Pie

邓波
头条研发业务架构

目录

- 是什么
- 为什么
- 怎么用

是什么

bb.byted.org/topic/26/我们现在说的runtime-具体来讲-到底是表示什么意义

应用 Bookmarks uniquestudio opengl product web android box design sth marcket

NoBBs

主页 / 开发框架 / 我们现在说的runtime，具体来讲，到底是表示什么意义?

我们现在说的runtime，具体来讲，到底是表示什么意义?

包春志 7天之前

是一个调用服务的标准吗?

还是说只要服务是基于某些框架的，就是runtime的?

这个比较迷茫。。。

求指导一下 :)

是什么

- 基于Django／Flask的HTTP框架
  - 继承原生框架的所有功能
  - 基础设施和一致原语的适配（nginx，cluster，env）
  - 统一的日志规范（https://wiki.bytedance.net/pages/viewpage.action?pageId=51348658）：格式，路径，插件等
  - 统一的监控规范（https://wiki.bytedance.net/pages/viewpage.action?pageId=51348664）
  - 统一的错误处理
  - 代码覆盖率统计，上报等功能

ps：目前日志是统一打在/opt/tiger/{product}/log下，按小时切分，其中app目录包含业务日志和框架的访问日志，rpc目录下包含rpc调用日志，python目录下包含启动错误日志和原生logging打印的日志，run目录下包含thrift服务标准输出日志，uwsgi包含http服务标准输出日志（原生logging没有被hack住的日志会打到标准输出）。其中错误码参考https://wiki.bytedance.net/pages/viewpage.action?pageId=80712471

是什么

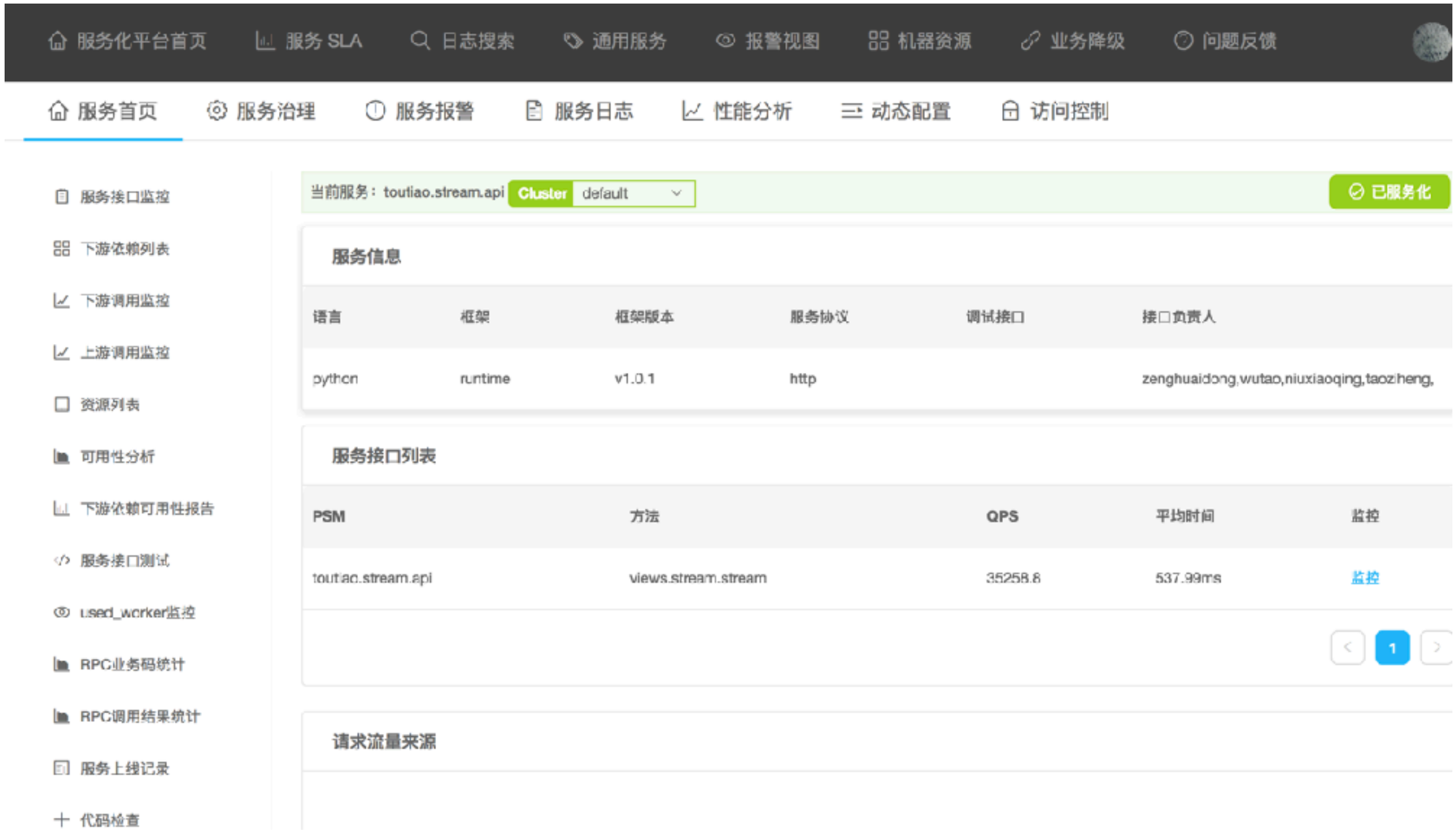
- 基于Thrift的RPC框架
  - 服务注册和发现（https://wiki.bytedance.net/pages/viewpage.action?pageId=72788035）
  - 服务治理
    - 服务降级，访问控制，流量调度，过载保护
  - 服务容灾（熔断和重试）
  - 动态配置
  - 在线调试（https://wiki.bytedance.net/pages/viewpage.action?pageId=80085062）
  - 0掉量重启（https://wiki.bytedance.net/pages/viewpage.action?pageId=78377157）
  - 统一的日志规范，监控规范，错误处理，代码覆盖率统计，基础设施和一致原语的适配，上报（运行时和静态）

为什么

- 功能完备，共享规范
- 基础设施统一规划和适配
- 日志自动被集中收集并可被搜索（目前是保证10天，由LIS负责）
- 减少重复建设，业务方可更加关注业务

为什么

- 自动接入服务化平台（http://ms.byted.org/page/home）
  - 服务上下游关系，服务治理，服务监控与报警，服务日志等



为什么

- 自动接入日志搜索平台（http://msp.byted.org/page/logsearch）
  - 搜索关键字，搜索错误栈，搜索调用链

首页　日志搜索　P.S.M管理　服务化平台　问题反馈

搜关键字
搜调用栈
搜请求链
搜客户端日志
任务列表
使用手册

只看我的任务　自动刷新（Every 5 seconds）

任务ID	任务名	开始时间	结束时间	任务类型	操作人	任务信息	执行结果	调用链详情
14522	54005640083	2017-04-17 22:07:46	2017-04-17 22:08:04	normal	wangdi	查看	完成	
14521	20170417214329010003048216526640C	2017-04-17 22:05:48	2017-04-17 22:07:02	chain	niuxiaoqing	查看	完成	调用链详情
14520	56301061992L	2017-04-17 22:05:20	2017-04-17 22:05:35	normal	wangdi	查看	完成	

为什么

调用链入口IP: 10.3.48.216 logid: 20170417214329010003048216526 40C 总耗时: 563ms 调用链总数: 4

业务访问链信息

服务名	节点编号	开始时间	处理总耗时(ms)	方法名	节点IP	节点类型	执行状态	操作
- nginx	0	2017-04-17 21:43:30	563	POST	10.3.48.216	nginx	OK	查看日志
- toutiao.discuss.ttdiscuss	1	2017-04-17 21:43:30	562	POST	10.3.48.216	api	OK	查看日志
toutiao.device.device_info	2	2017-04-17 21:43:29	0.048	GetInstallation	10.6.23.154	service	UNKNOW	查看日志
toutiao.article.article	3	2017-04-17 21:43:32	2228.19	MGetArticleMetaBylids	10.3.23.162	service	OK	查看日志

服务端视角,层级缩进关系表达访问链的向前推进,即前者访问了后者

RPC调用信息

发起方服务名	发起方编号	接收方编号	接收方服务名	接收方方法名	调用开始时间	调用耗时(ms)	调用执行状态
nginx	0	1	toutiao.discuss.ttdiscuss	POST	2017-04-17 21:43:30	563	OK
toutiao.discuss.ttdiscuss	1	-1	toutiao.device.device_info	GetInstallation	2017-04-17 21:43:29	1	OK
toutiao.discuss.ttdiscuss	1	2	toutiao.device.device_info	GetInstallation	2017-04-17 21:43:29	1	OK

怎么用

- 环境初始化（https://wiki.bytedance.net/pages/viewpage.action?pageId=81823933）
  - 配置gitr

    根据系统和版本的不同，ssh 配置文件的位置不同 (Windows 需自行查看设置)

    /etc/ssh/ssh_config, ~/.ssh/config （Linux)

    /etc/ssh_config, ~/.ssh/config (OSX 10.x)

    ~/.ssh/config中添加以下几行：

    Host gitr

        GSSAPIAuthentication no

        Hostname git.byted.org

        Port 29418

        User user_name

  - 初始化repo

    $> mkdir ~/repos/toutiao && cd ~/repos/toutiao

    $> git clone gitr:toutiao/runtime && ./runtime/update.sh

    执行完上面两个命令后，你的runtime环境就初始化完了（你可能会发现很多repo没有权限，但是没关系，必备的repo有）。现在你的runtime根路径就是~/repos/toutiao，下面默认的相对路径就是它（目录结构的含义参考https://wiki.bytedance.net/pages/viewpage.action?pageId=62423776）

怎么用

- 认识代码生成工具——pietool（https://wiki.bytedance.net/pages/viewpage.action?pageId=76126527）
  - 框架代码的生成统一采用pietool
  - pietool在gitr:toutiao/runtime目录下，对应到机器中的$RUNTIME_ROOT/runtime目录，使用-h命令就能看到所有的说明
  - 可以生成API代码、Thrift Client代码、Thrift Server（service）代码。其中API代码可以使用Flask或者Django框架

- 认识服务标识
  - 统一采用psm来作为RD之间的交流原语
  - https://wiki.bytedance.net/pages/viewpage.action?pageId=63237078

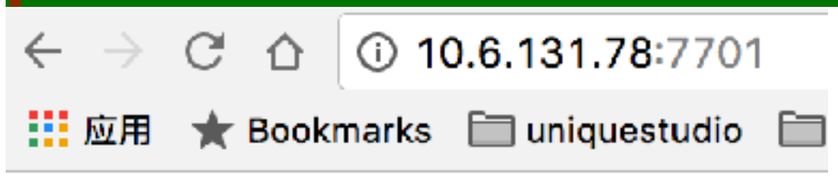
怎么用

- 我是 HTTP 服务提供者
  - 生成模板代码

```
dengbo@n6-131-078^:~/repos/toutiao$ ./runtime/pietool --type flask --psm toutiao.my.api --prefix my/api
欢迎使用Pie框架，wiki参考https://wiki.bytedance.net/pages/viewpage.action?pageId=70862800
app目录：/data04/home/dengbo/repos/toutiao/runtime/../app/my/api
uwsgi配置文件路径：/data04/home/dengbo/repos/toutiao/runtime/../conf/app/uwsgi_toutiao.my.api.xml
生成结束
dengbo@n6-131-078^:~/repos/toutiao$
```

  - 按照业务需求修改代码（主要看生成的模板代码里的settings），并添加业务逻辑
  - 运行代码

```
dengbo@n6-131-070^:~/repos/toutiao$ ./runtime/bin/python app/my/api/bootstrap.py
I will use in-memory cache
INFO 2017-04-18 11:17:44,809 /data04/home/dengbo/repos/ss_lib/python_package/lib/python2.7/site-packages/werkzeug/_internal.py:87 10.6.131.78 toutiao.my.api 0 - process=84168 thread
=140068553332480 data= * Running on http://0.0.0.0:7701/ (Press CTRL+C to quit)
INFO 2017-04-18 11:17:47,055 /data04/home/dengbo/repos/ss_lib/python_package/lib/python2.7/site-packages/werkzeug/_internal.py:87 10.6.131.78 toutiao.my.api 0 - process=84168 thread
=140068553332480 data=10.2.204.233 - - [18/Apr/2017 11:17:47] "GET / HTTP/1.1" 200 -
```



  - Hello world
    - 提交conf中uwsgi配置文件
    - 配置81 nginx（https://wiki.bytedance.net/display/TTRD/Nginx+Runtime）
    - 修改80 nginx，并上线（http://cloud.byted.org/tlb/service/apply）

怎么用

- 我是 Thrift 服务提供者
  - 定义IDL（参考https://wiki.bytedance.net/pages/viewpage.action?pageId=83074857），并放置在lib/idl目录中
  - 生成模板代码

```
dengbo@n6-131-078^:~/repos/toutiao$ ./runtime/pietool --type thriftserver --idl demo2.thrift --psm toutiao.my.server --prefix my/server
欢迎使用Pie框架，wiki参考https://wiki.bytedance.net/pages/viewpage.action?pageId=70862800
my/server
/data04/home/dengbo/repos/toutiao/runtime/runtime_service_gen.sh my/server demo2.thrift toutiao.my.server 0 0 0
Start generating...
生成service代码的thrift路径： /opt/tiger/thrift/bin/thrift
生成client代码的thrift路径： /opt/tiger/thrift/bin/thrift
```

  - 修改配置（settings, conf/app/service-{psm}.yml, conf/rpc/services/{ServiceName}.yml），添加逻辑（handler），其中psm是生成代码命令中指定的，ServiceName是在idl中定义的
  - 运行（修改settings里的DEBUG为True才可在控制台看到日志）

```
dengbo@n6-131-078^:~/repos/toutiao$ ./runtime/bin/python app/my/server/bootstrap.py
INFO 2017-04-18 15:50:06,615 /data04/home/dengbo/repos/toutiao/lib/frame/server_bootstrap.py:424 10.6.131.78 toutiao.my.server 0 - process=24540 thread=140064041277184 data=Using port: 7701 from settings.py
INFO 2017-04-18 15:50:06,617 /data04/home/dengbo/repos/toutiao/lib/frame/server_bootstrap.py:432 10.6.131.78 toutiao.my.server 0 - process=24540 thread=140064041277184 data=Starting ... with port: 7701

INFO 2017-04-18 15:50:06,618 /data04/home/dengbo/repos/toutiao/lib/frame/server_bootstrap.py:518 10.6.131.78 toutiao.my.server 0 - process=24540 thread=140064041277184 data=I can't find a appropriate debug port
```

怎么用

- 我是 Thrift 服务提供者
  - 上线
    - 提交服务代码和conf的配置
    - noops／tce上线
  - 测试服务
    - 发起调用
      - 编写client调用
      - 服务化平台（接口测试）
    - 查看返回值及日志，metrics（开发机的metrics和线上机器有区别，需要在最前面加test.）

怎么用

- 我是Thrift client使用者
  - 查找所调用服务的IDL，如果不在lib/idl下，则需要问服务提供者添加
  - 生成模板代码

```
dengbo@n6-131-078^:~/repos/toutiao$ ./runtime/pytool --type thriftclient --prefix my/test --idl demo_test.thrift
生成client代码的thrift路径: /opt/tiger/thrift/bin/thrift
OS type: Linux

2017-04-18 16:49:18,594 INFO namespace:demo_test, service:DemoTestService
2017-04-18 16:49:18,594 INFO gen service base ...
```

  - 构建客户端代码（参考生成手册）
  - 运行代码

```
dengbo@n6-131-078^:~/repos/toutiao$ ./runtime/bin/python app/my/test/test.py
INFO 2017-04-18 17:00:25,497 /data04/home/dengbo/repos/toutiao/lib/service_rpc/client/scmiddleware/FrameLogMiddleware.p
read=1399268850378 24 data= retry=0 business_retry_cost=20 called=toutiao.my.server method=GetItem rpc_status=success
Demo2Response(item=Demo2Item(ItemId=1, ItemName=u'123'), BaseResp=BaseResp(StatusMessage=u'SUCCESS', StatusCode=0))
```

怎么用

- 我是Thrift client使用者注意事项
  - 调用的服务端不是Pie框架，在生成client代码时会在conf/rpc/services生成一个用于提供RPC调用的yml文件，需要使用者询问服务提供者修改里面的配置，并在noops上线toutiao/conf目录（如果自己的机器比较少，可以直接到自己机器上手动拉代码，因为这个yml只有你会用，新版的client可以直接传consul_name，无需这个yml）
  - 一般Thrift client不是单独使用，而是在http服务／thrift服务／脚本中，这三类都有相应的bootstrap框架，因此一些初始化相关的工作都无须业务关心，只有自己想简单测试的时候要添加一些初始化相关的代码，具体见https://wiki.bytedance.net/pages/viewpage.action?pageId=70862819
  - RPC调用相关的日志在log/rpc目录下，metrics参考https://wiki.bytedance.net/pages/viewpage.action?pageId=51348664
  - 调用的下游cluster如果不是default，需要手动指定调哪个cluster（https://wiki.bytedance.net/pages/viewpage.action?pageId=86348214）
  - 关于cluster：tce在创建服务的时候，如果服务选择的集群不是default，tce会将这个信息注册到服务去，sd lookup看到了下游的实例，但在调用的时候却显示找不到下游实例看下下游所处的cluster是否不是default，如果不是则需要在创建client的时候传入to_cluster
  - 每个用client的服务都把pyrpc放到自己的根目录下，即和bootstrap.py同一级。
  - 测试阶段如何指定调用某个下游机器可以参考 https://wiki.bytedance.net/pages/viewpage.action?pageId=83921550，如果要hack某个环境变量可以直接在bootstrap.py一开始的时候就用os.environ设置
  - 在写try except的时候一定要指明类型，且级别不能大于Exception
  - 只要是用了框架，一般是不需要去显示调用框架的任何代码

Thanks

Q&A