**核心监控使用说明**

# 日报

## 日报功能说明

日报是为了将线上全部机器的运行情况进行统计，并将重要的信息展示出来。统计的范围都为应用的运行日志为主。  
目前主要包括的日志为:

**gc.log,** jvm 打出的日志，在jvm启动参数上添加  
-verbose:gc -Xloggc:/home/admin/logs/gc.log -XX:+PrintGCDetails -XX:+PrintGCDateStamps  
这样的参数就可以打出gc.log  
  
mbean.log, 这个主要包括数据库连接 线程池等。这个需要添加一个jar包依赖，不需要修改代码。  
<http://svn.taobao-toolkit.com/repos/binary-release/taobao/common/smonitor/> 这个是此jar包的svn地址  
  
monitor.log: 这个日志主要是一些接口调用的情况 如 tair hsf 等信息。  
这个monitor.log 也需要依赖jar   
<http://svn.taobao-toolkit.com/repos/binary-release/taobao/common/monitor/> 这个是svn   
地址，这个monitor.jar 是哈勃那边维护的，最好使用1.0.4以上的版本因为这个版本以上是支持日滚的  
  
hsf 的Client **版本是1.4.8的以上**  
  
一些应用自身特殊的log文件：  
这个为应用自己输出的信息，

除了上面几项外，还统计了 一些机器的运行状态，如CPU Load Pv rest 等信息 ，这个信息来自哈勃监控中心

## 日报接入方式

首先需要确认自己应用需要的汇总统计信息是否在上面内容中，如果没有将需要自己将应用的信息输出到日志中。然后告诉我应用的名称。我将帮你把收集应用加入配置中

# 实时监控

## 实时监控功能说明

实时监控通过实时采集系统相关的日志和系统信息 通过页面展示和信息告警反馈给用户。实时监控的信息采集通过ssh2 协议直接连接到目标机器上，利用scp 指令将目标机器上的日志信息拉到采集机器上分析入库，因此对于接入应用无需做任何代码嵌入工作。  
  
目前实时监控涉及的信息有:

CPU ,load :这个两项内容通过 TOP指令获取，执行的时间间隔为10s。  
gc.log: jvm 打出的日志，在jvm启动参数上添加  
-verbose:gc -Xloggc:/home/admin/logs/gc.log -XX:+PrintGCDetails -XX:+PrintGCDateStamps  
这样的参数就可以打出gc.log。它的采集是每分钟增量获取一次。统计了gc Fgc 的次数和消耗的时间  
  
mbean.log: 这个主要包括数据库连接 线程池等。这个需要添加一个jar包依赖，不需要修改代码。  
<http://svn.taobao-toolkit.com/repos/binary-release/taobao/common/smonitor/> 这个是此jar包的svn地址。 Mbean上的信息比较多 包含了数据库连接池，线程，内存，

### mBean配置说明

1.二方库：

taobao\smonitor\smonitor-1.0.jar

antx依赖：

<include uri="taobao/common/smonitor"  version="1.0"  />

mvn依赖：

<dependency>

       <groupId>com.taobao.common.smonitor</groupId>

        <artifactId>smonitor</artifactId>

        <version>1.0</version>

 </dependency>

2.配置，修改web.xml的配置，增加smonitor监控

   <servlet>

        <servlet-name>smonitor</servlet-name>

        <servlet-class>com.taobao.common.smonitor.SmonitorAdaptorServlet</servlet-class>

       <!-- 参数可选

         <init-param>

            <param-name>type</param-name>

            <param-value>1,2,3</param-value>

        </init-param>

        <init-param>

            <param-name>dataSource</param-name>

            <param-value>DB1DataSource,DB2DataSource</param-value>

        </init-param>

         <init-param>

            <param-name>threadNameKeys</param-name>

            <param-value>http,ajp</param-value>

        </init-param>

         <init-param>

            <param-name>intervalTime</param-name>

            <param-value>30000</param-value>

        </init-param>

        -->

        <load-on-startup>3</load-on-startup>

    </servlet>

 参数说明：

type:指定监控的种类, 1. 连接池 2. 线程池 3. 内存 ，多值之间用","分隔，默认值为 "1,2"

datasource: 监控的数据源，如：DB1DataSource,DB2DataSource，默认为所有数据源

threadNameKeys:线程监控的过滤关键字，如:http,ajp,hsf，线程名称里出现这3个key则会被监控，默认值"http,ajp,hsf"

intervalTime:监控的时间间隔，默认值为20s

3.日志目录：

监控日志默认写入服务器的$user.home/logs/mbean.log

4.日志格式：

连接池监控

数据源名称[MaxConnectionsInUseCount,InUseConnectionCount,AvailableConnectionCount,ConnectionCount,MaxSize,MinSize]

07-14:25:33 DBCDataSource[0,0,20,0,20,1]DB1DataSource\_cm2[0,0,10,0,10,2]JmsXA[0,0,20,0,20,0]CRMDataSource[0,0,15,0,15,15]DefaultDS[1,0,20,5,20,5]DBHDataSource[0,0,5,0,5,2]DBArkDataSource\_cm2[0,0,10,0,10,2]FORUMDatasource[0,0,5,0,5,2]DBCDataSource\_cm3[0,0,10,0,10,2]ETDataSource[0,0,15,0,15,15]DBQADataSource[0,0,10,0,10,2]DB1DataSource\_cm3[0,0,10,0,10,2]DB2DataSource[0,0,5,0,5,2]DB2DataSource\_cm3[0,0,10,0,10,2]DBArkDataSource[0,0,10,0,10,2]DBCDataSource\_cm2[0,0,10,0,10,2]CTUDatasource[0,0,20,0,20,1]DBFBDataSource[0,0,10,0,10,2]DBArkDataSource\_cm3[0,0,10,0,10,2]DB1DataSource[0,0,5,0,5,2]DB2DataSource\_cm2[0,0,10,0,10,2]

线程监控

Thread-dump:线程名[NEW,BLOCKED,RUNNABLE,WAITING,TERMINATED,TIMED\_WAITING]

线程池名-thread-pool[maxThreads,currentThreadCount]

07-14:25:33 Thread-dump:ajp[0,0,1,0,0,0]http[0,0,1,0,0,0]ajp-thread-pool[40,0]

内存监控

07-14:43:41 [PS Scavenge: Count=32 GCTime=2.2030sec][PS MarkSweep: Count=7 GCTime=4.6520sec][Code Cache: Used=2579K Committed=2624K][PS Eden Space: Used=2818K Committed=38912K][PS Survivor Space: Used=23656K Committed=38784K][PS Old Gen: Used=288160K Committed=378688K][PS Perm Gen: Used=40233K Committed=65536K][HeapMemoryUsage: Used=314635K Committed=456384K][NonHeapMemoryUsage: Used=42812K Committed=68160K]

monitor.log：这个日志是有各个客户端自己的埋点信息，如hsf Client ，tiar clinet等的自身打出的，但是应用必须依赖这个monitor.jar吧，svn地址是  
<http://svn.taobao-toolkit.com/repos/binary-release/taobao/common/monitor/>

这个最好是使用1.0.4以上的版本，因为这个版本以下是不会日滚的。  
  
应用的info 日志: 这个只要是收集应用的exception信息，这个需要应用将日志打到一个固定的目录下，且是统一的。  
  
应用自身的一个特殊信息: 如一些应用自身统计的 失败率等之类的信息，打到一个特定的文件，这个需要实现AnalyseFile类来完成对文件的解析.

## 实时监控的接入说明

目前实时监控是抽取目标应用的几台机器做为监控，现在还不能对应用全部机器做采样。未来规划将会采集全网机器，但是保存历史数据只会几台。

实时监控的接入也比较简单，告诉我监控的应用就可以了，我会看下应用目前拥有的日志情况，然后配置对应的配置项。  
  
监控数据采用的是key value 的结果存储在库表中，因此日志输出的信息业必须是key value的格式

# 告警设置

## 告警说明

Key的管理界面为: <http://cm.taobao.net:9999/monitorstat/alarm/manage_key.jsp>

目前 key主要分为下面几种名称格式  
PV\_ 开头的表示从apache 日志中获取  
PV\_PAGESIZE\_AVERAGEUSERTIMES 这个是页面平均大小（B）

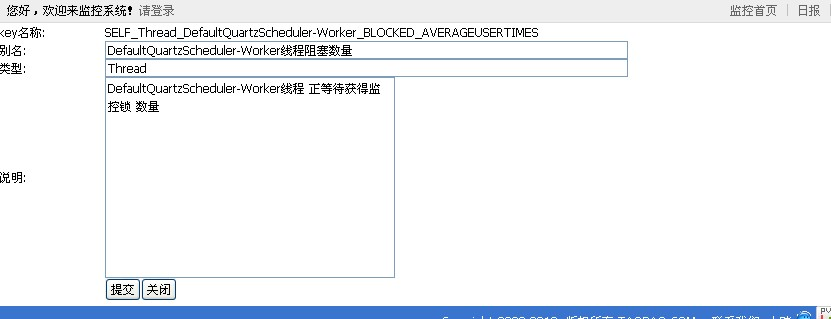
PV\_REST\_AVERAGEUSERTIMES 这个是平均响应时间 微秒  
PV\_VISIT\_COUNTTIMES 这个是平均每分钟的流量  
  
  
OUT\_开头的表示这个应用是对外调用,目前主要是下面几种类型  
OUT\_HSF-Consumer   
OUT\_forest\_client   
OUT\_PageCache  
OUT\_SearchEngine

在这个key 会已 \_ 符号作为分割，在最后的会存在两个标记  
\_AVERAGEUSERTIMES 这个配置平均执行时间  
\_COUNTTIMES 这个表示在每分钟的调用次数  
  
  
OTHER 开头的表示这个应用是自身一些特殊的埋点信息  
\_COUNTTIMES 表示这个值在同一个时间点内需要累加

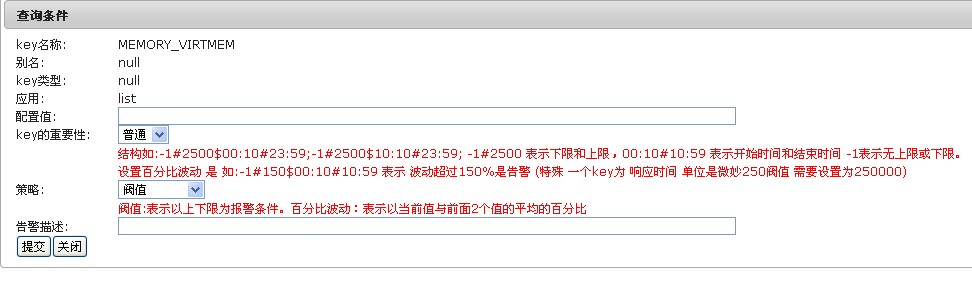
\_AVERAGEUSERTIMES 表示这个值在同一个时间点内需要平均  
  
  
  
  
  
exception开头的表示应用异常信息  
无后缀信息

## 如何添加告警

在<http://cm.taobao.net:9999/monitorstat/alarm/manage_key.jsp>页面中，选择自己的应用名称后，按查询可以得到下面结果：  


在图中有3列数据 ，第一列为key的名称（唯一），第二列为key的别名（可以点击修改），第三列为 操作加入告警  
  
点击一列或第二列名称将进入  
  
key的描述修改，其中

别名就是显示给大家看的，类型只是对同一类型的key进行归类，说明即对这个key的用户做一个简单的描述，方便别人理解  
  
  
“加入告警“将进入key的告警设置页面



在页面中需要填写的  
配置值: 这个就是告警值的设置。设置的格式为 value1#value2$time1#time2; 这个格式可以写多个 value1#value2$time1#time2; value1#value2$time1#time2;以分号作为分割  
其中value1 作为下限 -1表示无下限。Value2表示上线 -1表示无上限。Time1 和time2 表示在这个时间段内进行匹配,时间的格式为HH:mm。  
例1： -1#1000$12:00#23::00 表示这个值 在12点到23点之间如果大于1000将需要告警如果小于1000者不需要

例2：10#-1$12:00#23::00 表示这个值 在12点到23点之间如果小于10 需要告警 ，其它不需要  
  
Key的重要性：有3个选项 普通、严重、紧急.这个主要是在告警的时候会根据连续告警的情况升级，连续两次普通会升级为严重，两次严重升级为紧急。目前主要是体现在旺旺告警的标题上，未来会看情况对重要性进行判断来作为发送告警信息的判断  
  
策略：目前策略有两种一种是阀值，一种是波动。阀值:表示以上下限为报警条件。百分比波动：表示以当前值与前面2个值的平均的百分比

告警描述:即旺旺告警或短信告警 发送出去的信息，可以简要的填写。

# 告警接收者的添加

## 告警接收说明

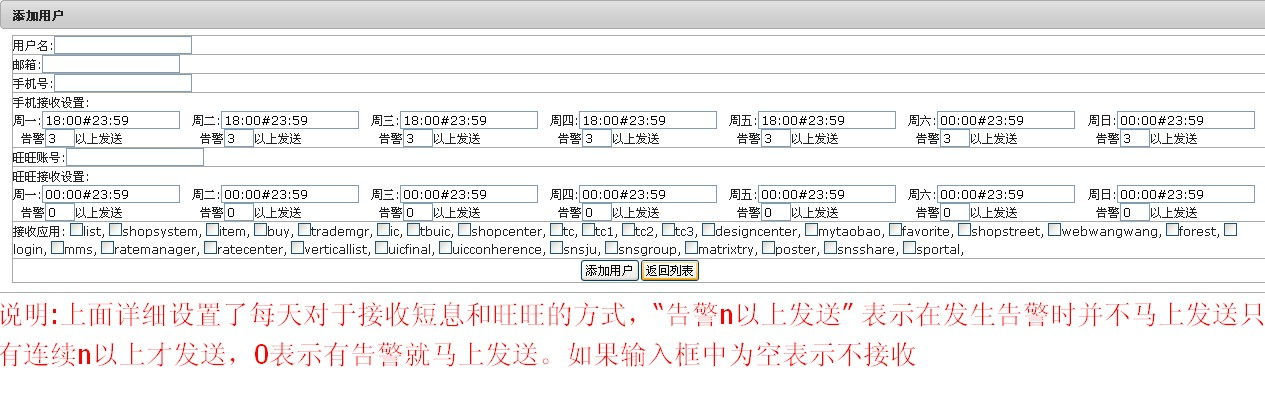
告警接收是已应用为单位进行区分，目前的细度也只是做到应用级别。也就是你选择了一个应用。那么你将接收到这个应用的全部告警。未来可能会再进一步细分到key的级别

## 接收者设置

<http://cm.taobao.net:9999/monitorstat/alarm/manage_user.jsp>

这个是总的管理界面，可以在这个上面进行添加和修改删除操作，如前还没有做权限控制，未来会加上控制

<http://cm.taobao.net:9999/monitorstat/alarm/add_alarm_user.jsp>

这个是添加用户界面  


用户名:就花名不需要高特殊  
邮箱: 发送报表使用

手机号:接收告警短信

旺旺号：接收旺旺告警信息

应用选择：需要接收那些应用的告警

这种重点说明下接收设置：  
旺旺和手机接收分为一个的接收时间和告警次数  
时间的格式为HH:mm#HH:mm 即表示在这个时间段内，告警次数表示只有连续发生几次以上才发送到旺旺或手机

连续告警的含义为：在第一次数据收集的时候发现key A 超过告警设置 ，那么这个算一个告警。在接下来的5分钟内 A 都没有发送告警。如果在第6分钟key A 有发送一次告警着这个不算连续。如果key A 在5分钟内发送了一次 那么Key A 算为连续告警2次

如：设置了周一 18:00#23:00 连续2次以上  
表示周一接收的时间为 晚上18点到23点且 告警连续发生2次，在第三次的时候将发送给手机或旺旺。其它时间段内者不会，或者没有发生连续告警也不会发送。

# 告警信息

## 历史告警查询

<http://cm.taobao.net:9999/monitorstat/alarm/alarm_record_detail.jsp>

这个链接可以查询到以往的告警信息

## 告警汇总查询

<http://cm.taobao.net:9999/monitorstat/alarm/alarm_record.jsp>

## 异常信息告警内容查询

<http://cm.taobao.net:9999/monitorstat/alarm/alarm_record_exception.jsp>

这这里就可以查询到应用异常告警的信息内容，这样就可以不用直接到后台日志查看

## 应用停止应用告警控制

<http://cm.taobao.net:9999/monitorstat/alarm/alarm_filter.jsp>

这里可以对应用暂时停止告警做一些简单的设置

# Jprof模块

## Jporf介绍

Jprof采用开源的asm （http://asm.ow2.org/）来实现时间采集程序嵌入到目标代码中，这个是无需对目标代码做任何修改。

工作的原理是在jvm load class文件之前，获取class文件流，并将这个文件流 转给asm 中的ClassReader 经过再加工再返回给jvm 。

## 如何使用jprof

首先现在jar包，svn地址是：  
<http://svn.taobao-develop.com/repos/coremonitor/trunk/coremonitor/monitorstat-jprof/release>

目前这个下面有连个版本。1.0.1版本支持细化到vm 模版的时间采集.  
jprof.properties 是作为jprof.jar的配置文件

### 环境变量设置

一、如果需要使用外置的配置（jprof.properties）需要使用环境变量环境变量

JPROF\_CONFIG\_PATH 这个可以用来设置 配置文件路径，如果不设置将采用默认配置文件  
  
二、如果需要在执行的时候 暂时中断采集 可以使用环境变量

JPROF\_PAUSE\_RUN

这个动态停止 true 暂停 false 执行

### 配置项说明

delayTime = 30 启动程序后 延时多少时间才开始 单位 s

debugJprof = false 测试模式 如果是true 下面4个时间配置就无效，dump 与jvm 停止做挂钩，只有在停止jvm才dump 文件

startJprofTime=00:00:00 开始做jprof 的时间段

endJprofTime = 23:12:12 结束做jprof 的时间段

eachJprofUseTime = 60 采样数据的时间 单位秒

eachJprofIntervalTime = 300 采样数据的间隔时间 单位秒

needNanoTime = false 是否使用纳秒 true 是 false 将使用毫秒

needPrintAllStack = false 是否将执行时间为0的dump下来

#out log file path

logFilePath = /tmp/jprof.txt duap 的输出路径

# separator ';'

includePackageStartsWith = com.taobao; 采集类名称已什么为开头

ignoreGetSetMethod = true 是否忽略get set 方法

excludePackageStartsWith = sun;javax;java; 过滤掉那些类

### 系统启动设置

在 jvm 的JAVA\_OPTS项中添加

-javaagent:\*\*\*\*\*\*/jprof-1.0.0.jar -javaagent:后面为jar包的绝对路径  
  
启动后就会在logFilePath 设置的路径下出现dump 文件。文件是定时缓存一段时间刷出的，所以不会马上就有

# 系统自动采集和报表发送

页面<http://cm.taobao.net:9999/monitorstat/jprof/manage_jprof_host.jsp>

这里可以查询自动采集的情况。  
如果需要使用这个功能必须目标机器已经在运行jprof  
  
后台再添加一条记录就可以

# 线上自动压测模块

## 压测介绍

自动压测 会采用两种方式，一种是http\_load 一种是apache jk 分流

http load 将会按照5用户 10用户 15用户20用户25用户30用户 递增压测应用 并得到压测结果， 结果包含GC CPU qps Rest 信息  
  
apache jk 分流 是修改jk 的配置文件将流量导向到目标机器，并在目标机器采集GC CPU qps Rest 信息。这里有一个需要注意的。这个需要分流机器和目标机器的admin 权限，用来重启apache

http\_load 模式是适用于 无需登录的和无需进行写操作的 应用  
  
apache jk 分流是适用于 需要登录或可能进行写操作的应用

## 如何接入

http\_load 模式比较简单，只需要应用名称就可以了

apache jk 分流模式就比较麻烦，首先需要对线上应用开发admin 权限，然后评估大致是需要多少台分流机器，最后配置

## 信息管理和查看

页面http://cm.taobao.net:9999/monitorstat/load/app\_loadrun\_list.jsp

分流模式的执行是需要登录的，因此需要执行可以联系我。  
  
  
在页面上我们可以看到压测类型分为   
压测目标: 就是线上应用的机器IP

压测类型：就是上面提到的 两种烈性  
是否自动压测:定时执行并将结果报表通过邮件发送  
历史趋势: 利用目前最新的数据进行估计，参数days 表示未来多少天 up 表示增长率  
  
自动执行：手工操作对目标机器进行压测，目前来说 分流模式不建议这么操作。  
  
半自动化工操作：这个只针对分流模式。主要是分流自动执行危险系数比较高。所以建议采用手工操作

## 分流半自动化操作说明

### 1.进入半自动页面



在这个页面里，我们可以看到分流机器ip列表，需要注意的是，这个列表不能随便增加机器，但是可以减少机器。主要是启动apache 是需要admin权限的，因此需要确认后才能用。

在这个页面我们还能看到5个配置文件，这5个配置文件是用来分流配置使用，在进行操作前可以确认一下

### 2.操作分流

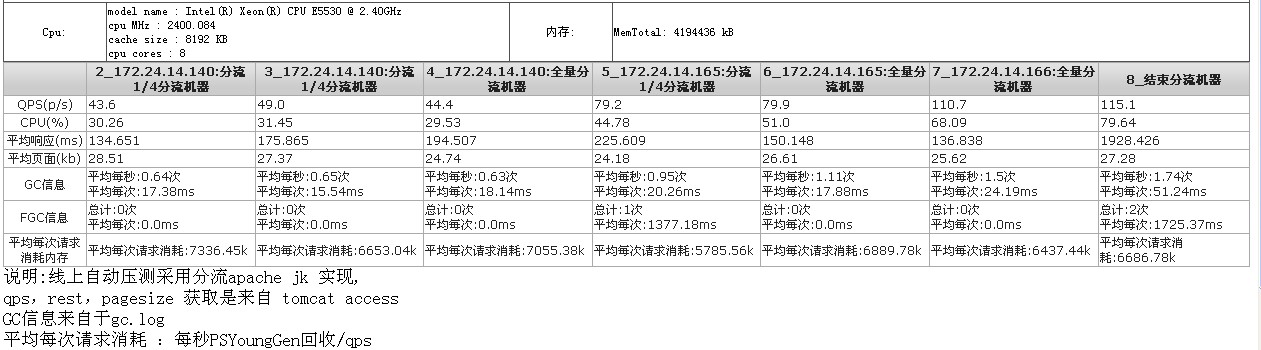


在这个页面的上面是一个操作区。下面是当前目标机器状态显示  
  
在上面共有 3个操作连接  
  
全量分流：就是将当前机器的全部流量切换到目标机器上去。

切换1/4流量：这个没按一次，将却换1/4流量过去，总共4次 ，最后一次是全量过去

结束本次分流: :一定要在结束后按 结束本次分流测试 不然连接不会断开apache 也不会恢复原始状态

### 3.结果显示页面



# 评分模块(线上线下性能跟踪体系)

## 评分功能说明

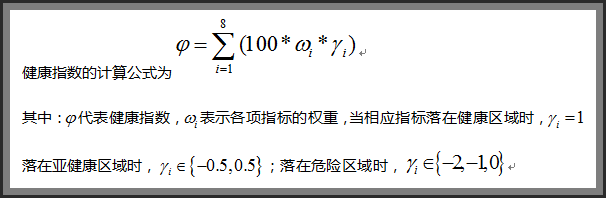
评分模块的用户入口： <http://cm.taobao.net:9999/monitorstat/health/index.jsp>

评分模块是线上线下性能跟踪体系的重要组成部分。我们期望通过数学建模的方式，在生产线基线与线下的性能测试基线之间建立可靠的衡量系数，进而通过线下性能测试指标评估出线上应用的健康状态。并给出性能健康指数的评分机制，一个分数说明性能健康与否，简洁直观。同时建立线上线下性能跟踪优化流程，并与生产线监控数据结合，出具优化方案，及时做好淘宝线上性能预警、优化、应急处理等性能保障工作。

评分首页会展现当前已有的评分应用的健康状况，健康分数以及历史分数链接入口：



将鼠标移到具体的分数上时，会弹出分数的计算公式：



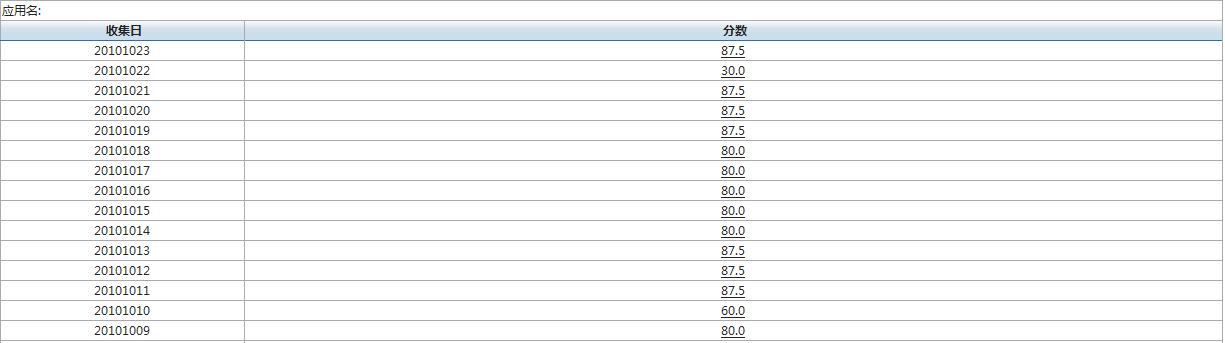
当鼠标点击具体分数时，会弹出详情框：



点击“得分历史”入口后，会展示该应用的历史得分图：



继续点击“查看应用评分历史细节信息”，会展示应用的历史分数：



在评分模块首页入口，如果点击“优化等级”列下的“急需优化”、“需优化”或“优化记录”，就可进入评分应用的优化记录历史列表：



该列表主要展示一些当前应用的性能问题，比如TPS过低，IO太高等等。

点击具体的“主题”，可以进入“优化解决方案”列表：



在该列表中可以填写具体的优化方案，比如问题的原因，建议采取的措施。

## 评分接入方式

评分模块设有一个简易的后台管理系统用于增加、修改、删除、查看当前的评分应用列表。目前该后台管理页面入口不对外开放，如果需要添加新的应用到评分模块，请联系性能测试组 耿电 [gengdian@taobao.com](mailto:gengdian@taobao.com) 或悟云 [wuyun.xl@taobao.com](mailto:wuyun.xl@taobao.com)