ĐỊNH NGHĨA SERVICE gRPC

"Không có áp lực, không có kim cương"

Thomas Carlyle

I. KHAI BÁO SERVICE

Tương tự các framework RPC khác, gRPC được xây dựng dựa trên ý tưởng là một chương trình có thể gọi method của một service, từ chương trình khác, được đặt tại một máy tính khác, mà không cần quan tâm các vấn đề về network. Mặc định, gRPC sử dụng protobuf để định nghĩa các service và message. Ví dụ: ta cần tạo một service, client gửi một chuỗi cho server và được server phản trả về một chuỗi khác.

```
service HelloService {
  rpc SayHello (HelloRequest) returns (HelloResponse);
}

message HelloRequest {
  string greeting = 1;
}

message HelloResponse {
  string reply = 1;
}
```

Hình 1: Định nghĩ service và message protobuf

II. CÁC LOẠI METHOD

gRPC cho phép định nghĩa 4 loại method của service. Cần xác định rõ nhu cầu sử dụng để chọn method phù hợp nhất cho service của mình.

1. **Unary RPC**: Sử dụng method này, client sẽ gửi **một** request duy nhất và cũng chỉ nhận **một** response duy nhất từ server. Cách gọi này tương tự như

các lệnh gọi hàm thông thường. Đa phần các request HTTP đều có thể chuyển sang dùng dạng này để thay thế. Ví dụ các request authentication (login, logout) có thể dùng method này.

```
rpc SayHello(HelloRequest) returns (HelloResponse){
}
```

Hình 2: Unary RPC

2. **Server streaming RPC**: Client gửi **một** request và nhận về **danh sách** các response từ server. Ví dụ như các request lấy danh sách bạn bè, danh sách giao dịch cũ, danh sách notify cũ. Để khai báo dạng này, ta chỉ việc thêm từ khóa **stream** trong phần return.

```
rpc LotsOfReplies(HelloRequest) returns (stream HelloResponse){
}
```

Hình 3: Server streaming RPC

3. **Client streaming RPC**: Client gửi **danh sách** các request và nhận về chỉ **một** response từ server. Ví dụ: backup dữ liệu từ client lên server. Để khai báo dạng này, ta chỉ việc thêm từ khóa **stream** trong phần param của service.

```
rpc LotsOfGreetings(stream HelloRequest) returns (HelloResponse) {
}
```

Hình 4: Client streaming RPC

4. **Bidirectional streaming RPC**: Client gửi **danh sách** request cho server và nhận về **danh sách** response. Cả hai stream này hoạt động độc lập với nhau nên client và server có thể read và write theo bất kì thứ tự nào tùy theo yêu

cầu xử lý. Để khai báo, ta thêm từ khóa **stream** vào phần param và return của service.

```
rpc BidiHello(stream HelloRequest) returns (stream HelloResponse){
}
```

Hình 5: Bidirectional streaming RPC