《基调网络听云PHP探针说明》

《基调网络听云PHP探针说明》

PHP探针运行环境

1、确认您的系统满足兼容性要求

2、如果您还没有听云server帐户，请注册听云server帐户

3、从听云server探针安装页面复制您的授权序号

4、安装PHP探针

5、配置PHP探针的授权序号

6、重启PHP宿主程序(Apache或者PHP-FPM)

7、通过管理页面察看数据，有问题请参考安装和排除故障章节。

兼容性和要求

安装PHP 探针之前，请确保您的系统满足如下这些条件。

	要求
操作系统	Linux (x86 and x86_64): • RedHat Enterprise Linux (RHEL) 5.0 或更高版本 • CentOS 5.0 或更高版本 • Suse Linux 10.0或更高版本 • Debian 5.0 ("lenny") 或更高版本 • Ubuntu 9.10 ("Karmic Koala") 或更高版本 • 任何其他内核版本高于2.6.13，GLIBC版本高于2.5并且提供本地Posix线程库支持的linux发行版。
PHP	5.2 5.3 5.4 5.5 5.6 7.0
Web服务器	Apache 2.2以上（apache module方式） Nginx + PHP-FPM （fastCGI的方式） 或者任何其它支持fastCGI的PHP服务器

PHP 探针架构

1. 探针的文件组成:

默认探针安装根路径 %prefix% = /usr, 使用bin安装包可以利用prefix参数指定探针安装路径
默认探针运行根路径 %runtime% = /var, 可以在php扩展配置文件内更改运行路径
探针安装后，对Linux系统的文件更改如下:

- 守护进程 %prefix%/bin/networkbench
- PHP扩展 %prefix%/lib/networkbench/
- 探针日志路径 %runtime%/log/networkbench/
- 辅助运行路径 %runtime%/run/networkbench/
- PHP扩展文件 {PHP扩展路径}/networkbench.so
- PHP扩展配置文件 {PHP附加配置文件路径}/networkbench.ini 或 {PHP主配置文件}
- 探针安装日志 /tmp/nbinstall-安装日期.tar

2. PHP扩展模块

PHP扩展模块负责收集PHP运行时的性能，并将性能数据发送给守护进程。
PHP扩展本身不与听云服务器进行数据交换，PHP扩展通过将数据发送到守护进程进行处理来减少对系统资源的消耗。

3.守护进程

守护进程是介于听云服务器和PHP扩展之间的一个汇总进程，接收php扩展模块捕获的运行时性能数据，汇总压缩后传送到听云系统后台。不启用代理守护进程，就不会有数据发送到听云系统。
汇总进程会被PHP自动启动，不需要用户手动启动;

常见问题

1. PHP探针都能采集哪些数据?

目前，探针可以采集php运行时的性能数据和错误信息。

性能数据包括：HTTP请求的总响应时间、从负载均衡服务器到web服务器的排队时间、数据库耗时、NOSQL耗时、web service耗时等性能；数据库性能统计；NOSQL性能统计；超过阈值的性能瓶颈函数；慢sql语句及调用堆栈;

错误信息包括：数据库连接错误、sql语法错误、php语法错误、php运行时未捕获的异常、http状态码为4xx 5xx的错误等;

2. 我在多台服务器上部署了应用探针，报表查看时，如何将不同的应用区分开?

默认数据汇总在同一个应用，应用名称为“PHP Application”，如果需要分开看，把配置文件内的nbs.app_name（应用名称） 改为不同名称就可以了。

3. 我在一台服务器上部署了多个Virtual Host，报表查看时，如何将不同的应用区分开?

把配置文件内的 nbs.auto_app_naming 修改为1或2 即可自动命名不同Virtual Host下的应用

4. 服务器权限要求比较严格，无法在/usr下安装探针，怎么办?

可以使用bin安装包指定安装路径, 假设您有 /home/tingyun 权限

./tingyun-agent-php-xxx.x86_64.bin -prefix=/home/tingyun

安装包会将探针先解压到指定路径，在安装过程中如果出现其他权限问题，请参考“故障排查”一节的“手工安装”步骤

安装包

对应基于x86/x86_64处理器的各个Linux发行版本，基调网络提供6个安装包，请选择适合您系统的安装包安装。

安装包	适用Linux 发行版	32位版本	64位版本
RPM安装包	• RedHat Enterprise Linux (RHEL) 5.0 或更高版本 • CentOS 5.0 或更高版本 • Suse Linux 10.0或更高版本	tingyun-agent-php-版本.i386.rpm	tingyun-agent-php-版本.x86_64.rpm
DEB安装包	• Debian 5.0 ("lenny") 或更高版本 • Ubuntu 9.10 ("Karmic Koala") 或更高版本	tingyun-agent-php-版本.i386.deb	tingyun-agent-php-版本.x86_64.deb
BIN安装包	• 任何其他内核版本高于2.6.13，GLIBC版本高于2.5并且提供本地Posix线程库支持的linux发行版。	tingyun-agent-php-版本.i386.bin	tingyun-agent-php-版本.x86_64.bin

RPM安装包(RHEL,CentOS,SUSE)

RPM安装包适用于已经安装了rpm系统的linux发行版，包括(但不限于) CentOS5及以后版本、RHEL5及以后版本、SUSE 10.0及以后版本。

安装步骤:

- 执行rpm安装

  32位版本rpm包安装

  ```
  sudo rpm -Uvh tingyun-agent-php-版本.i386.rpm
  ```

  64位版本rpm包安装

  ```
  sudo rpm -Uvh tingyun-agent-php-版本.x86_64.rpm
  ```

- 将探针关联到PHP

  ```
  sudo networkbench-install.sh
  ```

  在接下来的录入界面输入license

- 重启php宿主服务器(apache, php-fpm)

  此后，当您的php后台服务有http请求进入，几分钟后性能数据将发送到听云后台。

卸载步骤:

- 首先卸载php扩展文件和依赖的so文件

  ```
  networkbench-install.sh uninstall 或手工移除php扩展路径下
  的networkbench.so及networkbench.ini
  ```

- 然后移除探针

  ```
  sudo rpm -e tingyun-agent-php
  ```

- 重启php宿主服务器(apache, php-fpm)

更新步骤:

- 安装新版本php探针

  ```
  sudo rpm -Uvh tingyun-agent-php-版本.x86_64.rpm
  ```

- 停止守护进程

```
sudo killall networkbench
```

- 重启php宿主服务器(apache, php-fpm)

DEB安装包(Debian,Ubuntu)

RPM安装包适用于已经安装了dpkg系统的linux发行版，包括(但不限于) Debian5及以后版本、Ubuntu9.10及以后版本。

安装步骤:

- 执行deb安装

  32位版本rpm包安装

  ```
  sudo dpkg -i tingyun-agent-php-版本.i386.deb
  ```

  64位版本deb包安装

  ```
  sudo dpkg -i tingyun-agent-php-版本.x86_64.deb
  ```

- 将探针关联到PHP

  ```
  sudo networkbench-install.sh
  ```

  在接下来的录入界面输入license

- 重启php宿主服务器(apache, php-fpm)

  此后，当您的php后台服务有http请求进入，几分钟后性能数据将发送到听云后台。

卸载步骤:

- 首先卸载php扩展文件和依赖的so文件

  ```
  networkbench-install.sh uninstall 或手工移除php扩展路径下
  的networkbench.so及networkbench.ini
  ```

- 然后移除探针

  ```
  sudo dpkg -r tingyun-agent-php
  ```

- 重启php宿主服务器(apache, php-fpm)

更新步骤:

- 安装新版本php探针

  ```
  sudo dpkg -i tingyun-agent-php-版本.x86_64.deb
  ```

- 停止守护进程

```
sudo killall networkbench
```

- 重启php宿主服务器(apache, php-fpm)

BIN安装包(其他 linux发行版)

与RPM安装包和DEB安装包相比，BIN安装包的适用范围更广,BIN安装包适用于基于x86/x86_64、内核版本高于2.6.13,GLIBC版本高于2.5的linux服务器发行版。在没有dpkg和rpm的系统中可以使用bin包安装。

在需要探针以非root权限安装、运行或需要将探针安装到用户指定目录下时，必须使用bin包进行安装

安装步骤:

- 下载bin包

  32位版本bin包

  ```
  wget -q http://download.networkbench.com/agent/php/版本/tingyun-agent-php-版本.i386.bin && chmod 755 tingyun-agent-php-版本.i386.bin
  ```

  64位版本bin包

  ```
  wget -q http://download.networkbench.com/agent/php/版本/tingyun-agent-php-版本.x86_64.bin && chmod 755 tingyun-agent-php-版本.x86_64.bin
  ```

- 执行bin安装

  运行bin包进行安装，默认安装到 /usr 下

  ```
  ./tingyun-agent-php-版本.x86_64.bin
  ```

  如果您需要将bin包安装到指定位置，请使用 --prefix 参数

  ```
  ./tingyun-agent-php-版本.x86_64.bin --prefix=/path/to
  ```

- 在接下来的配置界面输入license和应用名称

  如果指定的安装路径没有写权限，安装过程中会提示更改安装路径，安装路径下为so文件和bin文件

  如果默认安装路径不在/usr, 安装过程中会提示指定运行路径，默认/var， 运行路径下为log文件和pid文件

  如果提示由于权限原因导致失败，可以走手工安装步骤，见“故障排查”中“手动安装PHP扩展”

- 重启php宿主服务器(apache, php-fpm)

此后，当您的php后台服务有http请求进入，几分钟后性能数据将发送到听云后台。

卸载步骤:

- 首先卸载php扩展文件和依赖的so文件

```
networkbench-install.sh uninstall 或手工移除php扩展路径下
的networkbench.so及networkbench.ini
```

- 然后移除探针

```
./tingyun-agent-php-版本.x86_64.bin uninstall
```

- 重启php宿主服务器(apache, php-fpm)

更新步骤:

- 更新php探针需要执行3个步骤
  - 停止守护进程

```
sudo killall networkbench
```

  - 卸载php探针
  - 安装新版本php探针并重启php宿主服务器(apache, php-fpm)

安装脚本

用途

PHP探针程序需要安装php扩展模块到php环境中才能正常工作，并且需要授权序号才能提供数据服务。

并且，各个系统的php版本和环境有很大差别，因此，我们需要有程序来做检测和安装的工作。

安装脚本充当了这个角色。

安装脚本需要在首次安装听云php探针、或者重新安装php之后运行。

安装脚本位置

```
/usr/bin/networkbench-install.sh
```

安装步骤及选项含义

- 1、启动安装脚本

  ```
  sudo sh /usr/bin/networkbench-install.sh
  ```

- 2、选择安装或解除安装

  安装脚本启动后，首先需要您选择是安装到php环境还是从php环境卸载探针。

  ```
  Please select from one of the following options:

  1)  Install
  2)  Uninstall

  0)  Exit

     Enter choice (1-2, 0 to exit):
  ```

  键盘输入"1" 将继续安装。

- 3、输入授权码和网站应用名称

  授权码是您从听云获得的服务授权号，请勿泄漏。

  ```
  Enter license key (请录入授权码):
  ```

```
```

输入您的授权序号，继续安装。

应用名称是给您的网站起一个名称，方便在报表显示。

```
Enter App name (请输入应用名称):
```

输入网站的应用名称（直接回车则默认应用名称为PHP Application），继续安装。

- 4、php版本选择

  如果您的系统中安装了多个版本的php, 安装脚本会提示您做出选择，安装到某个指定版本的php或全部安装。

```
Below is a list of the directories in which we found a copy of PHP.
Please
select the directory or directories for which you wish to install
Networkbench.
You can select either a single directory or multiple directories by
separating
each choice with either a space or a comma. To select all of the
directories
shown, please enter the special keyword 'all' (without the quotes).

   1)   /usr/bin
   2)   /usr/local/php/5.3.23/bin

   0)   Exit

   Selection (1-2, 0 to exit or all):
```

  键入您选择的序号，或者all全部安装。

- 5、安装结束

  如果一切顺利，您将看到如下提示:

```
Networkbench is now installed on your system. Congratulations!
```

- 6、重启apache或者您的fast-cgi服务器。

卸载

使用此脚本卸载php探针扩展:

```
sudo sh /usr/bin/networkbench-install.sh
```

选择 2。

或者:

```
sudo sh /usr/bin/networkbench-install.sh uninstall
```

守护进程启动方式

完成安装后，重启php的宿主程序(apache或者php-fpm)后，守护进程就已经被启动了。

- 随Apache或PHP-FPM启动而启动

  宿主程序(apache或者php-fpm)会自动生成守护的配置文件(/var/run/networkbench/.networkbench.cfg)，并且尝试自动启动守护进程(/usr/bin/networkbench)

  如果进程列表内缺少了networkbench进程，请使用下面的命令启动守护进程

  ```
  sudo networkbench -f /var/run/networkbench/.networkbench.cfg
  ```

卸载

卸载PHP探针需要3个步骤:

- 1、停止守护进程:

```
sudo killall networkbench
```

- 2、从PHP环境中卸载探针扩展:

```
sudo sh /usr/bin/networkbench-install.sh uninstall
```

如果为手工安装，请手工移除php扩展路径下的networkbench.so及配置文件networkbench.ini

- 3、移除安装包:

  rpm安装版本:

```
sudo rpm -e tingyun-agent-php
```

  deb安装版本:

```
sudo dpkg -r tingyun-agent-php
```

  bin安装版本:

```
sudo ./tingyun-agent-php-latest.x86_64.bin uninstall
```

- 4、重启php相关服务，例如apache或php-fpm

快速安装

安装步骤:

- 1、软件包安装
  - RPM

    适用于CentOS,RHEL,SUSE

    32位系统

    ```
    sudo rpm -Uvh tingyun-agent-php-版本.i386.rpm
    ```

    64位系统

    ```
    sudo rpm -Uvh tingyun-agent-php-版本.x86_64.rpm
    ```
  - DEB

    适用Debian, Ubuntu

    32位系统

    ```
    sudo dpkg -i tingyun-agent-php-版本.i386.deb
    ```

    64位系统

    ```
    sudo dpkg -i tingyun-agent-php-版本.x86_64.deb
    ```
  - BIN

    适用其他内核高于2.6.13的Linux发行版

    32位系统

    ```
    sudo ./tingyun-agent-php-版本.i386.bin
    ```

    64位系统

    ```
    sudo ./tingyun-agent-php-版本.x86_64.bin
    ```

- 2、安装配置

```
sudo sh /usr/bin/networkbench-install.sh
```

- 3、重启apache或者fast-cgi服务器，安装完毕

更新

- 1、停止web容器和守护进程

```
sudo service httpd stop 或 sudo service php-fpm stop

sudo killall networkbench
```

- 2、软件包更新
  - RPM

    适用于CentOS,RHEL,SUSE

    32位系统

    ```
    sudo rpm -Uvh tingyun-agent-php-版本.i386.rpm
    ```

    64位系统

    ```
    sudo rpm -Uvh tingyun-agent-php-版本.x86_64.rpm
    ```

  - DEB

    适用Debian, Ubuntu

    32位系统

    ```
    sudo dpkg -i tingyun-agent-php-版本.i386.deb
    ```

    64位系统

    ```
    sudo dpkg -i tingyun-agent-php-版本.x86_64.deb
    ```

  - BIN

    适用其他内核高于2.6.13的Linux发行版

    32位系统

    ```
    sudo ./tingyun-agent-php-版本.i386.bin update
    ```

    64位系统

```
sudo ./tingyun-agent-php-版本.x86_64.bin update
```

- 3、重启web容器

```
sudo service httpd start 或 sudo service php-fpm start
```

听云PHP探针配置

1）编辑配置文件

```
vi 配置文件
```

2）kill探针的守护进程

```
sudo killall networkbench
```

3) 重启web server

```
sudo service httpd restart 或 sudo service php-fpm restart
```

配置文件位置会根据软件环境不同而不同,几个典型的位置包括: /etc/php.d/networkbench.ini 或 /etc/php5/fpm/conf.d/networkbench.ini，如果php是通过源码编译安装的，那么配置文件的位置依赖于编译时选项指定的位置。

1） 如果编译php时指定了 --with-config-file-scan-dir 选项，那么networkbench.ini会放在编译时指定的位置

运行 php -i |grep .ini, 其中的 Scan this dir for additional .ini files 标明了配置文件的具体位置

例如 Scan this dir for additional .ini files => /opt/php.d/, 相应配置文件为 /opt/php.d/networkbench.ini

2) 如果编译php时禁止了 --with-config-file-scan-dir 选项，那么配置文件会合并在php主配置文件中

运行 php -i |grep .ini, 其中的 Scan this dir for additional .ini files =>(none) 表示编译php时禁止了 --with-config-file-scan-dir 选项, 其中的 Loaded Configuration File 标明了php主配置文件的具体位置

例如 Loaded Configuration File=> /opt/php55/lib/php.ini, 说明配置在 /opt/php55/lib/php.ini文件内

启用/禁用探针

配置选项格式:

```
nbs.agent_enabled = true
```

数据类型: boolean
取值: true/false

本选项设置为false时，php探针将不再采集性能数据。

默认情况下，当您想临时禁用探针时，应该通过性能报表内的开关来临时禁用和启用探针。

当多个虚拟主机的性能数据汇总在同一名称的应用下时，通过报表禁用应用会造成全部虚拟主机的数据都无法采集，此时可以通过下述配置来禁用其中某一虚拟主机的性能数据。

1.Apache 方式：修改httpd.conf 或 .htaccess，这需要有“AllowOverride Options”或“AllowOverride All”权限

对需要禁用的virtual hosts增加php_flag nbs.agent_enabled off,例如:

```
<VirtualHost 192.168.1.3>
 ServerName www.test2.com
 DocumentRoot "/path/to/vhost2/"
 ...
 <IfModule PHP_MODULE>
  php_flag nbs.agent_enabled off
 </IfModule>
</VirtualHost>
```

注意IfModule 语句中“PHP_MODULE”应该和引入php模块时的名称一致，如果您加载PHP时使用“LoadModule php5_module modules/libphp5.so”，则对应语句应该修改为"<IfModule php5_module>"

2. php-fpm方式：修改php-fpm.conf

```
[app2]
listen=/tmp/pool-app2.sock
php_flag[nbs.agent_enabled] = off
```

3. nginx方式：修改nginx.conf

```
location /app1{
fastcgi_param PHP_VALUE "nbs.agent_enabled=false";
 ...
}
```

授权序号

配置选项格式:

```
nbs.license_key = "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
```

数据类型: string

取值: 字符串

默认值: 无

说明

本条配置项是您在安装时输入的授权序号。

应用命名

自动命名

```
nbs.auto_app_naming = 0
```

数据类型: 整型
0 - 禁用自动命名，默认禁用
1 - 启动自动命名，命名规则：使用虚拟主机的域名+端口作为应用名称
2 - 启动自动命名，命名规则：使用nbs.app_name+端口作为应用名称

当您在同一台服务器上部署多个应用，并且使用了虚拟主机（Virtual Host）时，建议打开此选项
当使用域名区分虚拟主机时（即Virtual Host中配置了ServerName），建议使用
nbs.auto_app_naming = 1
当仅使用端口号区分虚拟主机时，建议使用 nbs.auto_app_naming = 2，否则由于未配置域名，可能会产生一些非预期之内的应用名称

应用名称

```
nbs.app_name = "PHP Application"
```

数据类型: string, 默认值: "PHP Application"

如果通过自动命名的方式获取的应用名称不满足您的要求，您需要将nbs.auto_app_naming设置为0，
然后通过修改APACHE或php-fpm或NGINX的配置文件来区分不同的应用。

1.Apache 方式：修改httpd.conf 或 .htaccess，这需要有“AllowOverride Options”或“AllowOverride All”权限

对每个virtual hosts增加 php_value nbs.app_name "my app" , 例如：

```
<VirtualHost 192.168.1.2>
 ServerName www.test.com
 DocumentRoot "/path/to/vhost/"
```

```
...
<IfModule PHP_MODULE>
 php_value nbs.app_name "my app name"
</IfModule>
</VirtualHost>

<VirtualHost 192.168.1.3>
ServerName www.test2.com
DocumentRoot "/path/to/vhost2/"
...
<IfModule PHP_MODULE>
 php_value nbs.app_name "another app"
</IfModule>
</VirtualHost>
```

注意IfModule 语句中“PHP_MODULE”应该和引入php模块时的名称一致，如果您加载PHP时使用“LoadModule php5_module modules/libphp5.so”，则对应语句应该修改为"<IfModule php5_module>"

2. nginx方式：修改nginx.conf

对virtual host每个站点内增加配置参数即可，例如：

```
server {
  listen 80;

  server_name www.web.com;

  location ~ /.php$ {
  fastcgi_param PHP_VALUE "nbs.app_name=www.web.com";
  ...

  }

}
```

```
server {
  listen 80;
```

```
 server_name bbs.web.com;

 location ~ /.php$ {
 fastcgi_param PHP_VALUE "nbs.app_name=bbs.web.com";
 ...

 }

}
```

或对按路径区分的站点，例如

```
location /app1 {

fastcgi_param PHP_VALUE "nbs.app_name=my app name";

 ...

}

location /app2{

fastcgi_param PHP_VALUE "nbs.app_name=another app";

 ...

}
```

3. php-fpm方式：修改php-fpm.conf

```
[app1]

listen=/tmp/pool-app1.sock

php_value[nbs.app_name] = "my app name"

[app2]

listen=/tmp/pool-app2.sock

php_value[nbs.app_name] = "another app"
```

日志管理

PHP探针有两个模块:守护进程和PHP扩展。每个模块都有单独的日志管理

1、PHP扩展日志:

- 日志文件路径

  配置选项格式:

  ```
  nbs.agent_log_file_name = "/var/log/networkbench/php-agent.log"
  ```

  数据类型: string
  默认值: "/var/log/networkbench/php-agent.log"

  ### 说明:

  指定php扩展的日志文件路径。

- 日志级别

  配置选项格式:

  ```
  nbs.agent_log_level = "info"
  ```

  数据类型: string
  取值: "OFF", "CRITICAL", "ERROR", "WARNING", "INFO", "VERBOSE", "DEBUG"
  默认值: "INFO"

  ### 说明:

  本选项是控制日志数据写入日志文件的级别。"DEBUG"是最低级，允许所有日志信息写入日志文件。"OFF"是最高级，禁止所有日志信息写入日志文件。

2、汇总进程日志:

- 日志文件路径

  配置选项格式:

  ```
  nbs.daemon_log_file_name = "/var/log/networkbench/daemon.log"
  ```

  数据类型: string
  默认值: "/var/log/networkbench/daemon.log"

  说明:

  指定守护进程的日志文件路径。

- 日志级别

  配置选项格式:

  ```
  nbs.daemon_log_level = "info"
  ```

  数据类型: string
  取值: "OFF", "CRITICAL", "ERROR", "WARNING", "INFO", "VERBOSE", "DEBUG"
  默认值: "INFO"

  说明:

  本选项是控制日志数据写入日志文件的级别。"DEBUG"是最低级，允许所有日志信息写入日志文件。"OFF"是最高级，禁止所有日志信息写入日志文件。

守护进程选项

- 守护进程路径

  配置选项格式:

  ```
  nbs.daemon.location = "/usr/bin/networkbench"
  ```

  数据类型: string

  默认值: "/usr/bin/networkbench"

  ## 说明:

  守护进程的路径

审计模式

配置选项格式:

```
nbs.audit_mode = false
```

数据类型: boolean

取值: true/false

默认值: false

说明:

本选项设定是否在日志文件中写入更详尽的信息，包括所有的向听云后台上传和下载的数据。

SQL数据选项

- 禁止发送SQL详情

  配置选项格式:

  ```
  nbs.action_tracer.log_sql = false
  ```

  数据类型: boolean

  取值: true/false

  默认值: false

  ### 说明:

  若本选项为true, 向服务器提交的数据中将只包含数据库汇总性能，不包含sql详情数据，sql详情记录在本地log中。

数据传输方式

是否启用http安全连接

```
nbs.ssl = true
```

数据类型: boolean
取值: true/false

是否使用安全连接(https)发送数据。若设定为true，则向服务器发送数据时期用https方式。否则，使用普通http方式。

代理服务器地址

```
nbs.proxy_host =
```

数据类型: string
默认值: 无

代理服务器的地址。若选项不为空，并且未启用安全连接，则本选项值为http代理服务器的ip地址。

代理服务器端口

```
nbs.proxy_port =
```

数据类型: 数字
默认值: 无

本选项指定代理服务器的端口。

代理服务器user

配置选项格式:

```
nbs.proxy_user =
```

数据类型: string

默认值: 无

同前，若代理服务器需要用户名密码，本选项指定代理服务器的登陆名。

代理服务器password

```
nbs.proxy_password =
```

数据类型: 数字

默认值: 无

同前，若代理服务器需要用户名密码，本选项指定代理服务器的登陆密码。

排队时间

定义：从负载均衡服务器到web应用服务器的时间

原理：负载均衡设备或容器收到请求后，增加请求的http头[X-QUEUE-START], 内容为收到请求的毫秒数，负载均衡将请求转发给WEB应用服务器。 WEB应用服务器在处理请求时，获取当前时间的毫秒数，两个时间差值即为请求排队时间。

例如web应用服务器配置了队列，大量请求在队列内排队后请求才被处理时，排队时间会变长

如果要配置排队时间，必须保证相关服务器的时钟同步。

配置方法

1. Apache 配置

```
RequestHeader set X-QUEUE-START "%t"
```

2. Nginx 配置（1.2.6版本以上）

A)使用 proxy_set_header

```
proxy_set_header X-QUEUE-START "s=$msec";
```

B)或使用 fastcgi_param

```
fastcgi_param HTTP_X_QUEUE_START "s=$msec";
```

C)或使用 Passenger

```
passenger_set_cgi_param HTTP_X_QUEUE_START "s=${msec}";
```

D)或使用 uWSGI

```
uwsgi_param HTTP_X_QUEUE_START "s=${msec}";
```

3. HaProxy 配置

```
http-request set-header X-Queue-Start t=%Ts%ms
```

4. F5 配置，使用以下脚本

```
when HTTP_REQUEST_SEND {
  set secs [clock seconds]
  set ms [clock clicks -milliseconds]
  set base [expr { $secs * 1000 }]
  set fract [expr { $ms - $base }]
  if { $fract >= 1000 } {
    set diff [expr { $fract / 1000 }]
    incr secs $diff
    incr fract [expr { -1000 * $diff }]
  }
  set micros [format "%d%03d000" $secs $fract]
  clientside {
    HTTP::header insert X-QUEUE-START "t=${micros}"
  }
```

```
}
```

排除安装故障

探针安装日志在/tmp/下，名称为nbinstall-xxxx.tar

1. 安装RPM包时，出现“error: Failed dependencies”提示

```
[nb@localhost tmp]$ rpm -Uvh tingyun-agent-php-latest.x86_64.rpm
error: Failed dependencies:
        ld-linux-x86-64.so.2()(64bit) is needed by tingyun-agent-php-
1.0.3-1.x86_64
        ld-linux-x86-64.so.2(GLIBC_2.3)(64bit) is needed by tingyun-agent-
php-1.0.3-1.x86_64
        libc.so.6()(64bit) is needed by tingyun-agent-php-1.0.3-1.x86_64
        libc.so.6(GLIBC_2.2.5)(64bit) is needed by tingyun-agent-php-
1.0.3-1.x86_64
        libc.so.6(GLIBC_2.3)(64bit) is needed by tingyun-agent-php-1.0.3-
1.x86_64
        libc.so.6(GLIBC_2.3.2)(64bit) is needed by tingyun-agent-php-
1.0.3-1.x86_64
        libc.so.6(GLIBC_2.4)(64bit) is needed by tingyun-agent-php-1.0.3-
1.x86_64
        libdl.so.2()(64bit) is needed by tingyun-agent-php-1.0.3-1.x86_64
        libdl.so.2(GLIBC_2.2.5)(64bit) is needed by tingyun-agent-php-
1.0.3-1.x86_64
        libgcc_s.so.1()(64bit) is needed by tingyun-agent-php-1.0.3-
1.x86_64
        libgcc_s.so.1(GCC_3.0)(64bit) is needed by tingyun-agent-php-
1.0.3-1.x86_64
        libgcc_s.so.1(GCC_3.3)(64bit) is needed by tingyun-agent-php-
1.0.3-1.x86_64
        libgcc_s.so.1(GCC_4.2.0)(64bit) is needed by tingyun-agent-php-
1.0.3-1.x86_64
        libm.so.6()(64bit) is needed by tingyun-agent-php-1.0.3-1.x86_64
        libpthread.so.0()(64bit) is needed by tingyun-agent-php-1.0.3-
1.x86_64
        libpthread.so.0(GLIBC_2.2.5)(64bit) is needed by tingyun-agent-
php-1.0.3-1.x86_64
```

出现此提示的原因是您的操作系统为32位，但您尝试安装64位的RPM包，请更换为32位的RPM包。

2. 安装PHP探针时，提示 permission denied

例如 error:unpacking of archive failed on file /usr/bin/networkbench: cpio:rename failed - permission denied
一般这种报错跟权限有关，大多是因为防火墙或一些安全类的软件，可以把相关安全软件关闭，再进行安装

3. 运行networkbench-install.sh时，出现“PHP install path not found, please enter the path.”提示

出现此提示的原因是在系统环境变量里面找不到php的安装路径，需要手工指定php的安装路径

录入php所在的文件夹路径，例如: php的位置是/opt/php5/bin/php，就录入/opt/php5/bin

4. 运行networkbench-install.sh时，出现“The Networkbench agent is not installed.”提示

PHP环境中缺少探针安装相关的必要信息，导致无法自动安装PHP扩展。
解决方法请参见：文档内“故障排查”的“手动安装PHP扩展”一节

5. 运行networkbench-install.sh时，PHP版本选择步骤中，“Below is a list of the directories in which we found a copy of PHP”列表中，没有用户期望的PHP路径

在系统环境变量里面找不到用户期望的PHP安装路径
解决方法：将此php的安装路径加入到PATH环境变量中，然后重新运行安装脚本

6、安装成功但报表内无数据
1）初次安装后需要一段时间才能看到数据，通常需要5分钟或更长时间，请多等一段时间
2）有的用户浏览器（比如猎豹）缓存有问题，即使有数据如果不强制刷新也可能看不到，请尝试清空缓存或强制刷新或更换浏览器
3）探针安装成功后，必须重启apache或php-fpm，否则探针不生效
4）确认PHP的扩展配置文件中授权序号正确
通常networkbench.ini位于/etc/php.d/networkbench.ini 或 /etc/php5/fpm/conf.d/networkbench.ini，
如果php是通过源码编译安装的，那么配置文件的位置依赖于编译时选项指定的位置。

```
nbs.license_key = "此处修改为您的授权序号"
```

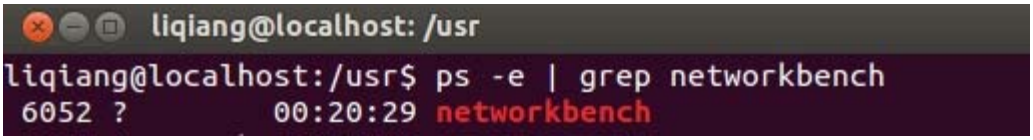
5）确认web服务器有没有用户访问
探针数据是基于http请求的性能，如果没有访问，就不会有性能数据。 如果没有用户访问，请使用浏览器访问一下相应的应用，再过5分钟后查看报表

检查守护进程

确认守护进程在运行:

```
ps aux | grep networkbench
```

正常情况下的信息和以下截图类似:



如果networkbench进程不存在,请重启web server

```
sudo service httpd restart
```

或

```
sudo service php-fpm restart
```

然后再次运行 ps -e | grep networkbench ，查看输出结果

如果networkbench进程仍不存在，请运行

```
ll /usr/bin/networkbench
```

查看输出结果，确认守护进程文件是否存在

如果 /usr/bin/networkbench 不存在，说明守护被误删了，请重新安装探针

检查php扩展模块

创建info.php,内容如下,并将info.php放在网站某个目录下

```
<?php phpinfo(); ?>
```

通过浏览器访问info.php，查找页面内是否存在关键字networkbench， 并且查看 nbs.app_name 内容，此值如果为空（no value）说明配置文件路径错误或内容错误， 查看 nbs.license_key 内容，此值应该和听云帐号内的授权码一致。
正常情况下的信息和以下截图类似

networkbench

networkbench	enabled
Version	Nov 19 2015
nbs.license_key	f373ea5dd553876
nbs.app_name	php5.6
nbs.prefix	/usr
nbs.runtime_root	/var
nbs.agent_log_level	debug
nbs.daemon_log_level	INFO
nbs.audit_mode	0

若没有networkbench模块信息，请以root运行networkbench-install.sh

```
#networkbench-install.sh
```

然后，请重启apache或您的fast-cgi程序以重新加载php扩展

执行上述操作后,刷新浏览器的info.php页面，如果仍未出现networkbench模块信息，
则很有可能您的php版本不符合安装条件，或者是经过裁剪修改的php版本.

查看日志

如果系统工作不正常或者没有数据，通常可以从日志里获取更多的信息来定位问题.

日志默认路径: /var/log/networkbench/

守护进程日志： /var/log/networkbench/daemon.log

php扩展日志：/var/log/networkbench/php-agent.log

运行

```
ll /var/log/networkbench
```

正常情况下会存在2个日志(daemon.log 和php-agent.log)

1）如果php-agent.log不存在说明PHP扩展有问题, 请检查php扩展模块

2）如果daemon.log不存在说明守护进程有问题, 请确认守护进程在运行

如果2个日志都存在说明安装没有问题，请运行

```
grep -E 'CRITICAL|ERROR|error' /var/log/networkbench/php-agent.log
```

如果php-agent输出有ERROR信息，说明PHP扩展运行时有错误

```
grep -E 'CRITICAL|ERROR|error' /var/log/networkbench/daemon.log
```

如果daemon输出有ERROR信息，说明可能和服务器通信有问题

关于日志的输出控制，请参考配置章节.

默认情况，日志级别为info, 审计模式是关闭的，这种情况得到的信息量比较小．通过修改日志级别为debug，开启审计模式，让守护进程和php扩展输出更多日志信息，通过日志得到的信息定位问题.

手动安装PHP扩展

当无法在标准位置查找到php或缺少php某些信息时，安装脚本无法自动安装php扩展，此时需要我们手动安装。

%prefix% 为探针安装路径，so文件和bin文件放在此路径下，默认探针安装路径为 /usr， bin包安装时可以使用prefix参数指定

1) 确认操作系统位数

确认操作系统是32位还是64位

```
uname -a
```

2) 确认PHP信息

在有权限生成PHP文件的情况下，最好使用 phpinfo() 函数来验证:

通过生成info.php文件，使用浏览器访问info.php查看php信息

```
echo "<?php phpinfo(); ?>" > /var/www/html/info.php
```

否则，使用php -i | grep 来验证php信息

信息名称	info.php 方式	php -i 方式
PHP API	PHP API 20090626	php -i | grep "PHP API"
是否启用了zts	Thread Safety disabled	php -i | grep "Thread Safety"
扩展so路径	extension_dir /usr/lib64/php/modules	php -i | grep "extension_dir"
ini配置文件路径	Configuration File (php.ini) Path /etc Loaded Configuration File /etc/php.ini Scan this dir for additional .ini files /etc/php.d	php -i | grep "Loaded Configuration File" php -i | grep "Scan this dir for additional"

3) 安装so扩展

进入到php扩展so路径(extension_dir指定的路径)， 做一个软链接 networkbench.so 到

%prefix%/lib/networkbench/agent/ 下的对应版本的so扩展

这里假设操作系统是64位，php版本是5.3，对应api版本为20090626，
php启用了zts（zend thread safe）模式, 扩展so路径为/usr/lib64/php/modules

```
ln -s /usr/lib/networkbench/agent/x64/networkbench-20090626-zts.so
/usr/lib64/php/modulesnetworkbench.so
```

PHP版本和探针so对应关系

PHP 版本	PHP API	启用zts	禁用zts
5.2	20060613	networkbench-20060613-zts.so	networkbench-20060613.so
5.3	20090626	networkbench-20090626-zts.so	networkbench-20090626.so
5.4	20100525	networkbench-20100525-zts.so	networkbench-20100525.so
5.5	20121212	networkbench-20121212-zts.so	networkbench-20121212.so
5.6	20131226	networkbench-20131226-zts.so	networkbench-20131226.so
7.0	20151012	networkbench-20151012-zts.so	networkbench-20151012.so

4) 增加配置文件，并修改授权码

a. 如果PHP信息中附加的配置文件路径不为空（即“Scan this dir for additional”不为"none"）， 将配置文件模版networkbench.ini.template复制到php扩展配置的文件夹，并改名为networkbench.ini

假设附加的php扩展配置路径为：/etc/php.d/

```
cp /usr/lib/networkbench/scripts/networkbench.ini.template /etc/php.d/networkbench.ini

vi /etc/php.d/networkbench.ini
```

b. 如果附加的配置路径为空（即“Scan this dir for additional” 值为 "none"），将配置文件模版networkbench.ini.template的内容添加到PHP主配置文件（"Loaded Configuration File"项指定的文件）后面。

假设PHP主配置文件路径为：/etc/php.ini

```
cp /etc/php.ini /etc/php.ini.bak

cat /usr/lib/networkbench/scripts/networkbench.ini.template >> /etc/php.ini

vi /etc/php.ini
```

将配置文件内 nbs.license_key = "REPLACE_WITH_REAL_KEY" 中 REPLACE_WITH_REAL_KEY 替换为您的授权码

如果您使用了bin包安装，还需要修改配置文件内nbs.prefix 为指定的安装路径，修改nbs.runtime_root为运行路径， 创建%runtime_root%/log/networkbench/ 和 %runtime_root%/run/networkbench/文件夹并赋予其足够的权限

```
假设 nbs.runtime_root=/home/mike

mkdir /home/mike/log/networkbench
chmod 777 /home/mike/log/networkbench

mkdir /home/mike/run/networkbench
chmod 777 /home/mike/run/networkbench
```

5）重启apache或php-fpm

版权申明

北京基调网络系统有限公司

www.networkbench.com

总部地址：北京市朝阳区京顺路5号曙光大厦C座207室

邮编：100028

联系电话：010-84440086