful.
    Create(request *Request) error
    // Upsert attempts to update or create a dns resource record in the zone specified in `request`.
    // Returns a nil error if successful.
    Upsert(request *Request) error
    // Delete attempts to delete a dns resource record in the zone specified in `request`,
    // by exact matching all attributes.
    // Always returns nil error due to implementation specifics in `dns` package.
    Delete(request *Request) error
    // Update attempts to update a dns resource record in the zone specified in `request`.
    // Returns a nil error if successful.
    Update(request *Request) error
    // Exists checks if a dns resource record exists in the zone specified in `request`
    // by exact matching all attributes.
    Exists(request *Request) bool
    // Exists checks if a dns resource record exists in the zone specified in `request`
    // by only matching by the `name` specified in `request`.
    ExistsByName(request *Request) bool
}
```

CoreDNS 的记录更新

vtolstov commented on 14 Sep

any progress?

syndbg commented on 14 Sep Collaborator

@vtolstov Didn't start anything. Will do it this weekend.

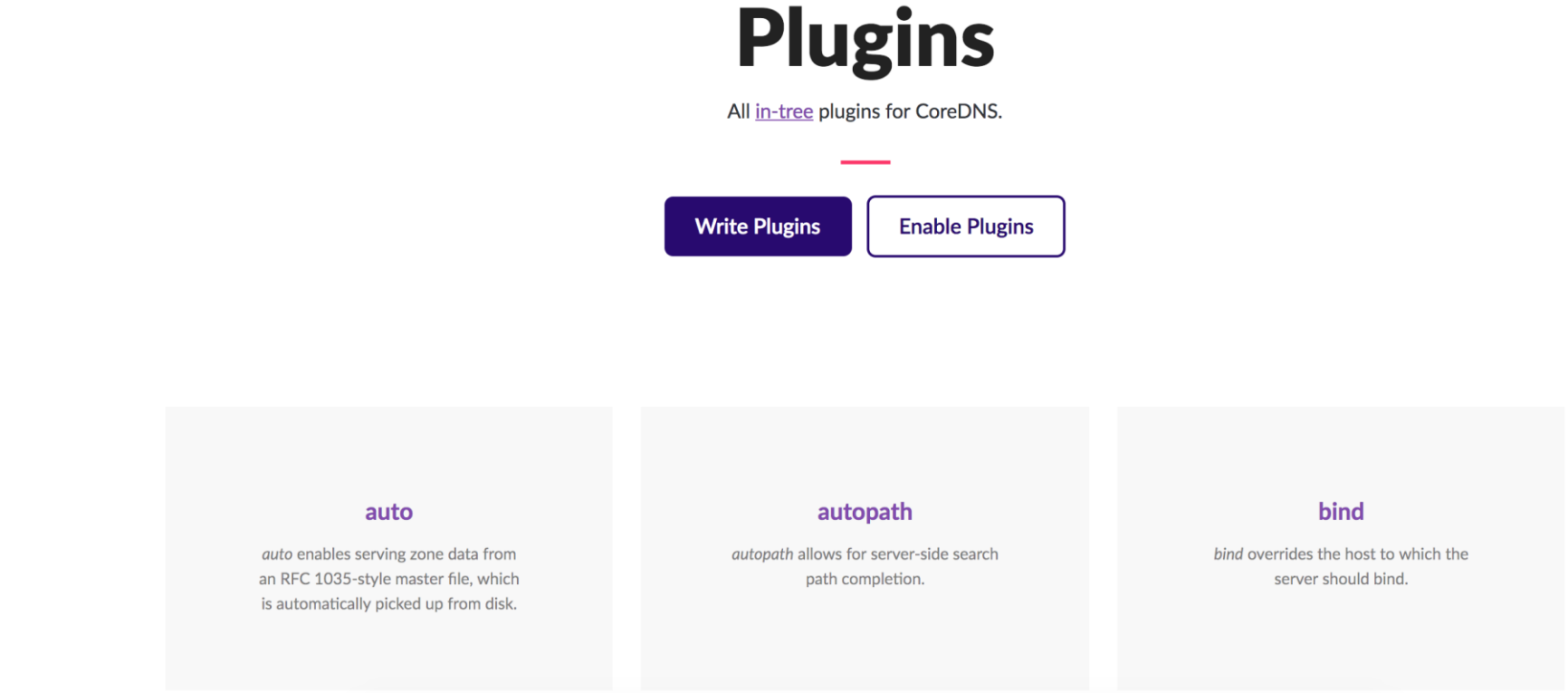
piaoyu commented on 9 Oct • edited

@syndbg it's very useful for us . And I put coredns/coredns#1822 merge into coredns and write the etcd plugin. Waiting for the dynapi plugin

CoreDNS 的记录更新

```
[redacted]
#cat record.json
{
  "zone": "rpc.szbd-uc.uaebd.local.",
  "name": "com.alipay.userservice.interface1",
  "type": "A",
  "address": "192.168.88.5",
  "TTL": 3600
}

[redacted]
#nslookup  com.alipay.userservice.interface1.rpc.szbd-uc.uaebd.local 192.168.3.10
Server:         192.168.3.10
Address:        192.168.3.10#53

** server can't find com.alipay.userservice.interface1.rpc.szbd-uc.uaebd.local: NXDOMAIN


[redacted]
#curl -X PUT   -d @record.json http://192.168.3.10:8090/dynapi
OK
[redacted]
#nslookup  com.alipay.userservice.interface1.rpc.szbd-uc.uaebd.local 192.168.3.10
Server:         192.168.3.10
Address:        192.168.3.10#53

Name:   com.alipay.userservice.interface1.rpc.szbd-uc.uaebd.local
Address: 192.168.88.5


[redacted]
#kubectl exec -it  etcd-[redacted]n kube-system sh
Defaulting container name to etcd.
Use 'kubectl describe pod/etcd-sz-kpi-s3-252 -n kube-system' to see all of the containers in this pod.
/ $
/ $ export ETCDCTL_API=3
/ $ etcdctl get /skydns/local/uaebd/szbd-uc/rpc/interface1/userservice/alipay/com
/skydns/local/uaebd/szbd-uc/rpc/interface1/userservice/alipay/com
{"host":"192.168.88.5","ttl":3600}
```

CoreDNS 的Plugins



https://coredns.io/plugins

CoreDNS 的性能

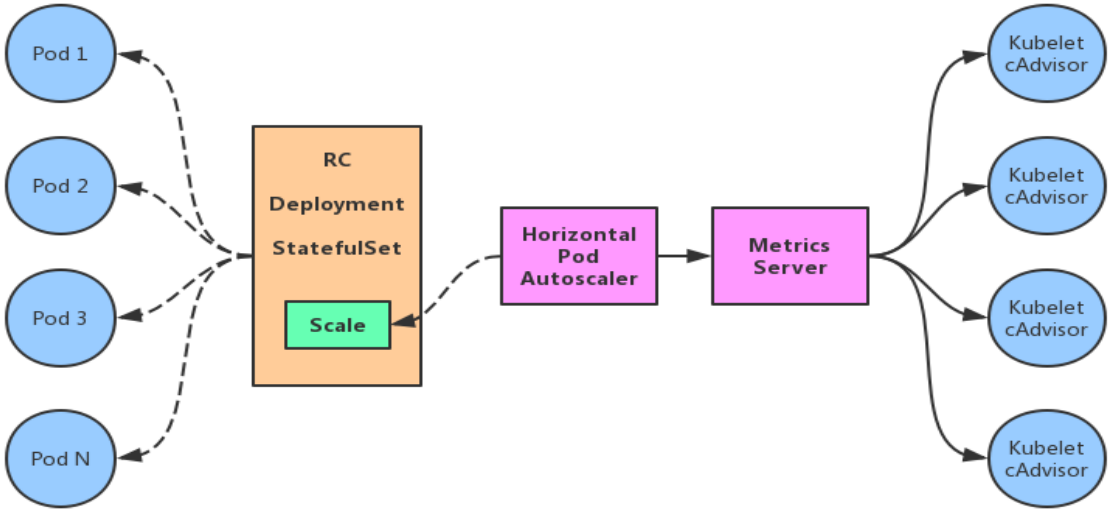
序号	对象	并发	QPS	总数	命中率/超时数
1	同机房	10	43899	2001674	(100%) |0
2	同机房	50	44369	2001674	(100%) |0
3	同机房	100	40815	2001674	(100%) |140
4	同机房	200	41094	2001674	(100%) |112
5	同机房	300	40664	2001674	(100%) |628
6	同机房	400	42226	2001674	(100%) |729
7	同机房	500	42156	2001674	(100%) |1192

Bind queryperf 测试　100个域名(5s timeout)

CoreDNS 的性能


Core-DNS性能和稳定性看板
Last 1 hour
8
6
4
2
0
19:10
19:20
19:30
19:40
19:50
20:00
{pod_name="core-dns-55c6b984d8-6dxrn"}
{pod_name="core-dns-55c6b984d8-dgbcg"}
{pod_name="core-dns-55c6b984d8-m5tx6"}
{pod_name="core-dns-test-58b8fc9687-9chvs"}
{pod_name="core-dns-test-58b8fc9687-lr8bs"}
95 MiB
76 MiB
57 MiB
38 MiB
19 MiB
0 B
2018-11-24 19:28:30
{le="+Inf"}: 76.0 K
{le="0.00025"}: 56.9 K
{le="0.0005"}: 70.6 K
{le="0.001"}: 75.0 K
{le="0.002"}: 75.8 K
{le="0.004"}: 76.0 K
{le="0.008"}: 76.0 K
{le="0.016"}: 76.0 K
{le="0.032"}: 76.0 K
{le="0.064"}: 76.0 K
{le="0.128"}: 76.0 K
{le="0.256"}: 76.0 K
{le="0.512"}: 76.0 K
{le="1.024"}: 76.0 K
{le="2.048"}: 76.0 K
{le="4.096"}: 76.0 K
{le="8.192"}: 76.0 K
5 Mil
3 Mil
0
-2 Mil
-5 Mil
-7 Mil
min
{pod_name="core-dns-55c6b984d8-6dxrn"} 684 K
{pod_name="core-dns-55c6b984d8-dgbcg"} 1.01 Mil
{pod_name="core-dns-55c6b984d8-m5tx6"} 741 K
QPS
80 K
60 K
40 K
20 K
0
Error
1.0
0.5
0
-0.5
-1.0

CoreDNS 的横向扩展



1、按照CPU 的维度

2、按照QPS 的维度(Custom Metrics)

CoreDNS 的正式发布

蚂蚁金服 ANT FINANCIAL

Kubernetes 1.13 之后 CoreDNS 作为Kubernetes默认的DNS服务

- https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG-1.13.md
- https://github.com/kubernetes/kubernetes/pull/69883

4

1 Service Mesh演进路线

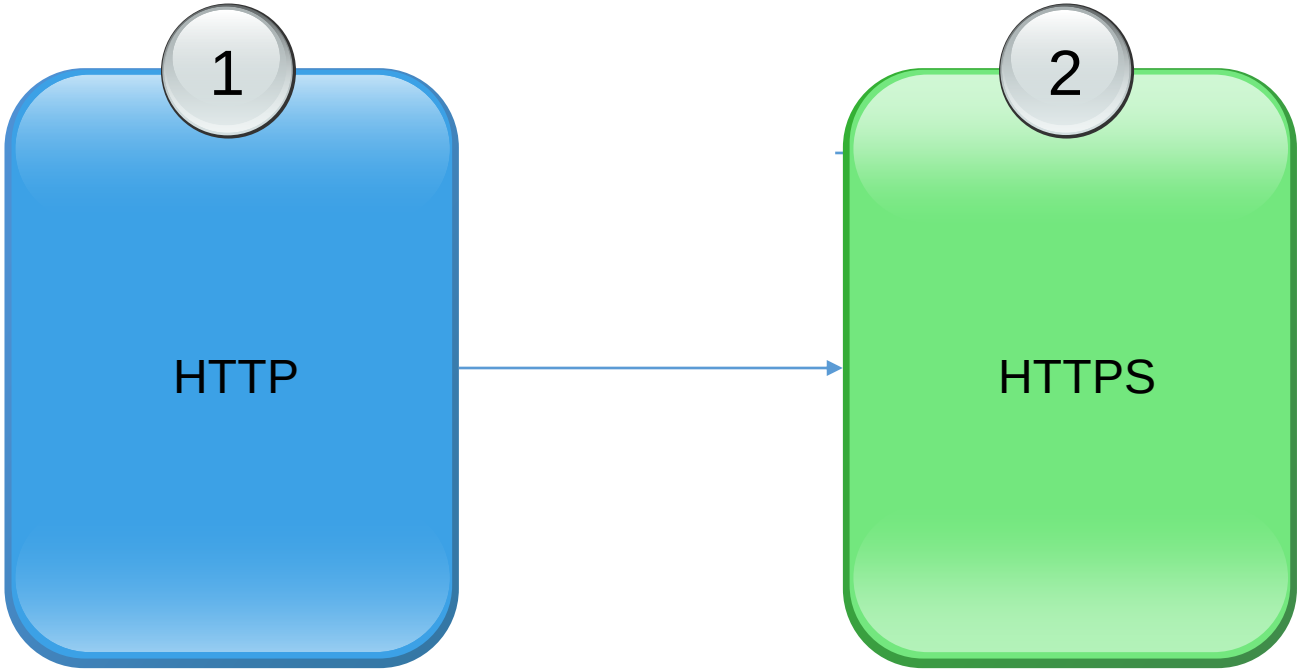
2 实现平滑迁移的关键

3 DNS寻址方案的演进

4 DNS寻址方案的后续规划

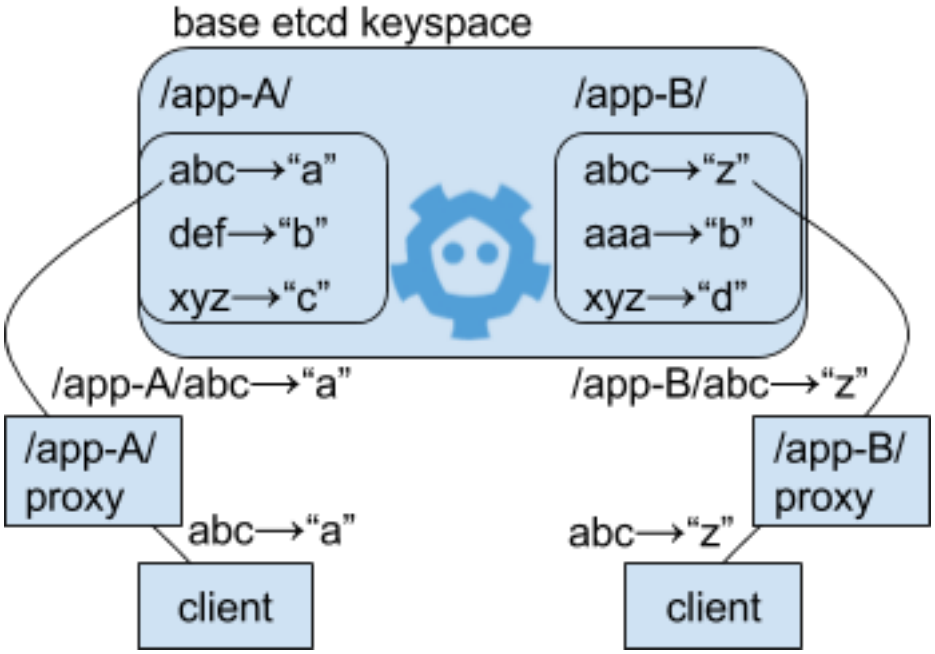
5 总结

CoreDNS DynAPI 安全加强




```
#curl -X PUT    -d @record.json http://192.168.3.10:8090/dynapi
OK
```

CoreDNS 后端Etcd Watch性能优化



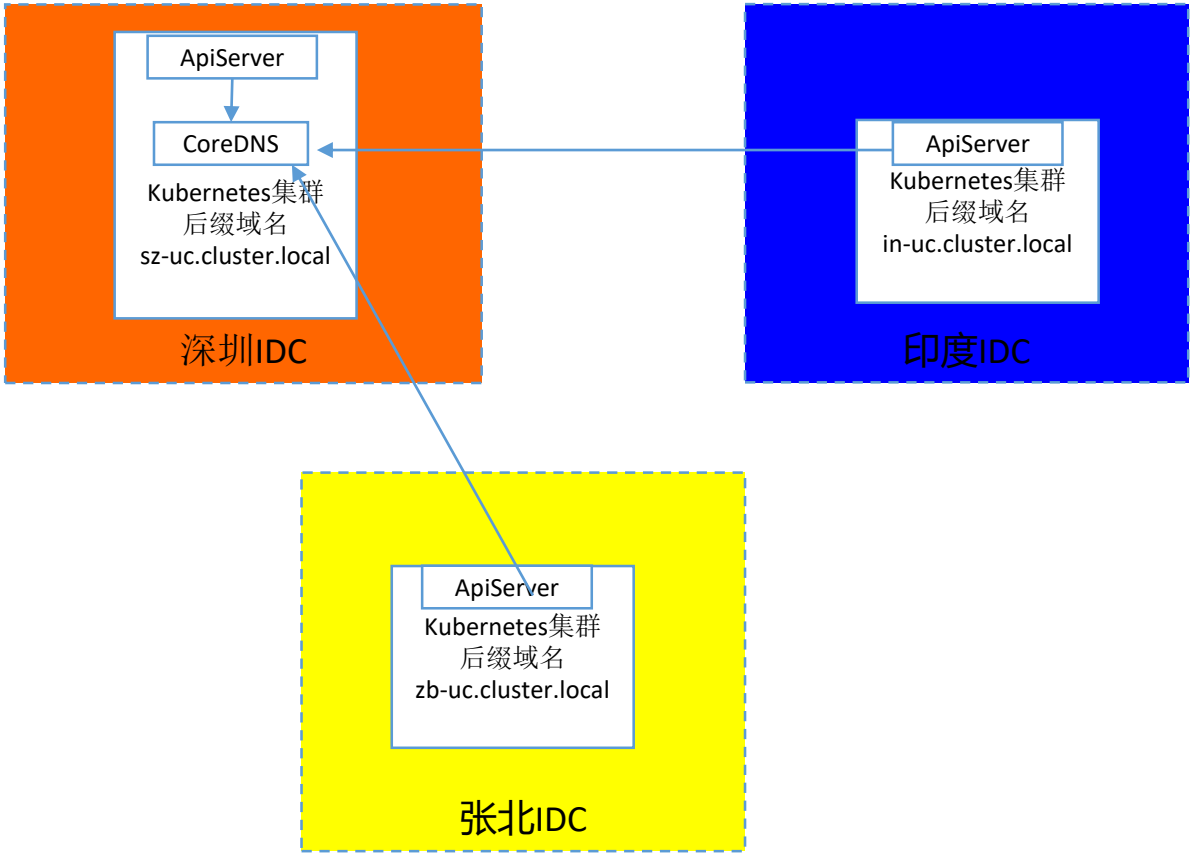
https://coreos.com/blog/etcd-3.2-announcement

全球域名整合

kubernetai

kubernetai - serve multiple Kubernetes within a Server.

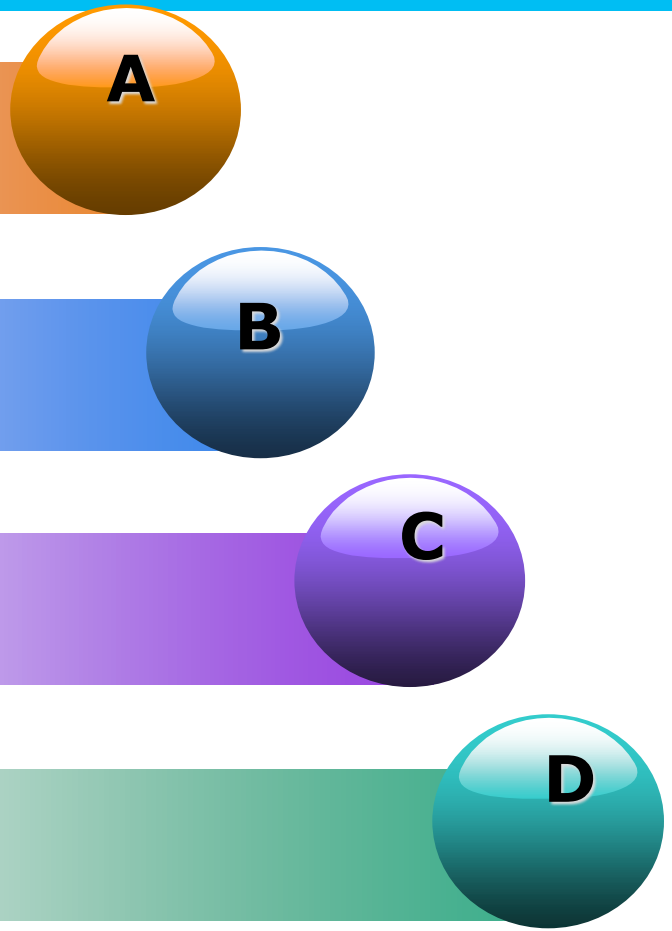
Source Home



5

1. Service Mesh演进路线
2. 实现平滑迁移的关键
3. DNS寻址方案的演进
4. DNS寻址方案的后续规划
5. 总结

总结


A
体: Service Mesh演进路线
B
面: 实现平滑迁移的关键
C
线: SOFAMesh 的DNS寻址
D
点: CoreDNS 的单点突破