1.TARS CPP 服务端与客户端开发

开发C++服务端

首先使用docker命令启动容器，这里我们可以用 `tarscloud/tars:dev` 或者 `tangramor/docker-tars:dev` ：

```
docker run -d --name mysql \
    -e MYSQL_ROOT_PASSWORD=password -p 3306:3306 \
    -v /c/Users/tangramor/mysql_data:/var/lib/mysql \
    mysql:5.6 --innodb_use_native_aio=0

docker run -d -it --name tars -p 3000:3000 \
    -v /c/Users/tangramor/Workspace/tars_data:/data \
    tarscloud/tars:dev
```

这个命令启动了 `tarscloud/tars:dev` 容器 **tars** 并将本地的一个目录 `/c/Users/tangramor/Workspace/tars_data` 挂载为容器的 /data 目录，同时它还把 3000 端口暴露出来了。

然后我们可以在宿主机的 `/c/Users/tangramor/Workspace/tars_data` 目录下看到有两个子目录被创建出来了：log、tars，前者是resin的日志目录，后者里面是Tars各系统进程的日志目录。同时 `/c/Users/tangramor/Workspace/tars_data` 目录下还有各个 Tars 子系统的部署 tgz 包，这些服务已经在镜像里自动安装完毕。

运行 `docker exec -it tars bash` 进入容器 **tars**， `cd /data` 进入工作目录，参考官方的 服务开发 文档，开发 TestApp.HelloServer，其中 testHello 方法修改如下：

```
int HelloImp::testHello(const std::string &sReq, std::string &sRsp,
tars::TarsCurrentPtr current)
{
    TLOGDEBUG("HelloImp::testHellosReq:"<<sReq<<endl);
    sRsp = sReq + " World!";
    return 0;
}
```

然后将编译完成的 HelloServer.tgz 部署到 tars 容器里

开发 C++服务端 的 PHP客户端

C++的客户端可以参考官方的 客户端同步/异步调用服务。

这里主要讲一下PHP的客户端开发与部署。

首先基于 **php** 标签的镜像创建我们需要的带web服务的新镜像，这里我们可以用 `tarscloud/tars-node:php` ：

Dockerfile

```
FROM tarscloud/tars-node:php

RUN yum -y install php httpd \
  && rm -rf /var/www/html && ln -s /data /var/www/html \
  && sed -i "s/tail -f \/dev\/null/rm -rf \/var\/run\/httpd\/*\n\t\thttpd\n\t
\ttail -f \/dev\/null/g" /sbin/entrypoint.sh

ENTRYPOINT [ "/sbin/entrypoint.sh", "start" ]
```

然后创建镜像： `docker build -t tars-node-php .`

```
docker run -d -it --name tars-node --link tars -e MASTER=tars -p 80:80 -v /c/
Users/tangramor/Workspace/tars_node:/data tars-node-php
```

这个命令启动了我们刚刚创建的 `tars-node-php` 容器 **tars-node** 并将本地的一个目录 `/c/Users/tangramor/Workspace/tars_node` 挂载为容器的 /data 目录，同时它连接了命名为 **tars** 的服务端容器，还把 80 端口暴露出来了。

我们从宿主机的 `/c/Users/tangramor/Workspace/tars_data/TestApp/HelloServer` 目录里找到 `Hello.tars` 文件，将它拷贝到宿主机的 `/c/Users/tangramor/Workspace/tars_node/web` 目录下。

运行 `docker exec -it tars-node bash` 进入容器 **tars-node**， `cd /data` 来到web目录，创建一个名为 `tarsclient.proto.php` 的文件：

```
<?php

    return array(
        'appName' => 'TestApp',
```

```
        'serverName' => 'HelloServer',
        'objName' => 'HelloObj',
        'withServant' => false, //决定是服务端,还是客户端的自动生成
        'tarsFiles' => array(
            './Hello.tars'
        ),
        'dstPath' => './',
        'namespacePrefix' => '',
    );
```

然后执行 `php /root/phptars/tars2php.php ./tarsclient.proto.php` ，我们可以在 web 目录下看到 TestApp 目录被创建出来， `TestApp/HelloServer/HelloObj` 目录下是生成的PHP的客户端类文件。

在 web 目录下再创建一个 `composer.json` 文件，内容如下：

```
{
  "name": "demo",
  "description": "demo",
  "authors": [
    {
      "name": "Tangramor",
      "email": "tangramor@qq.com"
    }
  ],
  "require": {
    "php": ">=5.3",
    "phptars/tars-client" : "0.1.1"
  },
  "autoload": {
    "psr-4": {
      "TestApp\\": "TestApp/"
    }
  },
  "repositories": {
    "tars": {
      "type": "composer",
      "url": "https://raw.githubusercontent.com/Tencent/Tars/master/php/dist/
tarsphp.json"
    }
  }
}
```

然后运行 `composer install` 命令， `vendor` 目录被创建出来了。这表明我们可以在PHP文件里使用 autoload 来加载 phptars。在 web 目录下新建 `index.php` 文件，内容如下：

```php
<?php
    require_once("./vendor/autoload.php");

    // 指定主控
    $config = new \Tars\client\CommunicatorConfig();
    $config->setLocator("tars.tarsregistry.QueryObj@tcp -h 172.17.0.3 -p
17890");
    $config->setModuleName("TestApp.HelloServer");
    $config->setCharsetName("UTF-8");
    $servant = new \TestApp\HelloServer\HelloObj\HelloServant($config);

    $start = microtime();

    try {
        $in1 = "Hello";

        $intVal = $servant->testHello($in1,$out1);

        echo "服务器返回：".$out1;

    } catch(phptars\TarsException $e) {
        // 错误处理
        echo "Error: ".$e;
    }

    $end = microtime();

    echo "<p>耗时：".($end - $start)." 秒</p>";
```

在宿主机上使用浏览器访问 http://127.0.0.1/index.php （linux、mac）或 http://192.168.99.100/index.php （windows），如果没有意外，页面应该返回类似下面的内容：

```
服务器返回：Hello World!

耗时：0.051169 秒
```