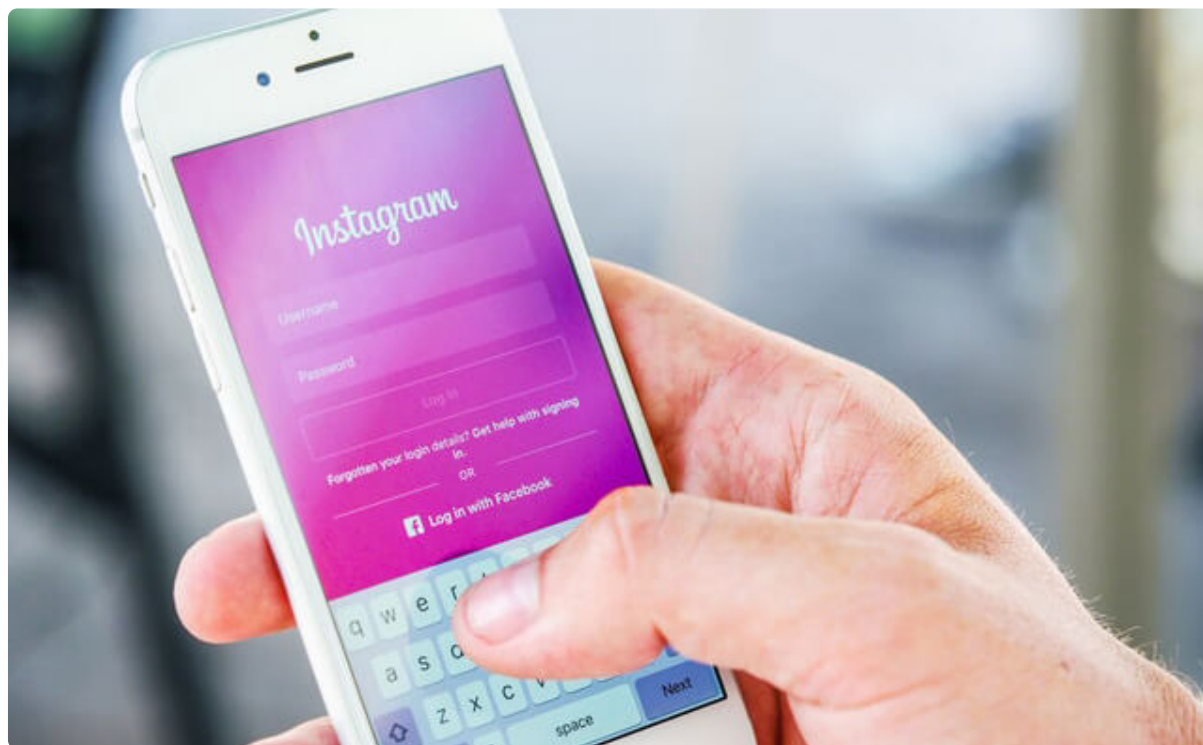


## 09 Consul 注册中心介绍

更新时间：2019-06-19 17:55:08



“我们活着不能与草木同腐，不能醉生梦死，枉度人生，要有所作为。

——方志敏”

在 Spring Cloud 体系中，几乎每个角色都会有两个以上的产品提供选择，比如在注册中心有：Eureka、Consul、zookeeper、etcd 等；网关的产品有 Zuul、Spring Cloud Gateway 等。在注册中心产品中，最常使用的是 Eureka 和 Consul，两者各有特点，企业可以根据自述项目情况来选择。

前面给大家详细介绍了 Eureka，本节给大家介绍 Consul 的使用。

### 什么是 Consul

Consul 是 HashiCorp 公司推出的开源产品，用于实现分布式系统的服务发现、服务隔离、服务配置，这些功能中的每一个都可以根据需要单独使用，也可以同时使用所有功能。Consul 官网目前主要推 Consul 在服务网格中的使用。

与其它分布式服务注册与发现的方案相比，Consul 的方案更“一站式”——内置了服务注册与发现框架、分布一致性协议实现、健康检查、Key/Value 存储、多数据中心方案，不再需要依赖其它工具。Consul 本身使用 go 语言开发，具有跨平台、运行高效等特点，也非常方便和 Docker 配合使用。

Consul 的主要特点有：

- **Service Discovery:** 服务注册与发现，Consul 的客户端可以做为一个服务注册到 Consul，也可以通过 Consul 来查找特定的服务提供者，并且根据提供的信息进行调用。
- **Health Checking:** Consul 客户端会定期发送一些健康检查数据和服务端进行通讯，判断客户端的状态、内存使用情况是否正常，用来监控整个集群的状态，防止服务转发到故障的服务上面。

- **KV Store:** Consul 还提供了一个容易使用的键值存储。这可以用来保持动态配置，协助服务协调、建立 Leader 选举，以及开发者想构造的其它一些事务。
- **Secure Service Communication:** Consul 可以为服务生成分布式的 TLS 证书，以建立相互的 TLS 连接。可以使用 intentions 定义允许哪些服务进行通信。可以使用 intentions 轻松管理服务隔离，而不是使用复杂的网络拓扑和静态防火墙规则。
- **Multi Datacenter:** Consul 支持开箱即用的多数据中心，这意味着用户不需要担心需要建立额外的抽象层让业务扩展到多个区域。

## Consul 角色

- **Server:** 服务端, 保存配置信息, 高可用集群, 在局域网内与本地客户端通讯, 通过广域网与其它数据中心通讯。每个数据中心的 Server 数量推荐为 3 个或是 5 个。
- **Client:** 客户端, 无状态, 将 HTTP 和 DNS 接口请求转发给局域网内的服务端集群。

Consul 旨在对 DevOps 社区和应用程序开发人员友好, 使其成为现代、弹性基础架构的理想选择。

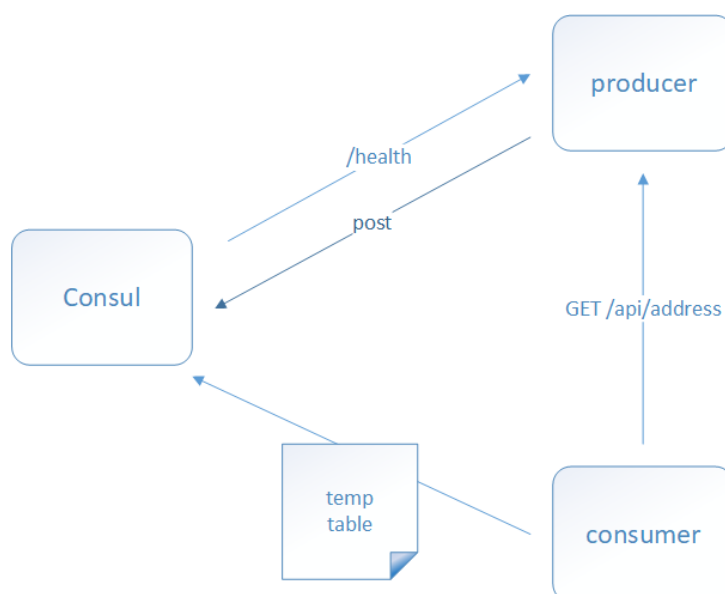
## 使用 Consul 的优势

- 使用 Raft 算法来保证一致性, 比复杂的 Paxos 算法更直接。相比较而言, zookeeper 采用的是 Paxos, 而 etcd 使用的则是 Raft。
- 支持多数据中心, 内外网的服务采用不同的端口进行监听。多数据中心集群可以避免单数据中心的单点故障, 而其部署则需要考虑网络延迟, 分片等情况等。zookeeper 和 etcd 均不提供多数据中心功能的支持。
- 支持健康检查。etcd 不提供此功能。
- 支持 http 和 dns 协议接口。zookeeper 的集成较为复杂, etcd 只支持 http 协议。
- 官方提供 Web 管理界面, etcd 无此功能。
- Consul 保持了 CAP 中的 CP, 保持了强一致性和分区容错性。
- Consul 支持 Http\gRPC\DNS 多种访问方式。

综合比较, Consul 作为服务注册和配置管理的新星, 比较值得关注和研究。

## Consul 调用过程

首先我们根据一张图来了解一下 Consul 服务调用过程:



1、当 Producer 启动的时候, 会向 Consul 发送一个 post 请求, 告诉 Consul 自己的 IP 和 Port;

2、Consul 接收到 Producer 的注册后，每隔 10s（默认）会向 Producer 发送一个健康检查的请求，检验 Producer 是否健康；

3、当 Consumer 发送 GET 方式请求 /api/address 到 Producer 时，会先从 Consul 中拿到一个存储服务 IP 和 Port 的临时表，从表中拿到 Producer 的 IP 和 Port 后再发送 GET 方式请求 /api/address；

4、该临时表每隔 10s 会更新，只包含有通过了健康检查的 Producer。

Spring Cloud Consul 项目是针对 Consul 的服务治理实现。Consul 是一个分布式高可用的系统，它包含多个组件，但是作为一个整体，在微服务架构中，为我们的基础设施提供服务发现和服务配置的工具。

## Consul 和 Eureka 的对比

我们先来通过一个表格做简单对比

Feature	Eureka	Consul
服务健康检查	可配支持	服务状态，内存，硬盘等
多数据中心	—	支持
kv 存储服务	—	支持
一致性	—	raft
cap	ap	cp
使用接口(多语言能力)	http (sidecar)	支持 http 和 dns
watch 支持	支持 long polling/大部分增量	全量/支持long polling
自身监控	metrics	metrics
安全	—	acl /https
编程语言	Java	go
Spring Cloud 集成	已支持	已支持

通过对比可以得知，Consul 功能更强大，Eureka 更容易使用。

Consul 强一致性带来的的是：

服务注册相比 Eureka 会稍慢一些。因为 Consul 的 raft 协议要求必须过半数的节点都写入成功才认为注册成功，Leader 挂掉时，重新选举期间整个 Consul 不可用。保证了强一致性但牺牲了可用性。

Consul 强烈的一致性意味着它可以作为领导选举和集群协调的锁定服务。

Eureka 保证高可用(A)和最终一致性：

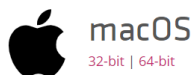
服务注册相对要快，因为不需要等注册信息 replicate 到其它节点，也不保证注册信息是否 replicate 成功。当数据出现不一致时，虽然 A, B 上的注册信息不完全相同，但每个 Eureka 节点依然能够正常对外提供服务，这会出现查询服务信息时如果请求 A 查不到，但请求 B 就能查到。如此保证了可用性但牺牲了一致性。

## 安装 Consul

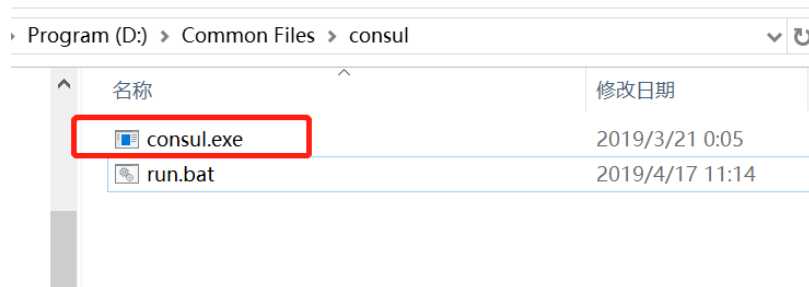
Consul 不同于 Eureka 是由 go 语言开发而成，因此需要我们单独来安装。

打开 [Consul 官网](#)，根据不同的操作系统选择最新的 Consul 版本，我们这里以 Windows 64 操作系统为例，可以看出 Consul 目前的最新版本为 1.4.4。

Check out the [v1.4.4 CHANGELOG](#) for information on the latest release.



下载下来是一个 `consul_1.4.4_windows_amd64.zip` 的压缩包，解压是一个 `consul.exe` 的执行文件。



`cd` 到对应的目录下，使用 `cmd` 启动 Consul:

```
cd D:\Common Files\consul
#cmd启动:
consul agent -dev    # -dev表示开发模式运行，另外还有-server表示服务模式运行
```

为了方便启动，可以在同级目录下创建一个 `run.bat` 脚本来启动，脚本内容如下:

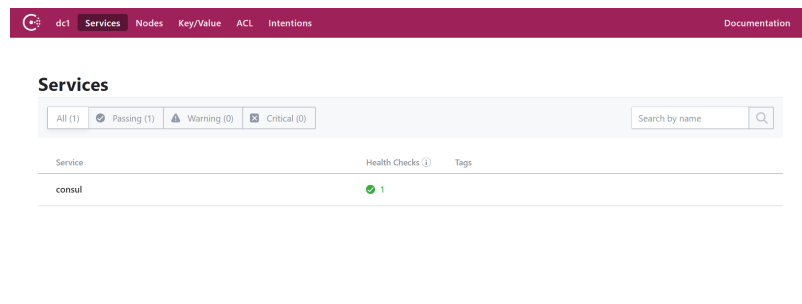
```
consul agent -dev
pause
```

下次启动的时候直接双击 `run.bat` 文件即可；当然也可以把 `consul` 的 `exe` 文件路径加入到本机的 `path` 路径下，这样后期只需要在 `cmd` 命令行下运行 `consul agent -dev` 命令即可。

执行命令后，命令行会输出如下信息:

```
选择C:\Windows\system32\cmd.exe
D:\Common Files\consul>consul agent -dev
=> Starting Consul agent...
=> Consul agent running!
    Version: 'v1.4.4'
    Node ID: '5fa0a0af-8c0f-25f5-0ae0-9d8987f93d93'
    Node name: 'LAPTOP-0536HN3V'
    Datacenter: 'dc1' (Segment: '<all>')
    Server: true (Bootstrap: false)
    Client Addr: [127.0.0.1] (HTTP: 8500, HTTPS: -1, gRPC: 8502, DNS: 8600)
    Cluster Addr: 127.0.0.1 (LAN: 8301, WAN: 8302)
    Encrypt: Gossip: false, TLS-Outgoing: false, TLS-Incoming: false
```

启动成功之后访问：<http://localhost:8500>，可以看到 Consul 的管理界面：



Consul 的 Web 管理界面有一些菜单，我们这里做一下简单的介绍：

- **Services**，管理界面的默认页面，用来展示注册到 Consul 的服务，启动后默认会有一个 **consul** 服务，也就是它本身。
- **Nodes**，在 **Services** 界面双击服务名就会来到 **Services** 对于的 **Nodes** 界面，**Services** 是按照服务的抽象来展示的，**Nodes** 展示的是此服务的具体节点信息。比如启动了两个订单服务实例，**Services** 界面会出现一个订单服务，**Nodes** 界面会展示两个订单服务的节点。
- **Key/Value**，如果有用到 **Key/Value** 存储，可以在界面进行配置、查询。
- **ACL**，全称 **Access Control List**，为访问控制列表的展示信息。
- **Intentions**，可以在页面配置请求权限。

当我们看到这个页面后，也就意味着 **Consul** 已经安装成功了。

## 小结

本节为大家介绍了注册中心的另外一个产品：**Consul**。介绍了 **Consul** 的特点、优势，以及和 **Eureka** 对比有什么不同的特性，最后为大家展示了如何在 **Windows** 下安装一个 **Consul** 服务。下节为大家介绍 **Consul** 的架构原理和实践。

参考链接：

<https://www.consul.io/intro/index.html>

本文作者：纯洁的微笑、江南一点雨