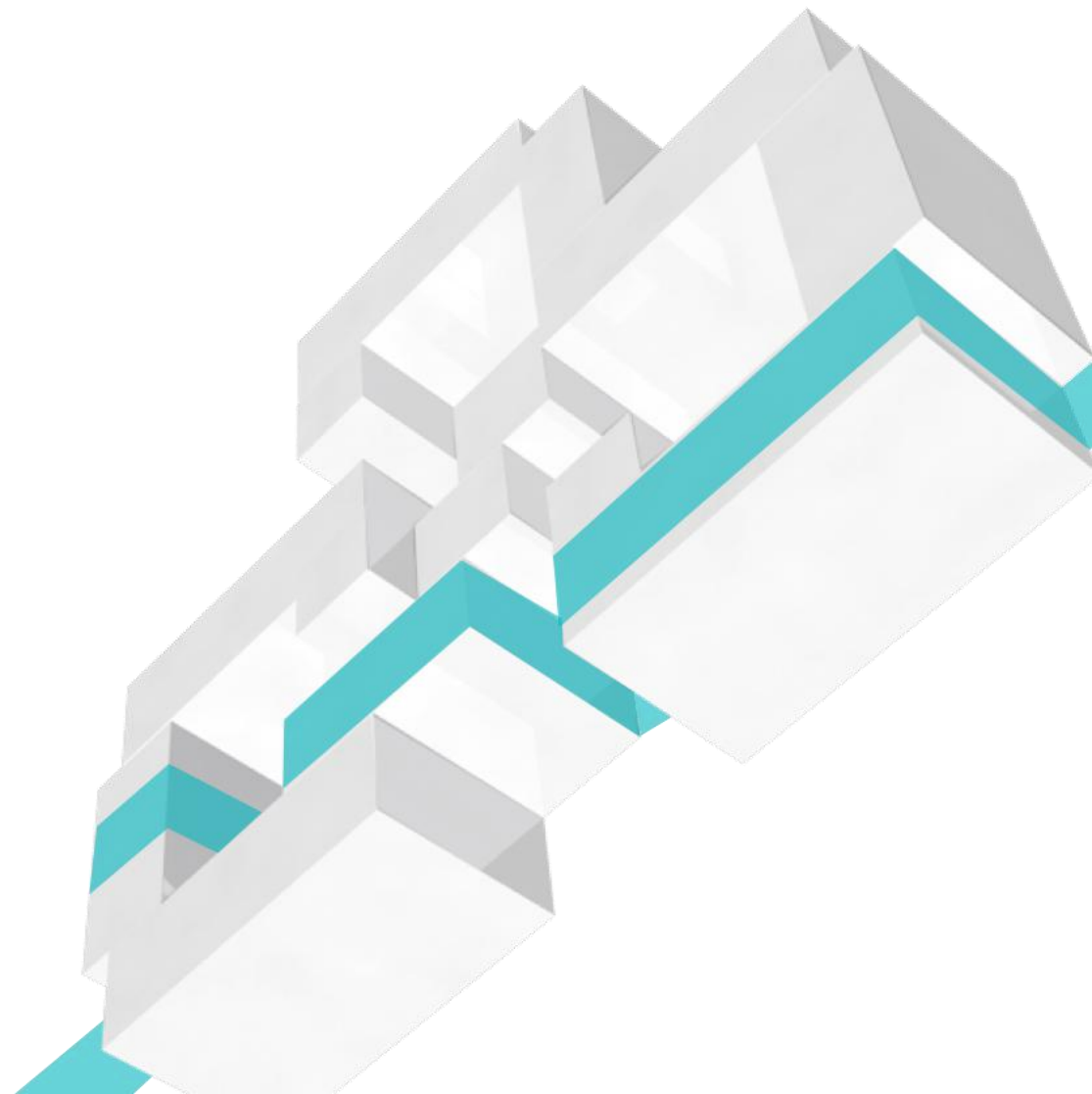


Kubernetes自动伸缩实现

演讲人：马老师

日期：2022/04/21

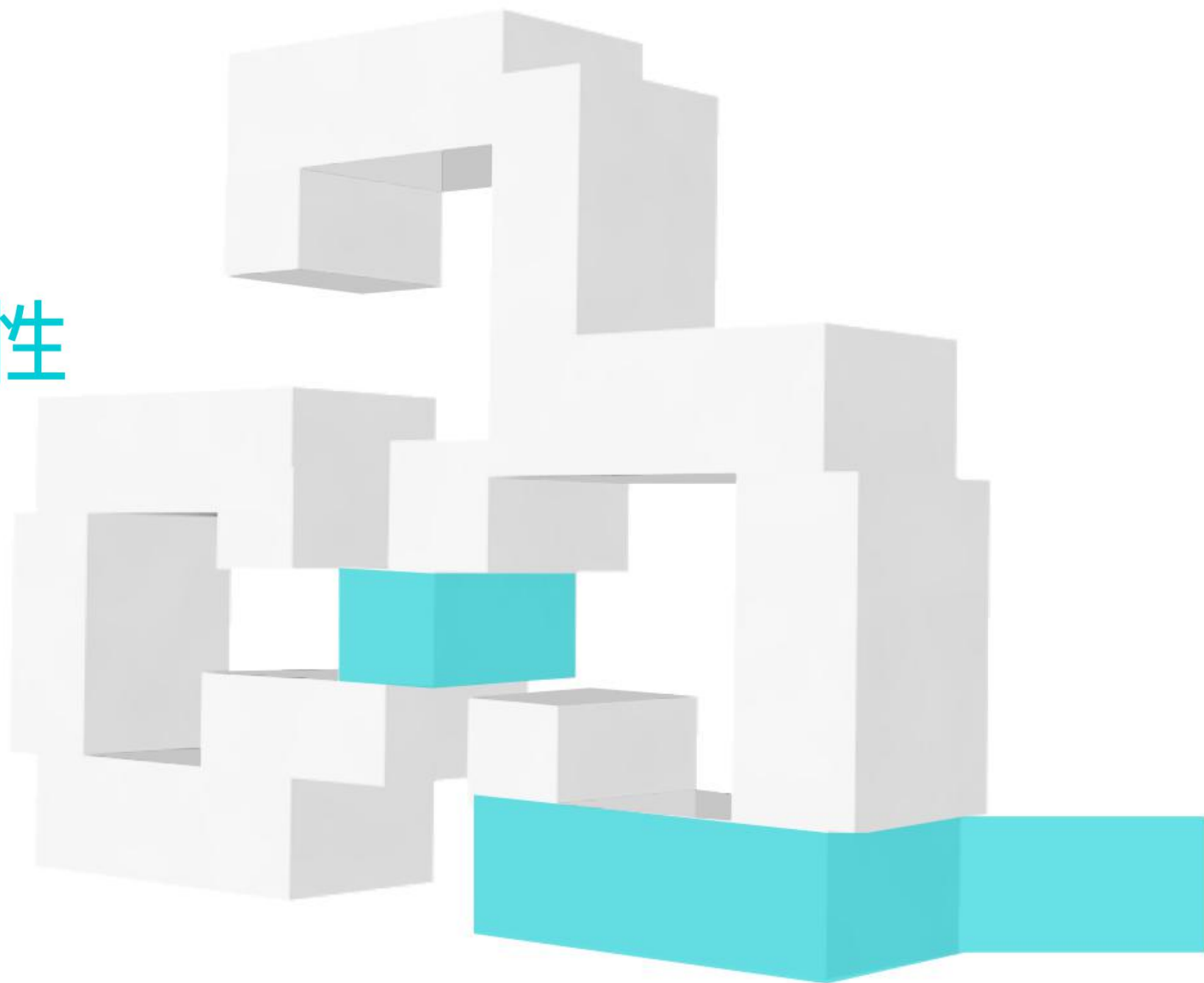


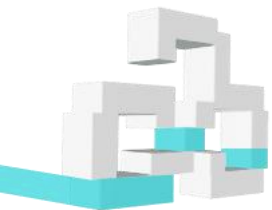
目录

- 01 | 自动伸缩能力的重要性
- 02 | HPA工作原理及基本用法
- 03 | 基于自定义指标的HPA实现



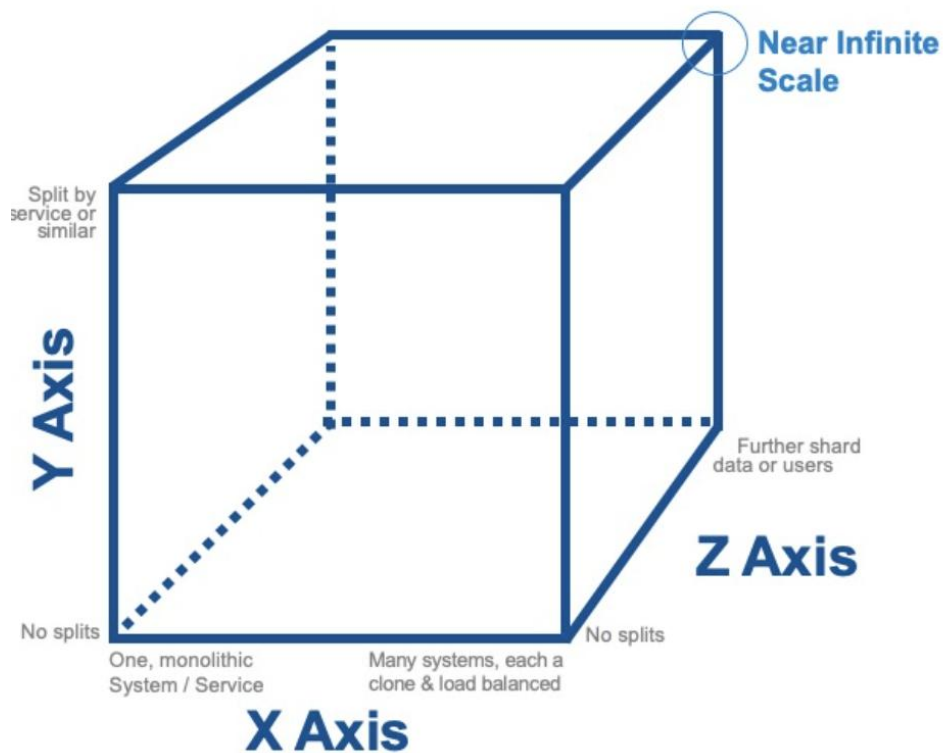
自动伸缩能力的重要性



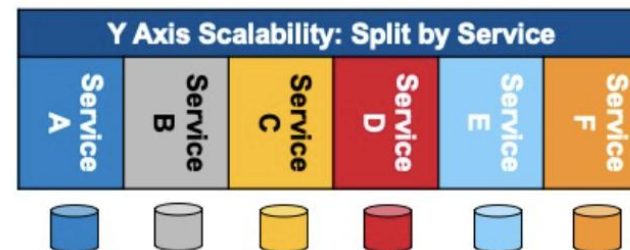


从可扩展性说起

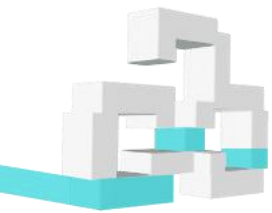
X轴：水平复制
Y轴：功能扩展
Z轴：数据分区



X Axis Scalability: Replicate & LB	
Web Tier	Replicate Web Servers & Load Balance
App Tier	Store Session in browser or separated Object Cache to horizontally scale app tier independent of web tier
DB Tier	Use Read-Replicas for read-only use cases like reporting, search, etc.

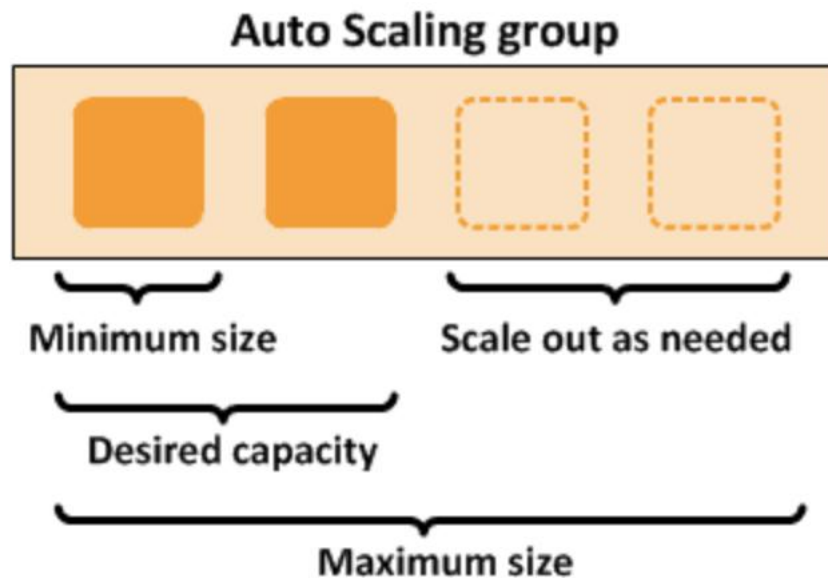


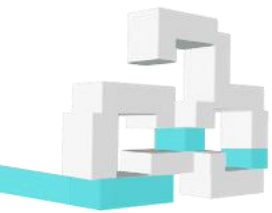
Z Axis Scalability: Segment by Customer			
NA		EU	
POD 1	POD 2	POD 3	POD 4



什么是自动伸缩 (Auto-scaling)

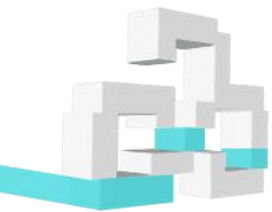
- 一种自动扩展计算资源的云计算技术，资源实例数量会基于用户需要动态变化。





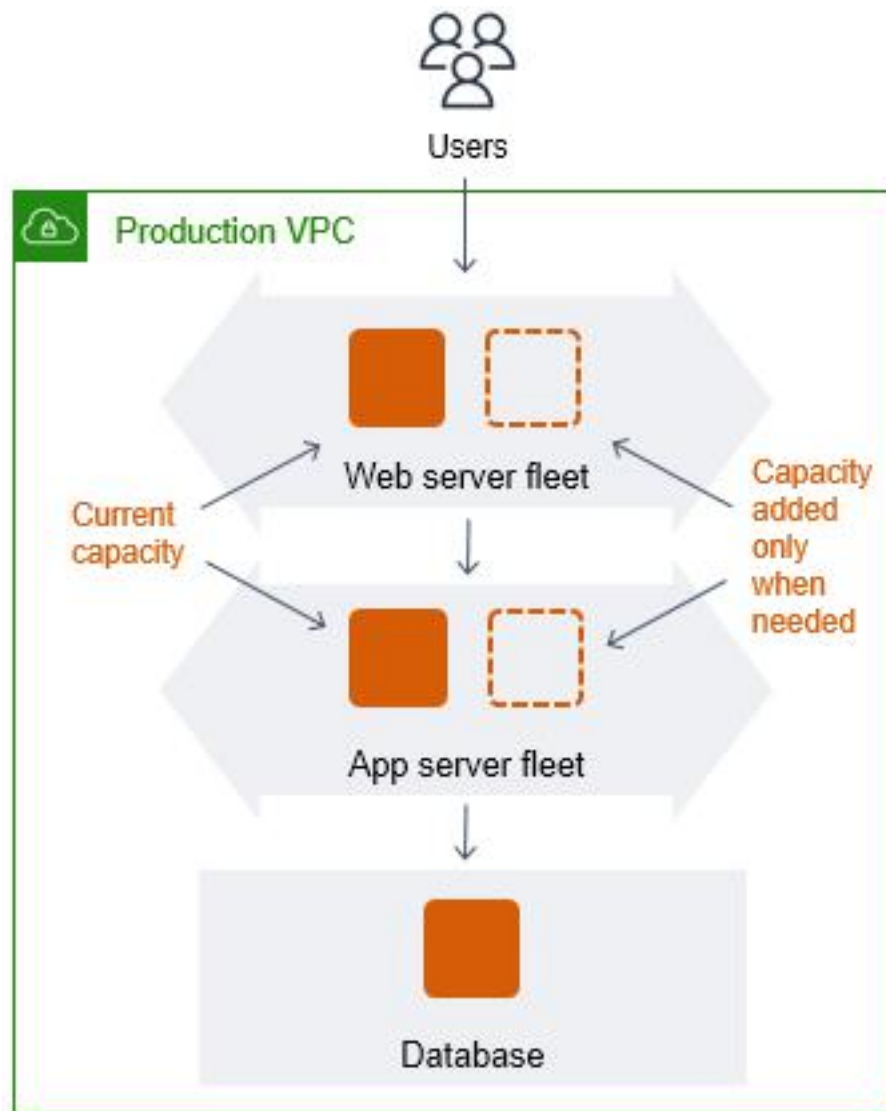
自动伸缩的重要性

- 更好的容错性（Fault-tolerance）
 - 及时、快速应对负载压力
- 更好的可用性（High Availability）
 - 高可用的本质：冗余
- 更好的成本管理（Cost saving）
 - 按需付费
- 云原生应用的必备能力
- 缺点：难以识别非正常流量（Ddos攻击）

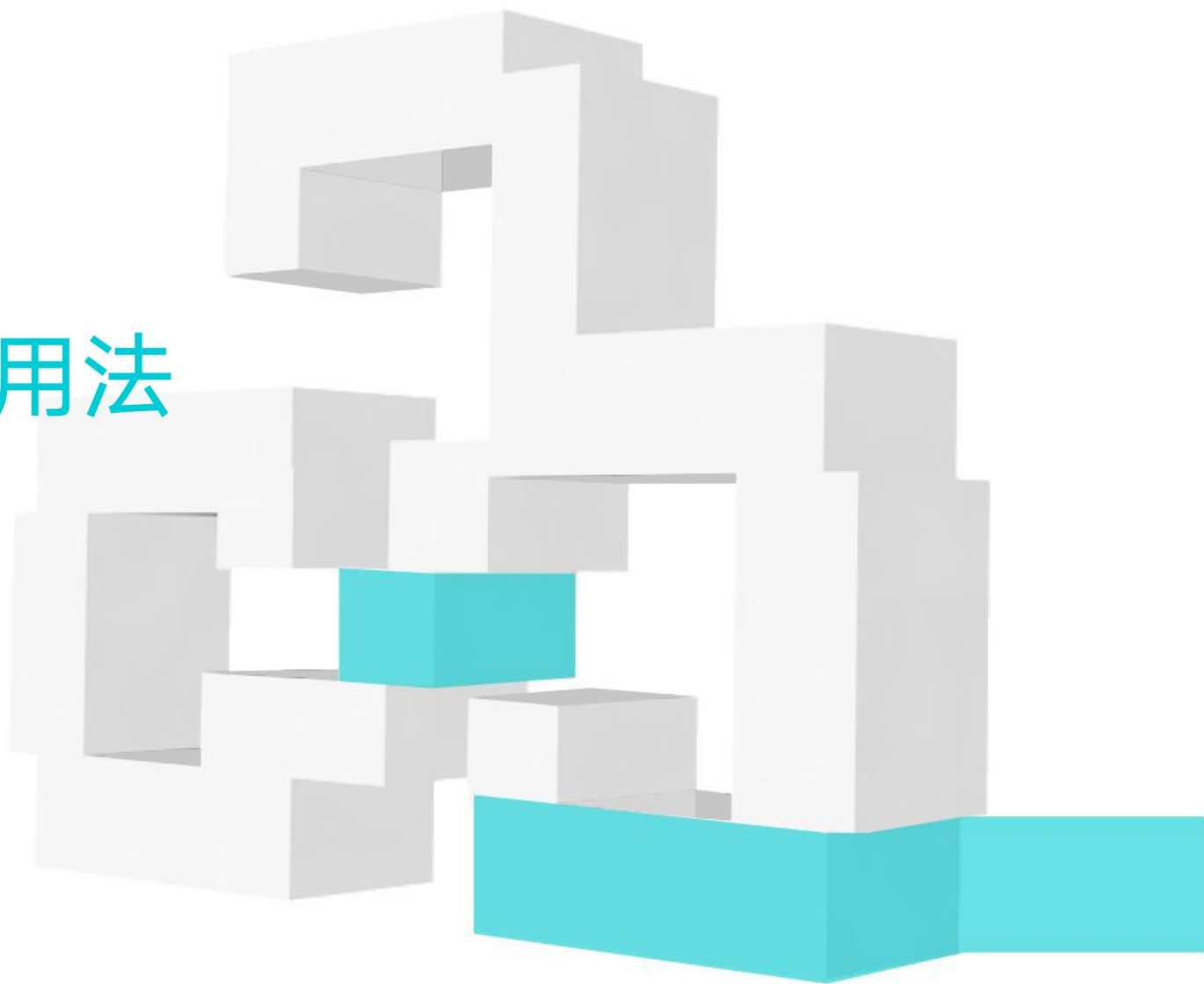


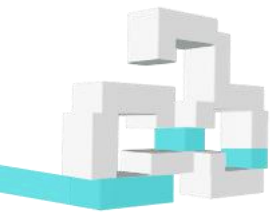
常见的自动伸缩的实现

- 云提供商的基本服务
- Serverless
- Kubernetes HPA



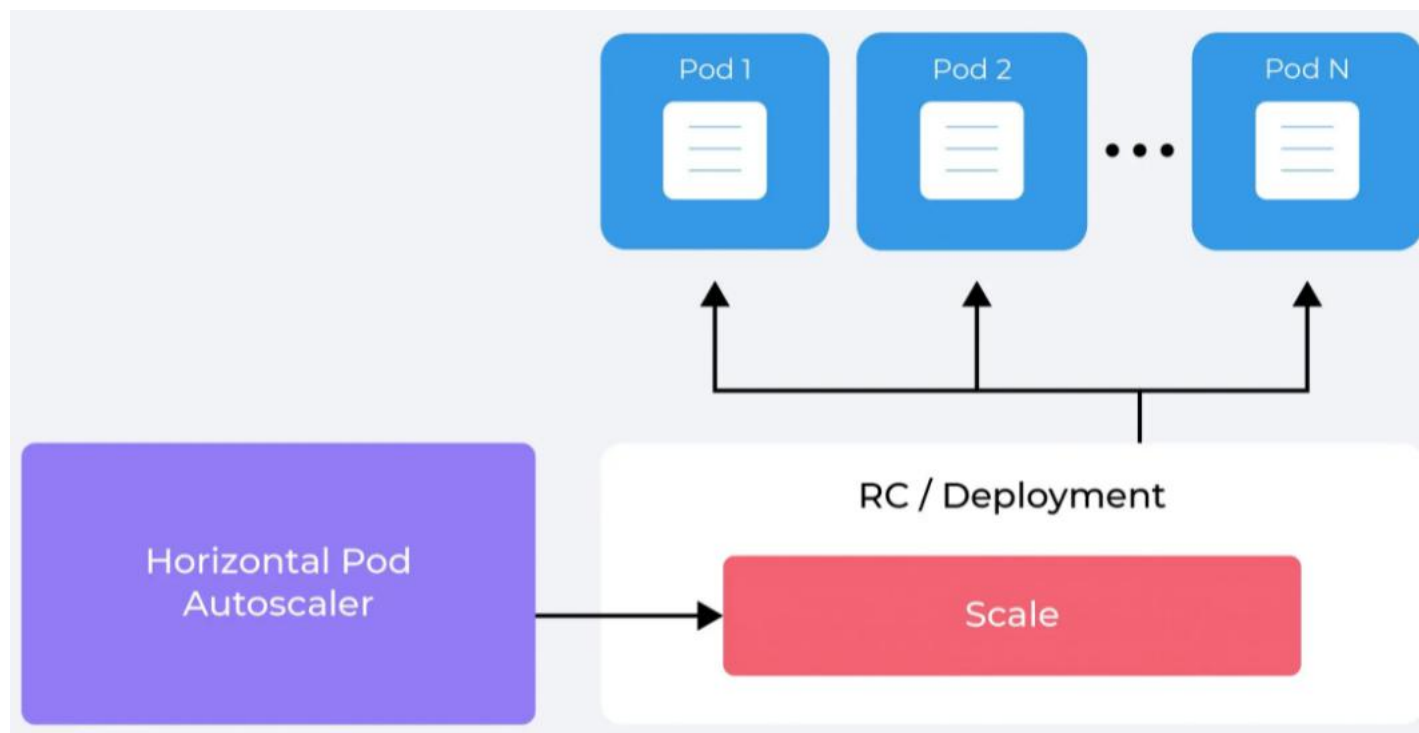
HPA工作原理及基本用法

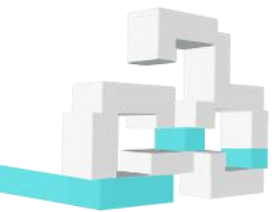




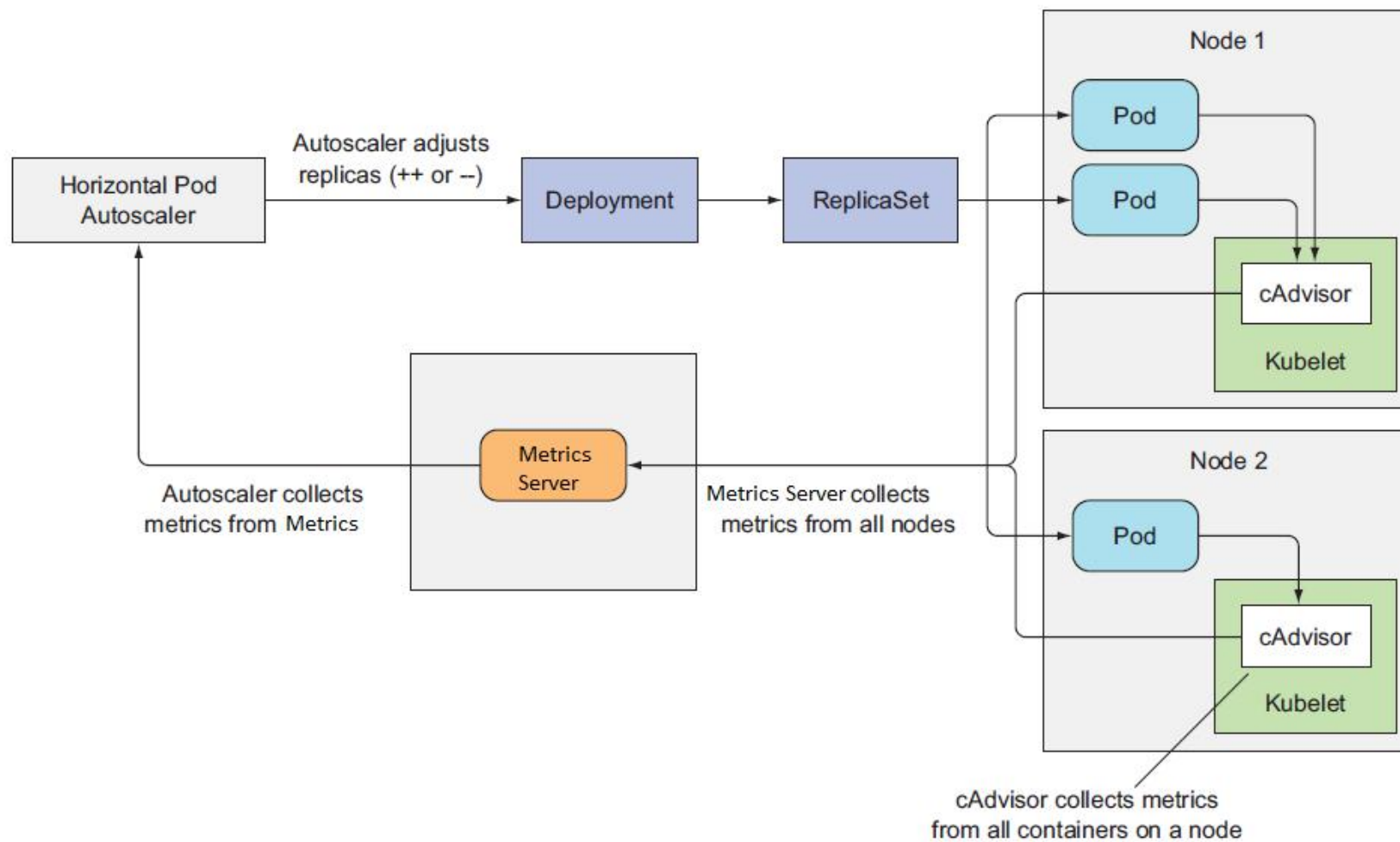
Kubernetes里的自动伸缩实现 - HPA

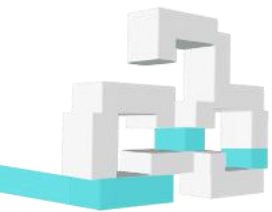
Kubernetes Horizontal Pod Autoscaler (HPA)





HPA工作原理





基于默认资源指标实现的HPA

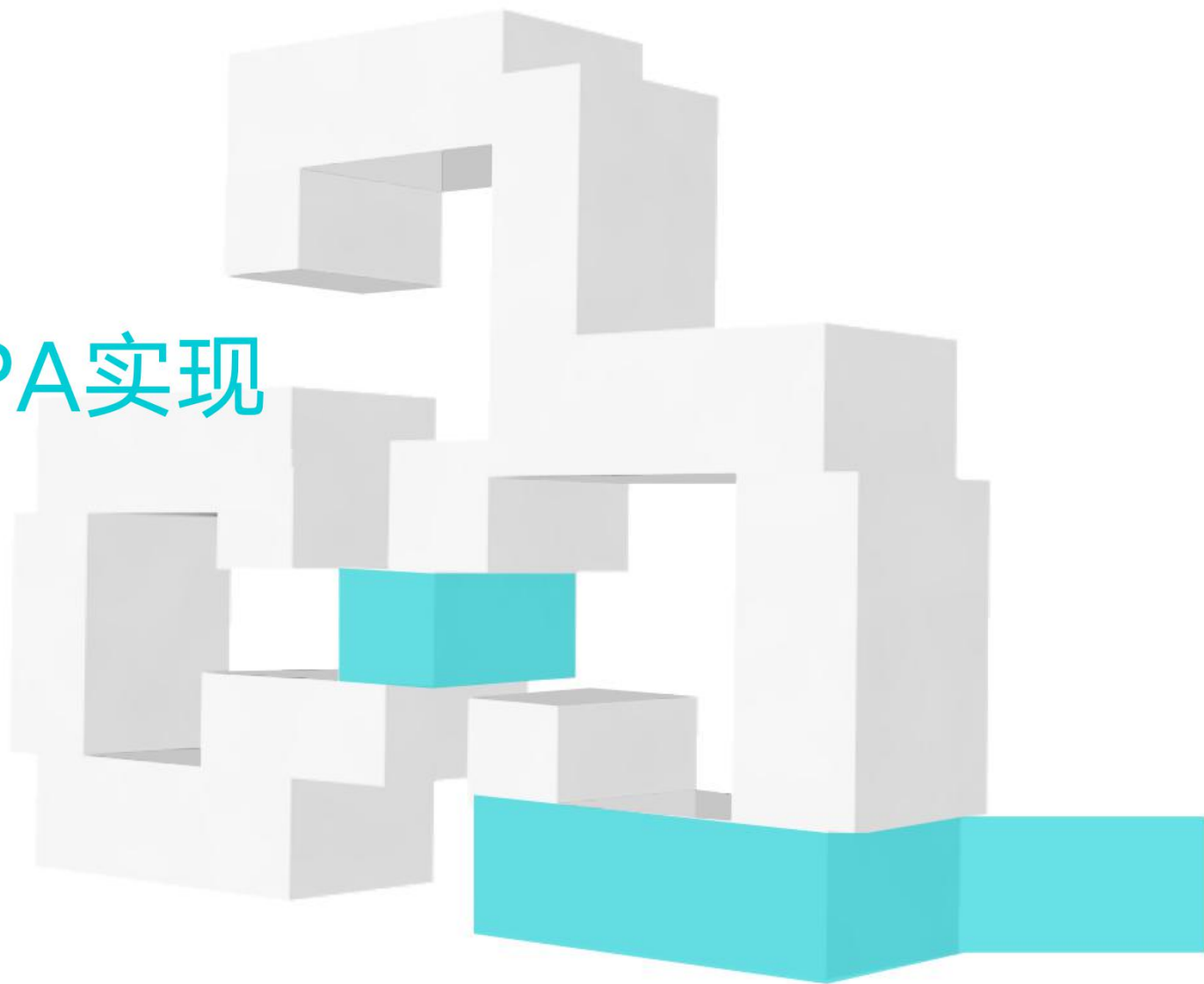
默认指标：CPU、内存

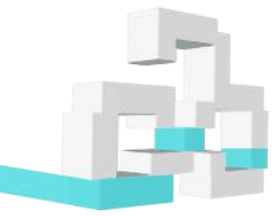
检查间隔：默认15s

公式： $\text{ceil}[\text{当前副本数} * (\text{当前指标} / \text{期望指标})]$

```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: php-apache
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: php-apache
  minReplicas: 1
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 50
```

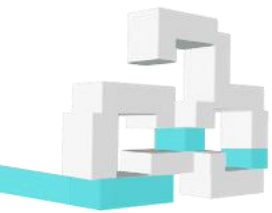
基于自定义指标的HPA实现



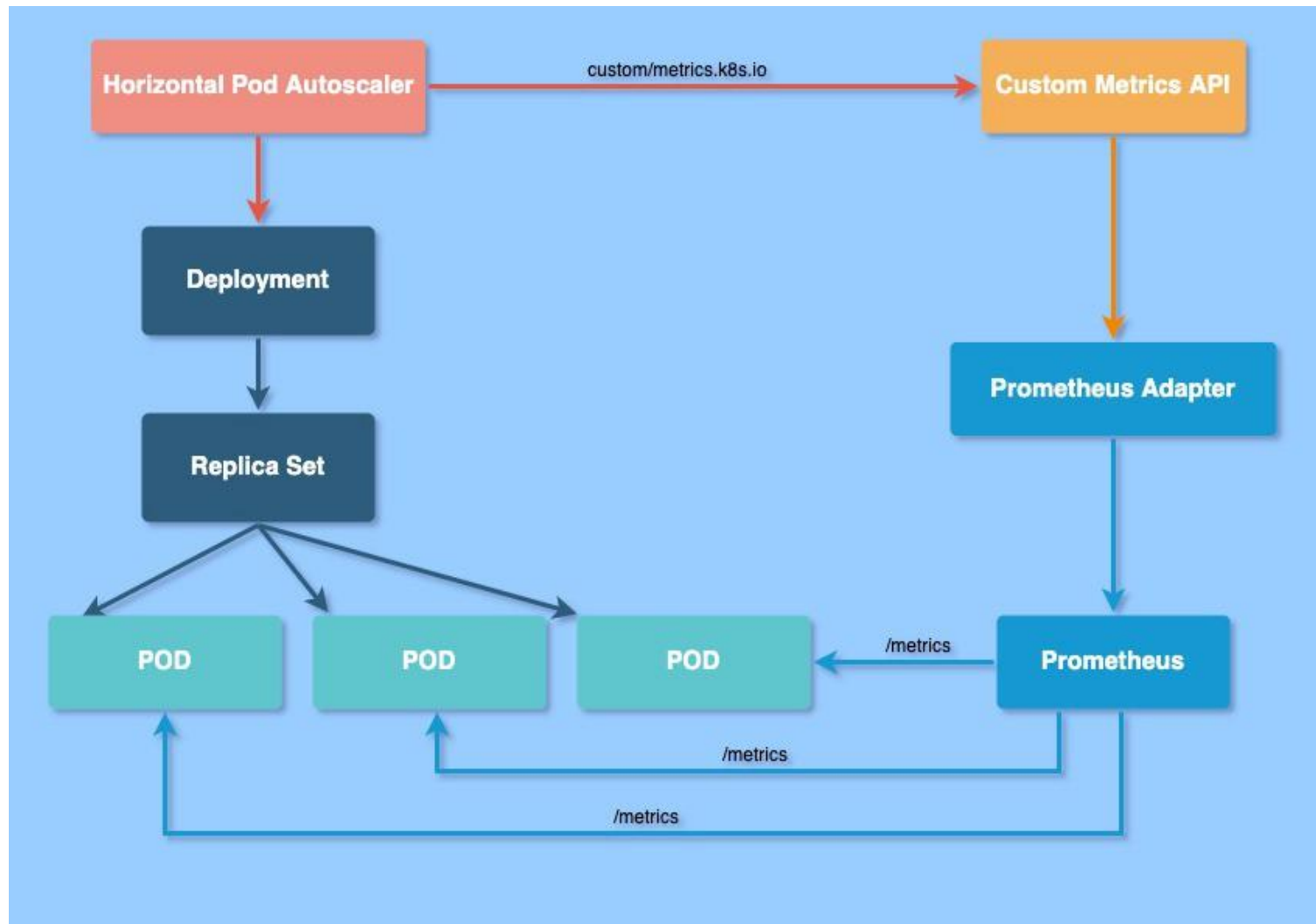


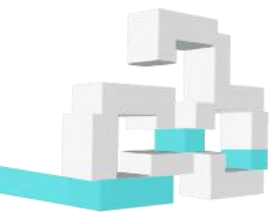
为什么需要自定义指标实现自动伸缩？

- 默认指标单一，不能反映真实的负载
- 无法满足定制需要
- 基于多指标扩容

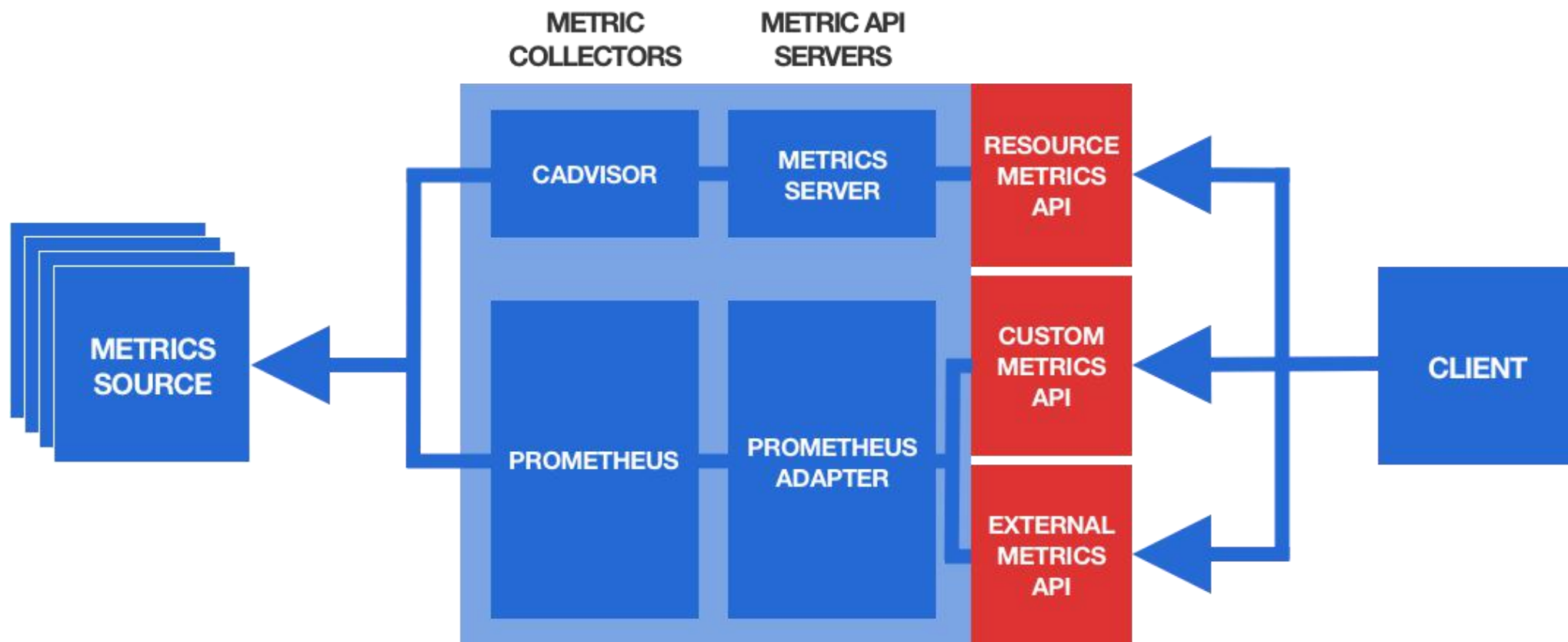


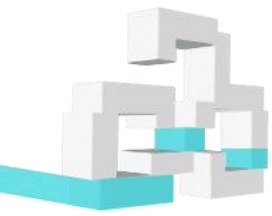
自定义指标实现及原理





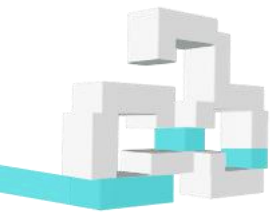
自定义指标实现及原理





基于QPS的HPA实现 - 定义指标

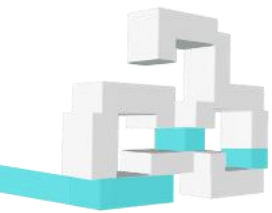
```
import (  
    //1. 引入依赖包  
    "github.com/prometheus/client_golang/prometheus"  
    "github.com/prometheus/client_golang/prometheus/promhttp"  
)  
  
var (  
    //2. 定义指标  
    HTTPRequests = prometheus.NewCounterVec(  
        prometheus.CounterOpts{  
            Name: "http_requests_total",  
            Help: "Number of the http requests received since the server",  
        },  
        []string{"status"},  
    )  
)  
  
func init() {  
    //3. 注册指标  
    prometheus.MustRegister(HTTPRequests)  
}  
  
func main() {  
    http.HandleFunc("/", func(w http.ResponseWriter, r *http.Request) {  
        //...  
        switch path {  
            //4. 暴露接口  
            case "/metrics":  
                promhttp.Handler().ServeHTTP(w, r)  
            //...  
        }  
    })  
    //5. 调用指标  
    HTTPRequests.WithLabelValues(strconv.Itoa(code)).Inc()
```

基于QPS的HPA实现 – 为adapter定义规则

```
rules:
  custom:
  - seriesQuery: 'http_requests_total'
    resources:
      template: <<.Resource>>
      name:
        matches: "http_requests_total"
        as: "qps"
      metricsQuery: sum(rate(<<.Series>>{<<.LabelMatchers>>}[1m])) by (<<.GroupBy>>)
prometheus:
  url: http://prometheus.monitoring.svc.cluster.local
  port: 9090
```

可通过kubectl get --raw /apis/custom.metrics.k8s.io/v1beta1测试



基于QPS的HPA实现 - 定义 HPA

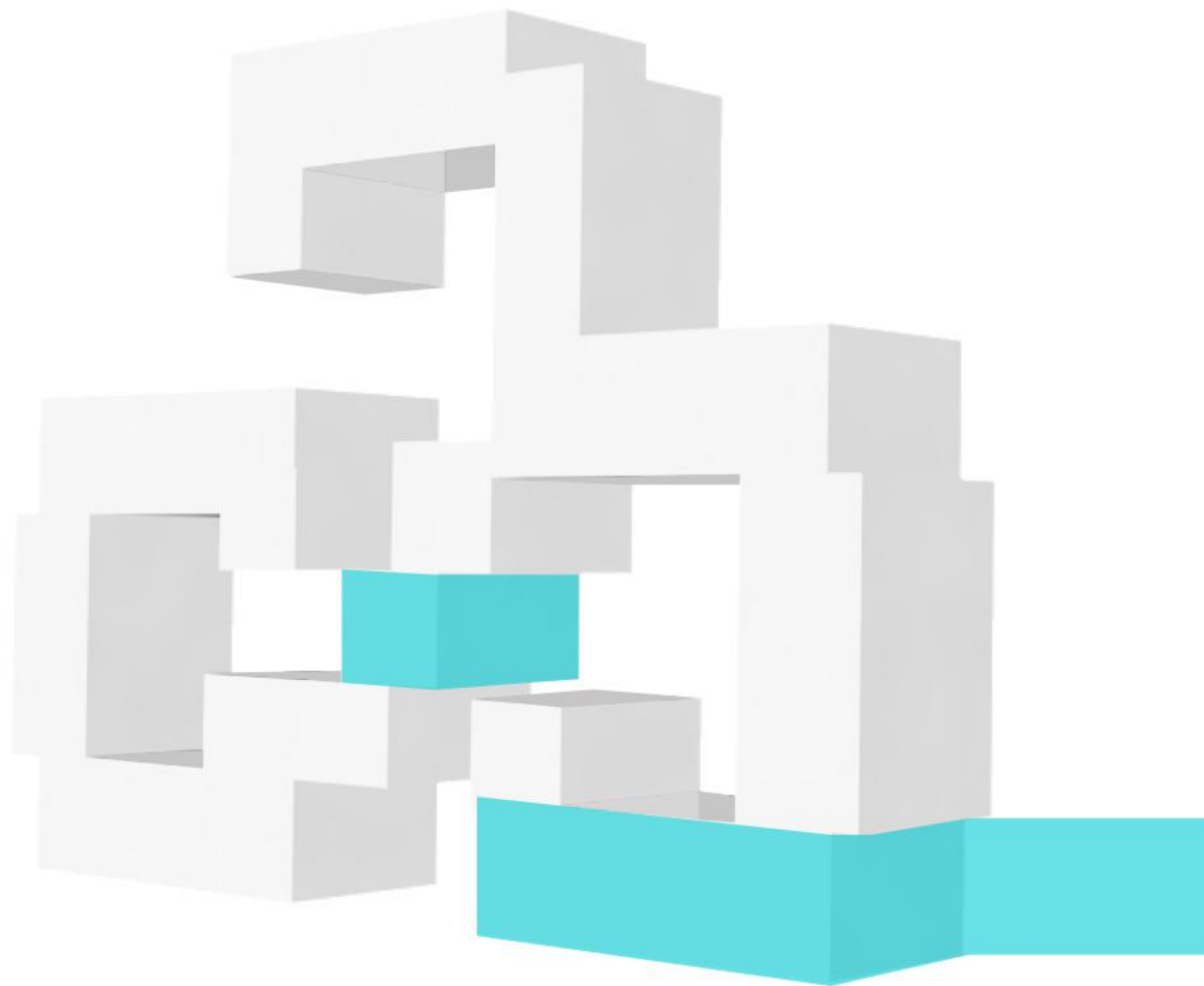
```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: httpserver
spec:
  minReplicas: 1
  maxReplicas: 10
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: httpserver
  metrics:
  - type: Pods
    pods:
      metric:
        name: qps
      target:
        averageValue: 1000
        type: AverageValue
```

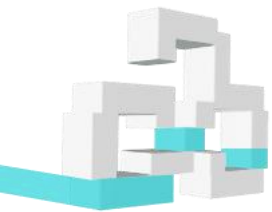
感谢观看

Q&A



4006661332
www.cloudwise.com





广告时间：云智慧透视宝 - 产品功能架构

数据展示

APM Server

APM APP RUM

APM Browser

APM Host

数据分析

应用拓扑

请求分析

页面分析

用户追踪

磁盘读写分析

JS分析

渲染性能分析

堆栈详情

流量监控

链式追踪

端到端分析

内存资源跟踪

Ajax分析

响应阶段分解

错误异常

CPU监控

CDN分析

中间件分析

进程占用分析

资源加载时序

.....

事件通知



告警

接口



API接口

数据存储

ClickHouse

HBase

Elasticsearch

PostgreSQL

GreenPlum

数据采集

Java

PHP

Python

.NET

Andriod

IOS

NodeJs

Javascript
†

Go