链滴

【GO-Micro】服务健康检查

作者：Allenxuxu
原文链接：https://ld246.com/article/1561444932828
来源网站：链滴

服务健康检查

在微服务架构中，每个服务都会存在多个实例，可能部署在不同的主机中。因为网络或者主机等不确因素，每个服务都可能会出现故障。我们需要能够监控每个服务实例的健康状态，当一个服务故障时及时将它从注册中心删除。

实现

micro提供两个方法可以直接实现健康检查功能

```
micro.RegisterTTL(time.Second*30),
micro.RegisterInterval(time.Second*20),
```

Interval就是间隔多久服务会重新注册

TTL就是注册服务的过期时间，如果服务挂了，超过过期时间后，注册中心也会将服务删除

micro内部服务注册的流程

当我们执行service.Run() 时内部会执行Start()

```
73  func (s *service) Start() error {
74      for _, fn := range s.opts.BeforeStart {
75          if err := fn(); err != nil {
76              return err
77          }
78      }
79
80      if err := s.opts.Server.Start(); err != nil {
81          return err
82      }
83
84      for _, fn := range s.opts.AfterStart {
85          if err := fn(); err != nil {
86              return err
87          }
88      }
89
90      return nil
91  }
92
93  func (s *service) Stop() error {…
113 }
114
115 func (s *service) Run() error {
116     if err := s.Start(); err != nil {
117         return err
118     }
119
120     ch := make(chan os.Signal, 1)
121     signal.Notify(ch, syscall.SIGTERM, syscall.SIGINT, syscall.SIGQUIT)
122
123     select {
124     // wait on kill signal
125     case <-ch:
126     // wait on context cancel
127     case <-s.opts.Context.Done():
128     }
129
130     return s.Stop()
131 }
132
```

原文链接：【GO-Micro】服务健康检查

在Start函数中又会执行s.opts.Server.Start()，方法的实现在go-micro/server/rpc_server.go中。

我们跳转到内部server的Start方法

```
504     go func() {
505         t := new(time.Ticker)
506
507         // only process if it exists
508         if s.opts.RegisterInterval > time.Duration(0) {
509             // new ticker
510             t = time.NewTicker(s.opts.RegisterInterval)
511         }
512
513         // return error chan
514         var ch chan error
515
516     Loop:
517         for {
518             select {
519             // register self on interval
520             case <-t.C:
521                 if err := s.Register(); err != nil {
522                     log.Log("Server register error: ", err)
523                 }
524             // wait for exit
525             case ch = <-s.exit:
526                 t.Stop()
527                 close(exit)
528                 break Loop
529             }
530         }
```

可以发现micro使用一个定时器按照间隔时间去自动重新注册。当服务意外故障，无法向注册中心重注册时，如果超过了设定的TTL时间，注册中心就会将服务删除。

修改源码

```
   service := grpc.NewService(
      micro.Name("go.micro.srv.hello"),
      micro.WrapHandler(ocplugin.NewHandlerWrapper(t)),
+       micro.RegisterTTL(time.Second*15),
+       micro.RegisterInterval(time.Second*10),
      // micro.Version("latest"),
   )

service := web.NewService(
      web.Name(name),
      web.Version("lastest"),
+       web.RegisterTTL(time.Second*15),
+       web.RegisterInterval(time.Second*10),
      web.MicroService(grpc.NewService()),
   )
```