Jenkins Pipeline 可持续化集成 戳此了解

转载

现有系统如何集成Consul服务发现

yzy121403725 关注

2018-08-31 14:40:19 15872人阅读 0人评论

Consul 简介

Consul 四大特性

1. Service Discovery (服务发现)
2. Health Check (健康检查)
3. Multi Datacenter (多数据中心)
4. Key/Value Storage

Consul Glossary(术语)

- **Agent**
  1. Agent 是一个守护进程
  2. 运行在Consul集群的每个成员上
  3. 有Client 和 Server 两种模式
  4. 所有Agent都可以被调用DNS或者HTTP API,并负责检查和维护同步
- **Client**
  1. Client 将所有RPC请求转发至Server
  2. Client 是相对无状态的
  3. Client 唯一做的就是参与LAN Gossip Pool
  4. Client 只消耗少量的资源和少量的网络带宽
- **Server**
  1. 参与 Raft quorum(一致性判断)
  2. 响应RPC查询请求
  3. 维护集群的状态
  4. 转发查询到Leader 或 远程数据中心
- **Datacenter**

  顾名思义其为数据中心, 如何定义数据中心呢? 需要以下三点
  1. 私有的
  2. 低延迟
  3. 高带宽

  故: 可以简单的理解为同属一个内网环境, 如北京机房和香港机房就不一定满足以上三个条件
- **Consensus (一致性)**

  Consul 使用consensus protocol 来提供CAP(一致性,高可用,分区容错性)
- **Gossip**

  一种协议:~~用来保证~~ **~~最终一致性~~**用来向集群广播消息
- **LAN Gossip**

  Local Area Network Gossip, 位于同一个局域网或者数据中心的节点
- **WAN Gossip**

  Wide Area Network Gossip, 只用于Server之间, 分部在不同的数据中心或广域网

在线客服

Consul Architecture 架构图

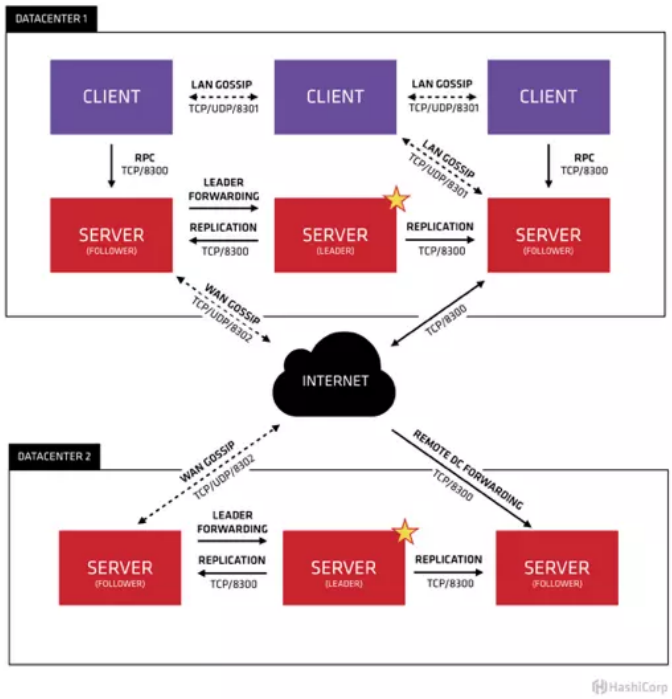


image.png

> 如上架构图可以看出几个关键点

1. Client和Client之间的LAN Gossip
2. Client将RPC请求发至Server, Follower Server将请求转发至Leader节点
3. Server之间的选举行为
4. Datacenter之间的WAN Gossip

Consul 与应用集成的几种姿势

首先来说说完整集成服务发现需要做哪些事:

1. 应用启动时需要向Consul 注册服务
2. 定时上报当前服务的状态
3. 应用销毁时需要De-Register 注销自身服务
4. 当Consul数据变化时需要修改本地配置

看上去很麻烦, 那么下面我们具体介绍三种姿势

1. 应用本身作为一个Agent , 适用于SDK相对完善的语言, 如有Spring-Cloud-Consul 的Java
2. 应用旁边(同一台主机)跑一个Client Agent
3. 使用Docker registrator

> 以上三种方式该如何选择呢?

1. 使用了Docker的话, 就用Docker registrator, 优点: 不用修改程序代码
2. 如果没有用Docker的话, 尽量选择同一台主机下起一个Client模式的Consul agent, 程序本身集成的话, 可能会造成一些不良的反应, 同时有可能还得为服务宕掉背黑锅

Docker registrator详解

下面来说说Docker registrator 是怎么一回事，registrator 自身为一个Container, 运行在其他容器身边(可以理解为同一台主机)

首先是运行时有这么三个步骤:

> 1. 启动Consul Server集群
> 2. 启动registrator container

在线客服

Docker Registrator 原理

1. Registrator is designed to be run once on every host, 也就是说同一台主机上只需要运行一个registrator即可, 并不用为每一个container起一个registrator
2. Registrator 会向consul集群register,deregister等操作
3. Registrator 通过读取同主机其他container的环境变量进行服务注册, 健康检查定义等操作

Docker Registrator 使用

集成方式很简单, 就是为你的应用添加需要的环境变量, 注意不是Registrator 的环境变量哦

Service Definition (服务定义)

环境变量Key	环境变量Value	说明
SERVICE_ID	web-001	可以为GUID或者可读性更强变量, 保证不重复
SERVICE_NAME	web	如果ID没有设置, Consul会将name作为id, 则有可能注册失败
SERVICE_TAGS	nodejs,web	服务的标签, 用逗号分隔,开发者可以根据标签来查询一些信息
SERVICE_IP	内网IP	要使用Consul 可访问的IP
SERVICE_PORT	50001	应用的IP, 如果应用监听了多个端口, 理应被视为多个应用
SERVICE_IGNORE	Boolean	是否忽略本Container, 可以为一些不需要注册的Container添加此属性

Health Check Definition (健康检查定义)

配置原则为: `SERVICE_XX_*` , 如果你的应用监听的是5000端口, 则改为 `SERVICE_5000_CHECK_HTTP` , 其他变量同理

环境变量Key	环境变量Value	说明
--- 以下为HTTP模式	---	---
SERVICE_80_CHECK_HTTP	/path_to_health_check	你的健康状态检查的路径如 `/status`
SERVICE_80_CHECK_INTERVAL	15s	15秒检查一次
SERVICE_80_CHECK_TIMEOUT	2s	状态检查超时时间
--- 以下为HTTPS模式	---	---
SERVICE_443_CHECK_HTTPS	/path_to_health_check	你的健康状态检查的路径如 `/status`
SERVICE_443_CHECK_INTERVAL	15s	15秒检查一次
SERVICE_443_CHECK_TIMEOUT	2s	状态检查超时时间
--- 以下为TCP模式	---	---
SERVICE_443_CHECK_TCP	/path_to_health_check	你的健康状态检查的路径如 `/status`
SERVICE_443_CHECK_INTERVAL	15s	15秒检查一次
SERVICE_443_CHECK_TIMEOUT	2s	状态检查超时时间
--- 使用脚本检查	---	---
SERVICE_CHECK_SCRIPT	`curl --silent --fail example.com`	如官方例子中的check_redis.py
-- 其他	---	---
SERVICE_CHECK_INITIAL_STATUS	passing	Consul默认注册后的服务为failed

Consul 与 Nginx集成的几种姿势

在线客服

0 分享 yzy121403725 关注

1. DNS
2. consul-template
3. nginx-upsync-module
4. Lua实现

1. DNS模式

如你的应用name=myweb, 再次*强调* 如果id不配置, consu会将name作为默认id, 所以定义服务时id务必要设置

```
upstream backend {
  server myweb.service.consul:20000;  #Service Name为myweb, 端口为20000}
server {
  resolver consul_host:8600;  # 告诉Nginx DNS解析使用consul的DNS查询
  listen 80;
  location / {
    proxy_pass http://myweb/;
  }
}
```

> DNS模式配置简单, 但在运行时会损耗一定性能

2. consul-template

- 首先consul-template是一个可执行程序, 而不是简单的模板
- consul-template的特点
  - 读取使用HCL(HashiCorp Configuration Language)语法的配置模板生成应用的配置
  - 可监听consul的事件, 当已注册service列表或key/value 发生变化时, consul-template会修改配置文件同时可执行一段shell, 如 nginx reload

如下模板文件:

```
apache {{range services}} upstream {{.Name}} { least_conn;{{range service .Name}} server {{.Address}}:{{.Port}};{{end}} } {{end}}
```

如下执行consul-template命令

```
consul-template \
    -template "/tmp/nginx.hcl:/var/nginx/nginx.conf:service nginx reload" \
```

> consul-template模式配置相对复杂, 性能相对较好, 仅需要reload nginx, 如果连接量小的话, 不会造成性能影响

3. 使用 nginx-upsync-module

nginx-upsync-module 可以同步consul的服务列表或key/value存储

安装步骤如下, 需要重新编译nginx

```
wget 'http://nginx.org/download/nginx-1.8.0.tar.gz'tar -xzvf nginx-1.8.0.tar.gzcd nginx-1.8.0/

./configure --add-module=/path/to/nginx-upsync-module
make
make install
```

完成以上步骤后, nginx配置如下

```
http {    upstream test {        # 先占个位子, 否则nginx在启动时会报错
        server 127.0.0.1:11111;        # 所有的后端服务列表会从consul拉取, 并删除上面的占位server
        upsync 127.0.0.1:8500/v1/catelog/service/test upsync_timeout=6m upsync_interval=500ms u
        upsync_dump_path /usr/local/nginx/conf/servers/servers_test.conf;
    }    server {        listen 8080;        location = /proxy_test {            proxy_pass htt
        }        # 输出目前指定的所有后端服务
        location = /upstream_show {            upstream_show;
        }

    }
```

在线客服

> 使用nginx-upsync-module 不需要reload nginx, 所以性能有保障, 同时配置相比consul-template相对简单

4. Lua实现

此方式适合熟悉openresty的同学, 相对nginx-upsync-module, 可以加入更多自己的逻辑, 毕竟lua语法普通程序员学习一两天就可以写, 具体步骤如下:

1. init_*_by_lua 阶段通过http api 获取服务列表载入Nginx 内存, 并设置timer轮训更新列表
2. balancer_by_lua 阶段 读取内存的列表, 设置后端服务器

> Lua实现 同样可以不reload nginx, 相比nginx-upsync-module 来说更加可扩展, 但缺点就是目前的版本的openresty中init_by_lua 和 init_worker_by_lua只能使用luasocket(阻塞), 不能使用cosocket, 所以在timer轮训时, 会造成一定性能下降

Consul 搭建配置中心

> 目标: 配置统一管理, 无需运维人员手动修改应用配置
> 方式: Consul + consul-template

第一步: 将应用配置分类

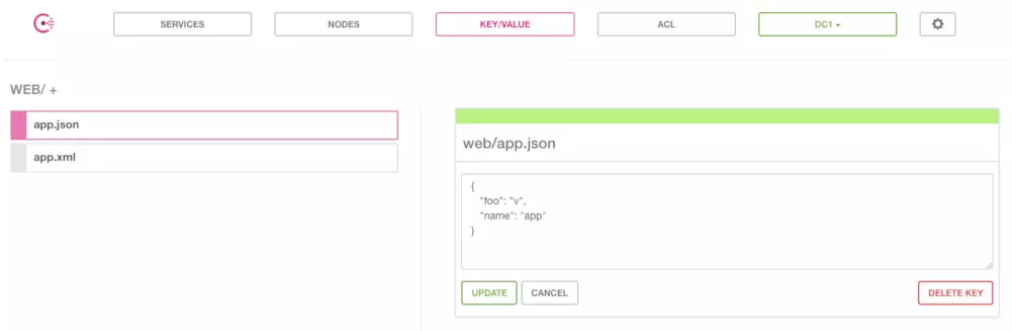
1. 调用其他服务的URL,如内部短信接口API, 配置为Service列表,这里还需要分两种情况
   a. 有API网关(如Nginx), consul-template 应配置为网关service
   b. 无API网关, 则需要程序内随机访问service列表
2. 应用自身配置参数, 如七牛key, 每天找回密码限制次数等
   - 利用consul 的key/value 作为存储

第二步: 另外程序本身需要提供一个url, 用来刷新配置

如应用的配置为json文件, 则模板类似如下:

```
{    sms_url: [
    {{range service "myweb"}}
      "{{.Address}}: {{ .Port }}",
    {{end}}
  ,    app: {{ key "myweb" }}
}
```

> 可利用consul提供的web ui 来管理配置, 可自由控制配置的颗粒度, 可以按需要将整个配置文件作为value或者某个单独的配置项作为value, 如下图:



链接：https://www.jianshu.com/p/28c6bd590ca0

参考通过consul_Template实现动态配置服务

https://www.bj-linux.com/posts/36421.html

在线客服

0 收藏 分享

上一篇：linux中sed在指定字符前后... 下一篇：RabbitMQ队列监控

yzy121403725
736篇文章，329W+人气，8粉丝

关注

提问和评论都可以，用心的回复会被更多人看到和认可

Ctrl+Enter 发布 取消 发布

推荐专栏 更多

基于Python的DevOps实战
自动化运维开发新概念
共20章 | 抚琴煮酒
¥51.00 388人订阅
订 阅

全局视角看大型园区网
路由交换+安全+无线+优化+运维
共40章 | 51CTO夏杰
¥51.00 1217人订阅
订 阅

网工2.0晋级攻略 ——零基础入门Python/A…
网络工程师2.0进阶指南
共30章 | 姜汁啤酒
¥51.00 1416人订阅
订 阅

负载均衡高手炼成记
高并发架构之路
共15章 | sery
¥51.00 474人订阅
订 阅

带你玩转高可用
前百度高级工程师的架构高可用实战
共15章 | 曹林华
¥51.00 446人订阅
订 阅

在线客服

猜你喜欢

ELK转EFK Ansible之进阶 templates模板

0 分享 yzy121403725 关注

在线
客服