



Python 探针用户指南

北京基调网络股份有限公司



简述

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。

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 <http://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

一、快速入门

1、环境要求

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	建议 v1.3-v1.6 稳定版本
使用范围	兼容 WSGI 1.0 (PEP 333)	

2、快速安装

注意：快速安装需要用到 pip（[推荐](#)） 或者 `setuptools` 工具，如果您的环境无法提供，请移驾[源码安装](#)。

- 安装探针：`pip install tingyun` 或者 `easy_install tingyun`
- 生成本地配置文件：`tingyun-admin generate-config YourLicenseKey outputFile`
如：`tingyun-admin generate-config 123-456-789-001 /tmp/tingyun.ini`
- 设置探针配置文件环境变量。
如：`export TING_YUN_CONFIG_FILE=/tmp/tingyun.ini`
- 嵌入探针：`tingyun-admin run-program 您应用的启动命令 您应用启动参数`
如：`tingyun-admin run-program ./uwsgi.sh start`

说明：以上步骤将会设置探针的 log 输出为 `/tmp/tingyun-agent.log`。关于探针更详细的用法和配置，参考章节 [探针工具](#)和[探针配置](#)

二、安装与配置

由于 python 极其灵活的安装方式，建议使用虚拟环境，如 **virtualenv**, **pyvenv** 避免污染系统环境：

1、探针的安装

1.1 探针源码安装

安装包地址为 <http://report.tingyun.com/download/serverSdk>，假设得到的安装包为 `tingyun-agent-python-x.x.x.tar.gz`。

- 解压安装包并进入安装目录，如：

```
tar -zxvf tingingun-agent-python-x.x.x.tar.gz -C /tmp
cd tingingun-agent-python-x.x.x
```
- 执行以下命令安装，即可将探针安装到当前的 python 环境。

```
python setup.py install
```

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[探针使用](#)章节

1.2 在线安装

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

- `pip install tingingun` 或者
- `easy_install tingingun`

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[探针使用](#)章节

2、探针升级

2.1 源码升级

源码方式升级探针，只需重新执行安装即可覆盖旧版本即可（详情参考[<探针源码安装>](#)），建议进入安装目录将安装的 `tingyun` 包给删除。

2.2 在线升级

注意： 在线升级时请注意对于安装时使用的工具，否则可能升级失败。

- **pip 方式升级**
`pip install -U tingyun`
- **easy_install 方式升级**
`easy_install -U tingyun`

3、探针卸载

3.1 源码安装方式的卸载

如果以源码方式安装的探针，卸载探针时，只需将探针安装目录下的 `tingyun` 包（通常在 `site-packages` 下）以及可执行目录下的 `tingyun-admin` 删除即可。

3.2 在线安装方式卸载

- **pip 方式卸载**

执行，`pip uninstall tingyun`

- **easy_install 方式卸载**

执行，`easy_install -m tingyun`

注意，由于该命令只是将安装包信息从 `.pth` 文件删除掉，安装包的脚本、包信息还在环境中，仅仅是导入时无法导入该安装包，若要完全移除安装包信息，需要手动删除安装包，详情参考[源码方式卸载](#)。

4、探针的配置

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 [探针工具 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	备注
tingyun	<code>license_key</code>	字符型
	<code>enabled</code>	Boolean 型
	<code>app_name</code>	字符型
	<code>log_file</code>	字符型
	<code>audit_mode</code>	Boolean 型
	<code>log_level</code>	字符型
	<code>ssl</code>	Boolean 型
	<code>auto_action_naming</code>	Boolean 型
	<code>action_tracer.log_sql</code>	Boolean 型

注：本地配置的修改需重启方能生效！

4.1 本地配置

4.1.1 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。

4.1.2 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。

4.1.3 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。

4.1.4 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。

4.1.5 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。

4.1.6 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。

4.1.7 audit_mode

功能：是否将提交上报的数据将会输出到 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。

4.1.8 auto_action_naming

功能： 设置是否开启自动命名，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。

4.1.9 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。

4.2 环境变量

4.2.1 TING_YUN_CONFIG_FILE

该环境变量用于设置配置文件的环境变量，探针在启动的时候需要加载该文件，如没有找到对应的配置文件，探针无法启动。

配置该环境变量后，可以使用[探针工具](#) `tingyun-admin check-config` 来检查该设置是否正确。

三、使用与维护

1、局限性、限制

※ 由类似 `greenlet` 等第三方协程、线程调用模块。其监测性能数据可能会有偏差。

※ 不支持未在本文档内注明支持的其他框架、模块。

※ 随着探针升级更多的框架、组件将得到支持，敬请期待。

2、探针使用

为了响应更多的需求，我们将会根据客户的反应、需求、以及探针自身的因素，进行不定期的升级。为了更好的用户使用体验，升级后我们会第一时间通知您，届时请升级到最新版。同时也欢迎您向我们提供宝贵的意见与建议！

2.1.1 探针启动

目前探针的启动方式为伴随着 `python` 解释器启动而启动，也就是说您只需要在您的应用程序启动命令前加上探针命令以及参数，探针就已经启动了，支持以下方式启动探针：

探针命令行命令	探针命令行参数	您的应用程序以及参数
tingyun-admin	run-program	

例如：

※ `tingyun-admin run-program python runserver`

※ `tingyun-admin run-program uwsgi_shell.sh start`

注意：探针仅支持满足 `WSGI` 协议的 `web` 应用，其他类型的应用的数据将不会被监测。

2.1.2 探针停止

探针会随着您应用程序的停止而停止，所以大致有以下两种正常的停止方式：

- ※ 直接使用您应用程序的停止方式停止探针，如 `uwsgi_shell.sh stop`
- ※ 使用探针命令方式停止探针，如 `tingyun-admin run-program uwsgi_shell.sh stop`

2.1.3 探针工具

由于探针使用了配置文件，以及环境变量方式配置探针，该工具用于检测探针配置的合法性，其使用方式如下：

探针命令行命令	探针命令行参数 1	探针命令行参数 2	探针命令行参数 3
tingyun-admin	check-config	[config_file_path]	
	generate-config	[license_key]	output_file

注意：以上命令不会对探针的 **license** 的合法性、有效性做检查

check-config 示例：

1、如果您对探针的配置文件，设置了[环境变量](#)，使用默认参数即可对配置文件进行检测和校验，其命令如下：

```
tingyun-admin check-config
```

2、对指定的配置文件进行校验（假设探针配置文件放置在/tmp/目录下）。

```
tingyun-admin check-config /tmp/tingyun.ini
```

3、**注意事项**，运行该命令的用户需要和您应用程序进程所属用户一致，才能有效的检查配置是否合理。

generate-config 示例：

1、提供 **license** ， 此时 **license** 会自动写入配置文件中，**output_file** 为**必须选项**。

```
tingyun-admin check-config 123-456-789-10 /tmp/tingyun.ini
```

2、稍后修改 **license**，**output_file** 为**必须选项**。

```
tingyun-admin check-config /tmp/tingyun.ini
```

若要修改更详细的配置，需手动修改，请参考[本地配置](#)>



3 探针的维护

如果配置了 log（建议配置 log），探针的所有信息将会根据设定的日志级别输出到 log 中，如果发现有任何异常，可根据 log 输出，获取更多的支持和帮助。

若需获取更多的帮助，请联系我们的技术支持。

四、支持框架

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,理论上基于只要是基于 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。

虽然所有基于下述框架下所有 wsgi 应用都可以支持,下述的所有支持,均经过验证和测试的。如果您发现有下述框架下的 wsgi 应用有不支持的情况,请向联系我们。

1 框架

1.1、django 的支持

django 符合 WSGI 标准,理论上所有的版本都能支持,这里建议使用 1.3.x 以上 1.7 以下版本。

- 其支持以下启动方式。

启动应用方式	说明
uwsgi	必须开启以下选项: --enable-threads --single-interpreter
gunicorn	
mod_wsgi	
AJP	暂支持 flup 下, threaded 模式
FASTCGIT	暂支持 flup 下, threaded 模式
SCGI	暂支持 flup 下, threaded 模式

- 支持以性能数据监测
 - ※ 支持请求中间件.
 - ※ 支持视图中间件.
 - ※ 支持模板响应中间件.



- ※ 支持响应中间件.
- ※ 支持异常中间件.
- ※ 支持视图函数、模板渲染等.
- ※ 支持 url 解析时产生的异常
- ※ 支持视图处理时产生的异常
- ※ 支持 500 错误时内部产生的异常
- ※ 其他部分

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，更多的模块将会支持。

数据库	数据库模块	备注
mysql	MySQLdb	
	pymysql	
oracle	cx_Oracle	
PostgreSQL	psycopg2	
	psycopg2ct	
	psycopg2cffi	
ODBC	pyodbc	
Memcached	python_memcached	
	bmemcached	
MongoDB	pymongo	
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	

五、故障与修复

1、探针常见问题

1、探针 log 文件未生成，如果配置了 log，但又没有生成 log 文件，可能有以下原因，请逐一排查。

- ※ 是否配置了探针的配置文件的环境变量，并正确设置路径、读取权限等。
- ※ 指定的文件名所在的目录不存在、或者该目录下您的应用程序进程的用户无权限写入。

- ※ python logging 模块的冲突。

如果 python logging 模块优先初始化了探针 log，而后监控的应用程序又调用了 logging.config.fileConfig() 函数，那么将会引发这个问题。

解决的方案之一是向该函数中传递 disable_existing_loggers=False 参数，让 logging module 允许存在初始化的 log。

2、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）。

3、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，详情参考 [<探针工具>](#)

附加说明

该章节对于一些探针的技术点、限制、特点等进行一些详细说明。

1、关于 uwsgi 的说明

`--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。