1.新的kv存储系统

2.新KV存储系统在intel非易失性内存和Intel ssd性能对比测试

3.新KV存储系统的架设

4.集群控制平台后端2.0的开发：pika-stark、stark-admin部分的单元测试、mysql数据库之间交互

Gin框架:<https://www.kancloud.cn/adapa/gingolang/1124987>

5.grafana新的监控页面和设计指标（集群的机器数、proxy节点数、data节点、集群总数据量），grafana页面的手动注册、自动注册

1. 注册服务到consul，多次调用幂等（通过api注册服务http://consul:8500/v1/agent/service/register ）

json、返回string、http.defaultclient.do把req变成resp

gin包rest api

**创建router对象(gin.Default())；通过router创建监听。设置监听回调；router.run()**

gin进行了封装，把request和response都封装到 gin.Context的上下文环境

sqlx包与mysql进行数据交流

交互命令**User:Pwd@tcp(Host:Port)**

const dsn = "root:123456@tcp(127.0.0.1:3306)/database"/?charset=utf8mb4

db, err = sqlx.Connect("mysql", dsn)

func Connect(driverName, dataSourceName string)

sql := `update user set name = ?, age = ? where id = ?`创表也是这个函数

res, err := DB.Exec(sql, "LGX", 28, 20)MustExec会返回panic

查询一行数据

var u user

if err := db.Get(&u, sqlStr, 1); err != nil {

查询多行数据

var users []user

if err := db.Select(&users, sqlStr, 0); err != nil {

Mysql时间戳用的timestamp，1970-01-01 00:00:01.000000到2038。有时区的自动转换。非小数部分4字节，datetime5字节，5.6.4之前是8字节。小数部分都是0到3字节。On update current\_timestamp()



http请求报文

1.请求方法（主要就是post(非覆盖，用于new新增)和get(覆盖，用于update资源)、delete、put。Head和trace、options）

2.url地址和报文头的host属性相加为完整的url

3.协议名称/协议版本

4.报文头

Accept: 告诉server，client接受什么类型的响应 application/json

Referer：req是从哪个url来的

Cache-control: 对缓存控制

Accept-encoding：告诉server，接受什么编码格式

Host：要请求resource所在的主机和端口

Authorization：当客户端接收到来自WEB服务器的 WWW-Authenticate 响应时，用该头部来回应自己的身份验证信息给WEB服务器

1. 报文体: kv组成的，多个请求参数的数据



http响应报文

1. 报文协议/版本
2. 状态码/状态描述

301资源被转移到其他URL 303重定向到其他页面304资源未修改 404请求资源不存在

1. 响应报文头

Date 设置消息发送的日期和时间

1. 响应报文体

http状态码报错

400 客户端错误(请求语法有问题、无法完成请求)

500 服务器错误(服务器在处理请求时发生了错误)

Get和Post区别

都是HTTP报文的一个参数

Get 请求参数在请求行，在url后，会被浏览器历史记录保留；受限于url的长度，不太安全；查询信息和想通过url分享的信息；幂等，同一请求的重复发送效果相同。本意是获取

Post

请求参数请求体中，请求参数不会被保留；传送数据量比较大，80kb/100kb，相对安全；私密性的信息请求；不幂等，重复发送效果不同。本意是更新

http包

Get data都是json的[]byte

Body是io.reader strings.NewReader(string)\*reader

http.NewRequest(method,url,body) (\*request,error)

http.DefaultClient.Do(\*request)(\*response,error)

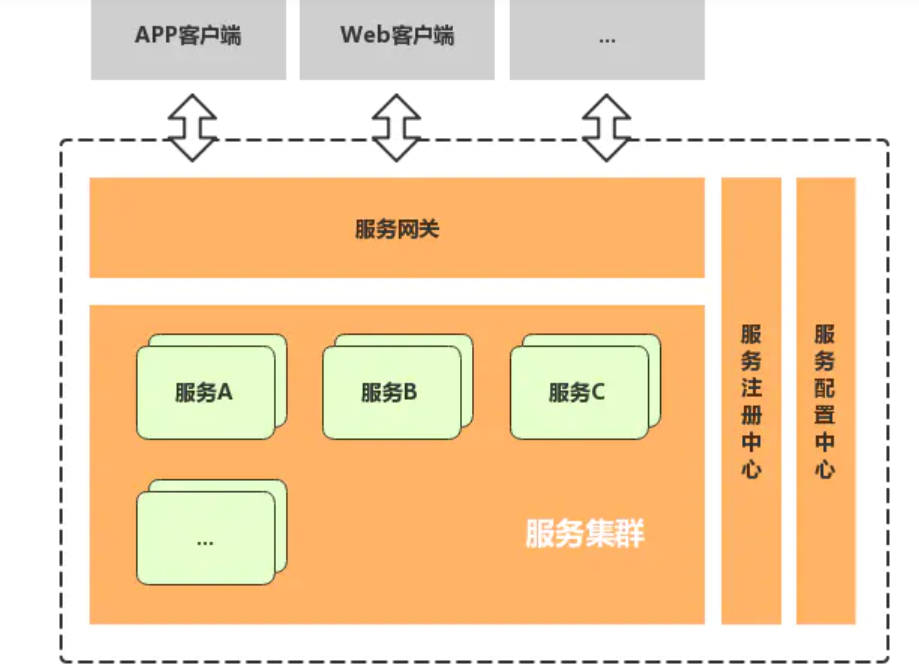
ioutil.ReadAll(response.Body)([]byte,error)

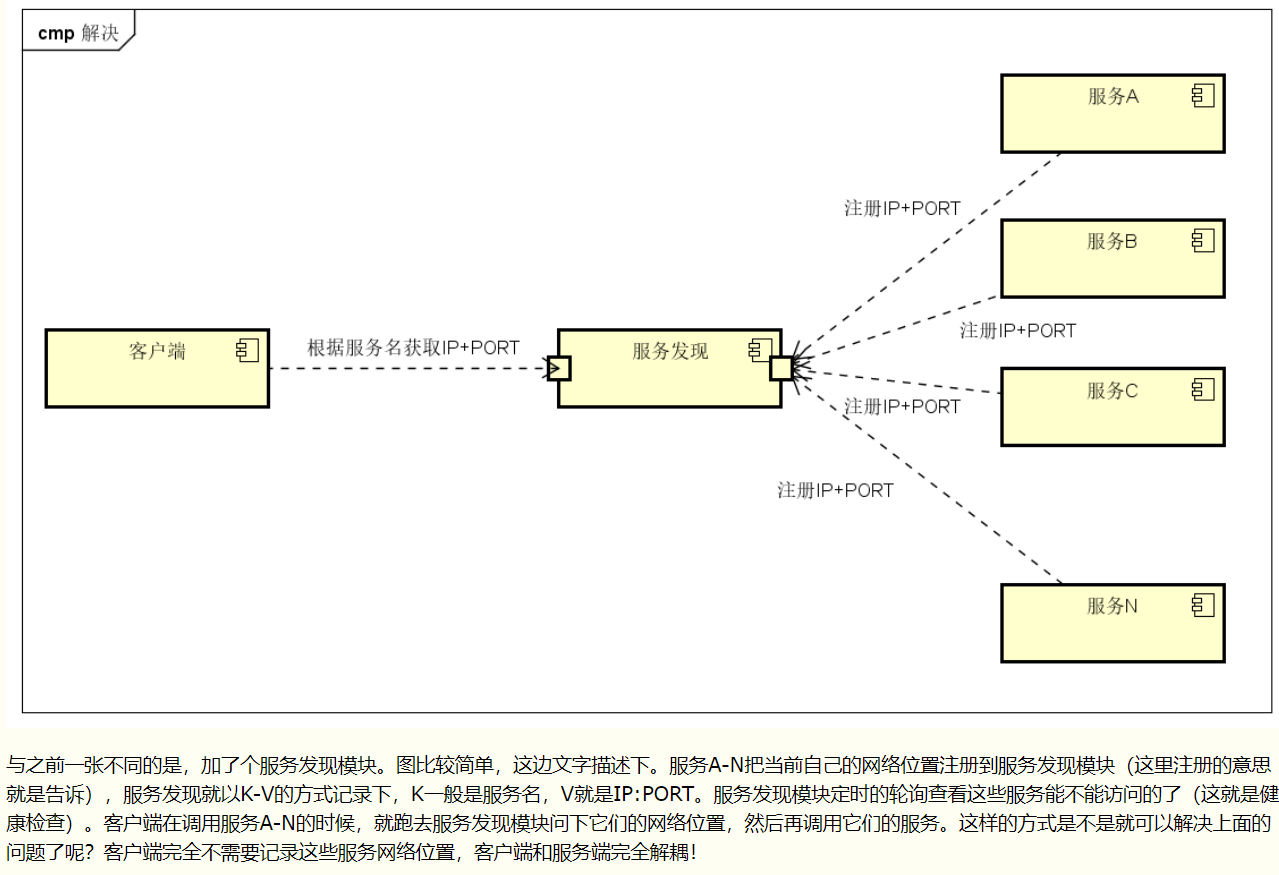
!!! 每次获取response的时候，需要close释放资源

Defer response.Body.Close()

单一职责的。一个微服务应该都是单一职责的，这才是“微”的体现，一个微服务解决一个业务问题（注意是一个业务问题而不是一个接口）。

面向服务的。将自己的业务能力封装并对外提供服务，这是继承SOA的核心思想，一个微服务本身也可能使用到其它微服务的能力。





consul-1.json



curl --request PUT --data @consul-1.json http://172.30.12.167:8500/v1/agent/service/register?replace-existing-checks=1

多次put更新 是幂等的