gRPC研究

工具

被高建龙添加，被高建龙最后更新于一月 14, 2016

Overview

gRPC 的协议采用了 HTTP2 标准，序列化协议使用了 Protocol Buffer。

HTTP2 由 HTTP1 的文本协议改为二进制协议，以帧（Frame）为基本交换单元。

Frame的格式基本如下：

Length	Type	Flags	Steam Identifier	Frame Payload

Type 可以是 HEADER、DATA 等等。

其中 Steam Identifier 来标识一个流，多个流可以并发地在一个连接上传输，以此实现多路复用。

服务端和客户端都可以主动发起流，客户端流ID都是奇数，服务端都是偶数。

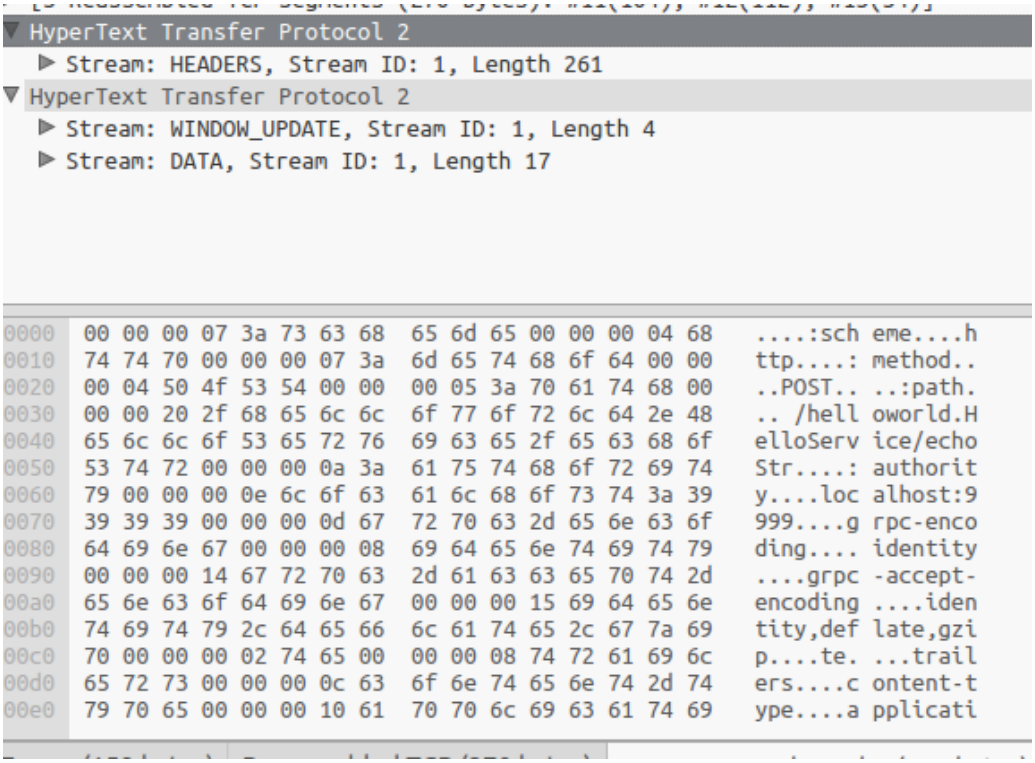
HTTP2 同时还定义了流优先级、流量控制等流控制标准、可选的TLS加密传输以及基于HPACK的头部压缩标准。

具体在 gRPC 中，把 Steam Identifier 当作消息 ID 来使用，每一次请求都发起一个新的流。

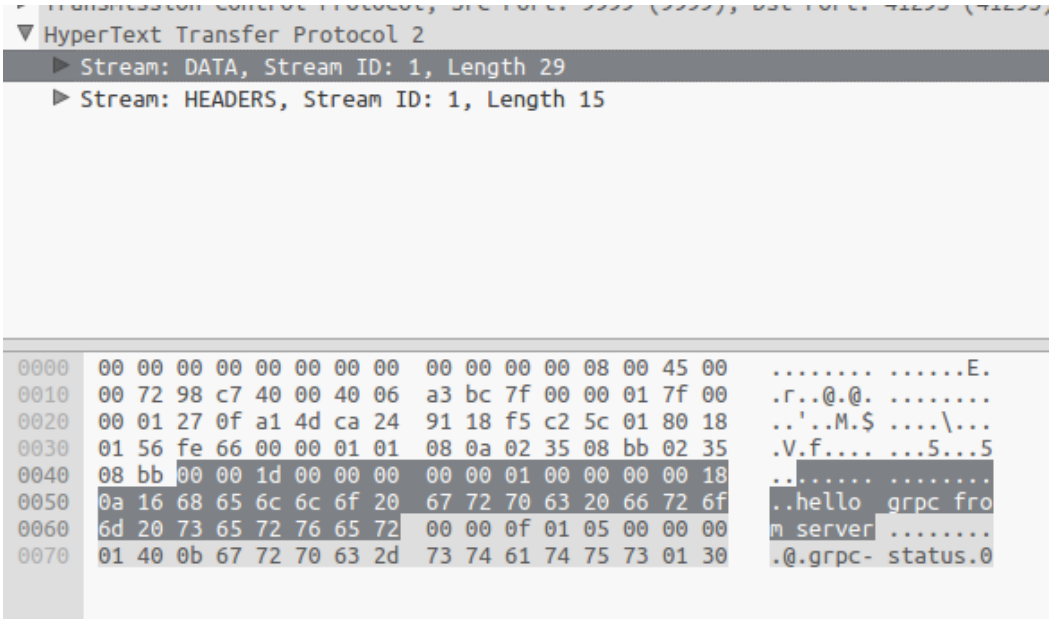
每个请求的调用哪个服务和方法、回应的调用结果状态码都在 HEADER Frame 中指定。

请求内容和回应内容由 Protocol Buffer 序列化后使用 DATA Frame 传输。

请求：



回应：



gRPC 还支持一种 Steaming RPC 的调用方式。

```
# proto 文件
service HelloService {
    rpc sayHello(HelloRequest) returns (stream HelloReply) {}
}
```

服务端可以返回多个 HelloReply，客户端连续读取直到流关闭。

多个 HelloReply 以 Steam Identifier 相同的多个 DATA Frame 的形式传输。

gRPC实现了 SSL/TLS 和 OAuth 2.0 两种认证机制。

启用 SSL/TLS 认证机制也同时会加密所有传输数据。

性能测试

单连接20个发送线程

环境一：本机 loopback

环境二：client 192.168.200.196

server 192.168.200.197

测试代码：http://source.jd.com/app/grpc-benchmark

测试proto文件：

```
syntax = "proto3";

package benchmark;

service BenchmarkService {
    rpc getSimpleObject(SimpleObject) returns (SimpleObject) {}
}

message SimpleObject {
    int32 id        = 1;
    string desc     = 2;
    bool boy        = 3;
    int64 salary    = 4;
    string money    = 5;
    int64 date      = 6;
    repeated int32 intArr = 7;
    int32 oneByte   = 8;
};
```

测试结果：

数据单位 qps 每秒请求数

	不启用 **SSL/TLS** 加密传输	启用 **SSL/TLS** 加密传输
环境一	3.2w	2.5w
环境二	1.3w	1.1w

无