Python Agent 用户指南

北京基调网络股份有限公司

概述

听云 python 探针（也称 Python Agent）是基于 Python 语言而研发的性能监测工具客户端，其主要目标为支持 WSGI 协议的 python web 框架。理论上只要是基于 WSGI 协议的 web 框架都能对其监测， 比如 django、tornado、flask 等。当然其中也包括一些该 web 框架支持的一些外围组件，例如 sql 数据库、nosql 数据库、第三方 http 调用等。其功能会随着版本的升级而得到更多的支持。

关于 Tornado 的支持，目前仅支持 Tornado4.x 版本，更多关于 Tornado 支持请参考<部署与框架>章节。

本文档所列明的支持的框架、数据库、软件环境等，是经过测试，且完全支持的。对于没有在本文档列出的框架以、组件、安装方式等将不被支持，但随着版本的更新会逐步完善。在使用听云 python 探针前请务必仔细阅读此文档，同时我们也极力保持文档与探针功能的同步更新。

探针在性能监测时，将会对页面加载、模板渲染、中间件调用、视图层、数据层、系统软件环境、硬件环境等数据进行跟踪和监测，更多关于监测的数据参考 https://report.tingyun.com/平台的数据监测。

欢迎使用听云，因为有你，我们将会做的更好！！

听云 python 团队

快速体验

1、环境要求

Python 探针目前不能兼容所有的 python 版本以及框架等，基本环境要求如下表 1-1 所示：

表 1-1

环境	要求	备注
操作系统	类 unix 系统，如 linux、FreebSD、MacOS X 等	windows 未经测试
Python	Cpython 2.6.x，2.7.x	
web 框架	Django、flask 等	WSGI 1.0 框架 (PEP 333)

更详细的支持列表以及信息，请参考《兼容与支持》

2、快速部署

快速安装需要用到 pip，并需要连接互联网，如您的环境无法提供，请移驾探针安装。以下步骤在命令行操作,示例应用为 django 应用。

1、第一步，安装探针:

pip install tingyun

2、第二步、生成本地配置文件:

tingyun-admin generate-config YourLicenseKey outputFile.ini

例: *tingyun-admin generate-config* 123-456-789-001 /tmp/tingyun.ini

3、第三步、部署探针

TING_YUN_CONFIG_FILE=/tmp/tingyun.ini *tingyun-admin run-program* 应用的启动命令 应用启动参数

例：TING_YUN_CONFIG_FILE=/tmp/tingyun.ini *tingyun-admin run-program* python manage.py runserver

注：探针默认输出 log 为 /tmp/tingyun-agent.log

4、第四步

访问应用，探针需要有应用的访问才能驱动探针工作，在部署后需要对应用进行访问。

部署与使用

Python 探针是根据 WSGI 协议而为 web 框架定制的性能监测客户端，理论上只要是基于 WSGI 协议的 web 框架都能对其进行监测。

在使用探针之前请确认当前探针版本支持您目前的应用以及部署方式，详情参考《兼容与支持》章节。

使用探针有两个前置条件，**一是注册听云账户，取得 license-key，二是设置探针的配置文件，此两者缺一不可，否则探针无法正常工作**。

1、一键嵌入

通过命令行方式一键嵌入探针，不需要修改任何代码即可完成探针部署，此方式也支持将其封装在 shell 里进行操作。

生成配置文件

tingyun-admin generate-config YourLicenseKey outputFile.ini

提示：配置文件有很多配置项可提供不同功能，详情参考《探针配置》章节

嵌入探针

假设配置文件放置于/tmp/tingyun.ini

TING_YUN_CONFIG_FILE=/tmp/tingyun.ini *tingyun-admin run-program* 应用启动命令 应用启动参数

参数： “应用启动命令” 和 “应用启动参数” 为正常情况下没有使用探针的启动方式

设置阻塞时间（可选）

阻塞时间是用于测量前端负载均衡接受到请求，到应用层接收到请求耗时。当并发访问量大的时候，对于衡量负载均衡 到 应用层时间非常有用。
详细设置方式参考阻塞时间章节

2、API 手动嵌码

探针除了支持一键嵌入之外，还支持**手动嵌码**操作，建议参阅源代码，里面有详细的使用示例。支持的 API 如表 2-2-1 所示。

表 2-2-1

所在模块以及函数	功能
tingyun.api.init_tingyun_agent	初始化探针
tingyun.api.wsgi_app_decorator	打包 WSGI 程序入口
tingyun.api.function_trace_decorator	函数/方法性能追踪

2.1 init_tingyun_agent

init_tingyun_agent(config_file=None):
功能：初始化探针程序。
参数：
config_file: 探针的配置文件路径，建议使用绝对路径。如果未提供该参数，探针会寻找环境变量`TING_YUN_CONFIG_FILE`指向的配置文件。若两个地方都没有提供，探针将不会工作。
返回值：无
使用方式：直接调用
注意事项：
必须在所有用户代码之前调用。

2.2 wsgi_app_decorator

wsgi_app_decorator(framework='xx', version='xx')
功能：打包 WSGI 应用入口
参数：
framework：探针所在框架，默认为`xx', 若无框架类别可不填或自定义。
version: 该框架（或自定义框架）使用的版本号，默认值为 `xx`。
返回值: 函数
使用方式:
装饰器调用,如@wsgi_app_decorator(framework='customer', version='0.1')

2.3 function_trace_decorator

function_trace_decorator(name=None, group=None)
功能：函数性能追踪
参数：
name：被追踪函数的名字，如未提供，则使用当前被追踪函数名字作为该性能追踪的名称。
group: 被追踪函数的分组， 未提供则分组为`Function`
返回值: 函数
使用方式:
装饰器调用,如@function_trace_decorator(name='home_page', group='views')

3、探针停止

探针会随着 python 解释器的停止而停止，所以按应用正常的停止方式停止即可。

4、私有化部署

私有化，是指用户应用性能数据不方便通过公网上传到听云服务器时，需要在自己“内网”安装探针和服务器的一种部署方式。

敬告：普通用户无需关心此选项，如需私有化请联系我们的私有化团队。

Python 探针支持私有化部署，只需在配置文件中增加以下配置即可：

```
[tingyun:private]
host=YourprivateServer
port=YourprivatePort
```

更多详情参考《探针配置》章节。

5、探针命令详解

tingyun-admin 是探针的主命令，需要提供参数才能工作，在命令行直接键入该命令，会有相应提示。

5.1 生成配置文件

配置文件使用并遵循标准的 ini 文件标准，详情参考”Python ini 文件解析”官方文档，生成配置文件命令如下：

tingyun-admin generate-config **[licenseKey] filename.ini**

licenseKey[可选]， 如果生成配置文件时没有提供该参数，需要后续手动修改生成的配置文件。

filename.ini, 配置文件名称，建议使用绝对路径。

5.2 检查配置文件

用于对配置文件内设置的参数进行检查，但不会检查 license 的合法性，其命令格式如下：

tingyun-admin check-config [filename.ini]

filename.ini[可选]，探针使用的配置文件。如未提供,该命令会检测系统环境变量 TING_YUN_CONFIG_FILE

3.3 启动探针命令

使用该命令对应用进行一键嵌码操作，使用它的前提是需要一份配置文件，提供配置文件有两种方式，一种是将配置文件环境变量直接放在启动的探针命令前面，另外一种是设置操作系统环境变量。

假设已配置的探针配置文件位于/tmp/tingyun.ini

方式一：

TING_YUN_CONFIG_FILE=/tmp/tingyun.ini tingyun-admin run-program 启动命令 应用启动参数

方式二：

Export TING_YUN_CONFIG_FILE=/tmp/tingyun.ini
tingyun-admin run-program 启动命令 应用启动参数

两种方式都目的都是设置配置文件的环境变量，探针在启动的时候会检测该环境变量，如不存在或者没有读取权限探针将不 能正常工作。

6、部署方案建议

下表为常见的 Python 应用部署方案的组合，并描述其优劣，以及使用建议等信息。

分类	应用服务器	优劣	框架
WSGI	UWSGI	多进程，多线程，同步	所有 WSGI 协议框架
	Gunicorn	多进程，io 异步	所有 WSGI 协议框架
	Gevent	单进程，io 异步	所有 WSGI 协议框架
非 WSGI	Tornado	单进程，网络异步	Tornado
			WSGI App 不建议

Tips：使用 shell 部署时，注意权限问题、shell 文件第一行是否指定了 shell 类型

6.1 UWSGI 部署

6.1.1 优势

UWSGI 部署应用的优势在于，能够精确的控制应用启动的参数，启动/重启条件，精确的调试信息，以及更多的控制、调试、信号接口。其属于传统的多进程，*多线程*应用部署方案，使用广泛，配置灵活且多样性。

6.1.2 劣势

完全依赖于硬件资源，一旦达到硬件瓶颈，可能出现无响应，假死等现象，但如果能精确预测以及控制应用访问（比如负载均衡以及后备硬件资源），也未尝不是好的选择。

6.1.3 部署建议

为了更好的配合探针工作，建议开启 UWSGI 以下选项：

--enable-threads，uWSGI 默认情况下，没有开启多线程的支持，由于探针基于多线程模式，所以必须开启此模式，否则探针无法正常启动。如果 uwsgi 使用

了选项—threads，将自动开启选项--enable-threads。

--single-interpreter， 默认情况下 uWSGI 为了隔离应用环境，以便运行多个应用，在启动的时候会启动子进程来运行某个应用，而不是在主进程中执行。 为了更好的适应探针环境请开启这个选项， 开启后不会有其他任何的副作用，它也是安全而有效的方式。

--lazy，该选项开启后，uwsgi 将会在 worker 进程中加载 app，而不是在 master 中加载 app。 这样做完全不受影响，因为 uwsgi 在工作时，master 是不工作的，它只负责调度。

6.2 Gunicorn&Gevent

6.2.1 优势

两者从本质上讲差别并不大，Gevent 是基于 libevent io 库，而 Gunicorn 可以直接使用 Gevent 或者 greenlet 作为其 worker 的 service。 Gunicorn 以多进程方式管理 worker（可以是 gevent 等），而 Gevent 部署时一般都是以单进程部署的。

两者可以认为都是基于事件驱动的 io 异步模型，在并发上要比 uwsgi 同步的方式要高，且能更好的利用服务器资源。

6.2.2 劣势

当然相比 uwsgi 而言，劣势也很明显，应用过程状态监控以及过程信息不透明，当然灵活性也不那么高。

6.2.3 部署建议

探针目前兼容异步应用服务器的部署方案， 由于其异步特性，在开发应用时，尽量不要有过多、多大的 io 以及网络阻塞操作，并且做好负载均衡。一旦请求量大时，会导致过多的协程数据驻留内存，导致内存消耗过大。 特别是慢的请求太多时，会导致探针采集过多的信息，增加机器资源消耗。

6.3 Tornado

6.3.1 优势

Tornado 的优势在于网络异步并发，可以在单机、单进程上轻松支持成千上万的网络并发请求。并且对长连接有很好的支持。

6.3.2 劣势

基于 Tornado 的应用，其发开难度大，要完全发挥 tornado 网络异步的特性，应用需要利用其异步接口支持其异步特性，比如数据库操作、文件 io 等，否则使用该框架性能会大打折扣。

6.3.3 部署建议

Tornado 常见部署方案如下所示：

方案	应用服务器	框架/应用	说明
A.	Tornado(beta)	Tornado 应用	原生部署方式
B.	可能会导致 QPS 降低	WSGI 应用	不建议，可以参考《UWSGI 部署》或者《Gunicorn&Gevent》
C.	WSGI 服务器	Tornado 应用	不建议

方案 A：

探针目前仅支持 Tornado4.X 版本，按常规方式使用探针即可。

方案 B & 方案 C：

需要开启 Tornado wsgi 兼容模式，请在探针配置文件中设置 tornado_wsgi_adapter_mode 选项。

安装与配置

1、 探针安装

探针支持 python 标准的安装方式，其中内置了 C 模块，建议安装 gcc 编译器（没有 gcc 也能正常安装）， 如果您为**非小白**用户，请忽略该章节。

1.1 在线安装

在线安装使用 python 官方源，需要用到 setuptools，或者 pip，更详细的用法请参考其官方文档。

pip install tingyun 或者
easy_install tingyun

提示： 安装完成后要配置探针才能正常使用，详情请参考探针配置章节，具体使用参考部署与使用章节。

1.2 离线安装

探针下载地址有两个，一个 python 官方源，一个 tingyun 官方网站，分别如下：

官方源地址：https://pypi.python.org/pypi/tingyun
tingyun 官网地址：https://report.tingyun.com/server/download/serverSdk

假设得到的安装包为 tingyun-1.1.0.tar.gz，直接下述操作即可安装（如使用虚拟环境，请先激活）

tar –zxvf tingyun -1.1.0.tar.gz –C /tmp
python /tmp/ tingyun-1.1.0 /setup.py install

2、 探针卸载

探针卸载方式，遵循 python 安装包标准的安装/卸载方式，详情参考 python 官方网站

2.1 工具卸载

如果使用的是 pip 等工具安装，可使用工具自带的卸载命令可将探针，如使用 pip 卸载方式为：pip uninstall packagename

2.2 手动卸载

由于 python 程序为解释性语言，手动删除探针安装文件即可将探针卸载。

3、探针配置

本小节主要讨论探针的本地配置，在安装包中提供了一份默认的配置文件 tingyun.ini（或者通过命令 *generate-config* 生成），该配置为标准 python ini 文件配置，详情参考 http://docs.python.org/library/configparser.html

注意：在使用时需要为该配置文件设置环境变量（TING_YUN_CONFIG_FILE），探针方能启动。 配置文件 tingyun.ini 中除了 license key 之外均有默认值，为了正常使用探针，license key 是必填项， 更多配置请参考下述说明。

Section	配置项（区分大小写）	备注
tingyun	license_key	字符型
	enabled	Boolean 型
	app_name	字符型
	log_file	字符型
	audit_mode	Boolean 型
	log_level	字符型
	ssl	Boolean 型
	auto_action_naming	Boolean 型
	action_tracer.log_sql	Boolean 型
	tornado_wsgi_adapter_mode	Boolean 型
tingyun:private	host	字符型
	port	正整数
tingyun:proxy	proxy_scheme	代理主机数据传输协议
	proxy_host	代理主机
	proxy_port	代理主机端口号
	proxy_user	代理账户
	proxy_pwd	代理账户所需密码

注：本地配置的修改需重启方能生效！

下述列表采用标准的 ini 文件配置，标题书写格式为： [secionName] optionName

[tingyun] license_key

功能：账户认证标识，**必填项，使用探针前请务必填写该项**。
默认值： 无
说明：如缺少该配置项、或者配置项错误，探针能正常启动，log 输出会提示 license 无效，将不会采集并上报数据。

[tingyun] enabled

功能：客户端开启、禁用探针开关。
默认值：True，可选值 True, False, 以及 on, off，不区分大小写。
说明：如缺少该配置项、或者配置项错误，探针将使用值 True，开启探针功能。

[tingyun] app_name：

功能： 监控的应用名字
默认值：Python App
说明： 如缺少该配置项、空值、错误值等，将使用值 Python App 作为应用的名字上报数据。
探针支持多应用关连，该值可以设置为**英文分号**分割的多个应用名称。

[tingyun] log_file

功能： 指定探针 log 写入的文件名以及路径，**推荐使用绝对路径**。
默认值：/tmp/tingyun-agent.log，支持自定义系统文件路径、stdout、stderr
说明：若缺少该配置、空值、错误值等，探针可正常启动，log 将会输出到 stderr。
若指定了 stdout、或者 stderr，将会定向到系统标准输出。
由于 python 的 log 分割机制有缺陷，探针将会向一份日志文件中输出日志，请做好日志处理。
请确保您应用进程的用户对该目录和 log 文件有写入权限，否则将会输出到 stderr。

[tingyun] log_level

功能：指定探针 log 的日志级别
默认值：INFO，可选值 NOTSET, DEBUG，INFO, WARNING，WARN, FATAL ,ERROR，CRITICAL (不区分大小写)。
说明：若缺少该配置，或错误配置将默认使用 INFO 级别。

[tingyun] ssl

功能： 指定使用 http 或 https 传输协议
默认值：True，可选值 True, False, 以及 on, off，不区分大小写。
说明： 若缺少该配置、配置错误等，将使用值 True，使用 https 协议传输数据。

[tingyun] audit_mode

功能：是否将提交上报的数据会输出到 log 中以备审计，False 关闭审计，反之开启。（该 log 以 INFO 级别输出）
默认值：False，可选值 True, False, 以及 on, off，不区分大小写。
说明：若缺少该配置、配置错误等，将使用值 False，关闭审计模式。该部分 log 将以“Agent capture”开头， info 级别输出。

[tingyun] auto_action_naming

功能：设置是否开启自动事务命名，如开启自动命名，uri 名字将会做为 action 的名字。
默认值： True，可选值 True, False, 以及 on, off，不区分大小写。
说明：若缺少该配置、配置错误等，将使用值 True，开启自动命名。

[tingyun] action_tracer.log_sql

功能：事务跟踪时 SQL 语句的记录只写到本地日志文件中，不提交到数据采集服务上。（输出级别为 INFO 级别 log）
默认值： False，可选值 True, False, 以及 on, off，不区分大小写。
说明：若缺少该配置、配置错误等，将使用值 False，该部分 log 将以“Log sql is opened”开头。

[tingyun] urls_merge

功能：在关闭自动事务命名时，合并 uri 作为事务名称。
默认值： True，可选值 True, False, 以及 on, off，不区分大小写。

说明：
该参数用于开启 url 自动合并，仅当配置自动事务命名关闭，该选项开启时 url 自动合并才生效，其命名规则如下：
- uri 中除字符'/'外，出现的连续数字的将会被字符'*'替换。
- uri 中字符'/'中间部分的值为数字的将会被字符'*'替换。

[tingyun] verify_certification

功能：是否验证探针服务器网站证书。
默认值：True，当启用 ssl 加密传输数据时，不验证探针服务器的网站的证书信息。可选值 True, False, 以及 on, off，不区分大小写。

说明：
如果 python 包 certifi 的版本大于 2015.04.28 时，该证书使用 sha256 加密认证网站证书信息，但由于探针服务器证书颁发机构的根证书使用的是 sha1 加密，所以使用该版本之后的版本将会导致认证服务器证书失败的，以至于上传探针监测的数据失败，所以暂时的解决方案是，关闭服务器证书的认证。

[tingyun] tornado_wsgi_adapter_mode

功能：启用 tornado wsgi 应用模式
默认值：False， 支持 false， true， on， off，不区分大小写。
说明：
当使用第三方容器部署（如 gevent，uwsgi）tornado 应用（tornado wsgi 应用）时，需要开启该选项，探针才能正常工作

[tingyun:private] host

敬告：需要私有化探针时才需配置此选项，常规用户无需理会该选项！

功能：私有化时，用于配置内网的服务器重定向地址。
提示：如果配置文件没有 section [tingyun:private]，请手动添加该 section，然后再配置 host 等选项。

[tingyun:private] port

敬告：需要私有化探针时才需配置此选项，常规用户无需理会该选项！

功能：私有化时，用于配置内网的服务器用于重定向地址开放的端口号

[tingyun:exclude] plugins

功能：配置不需要监测的 python 包
默认值：空
说明：plugin 已英文半角逗号分隔，其支持的**包名**称如下表所示,使用示例:
[tingyun:exclude]
plugins=mysql,memcahced

数据库包名	框架包名	对外调用包名	模板以及其他
mysql	django	urllib	jinja2
pymysql	flask	urllib2	mako
oursql	web2py	urllib3	gevent
oracle	webpy	thrift	
postgresql	tornado4	requests	
psycopg2	nova	httplib2	
psycopg2ct			
psycopg2cffi			
pyodbc			
mongodb			
redis			
memcached			
pymemcache			
bmemcached			

[tingyun:proxy] proxy_host

功能：配置 http/https 代理服务器主机地址,用户探针链接外网上报性能数据。
默认值： 无
说明：该配置可与配置项 proxy_port, proxy_user, proxy_pwd 配合使用， 该选项支持格式：schme://user:password@host/path

[tingyun:proxy] proxy_port

功能：配置代理服务器主机端口号
默认值：无

[tingyun:proxy] proxy_user

功能：配置代理服务器用户名
默认值：无

[tingyun:proxy] proxy_pwd

功能：配置代理服务器用户密码，该选项和 proxy_user 对应。
默认值：无

[tingyun:proxy] proxy_scheme

功能：配置代理服务器数据传输协议。
默认值：http

4、环境变量

4.1 TING_YUN_CONFIG_FILE

该环境变量用于指向探针的配置文件，当用命令行启动探针，探针会自动在操作系统环境变量中寻找该变量，读取配置文件，该变量区分大小写。

如果应用的启动方式通过 shell 来控制，可将其植入到 shell 中。

4.2 DISABLE_TING_YUN_AGENT

该变量如果出现在 HTTP 请求的 HEADER 中，并且值为"真"，则该次请求的性

能值将被忽略。

5、请求阻塞时间

阻塞时间用于提取客户端发起请求到应用处理请求之间阻塞的时间， 即：

请求阻塞时间 = 请求到达前端服务器 到 应用服务器收到请求的时间

只需在 web Server 中配置相应的头即可，根据不同的应用服务器配置方案如下：

Web Server	App Server	Web Server 配置项
Nginx	uwsgi	uwsgi_param HTTP_X_QUEUE_START "s=$msec";
	tornado	proxy_set_header X-QUEUE-START "s=$msec";
	gunicorn	proxy_set_header X-QUEUE-START "s=$msec";
	gevent	proxy_set_header X-QUEUE-START "s=$msec";
	FASTCGI	fastcgi_param HTTP_X_QUEUE_START "s=$msec";
	SCGI	scgi_param HTTP_X_QUEUE_START "s=$msec";
Apache	mod_wsgi	RequestHeader set X-QUEUE-START "%t"

兼容与支持

听云 python 探针是根据 WSGI 协议而为 web 框架定制的性能监测客户端，理论上只要是基于 WSGI 协议的 web 框架都能对其监测。当然其中也包括一些该 web 框架支持的一些 python 的第三方包，例如 sql 数据库、nosql 数据库、第三方 http 调用等。其功能会随着版本的升级而得到更多的支持，目前其支持的组件以及框架结构如下所述。

目前所有的组件，包括数据库、第三方调用等，暂时没有提供独立的 api，都依赖于 web 框架而存在。

基于下述框架下所有 wsgi 应用都可以支持， 均经过验证和测试。如果您发现有下述框架下的 wsgi 应用或其他框架应用有不支持的情况，请联系我们。

1、 框架

1.1 支持的 web 框架

支持框架	建议版本	备注
django	1.4.x - 1.7.x	
flask	0.6 以上	
Webpy	0.3.x	
Web2py	2.8.1-2.12.x	
Tornado	v4.x.x	不支持 browser 探针自动嵌码
bottle	0.10.x-0.12.x	

1.2 支持组件以及部署方式

框架性能监测组件	部署方式
请求中间件	Uwsgi（须开启--enable-threads 和 --single-interpreter 选项）
视图中间件	gunicorn
模板响应中间件	mod_wsgi
响应中间件	AJP
异常中间件	FASTCGIT
视图函数、模板渲染等	SCGI
异常捕获	独立的 WSGI 容器
其他	CGI

2 数据库支持

Python 探针目前支持以下数据库性能监测，若您站点使用其他类型的数据库 api，其数据库的性能将不会被采集。随着探针的升级，更多的模块将会被支持。

数据库	数据库模块	备注
mysql	mysql-python	(1.2.3-1.2.5)
	pymysql	(0.6.x-0.7.x)
oracle	cx_Oracle	(5.1.x-5.2.x)
PostgreSQL	psycopg2	(2.3.x-2.6.x)
	psycopg2ct	psycopg2ct(0.2.x-2.x)
	psycopg2cffi	(2.5.x-2.7.x)
ODBC	pyodbc	(2.1.x-3.0.x)
Memcached	python-memcached	(1.4.x-1.5.x)
	python-binary-memcached	(0.17-0.24.6)
	pymemcache	(1.0.3-1.3.6)
MongoDB	pymongo	(2.x-3.x)
Redis	redis	

3 外部调用支持

仅支持以下列表中的组件的性能采集，若您站点使用其他类型的外部调用模块，其性能数据库将不会被采集。随着探针的升级，更多的模块将会支持。

模块	备注
urllib	
urllib2	
urllib3(v1.6-v1.14)	对外调用时，支持跨应用追踪
requests(2.0.0-2.10.0)	对外调用时，支持跨应用追踪
thrift	
httplib2(0.7.5-0.9.2)	对外调用时，支持跨应用追踪
pika(0.10.x)	采集 RabbitMQ 的消费者、生产者性能数据

4 python 与系统平台

python	平台
Cpython2.6.x 2.7.x	类 unix 系统，如 linux、FreebSD、MacOS X 等

5 其他部分

模块	备注
Jinja2	2.3 以上版本
mako	v0.7.x-v1.0.x
django-piston	0.2.x

异常排查

1、 没有 log 信息输出

1.1 现象描述

没有生成探针 log 或者 log 中没有任何 log 输出。或者除了”agent check log file config input message”外没有其他 log（由 tingyun-admin check-config 命令生成）。

1.2 解决方案

①、将探针装在与用户应用使用的同一个 python 环境中，须放在一个环境才能工作。如果使用了虚拟环境需要特别注意。

② 为探针配置文件设置读取权限。

执行应用进程的 linux 用户需要对探针配置有读取权限，否则可能因为权限问题无法读取。

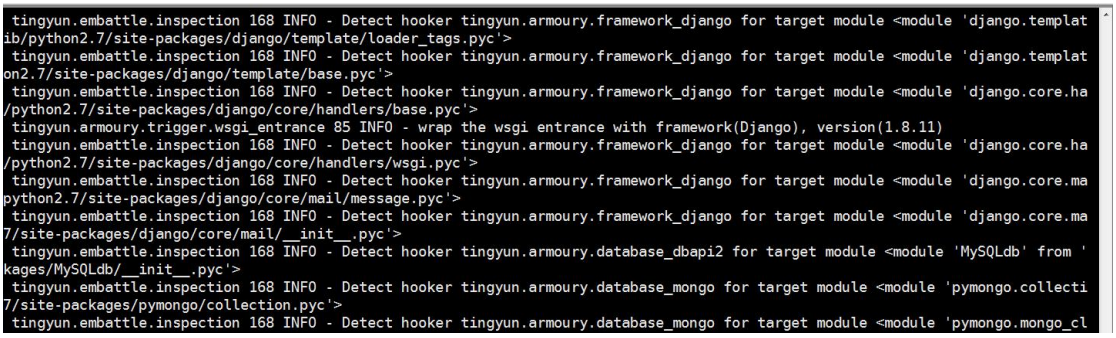
③ 对配置文件中的 log 文件设置合理的写入权限。

应用程序运行用户需要对配置的探针 log 有写入权限，否则不会有任何 log 出现。

2、 仅有探针初始化部分 log

2.1 现象描述

探针 log 只有探针初始化部分 log 如下图示例， 后续没有数据上报等 log 输出，但报表里却有数据显示。



2.2 解决方案

用户应用启动后，对 python 的 logging 模块做了配置，使用了 disable_existing_loggers=True 参数，禁止掉了第三方 log 的输出。
或者，python logging 模块优先初始化了探针 log，而后被监控的应用程序又调用了 logging.config.fileConfig()函数。

此时只需将 python logging 配置中的 disable_existing_loggers 参数设置为 False 即可。

其他常见问题

1、更新了本地配置，为什么没有生效？

如果更新本地配置文件，探针系统不能自动识别，目前解决方案为：重启应用（探针也被重启）。

2、怎样确认探针配置文件、配置选项是否正确、合理

我们提供了命令行工具 *tingyun-admin check-config* 检查配置文件，能检查常见选项值是否有误，但不能检查 license 有效性。

3、私有化时，探针部署成功，log 中一堆 license 无效的错误

部署私有化时，没有对 host、port 参数，设置 section，导致数据往公网服务器上报所致。

解决方案，针对私有化进行专门配置。

4、报表显示采集的请求响应时间信息误差偏大

这种情况很可能是因为探针没有采集到应用阻塞时间，整个请求响应时间 阻塞时间 + 应用层处理时间。

解决方案，在 web 服务器设置上请求头，以便探针处理阻塞时间。详情参考请求阻塞时间