基于 Thrift 框架 RPC 的研究与实现

田翠珍

（同济大学软件学院，上海 200092）

摘 要：Apache Thrift 是一种高效的、可扩展的并且支持多种编程语言的跨语言通讯框架。2007 成为 Apache 基金的开源项目。先详细介绍了 Apache Thrift 的整体架构、数据类型、协议和内部原理，又基于 Java 语言的 Thrift 客户端和服务器的实现给予处理。

关键词：Thrift；RPC；数据类型；数据协议

中图分类号：TP311.52 **文献标识码：**A **文章编号：**1003-9767（2016）01-083-02

目前远程服务调用方式有很多种，主要有以下两种，一种是基 SOAP 消息格式的 Web Service，SOAP 是一个重量级的协议，基于 XML 格式进行数据传输，其安全性好。另外一种是 RESTful 服务，RESTful 是基于 HTTP 之上建立的一种接口规范，核心是资源，使用 JSON 进行数据传输，其安全性较差。本文介绍的 Thrift 远程服务调用框架，采用生成目标代码的方式，简单易用；同时具有数据结构与传输表现的分离，支持多种消息格式，包含完整的客户端 / 服务端堆栈等特点，可快速实现 RPC，Thrift 使用二进制格式进行数据传输，效率更高，更适合高并发场景。

1 Thrift 架构

Thrift 是一个客户端和服务端的架构，即实现了 C/S 模式，通过 Thrift 的 IDL 编译器代码生成工具，生成相应语言的接口代码。Thrift 通过自定义的传输协议规范 (TProtocol) 和传输数据标准 (TTransports)，使用自己内部的序列化机制对传输的数据（IDL 生成的接口代码）进行简化和压缩，从而提高传输效率。

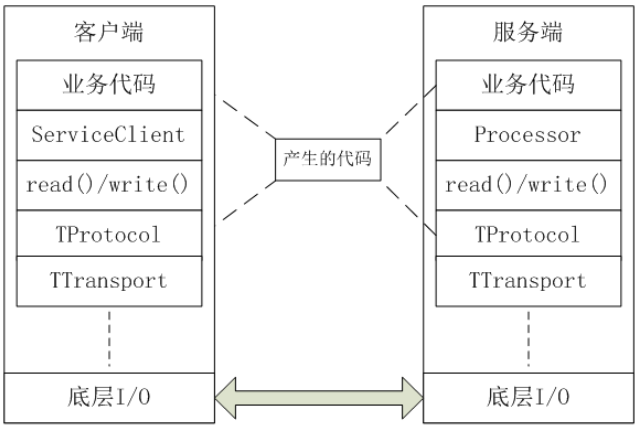

图 1 Thrift 整体架构图

Thrift 整体的架构，如图 1 所示，在最上层是用户自行实现的业务逻辑代码。第二层是由 Thrift 编译器自动生成的代码，主要用于结构化数据的解析，发送和接收。TServer 主要任务是高效的接受客户端请求，并将请求转发给 Processor 处理。Processor 负责对客户端的请求做出响应，包括 RPC 请求转发，调用参数解析和用户逻辑调用，返回值写回等处理。从 TProtocol 以下部分是 Thirft 的传输协议和底层 I/O 通信。TProtocol 是用于数据类型解析的，将结构化数据转化为字节流给 TTransport 进行传输。TTransport 是与底层数据传输密切相关的传输层，负责以字节流方式接收和发送消息体，不关注是什么数据类型。底层 IO 负责实际的数据传输，包括 Socket、文件和压缩数据流等。Thrift 还提供阻塞、非阻塞、单线程和多线程的模式运行，配合一些服务器 / 容器一起运行，如 Tomcat 等。

2 数据类型及协议

2.1 数据类型

Thrift 支持的数据类型有以下几种：

基本类型：Thrift 的基本数据类型和 Java 的基本数据类型相似，有 bool，byte，i16（16 位有符号整数），i32（32 位有符号整数），i64（64 位有符号整数），Double，String。

集合类型：list（单一类型的有序集合，许元素重复），Set（无序元素机车且集合内的元素具有唯一性），Map<t1,t2>（键类型为 t1，值类型为 t2 的 kv 对，键不容许重复）。

结构体和异常类型：Thrift 结构体在概念上类似于 C 语言结构体类型 -- 将相关属性封装在一起的简便方式。

异常在语法和功能上类似于结构体，差别是异常使用关键字 Exception 而不是 Struct 声明。但它在语义上不同于结构体：当定义一个 RPC 服务时，开发者可能需要声明一个远程方法抛出一个异常。

服务类型：Service：定义对象的接口和一系列方法。

2.2 协议

Thrift 协议层抽象了数据结构的定义，描述了如何组织数据以进行传输，包括 Encode 和 Decode 数据处理。所以，

作者简介：田翠珍（1978–），女，内蒙古呼和浩特人，本科，小教高级。研究方向：软件工程。

协议层负责实现数据的序列化和反序列化机制，例如序列化 Json, XML, Plain Text, Binary, Compact Binary 等。总体划分为文本和二进制传输协议，为提高传输效率，大多采用二进制类型的传输协议，常用协议有以下几种：TBinaryProtocol 协议，二进制编码的传输协议，Thrift 默认支持该协议；TCompactProtocol 协议，采用 Zigzag 编码可以节省传输空间，使数据的传输效率更高；TJSONProtocol 协议，JSON 数据编码协议；TSimpleJSONProtocol，只提供 JSON 只写的协议，适用于通过脚本语言解析；TDenseProtocol 密集协议，尽可能使用小的空间，不建议生产环境使用。

3 Thrift 内部原理

Thrift 由两部分组成：编译器（在 Compiler 目录下，采用 C++ 编写）和服务器（在 lib 目录下），其中编译器的作用是将用户定义的 Thrift 文件编译生成对应语言的代码，而服务器是事先已经实现好的、可供用户直接使用的 RPC Server（当然，用户也很容易编写自己的 Server）。同大部分编译器一样，Thrift 编译器（采用 C++ 语言编写）也分为词法分析、语法分析等步骤，Thrift 使用了开源的 flex 和 Bison 进行词法语法分析（具体见 thrift.ll 和 thrift.yy），经过语法分析后，Thrift 根据对应语言的模板（在 compiler\cpp\src\generate 目录下）生成相应的代码。对于服务器实现而言，Thrift 仅包含比较经典的服务器模型，比如单线程模型（TSimpleServer），线程池模型（TThreadPoolServer）、一个请求一个线程（TThreadedServer）和非阻塞模型 (TNonblockingServer) 等。

4 基于 Java 的 Thrift 实现

4.1 生成代码

编写 Thrift 文件（定义接口，结构，异常等）, 保存为 test.thrift

```
namespace java net.johnc.thrift
service Test{
void ping(1： i32 length)
}
```

把 thrift.exe 和 test.thrift 文件放在同一个目录，当然也可以把 thrift.exe 文件放进环境变量

进入 DOS 命令执行：thrift.exe --gen java test.thrift

生成文件 gen-java/net/johnc/thrift/Test.java

4.2 服务端代码实现

服务端代码要实现 Iface 接口：

```
package net.johnc.thrift;
import org.apache.thrift.TException;
public class ServerImpl implements Server.Iface {
public void sayHello(String msg) throws TException {
System.out.println("calling sayHello ,msg=" + msg);
}
}
```

4.3 启动服务端代码

通过实例化 TServerSocket 对象，绑定 8088 端口来启动 Thrift 的服务程序：

```
TServerSocket serverTransport = new TServerSocket(8088);
Test.Processor process = new Processor(new ServerImpl());
Factory factory = new TBinaryProtocol.Factory(true, true);
Args args = new Args(serverTransport);
args.processor(process);
args.protocolFactory(factory);
TServer server = new TThreadPoolServer(args);
server.serve();
```

4.4 客户端实现

客户端通过与服务端 8088 端口建立连接，访问服务端方法：

```
TTransport ttransport = new TSocket("localhost", 8088);
TProtocol tpProtocol = new TBinaryProtocol(ttransport);
Client client = new Client(tpProtocol);
ttransport.open();
client.sayHello("Hello World!");
ttransport.close();
```

5 结 语

本文介绍了 Thrift 框架的架构体系以及支持的数据类型，并对 Thrift 的内部实现原理进行了简单的剖析。最后通过一个简单的客户端 / 服务端的构建来展示了 Thrift 的客户端 / 服务端应用。由于 Thrift 具有传输效率高，支持多语言的特点，目前在互联网公司中具有广泛的应用，例如：Facebook 开源日志收集系统、淘宝的实时数据传输平台、Evernote 开放接口等。

参考文献

[1] 韩冰 , 祝永志 . 一种基于 Thrift 的跨平台单点登录实现方法 [J]. 软件导刊 ,2014(13).

[2] 梁明炯 . 基于 Thrift 框架的数据交换方案 [J]. 科技创新与应用 ,2015(13).