bobby · 更新于 2022-11-08

上一节 2. go控制grpc... 4. token认证 下一节

1. proto

代码块 预览 复制

```
syntax = "proto3";
option go_package = ".;proto";
service Greeter {
    rpc SayHello (HelloRequest) returns (HelloReply);
}
//将 sessionid放入 放入cookie中 http协议
message HelloRequest {
    string name = 1;
}

message HelloReply {
    string message = 1;
}
```

2. 客户端

代码块

```
package main

import (
    "context"
    "fmt"
    "google.golang.org/grpc"
    "time"

    "start/grpc_interceptor/proto"
)

func interceptor(ctx context.Context, method string, req, reply interface{}, cc *
    start := time.Now()
    err := invoker(ctx, method, req, reply, cc, opts...)
    fmt.Printf("method=%s req=%v rep=%v duration=%s error=%v\n", method, req, rep
    return err
}

func main(){
    //stream
    var opts []grpc.DialOption

    opts = append(opts, grpc.WithInsecure())
    // 指定客户端interceptor
```

```
26
27      conn, err := grpc.Dial("localhost:50051", opts...)
28      if err != nil {
29          panic(err)
30      }
31      defer conn.Close()
32
33      c := proto.NewGreeterClient(conn)
34      r, err := c.SayHello(context.Background(), &proto.HelloRequest{Name:"bobby"})
35      if err != nil {
36          panic(err)
37      }
38      fmt.Println(r.Message)
39  }
```

3. 服务端

代码块

```
1   package main
2
3   import (
4       "context"
5       "fmt"
6       "net"
7
8       "google.golang.org/grpc"
9
10      "start/grpc_interceptor/proto"
11  )
12
13
14  type Server struct{}
15
16  func (s *Server) SayHello(ctx context.Context, request *proto.HelloRequest) (*pro
17      error){
18      return &proto.HelloReply{
19          Message: "hello "+request.Name,
20      }, nil
21  }
22
23
24  func main(){
25      var interceptor grpc.UnaryServerInterceptor
26      interceptor = func(ctx context.Context, req interface{}, info *grpc.UnarySer
27          // 继续处理请求
28          fmt.Println("接收到新请求")
29          res, err := handler(ctx, req)
30          fmt.Println("请求处理完成")
31          return res, err
32      }
33      var opts []grpc.ServerOption
34      opts = append(opts, grpc.UnaryInterceptor(interceptor))
35
36      g := grpc.NewServer(opts...)
37      proto.RegisterGreeterServer(g, &Server{})
38      lis, err := net.Listen("tcp", "0.0.0.0:50051")
39      if err != nil{
40          panic("failed to listen:"+err.Error())
41      }
42      err = g.Serve(lis)
```

```
44          panic("failed to start grpc:"+err.Error())
45      }
46  }
47
```

4. 拦截器的应用场景

go-grpc-middleware

2. go控制grpc的metadata ◂ 上一节 下一节 ▸ 4. token认证

我要提出意见反馈