QQ截图20151201103408.png

**PinPoint APM系统搭建手册**

信息技术中心

2016年10月26日

**文档修订记录**

|  |  |  |  |
| --- | --- | --- | --- |
| 日 期 | 版本 | 描述信息 | 作者 |
| 2016-10-26 | 1.0 | 新建 | 陈俊彦 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# 前言

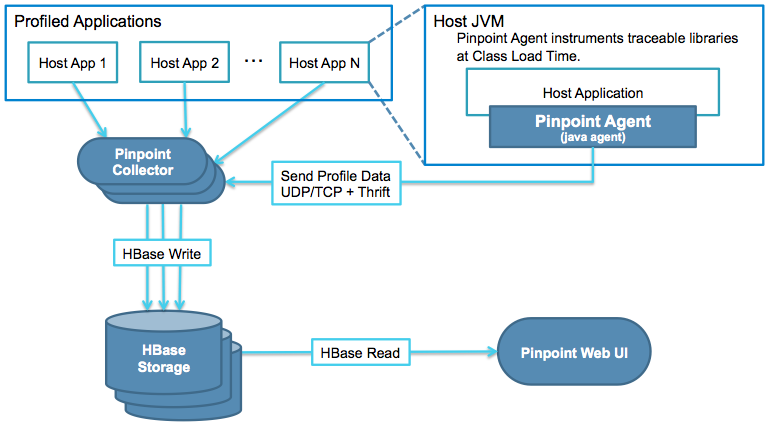
国内的APM行业这两年刚刚起步，但是在国外却比较成熟了，并且由于这两年人力成本的快速提高，国内外涌现了几家非常不错的APM企业，例如APPdynamic，Dynamic，NewRelic等。

下面给大家介绍一个开源的 APM (Application Performance Management/应用性能管理)工具 - Pinpoint。一个分布式事务跟踪系统的平台，思路基于google Dapper，用于基于java的大规模分布式系统，通过跟踪分布式应用之间的调用来提供解决方案，以帮助分析系统的总体结构和内部模块之间如何相互联系。Pinpoint的目标就是为n(n>=1)层架构开发新的跟踪平台，为n层架构的系统提供解决方案。

## 1.1 Pinpoint的特点如下

* 分布式事务跟踪，跟踪跨分布式应用的消息
* 自动检测应用拓扑，帮助你搞清楚应用的架构
* 水平扩展以便支持大规模服务器集群
* 提供代码级别的可见性以便轻松定位失败点和瓶颈
* 使用字节码增强技术，添加新功能而无需修改代码
* 安装探针不需要修改哪怕一行代码及trace server端部署简单，支持hdfs存储
* 具有简单的阀值触发报警功能
* 移植性比较强的，会比较讨人喜欢（相比cat）
* 插件化功能可扩展（https://github.com/naver/pinpoint/wiki/Pinpoint-Plugin-Developer-Guide）

## Pinpoint 架构



# 环境准备

为了省事可以直接就到这里(<https://get.k8scn.org/packages/others/pinpoint/>)下载所需安装包，统一放置在 /root/pp/ 目录下面。

apache-tomcat-8.0.36.tar.gz

hbase-1.0.3-bin.tar.gz

jdk-8u92-linux-x64.rpm

pinpoint-agent-1.5.2.tar.gz

pinpoint-collector-1.5.2.war

pinpoint-web-1.5.2.war

pp-collector

pp-web

zookeeper-3.4.6-10.el6.x86\_64.rpm

## 2.1 部署环境约定

10.163.34.22 CentOS7.2(jdk1.8.0) Pinpoint-collector, Pinpoint-web, zookeeper Hbase

10.163.84.40 CentOS6(jdk1.8.0) Pinpoint-agent(demo应用)

## 2.2 系统环境说明

$ /etc/init.d/iptables stop

$ setenforce 0

这里我们为了不影响模块之间的相互通信，暂时先把几台设备的iptables都关闭了。

# 开始部署

我们采用rpm方式安装jdk，这里就略过安装过程了，主要说一下Tomcat 和Bbase的安装和配置

## 3.1 部署和初始化 Hbase

3.1.1添加JAVA环境变量

$ cat /etc/profile

# export JAVA\_HOME=/usr/java/

export JRE\_HOME=$JAVA\_HOME/jdk1.8.0\_92/jre

export CLASSPATH=.:$JAVA\_HOME/lib:$JRE\_HOME/lib:$CLASSPATH

export PATH=$PATH:$JAVA\_HOME/bin:$JRE\_HOME/bin

3.1.2初始化 Hbase

$ tar xf /root/pp/hbase-1.0.3-bin.tar.gz /usr/local/

$ cd /usr/local/hbase-1.0.3/conf

$ vim hbase-env.sh

export JAVA\_HOME=/usr/java/jdk1.8.0\_92/ # 指到自己对应的JAVA\_HOME，默认使用java6

3.1.3 添加以下配置到hbase-site.xml

$ vim hbase-site.xml

<configuration>

<property>

<name>hbase.rootdir</name>

<value>file:///data/hbase</value> # 这里我们指定Hbase本地来存储数据，生产环境将数据建议存入HDFS中。

</property>

</configuration>

3.1.4 启动 Hbase

$ ../bin/start-hbase.sh // 启动 Hbase

注意: 这里说明一下，我们是最简方式启动的Hbase，没有单独部署Hbase集群管理的zookeeper，而是直接使用Hbase自带的zk模块。另外，Hbase墙裂建议将数据存入HDFS里面，目前我们是存在Hbase本地的。

## 3.2 验证Hbase是否启动成功

$ jps

8114 Jps

7820 HMaster

## 3.3下载并执行pp的Hbase初始化表语句

$ wget -c https://raw.githubusercontent.com/naver/pinpoint/master/hbase/scripts/hbase-create.hbase -P /root/pp/

$ cd /usr/local/hbase-1.0.3

$ ./bin/hbase shell /root/pp/hbase-create.hbase # 这里需要大概半分钟时间完成pp的表初始化

$ ./bin/hbase shell

2016-07-14 20:57:10,606 WARN [main] util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

HBase Shell; enter 'help<RETURN>' for list of supported commands.

Type "exit<RETURN>" to leave the HBase Shell

Version 1.0.3, rf1e1312f9790a7c40f6a4b5a1bab2ea1dd559890, Tue Jan 19 19:26:53 PST 2016

hbase(main):001:0> status 'detailed' # 我这里已经初始化pp相关表，所以数据展示比较多

version 1.0.3

0 regionsInTransition

master coprocessors: []

1 live servers

c612:46781 1468494831781

requestsPerSecond=0.0, numberOfOnlineRegions=194, usedHeapMB=18, maxHeapMB=241, numberOfStores=322, numberOfStorefiles=3, storefileUncompressedSizeMB=0, storefileSizeMB=0, memstoreSizeMB=0, storefileIndexSizeMB=0, readRequestsCount=9272, writeRequestsCount=385, rootIndexSizeKB=2, totalStaticIndexSizeKB=1, totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, coprocessors=[MultiRowMutationEndpoint]

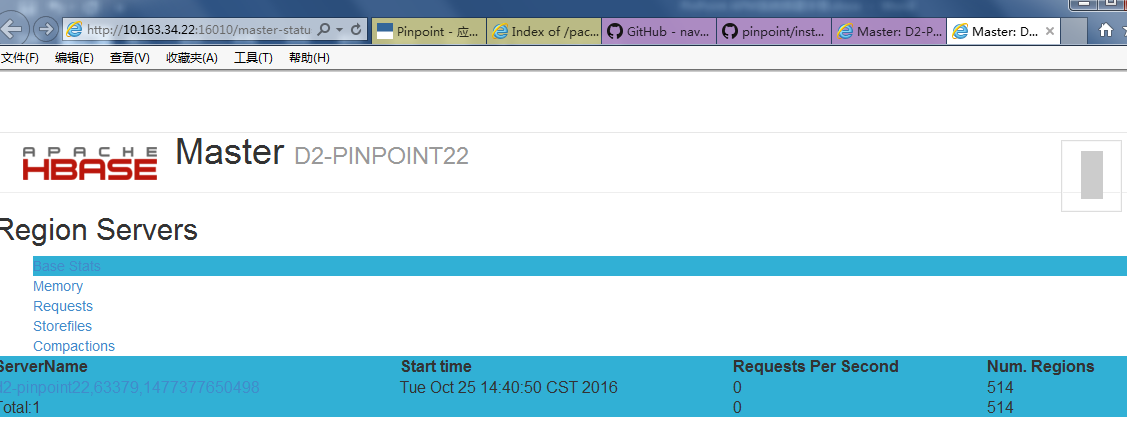
"AgentEvent,,1468495003477.b13eb3203210fb9d94feb1a445e0ad1b."

numberOfStores=1, numberOfStorefiles=0, storefileUncompressedSizeMB=0, storefileSizeMB=0, memstoreSizeMB=0, storefileIndexSizeMB=0, readRequestsCount=0, writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0, totalStaticBloomSizeKB=0, totalCompactingKVs=0, currentCompactedKVs=0, compactionProgressPct=NaN, completeSequenceId=-1, dataLocality=0.0

"AgentInfo,,1468494998919.fa9d36f7b52b95ee5216a1ec757ad690."

...

我们同时可以登录Hbase Web管理控制台来验证相关数据<http://10.163.34.22:16010/master-status>



## 3.4 部署和配置pp-collector和pp-web

### 3.4.1部署zookeeper

从pinpoint1.5.0开始，pp-web到pp-agent的request请求直接通过pp-collector来统筹管理，包括一些real-time级别的信息监测，而zookeeper在其中间协调三者之间的通信沟通。

如果zookeeper部署在独立的一台设备，这里我就需要修改一下pp-colletor & pp-web的配置文件pinpoint-collector.properties，pinpoint-web.properties，不然会导致俩个模块启动失败。详细描述参见[官方文档](https://github.com/naver/pinpoint/blob/master/doc/installation.md" \l "routing-web-requests-to-agents)

$ rpm -ivh /root/pp/zookeeper-3.4.6-10.el6.x86\_64.rpm

$ /etc/init.d/zookeeper start # 保证zookeeper的2181端口起来就可以了

### 3.4.2 pp-collector的配置

$ mkdir -p /data/service/

$ tar xf /root/pp/apache-tomcat-8.0.35.tar.gz -C /data/service/

$ cd /data/service/

$ mv apache-tomcat-8.0.35/ pinpoint-collector

$ vim /data/service/pinpoint-collector/conf/server.xml # 因为我们的pp-collector和pp-web部署在同台设备，所以请确认tomcat启动端口不会冲突

<Server port="8005" shutdown="SHUTDOWN">

<Connector port="8085" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443" />

<!-- <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" /> --> # 注释该行

$ rm -rf /data/service/pinpoint-collector/webapps/\*

$ unzip pinpoint-collector-1.5.2.war -d /data/service/pinpoint-collector/webapps/ROOT/

$ cd /data/service/pinpoint-collector/webapps/ROOT/WEB-INF/classes

$ ls -al

-rw-rw-r-- 1 root root 15835 Apr 7 14:43 applicationContext-collector.xml

-rw-rw-r-- 1 root root 7485 Apr 7 14:43 applicationContext-hbase.xml

-rw-rw-r-- 1 root root 757 Apr 7 14:43 hbase.properties

-rw-rw-r-- 1 root root 1845 Apr 7 14:43 log4j.xml

-rw-rw-r-- 1 root root 1360 Apr 7 14:43 pinpoint-collector.properties # 主要针对pp-agent的配置，同时支持tcp和udp协议数据传输

-rw-rw-r-- 1 root root 3292 Apr 7 14:43 servlet-context.xml

...

这里大概说一下: applicationContext-collector.xml, applicationContext-hbase.xml 这俩个配置文件时collector与agent和Hbase之间通信时需要设定的一些参数，在后续调优的时候需要用到，hbase.properties 主要是设定后端存储的连接配置，log4j.xml那就是log相关了。

$ vim hbase.properties

hbase.client.host=10.163.34.22 # 这里我们只修改这一项配置就好，能够让collector将汇报上来的数据存储到Hbase

hbase.client.port=2181

这里tomcat我们就用默认的启动端口了，实际的生产环境可能需要根据实际情况修改tomcat侦听端口。OK，到此，pp-collector我们就基本上配置完成了。接下来我们开始配置pp-web，它与collector很类似。

### 3.4.3 pp-web的配置

$ cd /data/service/

$ tar xf /root/pp/apache-tomcat-8.0.35.tar.gz -C /data/service/

$ mv apache-tomcat-8.0.35 pinpoint-web

$ cd pinpoint-web/webapps/;rm -rf \*

$ mkdir ROOT

$ cd ROOT/

$ unzip /root/pinpoint-web-1.5.2.war

$ vim /data/service/pinpoint-web/conf/server.xml # 因为我们的pp-collector和pp-web部署在同台设备，所以请确认tomcat启动端口不会冲突

<Server port="8006" shutdown="SHUTDOWN">

<Connector port="8086" protocol="HTTP/1.1" connectionTimeout="20000" redirectPort="8443" />

<!-- <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" /> --> # 注释该行

$ cd /data/service/pinpoint-web/webapps/ROOT/WEB-INF/classes

$ vim hbase.properties

hbase.client.host=10.163.34.22 # 设置Hbase数据源

hbase.client.port=2181

...

$ ls -al

total 88

-rw-rw-r-- 1 root root 2164 Apr 7 14:43 applicationContext-cache.xml

-rw-rw-r-- 1 root root 3649 Apr 7 14:43 applicationContext-dao-config.xml

-rw-rw-r-- 1 root root 1490 Apr 7 14:43 applicationContext-datasource.xml

-rw-rw-r-- 1 root root 6680 Apr 7 14:43 applicationContext-hbase.xml

-rw-rw-r-- 1 root root 1610 Apr 7 14:43 applicationContext-websocket.xml

-rw-rw-r-- 1 root root 6576 Apr 7 14:43 applicationContext-web.xml

-rw-rw-r-- 1 root root 106 Apr 7 14:43 batch.properties

-rw-rw-r-- 1 root root 682 Apr 7 14:43 ehcache.xml

-rw-rw-r-- 1 root root 1005 Jul 14 21:45 hbase.properties

-rw-rw-r-- 1 root root 153 Apr 7 14:43 jdbc.properties

-rw-rw-r-- 1 root root 3338 Apr 7 14:43 log4j.xml

-rw-rw-r-- 1 root root 1420 Apr 7 14:43 mybatis-config.xml

-rw-rw-r-- 1 root root 630 Apr 7 14:43 pinpoint-web.properties

-rw-rw-r-- 1 root root 141 Apr 7 14:43 project.properties

-rw-rw-r-- 1 root root 3872 Apr 7 14:43 servlet-context.xml

drwxrwxr-x 2 root root 4096 Jul 14 21:46 sql

...

这里说明一下:

* hbase.properties 配置我们pp-web从哪个数据源获取采集数据，这里我们只指定Hbase的zookeeper地址。
* jdbc.properties pp-web连接自身Mysql数据库的连接认证配置。
* sql目录 pp-web本身有些数据需要存放在MySQL数据库中，这里需要初始化一下表结构。
* pinpoint-web.properties 这里pp-web集群的配置文件，如果你需要pp-web集群的话。
* applicationContext-\*.xml 这些文件在后续的调优工作中会用到。
* log4j.xml 日志相关配置。

好了，pp-web终于算是配置完成了。

### 3.4.4启动pp-colletor和pp-web

$ cp /root/pp/pp-\* /etc/init.d/

$ chmod +x /etc/init.d/pp-\*

$ mv /etc/init.d/pp-web.init /etc/init.d/pp-web

$ mv /etc/init.d/ pp-collector.init /etc/init.d/ pp-collector

$ /etc/init.d/pp-web restart

$ /etc/init.d/pp-collector restart

# 部署pp-agent

以下是pp目前支持埋点的java模块：

JDK 6+

Tomcat 6/7/8, Jetty 8/9

Spring, Spring Boot

Apache HTTP Client 3.x/4.x, JDK HttpConnector, GoogleHttpClient, OkHttpClient, NingAsyncHttpClient

Thrift Client, Thrift Service, DUBBO PROVIDER, DUBBO CONSUMER

MySQL, Oracle, MSSQL, CUBRID, DBCP, POSTGRESQL, MARIA

Arcus, Memcached, Redis, CASSANDRA

iBATIS, MyBatis

gson, Jackson, Json Lib

log4j, Logback

## 4.1 部署pp-agent (探针)

### 4.1.1安装 pp-agent

客户机 10.163.84.40

JDK安装位置为 /home/pipre/jdk1.6.0\_45

tomcat容器安装位置 /home/pipre/apache-tomcat-6.0.45

这里我们就将pp-agent解压至 /home/pipre/apache-tomcat-6.0.45/pinpoint-agent-1.5.2下。

$ tar xf /root/pp/pinpoint-agent-1.5.2.tar.gz -C /home/pipre/apache-tomcat-6.0.45/

$ cd /home/pipre/apache-tomcat-6.0.45/pinpoint-agent-1.5.2/

$ vim pinpoint.config

profiler.collector.ip=10.163.34.22

这里说明一下： 我们只要指定pp-colletor的地址就可以了，至于端口如果之前pp-colletor没有变更这里也不需要修改。如果你的生产环境特殊，pinpoint支持还是很丰富的，同时支持tcp和udp协议传输。

$ pwd

/data/projects/service/pinpoint-agent-1.5.2

$ ls -al

total 80

drwxrwxr-x 6 ceph ceph 4096 Jul 14 22:11 .

drwxr-xr-x 3 root root 4096 Jul 14 22:08 ..

drwxrwxr-x 2 ceph ceph 4096 Apr 7 13:43 boot

drwxrwxr-x 2 ceph ceph 4096 Apr 7 13:43 lib

-rw-rw-r-- 1 ceph ceph 27603 Apr 7 13:42 pinpoint-bootstrap-1.5.2.jar

-rw-rw-r-- 1 ceph ceph 14278 Apr 7 13:30 pinpoint.config

-rw-rw-r-- 1 ceph ceph 11988 Apr 7 13:30 pinpoint-real-env-lowoverhead-sample.config

drwxrwxr-x 2 ceph ceph 4096 Apr 7 13:43 plugin

drwxrwxr-x 2 ceph ceph 4096 Apr 7 13:43 script

### 4.1.2 在应用里部署探针

添加以下CATALINA\_OPTS到catalina.sh

$ vim /home/pipre/apache-tomcat-6.0.45/bin/catalina.sh

CATALINA\_OPTS="-Djava.library.path=/usr/local/apr/lib"

CATALINA\_OPTS="$CATALINA\_OPTS -javaagent:/home/pipre/apache-tomcat-6.0.45/pinpoint-agent-1.5.2/pinpoint-bootstrap-1.5.2.jar"

CATALINA\_OPTS="$CATALINA\_OPTS -Dpinpoint.agentId=D2-PIPRE40ID"

CATALINA\_OPTS="$CATALINA\_OPTS -Dpinpoint.applicationName=D2-PIPRE40"

这里说明一下：

* 探针配置项写到 CATALINA\_OPTS 变量下一行即可。
* AGENT\_ID 命名必须在生产环境里唯一，否则后面将会混乱。
* APPLICATION\_NAME 配置为一个服务组的ID号，即服务组下的实例ID。
* 此外参考[官方agent文档说明](https://github.com/naver/pinpoint/blob/master/doc/installation.md#installation-3)

## 重启被采集端的应用，使探针生效

重启tomcat

/home/pipre/apache-tomcat-6.0.45/bin/shutdown.sh

/home/pipre/apache-tomcat-6.0.45/bin/startup.sh

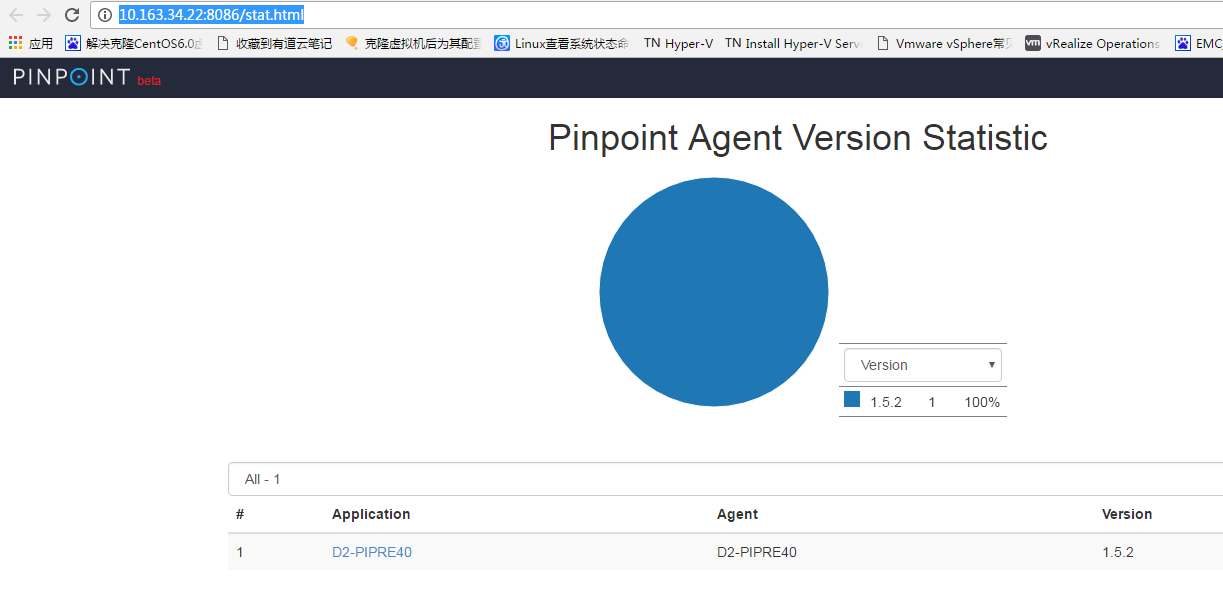
# 验证探针部署

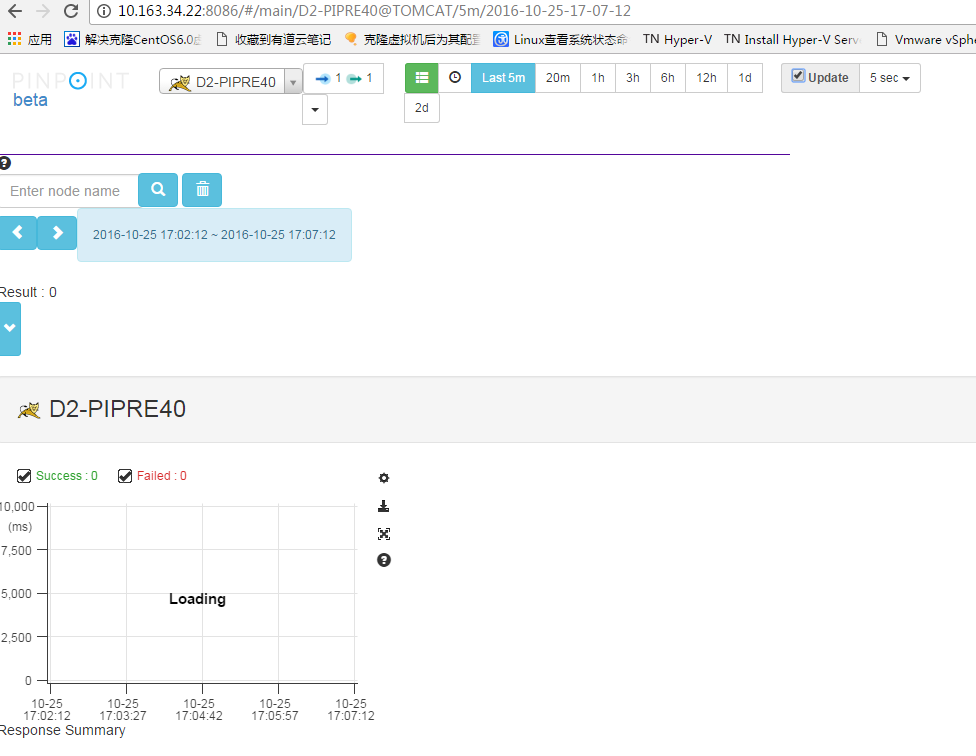
打开浏览器，访问如下pp-web的页面验证我们的探针部署是否成功。其中，第二个链接是pp的APM主页面可以看到探针汇报上来的信息了。

http://10.163.34.22:8086/stat.html # pp-agent 信息统计页面，即探针部署情况的统计信息

http://10.163.34.22:8086/#/main # pp APM信息主页面

页面访问效果如下





# 参考文档

https://sconts.com/11https://github.com/naver/pinpoint/blob/master/doc/installation.md  
http://hbase.apache.org/book.html#standalone\_dist Apache HBase ™ Reference Guide