# JMX自动监控Java应用 解决well-known file is not secure - 文艺青年 - 博客频道 - CSDN.NET

http://blog.csdn.net/asdfasdfafasdf/article/details/52203645

JMX自动监控Java应用 解决well-known file is not secure

### 前言

最近在做一个监控的Agent组件开发工作，主要就是收集各种系统Metrics信息，然后上报给监控系统，以便进行监控和告警。目标就是要做到自动化：自动发现服务，动态探测服务，监控自动运维化，插件化等等。   
在[**Java**](http://lib.csdn.net/base/javase)应用监控这一块，主要就是监控Java应用的JMX信息。为了能够自动发现Java应用，自动连接JMX，做了不少尝试。本文就是分享，其中的一个问题的解决，把问题的解决思想记录下来。

### 什么是well-known file is not secure

当我们要连接VM的时候，如JMX，jmap，jstack时，VM会检查/tmp/hsperfdata\_$USER/$PID这样的一个文件，它会检查执行该命令的用户，是否和hsperfdata\_$USER这个文件所属的用户一致，如果不一致，就会出现well-known file is not secure这个异常，并拒绝连接

### 怎样通过JMX连接Java应用并监控

JMX连接都是通过一个连接字符串，调用JMXConnectorFactory.connect()方法建立连接。一般而言，JMX的连接字符串有两种获取方法：

* 第一种是手动指定形如service:jmx:rmi:///jndi/rmi://host:ip/jmxrmi的JMX Remote URL。
* 第二种则是先使用VirtualMachine.list()方法得到本地的所有JMX信息，然后通过Attach得到相关Pid的com.sun.management.jmxremote.localConnectorAddress进行连接。

第一种方法，需要在java应用启动时，指定开启JMX Remote，如可添加如下参数

-Dcom.sun.management.jmxremote

-Dcom.sun.management.jmxremote.port=${JMXport}

-Dcom.sun.management.jmxremote.ssl=false

-Dcom.sun.management.jmxremote.authenticate=false

这种方法的好处就是没有well-known file is not secure这种问题，但是这样就需要手动配置连接信息，也就失去了标题所述的自动连接，自动监控。

第二种方法，能够实现自动探测本地的应用，无论应用是否有开启JMX Remote的功能，都是可以连接。但是缺点就是会存在well-known file is not secure异常，就是当目标Java应用和本程序启动的用户不一样时，就会出现这个异常。所以就需要结合两个方法一起使用。

### 怎样结合两者？

话不多说，开门见山。

两者结合的方案就是，先使用第二种方法进行本地的Java应用扫描，无论目标应用是不是同一个用户启动，都是可以获取得到，并且也能够获取到它的Pid。然后针对每个应用，先进行第二中方法的连接，如果连接失败，则代表该应用存在well-known file is not secure异常。然后，就可以根据该应用的Pid，使用 ps -aux | grep Pid的方式，获取到应用的启动命令，然后解析启动命令，获取启动命令中的com.sun.management.jmxremote.port参数值port，然后根据该port，使用本地的IP地址127.0.0.1自动构建JMX Remote URL连接。

当然，这其中如果应用启动时，设置了参数-Dcom.sun.management.jmxremote.authenticate=true，则代表该JMX有授权认证，需要登录名和密码才能连接。我的解决方法是解析/etc/profile文件，获取JAVA\_HOME地址，然后解析JAVA\_HOME/jre/lib/management/jmxremote.access和JAVA\_HOME/jre/lib/management/jmxremote.password文件的内容获取用户名和密码。

但是，由于java对这两个文件有严格的用户权限控制：必须是运行该程序的用户才能进行读写，不然java应用会拒绝启动。所以即使是root用户启动，只要不是相同的用户，也无法访问这两个文件。   
凡事总会有解决办法，我的方案就是对对应的配置文件进行拷贝到一个临时文件，并授权777（因为是监控，所以本程序基本上都是root身份运行），然后获取临时文件内容，解析获取用户名和密码，解析完了再删除临时文件。这样就可以获取到JMX Remote连接的授权用户和密码。

最后补充一下，如果目标应用没有开启JMX Remote Connect的功能而且启动的用户也不同，那么应该是无法使用JMX连接了，除非使用相同的用户，再启动一个进程进行连接。不过，正常而言，公司应该都会开启Java应用的JMX Remote功能。

不过，又有一个问题来了   
如果Java是跑在[**Docker**](http://lib.csdn.net/base/docker) [**Container**](http://lib.csdn.net/base/docker)上怎么办？怎样同时监控Docker容器和容器内的应用呢？有时间再分享这个的解决方案。

# 开源一个监控数据采集Agent：OpenFalcon-SuitAgent - 文艺青年 - 博客频道 - CSDN.NET

http://blog.csdn.net/asdfasdfafasdf/article/details/52468240?locationNum=4&fps=1

OpenFalcon-SuitAgent

**项目地址：**[**github**](https://github.com/cqyijifu/OpenFalcon-SuitAgent)

### 版本说明

本系统版本划分如下

* alpha：内部测试版（不建议使用于生产环境）
* beta：公开测试版（不建议使用于生产环境）
* final：最终正式版（可用于生产环境）

当前版本请查看pom.xml信息。

### 使用之前

* 此系统是和[OpenFalcon](http://book.open-falcon.org/zh/index.html)监控系统一起使用,是为了更方便的进行运维监控。若不了解,可以先点击链接去OpenFalcon的官方社区进行了解。
* 目前此系统仅支持类Unix系统下使用，不支持Windows系统

### 什么是SuitAgent

这是一个获取各种系统的监控数据的Agent。   
其中内置了OpenFalcon的社区组件:FalconAgent,因所有的监控数据都必须要上报到FalconAgent，所以为了部署方便和管理，SuitAgent集成了FalconAgent，若agent.falcon.push.url配置为本地地址,则启动时会同时启动自带的FalconAgent，关闭时也会同时关闭FalconAgent。

### 为什么有这个项目

OpenFalcon监控系统，是由一系列的组件构成。对于[**操作系统**](http://lib.csdn.net/base/operatingsystem)的监控，官方有FalconAgent，但是对于其他系统的监控，   
都是使用脚本或者其他方式进行单独采集，然后将数据上报给FalconAgent系统。   
公司为了运维监控的方便和自动化，便有了这个项目，将系统的采集，集中在一个系统中，并且尽可能的采集便捷。

### SuitAgent特点

* 自动探测部署机上的服务,自动监控
* 监控配置动态生效,无需重启
* 能够动态发现部署机上新启动的服务
* 支持Mock接口功能,有自动化运维的公司,可利用此特性进行监控自动化开发
* 监控服务全部插件化实现，可自定义开发自己需求的监控服务的插件，只需要开发采集逻辑和插件运行配置。SuitAgent可自动发现插件，并根据配置自动运行，无需关心插件的启动。

### 目前支持的监控服务

* Docker（包括Docker内的应用存活情况监控）
* ElasticSearch
* Http
* Logstash
* Mysql
* 交换机
* Ntp
* Oracle
* Ping
* Standalone Jar（Java通过java -jar xxx.jar运行的服务）
* Tcp
* Tomcat
* Zookeeper

### SuitAgent编译

构建工具:maven   
[**Java**](http://lib.csdn.net/base/javase)版本：jdk 1.8

选择对应的版本进行编译:

- linux-64 : mvn clean package -Plinux-64 -Dmaven.test.skip=true

- linux64-noJar : mvn clean package -Plinux64-noJar -Dmaven.test.skip=true

- osx-noJar : mvn clean package -Posx-noJar -Dmaven.test.skip=true

编译后，在target目录下可看到编译后的压缩包，解压缩，改改配置，即可使用

或者直接下载编译后的文件，地址：

https://yunpan.cn/cMTHpgsQ3e5DV （提取码：9f74）

版本说明:

* linux-64 : linux-64位自带Jre运行环境。不需要目标系统安装java
* osx-noJar : 苹果系统。需要安装java环境
* linux64-noJar : linux-64位系统。需要目标系统安装个java环境

修改相关配置后,直接启动SuitAgent即可

建议：推荐编译使用，因目前*SuitAgent*尚在完善中，代码和配置难免会有所变动。网盘中的文件不能保证随时都是最新的。

#### SuitAgent配置

* FalconAgent配置文件(配置官方社区的FalconAgent)   
  + {SuitAgentHome}/conf/falcon/agent.cfg.json
* SuitAgent配置文件   
  + {SuitAgentHome}/conf/agent.properties
* 监控服务授权配置   
  + {SuitAgentHome}/conf/authorization.properties
* 监控服务插件配置   
  + {SuitAgentHome}/conf/plugin目录下

#### SuitAgent相关命令

* 启动：./bin/agent.sh start (使用root用户)   
  + 启动日志查看   
    - 可通过tail -f conf/console.log   
      观察SuitAgent的运行情况
* 关闭：./bin/agent.sh stop
* 状态：/bin/agent.sh status

### SuitAgent日志

* SuitAgent运行中的日志分为四种：   
  1、console（debug）   
  2、info   
  3、warn   
  4、error   
  每种日志均自动控制日志大小，每到5MB就自动进行日志分割，最多有10个同类文件。既所有的日志文件，最多只会达到200MB，无需担心日志文件过于庞大。

### 监控数据上报

#### 公有的Tag

跟OpenFalcon官方社区的思想一致,FalconAgent采集的系统监控信息(如内存,CPU等等一百多种)没有任何tag信息

其他的业务系统的监控,都会打上tag。`SuitAgent`采集的系统，基本都是业务系统（非Linux操作系统监控信息），所以都会有对应的Tag

* tag来区分服务

例如：

service={value} ：（內建）服务产品名，如tomcat

service.type={value} ：（內建）监控服务类型，如jmx,database

agentSignName={value} ：（內建）agent提供的标识字符串，如 allUnVariability（代表该服务SuitAgent启动时就不存在），标识服务的占用端口号，服务名等。

metrics.type={value} ：（內建）监控值类型，如availability，jmxObjectConf，jmxObjectInBuild，httpUrlConf，sqlConf，sqlInBuild，snmpConnomInBuild，snmpPluginInBuild

* 可用性(availability)会自动打上标签：

metrics.type=availability,service={value}

* 若某个服务有自定义的endPoint（如SNMP V3），则会加上customerEndPoint=true的tag
* Tomcat监控值会打上dir={dirName}的tag，方便同一个物理机启动多个tomcat时，更好的识别具体的tomcat

### SuitAgent Plugin 说明

SuitAgent所有的监控服务都是插件式开发集成

#### 如何自定义插件开发

SuitAgent支持的插件一共以下几种：

* JDBC的监控服务，实现com.yiji.falcon.agent.plugins.JDBCPlugin接口
* JMX的监控服务，实现com.yiji.falcon.agent.plugins.JMXPlugin接口
* SNMP V3的监控服务，实现com.yiji.falcon.agent.plugins.SNMPV3Plugin接口
* 探测监控服务，实现com.yiji.falcon.agent.plugins.DetectPlugin接口

若要开发自己的监控服务，想好自己的监控服务是哪种类型，参照SuitAgent目前已实现的插件结构，   
在包com.yiji.falcon.agent.plugins.plugin下建立自己的插件目录，将插件类放在该目录中，   
然后根据规则，在src/main/resources\_ext/conf/plugin/目录下建立自己的插件配置文件即可。

### SuitAgent目前集成的监控服务

#### JMX监控服务

##### 特殊的metrics说明

- 若SuitAgent在启动时，需要进行监控的服务（对应的work配置为true的）未启动，则将会上报一个名为`allUnVariability`的metrics监控指标，值为`0`。tag中有metrics的详情（参考tag命名），代表为该服务全部不可用

##### JMX监控属性组成

JMX监控的属性，由以下三部分组成

* SuitAgent内置的JMX监控属性
* - `HeapMemoryUsedRatio` - 堆内存使用比例
* - `HeapMemoryCommitted` - 堆内存已提交的大小
* - `NonHeapMemoryCommitted` - 非堆内存已提交的大小
* - `HeapMemoryFree` - 堆内存空闲空间大小
* - `HeapMemoryMax` - 堆内存最大的空间大小
* - `HeapMemoryUsed` - 堆内存已使用的空间大小
* - `NonHeapMemoryUsed` - 非堆内存已使用的空间大小
* JMX 公共的监控属性自定义配置
  + 定义于conf/jmx/common.properties文件
* 自定义的监控属性   
  + 每个插件自定义的属于自身的监控属性

##### Java应用的停机处理说明

正常情况下，若Java应用停机，则它的JMX连接将会不可用，此时，SuitAgent将会上报该应用不可用的监控报告，并且，在每一次重新获取监控值时，都会尝试重新连接此应用。

若该应用是被下线了，就是废弃了，那么岂不是会永远上报不可用状态？所以，SuitAgent有一个处理机制，在SuitAgent启动时，它会记录每一个Java应用的应用路径，如果该应用被发现停机了，它会检查该路径还是否存在有效，如果路径无效，SuitAgent将会清除此下线应用的监控信息，就不会上报不可用了。

##### 目前支持的监控组件

* zookeeper
* tomcat
* elasticSearch
* logstash
* standaloneJar（单独Jar包运行的应用，如SpringBoot)

#### JDBC监控的服务

##### 目前支持的监控组件

- Oracle

- Mysql

#### SNMP监控服务

##### 公共的metrics列表

每个接口的Metrics：

- {ifName}.if.HCInBroadcastPkts

- {ifName}.if.HCInMulticastPkts

- {ifName}.if.HCInOctets

- {ifName}.if.HCInUcastPkts

- {ifName}.if.HCOutBroadcastPkts

- {ifName}.if.HCOutMulticastPkts

- {ifName}.if.getIfHCOutUcastPkts

- {ifName}.if.HCOutOctets

- {ifName}.if.OperStatus : (接口状态，1 up, 2 down, 3 testing, 4 unknown, 5 dormant, 6 notPresent, 7 lowerLayerDown)

ping Metrics：

- pingAvgTime : Ping延时（正常返回延时，超时返回 -1）

##### 交换机（SNMP V3）

说明

监控的设备采集信息和采集逻辑主要参考了Falcon社区的swcollector项目，因swcollector不支持SNMP V3协议。   
<https://github.com/gaochao1/swcollector>

采集的私有metric列表

- 公共的metrics数据

- CPU利用率

- 内存利用率

内存和CPU的目前[**测试**](http://lib.csdn.net/base/softwaretest)的支持设备

- Cisco IOS(Version 12)

- Cisco NX-OS(Version 6)

- Cisco IOS XR(Version 5)

- Cisco IOS XE(Version 15)

- Cisco ASA (Version 9)

- Ruijie 10G Routing Switch

- Huawei VRP(Version 8)

- Huawei VRP(Version 5.20)

- Huawei VRP(Version 5.120)

- Huawei VRP(Version 5.130)

- Huawei VRP(Version 5.70)

- Juniper JUNOS(Version 10)

- H3C(Version 5)

- H3C(Version 5.20)

- H3C(Version 7)

#### 探测监控服务

* HTTP监控

监控Metrics:

- availability

- response.code : 响应状态码

- response.time : 响应时间 毫秒

* Ping监控

监控Metrics

- availability

- pingAvgTime : ping的平均延时（当前为每次ping5次，取绝对值）

- pingSuccessRatio : ping的成功次数占比，如ping了`5次`，只成功返回`4次`，则为`0.8`

* TCP（Socket）监控

监控Metrics :

- availability

- response.time : 响应时间 毫秒

* Yiji NTP 监控

监控Metrics

- availability

- `0`：NTP监控失败（如ntpdate命令执行失败）

- `1`：NTP监控成功

- ntpOffset : ntpdate命令解析的offset值

* [**Docker**](http://lib.csdn.net/base/docker) 监控

监控Metrics

- availability

- `0`：Docker daemon 探测失败

- `1`：Docker daemon 探测成功

- availability-container

- 说明：在SuitAgent第一次运行Docker插件时，会将第一次检测到的容器名称保存到内存缓存中，在以后的每一次监控时，会上报内存缓存中的容器的可用性状态

- `0`：容器已停止运行

- `1`：容器正在运行

- availability.container.app

- 说明：在SuitAgent第一次运行Docker插件时，会将第一次检测到的容器内的进程情况进行缓存到内存中，在以后的每一次监控时，会重新获取容器内的进程情况，若与第一次一致，则为可用

- `0`：容器内应用运行不正常

- `1`：容器内应用正常运行

- `-1`：容器内的应用状态数据获取失败

- has\_cpu : 是否有CPU使用权限

- total.cpu.usage.rate : CPU总使用率百分比

- user.cpu.usage.rate : CPU用户级别使用率百分比

- system.cpu.usage.rate : CPU系统级别使用率百分比

- has\_memory : 是否有内存使用权限

- mem.size.usage : 当前已使用的内存大小（单位兆）

- mem.size.cache : 当前已缓存的内存大小（单位兆）

- mem.usage.rate : 内存已使用与容器总内存百分比

- mem.cache.rate : 内存缓存与容器总内存的百分比

- has\_network : 是否有网络使用权限

- net.if.in.bytes : 网络IO流入字节数

- net.if.in.packets : 网络IO流入包数

- net.if.in.dropped : 网络IO流入丢弃数

- net.if.in.errors : 网络IO流入出错数

- net.if.out.bytes : 网络IO流出字节数

- net.if.out.packets : 网络IO流出包数

- net.if.out.dropped : 网络IO流出丢弃数

- net.if.out.errors : 网络IO流出出错数

### 自动发现功能说明

* zookeeper

无需配置，可自动发现

* tomcat

无需配置，可自动发现

* elasticSearch

无需配置，可自动发现

* logstash

无需配置，可自动发现

* standalone应用(java -jar 方式的Jar包运行)

需要配置conf/plugin/standaloneJarPlugin.properties配置文件的jmxServerName或jmxServerDir，   
然后SuitAgent会自动发现已配置的standalone应用

* [**Oracle**](http://lib.csdn.net/base/oracle)

需要配置conf/authorization.properties配置中的Oracle连接信息，然后会根据配置的连接信息，进行自动发现Oracle应用，并监控

* [**MySQL**](http://lib.csdn.net/base/mysql)

需要配置conf/authorization.properties配置中的Mysql连接信息，然后会根据配置的连接信息，进行自动发现Mysql应用，并监控

* SNMP V3 交换机

需要配置conf/authorization.properties配置中的交换机的SNMP V3的连接信息，然后会根据配置的连接信息，进行自动发现交换机并监控。

* HTTP监控

只要配置了被探测的地址，就会触发监控服务

* Ping监控

只要配置了被探测的地址，就会触发监控服务

* TCP监控

只要配置了被探测的地址，就会触发监控服务

* NTP监控

只要配置了NTP服务器地址，就会触发监控服务

* Docker监控

无需配置地址，自动探测本机的Docker服务：若发现有命令/usr/bin/docker存在，就会启动Docker监控插件。并自动连接本机的cAdvisor服务或启动内置的vAdvisor服务

### SuitAgent 动态配置

SuitAgent支持部分配置的动态生效，支持的范围见如下说明

authorization.properties文件的改动

* 若对应插件未启动，则文件修改后，将在SuitAgent下一次的自动服务发现时生效。
* 若对应的插件已启动，因系统不会重复启动相同的监控服务，故虽然插件配置会生效，但是不会重新启动服务

plugin目录下的插件配置文件的改动

* 若改动的是未启动的监控服务配置（如StandaloneJarPlugin插件的standaloneJarPlugin.properties文件，添加了一个服务名。或更改了插件启动类型等），将在SuitAgent下一次的自动服务发现时生效
* 若改动的是插件的监控配置（如Tomcat插件的tomcatPlugin.properties文件的服务器监控参数配置），下一次监控扫描就能够生效。
* 若改动的是插件的自定义配置文件，它的改动将不会触发插件的配置更新事件，不过可以利用改动它的插件配置文件，触发配置更新。

注意

* 已启动的插件服务，不会因为配置文件的改动而停止服务
* 插件启动时的配置项（如插件的step，pluginActivateType等）被改动时，若插件已启动，虽然改动的配置插件会实时更新，但由于服务已启动，这些属性已经在启动时固定，所以将不会因为改动而生效

### SuitAgent mock 接口

接口说明

SuitAgent提供可用性(availability)的mock服务。

mock服务具有有效性,有效性时长,通过配置 agent.properties 文件的 agent.mock.valid.time 配置项

有效的mock,将会让mock的目标服务即使已经停止运行,也会上报一个可用的监控数据,并且带上 mock=true 的 tag

若mock的时间超时,既无效,则会上报目标服务不可用的监控数据,并且带上 mock=timeout-{time} 的 tag 。其中{time}是停机时间

接口使用

* <http://ip:port/mock/list>
  + 查看SuitAgent当前所有的mock配置，返回json格式的数据,示例:

{

"jmx": {

"service": "tomcat",

"isTimeout": false,

"shutdownTime": 0

}

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 1
* 2
* 3
* 4
* 5
* 6
* 7
* <http://ip:port/mock/add/>serviceType/service
  + 添加一个mock配置
* <http://ip:port/mock/remove/>serviceType/service
  + 删除一个mock配置

参数说明

* serviceType : 对应于监控值tag中的service.type属性
* service : 对应于监控值tag中的service或agentSignName属性

# plugin\_ping · cqyijifu/OpenFalcon-SuitAgent Wiki

https://github.com/cqyijifu/OpenFalcon-SuitAgent/wiki/plugin\_ping

### 运行说明

只要配置了pingPlugin.properties中的探测地址配置，就会启动监控。

### 配置说明

* 配置文件位置：conf/plugin/pingPlugin.properties
* 配置项
  + step：采集周期，单位秒。【不支持动态配置生效】
  + address：需要被监控的HTTP地址。【支持动态配置生效】
  + # 注:pluginActivateType属性在此插件将不起作用,只要配置了监控地址,就会触发监控启动
  + # Ping地址探测监控,多个地址用英文逗号分隔,[tags] 为可选配置，用于指定自定义的上报tag于监控的地址上
  + # 格式：（suffix为可选后缀）
  + # address<suffix> = ip[tags]
  + #
  + # 【行默认tag】：每一行的第一个地址的tag为该行的 【行默认tag】，若第一个地址没有自定义tag，则该行无【行默认tag】
  + # 每一行的tag值规律：ip 形式的：将会打上【行默认tag】；ip[] 形式的，代表该IP无tag，ip[tag=tagValue] 形式的，代表该地址将使用定义的tag，不使用【行默认tag】
  + #
  + # 例如 （192.168.56.110将会自动打上【行默认tag】,192.168.56.111 将会不会打上自定义的tag）
  + #
  + # address = 192.168.46.22[tag1=tag1Value;tag2=tag2Value],192.168.56.110,192.168.56.111[]

# address.1 = 192.168.46.22[tag1=tag1Value;tag2=tag2Value],192.168.56.110

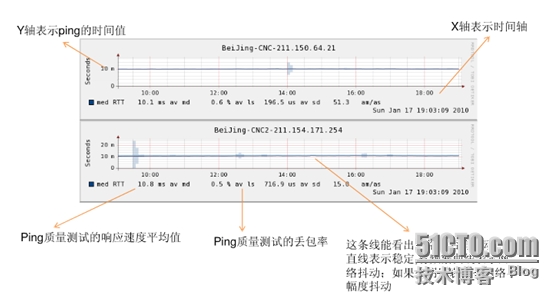
​

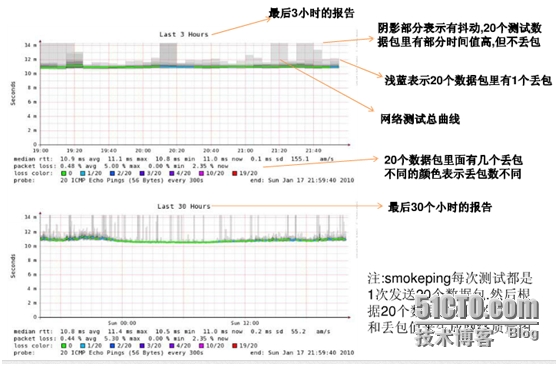
### 监控采集说明

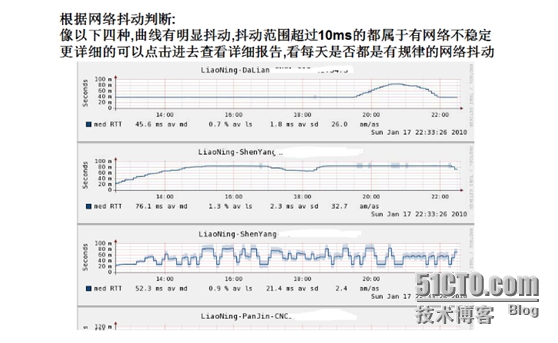
* availability：可用性
* pingAvgTime：平均延时
* pingSuccessRatio：ping的成功比率（如ping了5次，只有4次成功，则该值为0.8）

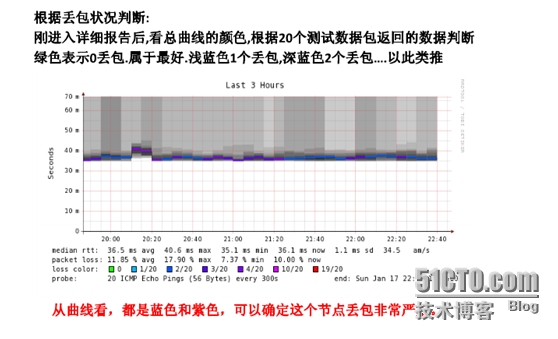
# Smokeping图表参数说明 - 技术成就自我 - 51CTO技术博客

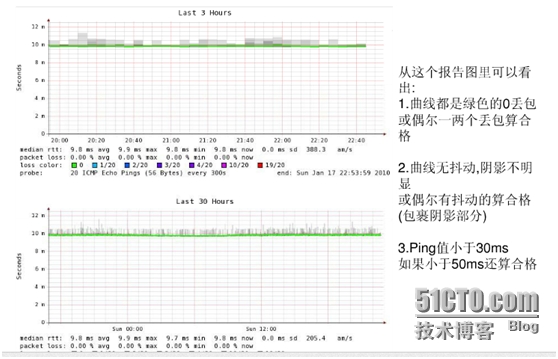
http://slzyer.blog.51cto.com/692866/1698136

[](http://s3.51cto.com/wyfs02/M02/73/B3/wKioL1YEwtmCJRN4AAEREArBhak769.jpg)

[](http://s3.51cto.com/wyfs02/M00/73/B3/wKioL1YEwtrhwdLtAAHvbX8HCG8932.jpg)

[](http://s3.51cto.com/wyfs02/M01/73/B6/wKiom1YEwtbQa6uGAAGzJsfqulQ923.jpg)

[](http://s3.51cto.com/wyfs02/M00/73/B3/wKioL1YEwtqRJ5XoAAGRfoxbdVk088.jpg)

[](http://s3.51cto.com/wyfs02/M02/73/B6/wKiom1YEwtfxrqjUAAG4gSACcKo881.jpg)

本文出自 “[技术成就自我](http://slzyer.blog.51cto.com/)” 博客，请务必保留此出处<http://slzyer.blog.51cto.com/692866/1698136>

# Linux 配置 SmokePing\_独树一帜叶公好龙z\_新浪博客

http://blog.sina.com.cn/s/blog\_8ace12de01014ohk.html

【实际需求】  
  
做为网站公司一般都会要求我们 IT 部提供全国各地 IDC 网络速度如何？质量？稳定性等等？  
一般 ISP 或者代理公司会给出测试服务器供我们测试该 IDC 的网络情况如何？  
比如 windows 服务器中有雷傲测试工具等等、就是通过在高峰时间段 ping 各地区的信息港或者骨干网络的 IP 地址；最后分析统计该时间段内的 ping 值数据的平均值和丢包率等；  
===================================================================================SmokePing

【Smokeping 简介】

是 rrdtool 的作者 Tobi Oetiker 的作品，所以它在图形显示方面有很大优势，也是一个很有特点的 opensource 工具：  
多种探测方式，包括 fping、echoping、dig、curl 等；  
易用可扩展的插件；  
master/slave 的工作方式，可以在多个节点收集同一个监测点的数据；  
很有特色的 alert 设置，不只是简单的设置一个阀值。  
====================================================

# 5款监控利器介绍(smokeping、cacti、nagios、zabbix、ntop)

http://www.360doc.com/content/16/0605/01/33848986\_565123487.shtml

## 1. smokeping

Smokeping 是rrdtool 的作者Tobi Oetiker 的作品，是用[Perl](http://www.ttlsa.com/perl/) 写的，主要是监视网络性能，包括常规的ping，用echoping监控www 服务器性能，监视dns 查询性能，监视ssh 性能等。底层也是rrdtool做支持，特点是画的图非常漂亮，网络丢包和延迟用颜色和阴影来表示。在企业中运用较少。

## 2 cacti

Cacti是一套基于[PHP](http://www.ttlsa.com/php/),MySQL,SNMP及RRDTool开发的网络流量监测图形分析工具。  
简单的说Cacti 就是一个PHP 程序。它通过使用SNMP 协议获取远端网络设备和相关信息，（其实就是使用Net-SNMP 软件包的snmpget 和snmpwalk 命令获取）并通过RRDTOOL 工具绘图，通过PHP 程序展现出来。我们使用它可以展现出监控对象一段时间内的状态或者性能趋势图。  
Cacti是通过 snmpget来获取数据，使用 RRDtool绘画图形，而且你完全可以不需要了解RRDtool复杂的参数。它提供了非常强大的数据和用户管理功能，可以指定每一个用户能查看树状结构、host以及任何一张图，还可以与LDAP结合进行用户验证，同时也能自己增加模板，功能非常强大完善。界面友好。 CACTI  
软件 Cacti 的发展是基于让 RRDTool 使用者更方便使用该软件，除了基本的 Snmp 流量跟系统资讯监控外，Cacti 也可外挂 Scripts 及加上 Templates 来作出各式各样的监控图。

## 3 [nagios](http://www.ttlsa.com/monitor/nagios/)

Nagios是一款开源的免费网络监视工具，能有效监控[Windows](http://www.ttlsa.com/windows/)、[Linux](http://www.ttlsa.com/linux/)和Unix的主机状态，交换机路由器等网络设置，打印机等。在系统或服务状态异常时发出邮件或短信报警第一时间通知网站运维人员，在状态恢复后发出正常的邮件或短信通知。  
Nagios 可以监控的功能有：

* 监控网络服务（SMTP、POP3、HTTP、NNTP、PING等）；
* 监控主机资源（处理器负荷、磁盘利用率等）；
* 简单地插件设计使得用户可以方便地扩展自己服务的检测方法；
* 并行服务检查机制；
* 具备定义网络分层结构的能力，用"parent"主机定义来表达网络主机间的关系，这种关系可被用来发现和明晰主机宕机或不可达状态；
* 当服务或主机问题产生与解决时将告警发送给联系人（通过EMail、短信、用户定义方式）；
* 可以定义一些处理程序，使之能够在服务或者主机发生故障时起到预防作用；
* 自动的日志滚动功能；
* 可以支持并实现对主机的冗余监控；
* 可选的WEB界面用于查看当前的网络状态、通知和故障历史、日志文件等；

## 4. ntop

Ntop是一种监控网络流量工具，用ntop显示网络的使用情况比其他一些网络管理软件更加直观、详细。Ntop甚至可以列出每个节点计算机的网络带宽利用率。他是一个灵活的、功能齐全的，用来监控和解决局域网问题的工具；尤其当ntop与nprobe配合使用，其功能更加显著。它同时提供命令行输入和web页面，可应用于嵌入式web服务  
Ntop主要提供以下一些功能:

* 自动从网络中识别有用的信息；
* 将截获的数据包转换成易于识别的格式；
* 对网络环境中通信失败的情况进行分析；
* 探测网络通信的时间和过程

## 5. [zabbix](http://www.ttlsa.com/monitor/zabbix/)

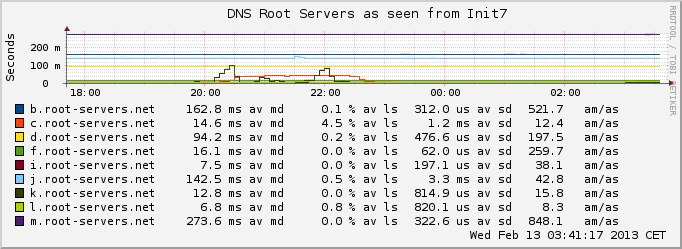
zabbix是一个基于WEB界面的提供分布式系统监视以及网络监视功能的企业级的开源解决方案。zabbix能监视各种网络参数，保证服务器系统的安全运营；并提供柔软的通知机制以让系统管理员快速定位/解决存在的各种问题。zabbix由2部分构成，zabbix server与可选组件zabbix agent。zabbix server可以通过SNMP，zabbix agent，ping，端口监视等方法提供对远程服务器/网络状态的监视，数据收集等功能，它可以运行在Linux, Solaris, HP-UX, AIX, Free BSD, Open BSD, OS X等平台上。

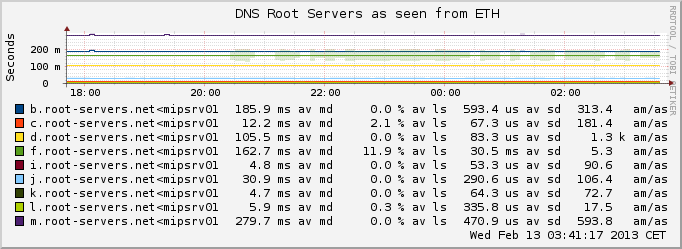
# SmokePing首页、文档和下载 - 网络性能监视 - 开源中国社区

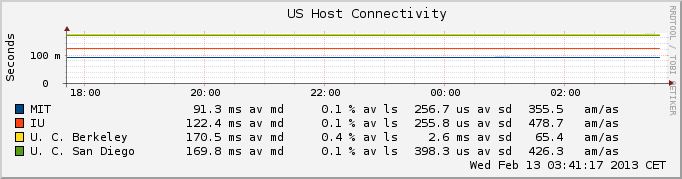
http://www.oschina.net/p/smokeping/similar\_projects?lang=0&sort=time&p=17

Smokeping 主要是监视网络性能，包括常规的 ping，用 echoping 监视 www 服务器性能，监视 dns 查询性能，监视 ssh 性能等。底层也是 rrdtool 做支持，特点是画的图非常漂亮，网络丢包和延迟用颜色和阴影来表示。

最新版本的 Smokeping 支持多个节点的检测结果从一个图上画出来。比如从 A、B 两个监视点检测 C 点的 ping 效果。可以把 A、B 的检测结果在一个图上表示出来，便于比较。







# 使用smokeping监控idc机房网络质量情况 - 吟—技术交流 - 51CTO技术博客

http://dl528888.blog.51cto.com/2382721/1266598

近工作比较忙，也没有时间写博客，看到好友[芮峰云](http://rfyiamcool.blog.51cto.com/)最近一直在写博客，所以也手痒了，就先把之前的一些积累下来的文章分享给大家。

本文是介绍如何的使用smokeping来监控idc机房的网络质量情况，从监控图上的延时与丢包能分辨出你机房的网络是否稳定，是否为多线，是否为BGP机房，到各城市的3个运行商网络各是什么情况，如果出现问题，如果有针对的解决。而且如果选择新机房的时候，你可以根据smokeping的监控结果来判断这个机房是否适合。

1、安装其他源

|  |  |
| --- | --- |
| 1 | rpm -Uvh http://apt.sw.be/redhat/el6/en/x86\_64/rpmforge/RPMS/rpmforge-release-0.5.3-1.el6.rf.x86\_64.rpm |

2、安装rrdtool与依赖库

|  |  |
| --- | --- |
| 1 | yum -y install perl perl-Net-Telnet perl-Net-DNS perl-LDAP perl-libwww-perl perl-RadiusPerl perl-IO-Socket-SSL perl-Socket6 perl-CGI-SpeedyCGI perl-FCGI perl-CGI-SpeedCGI perl-Time-HiRes perl-ExtUtils-MakeMaker perl-RRD-Simple rrdtool rrdtool-perl curl fping echoping  httpd httpd-devel gcc make  wget libxml2-devel libpng-devel glib pango pango-devel freetype freetype-devel fontconfig cairo cairo-devel libart\_lgpl libart\_lgpl-devel mod\_fastcgi |

3、下载与安装smokeping

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | wget http://oss.oetiker.ch/smokeping/pub/smokeping-2.6.8.tar.gz  tar zxvf smokeping-2.6.8.tar.gz  cd smokeping-2.6.8  ./configure --prefix=/usr/local/smokeping  出现问题是因为需要安装perl的模块，所以运行下面内容即可  ./setup/build-perl-modules.sh /usr/local/smokeping/thirdparty  ./configure --prefix=/usr/local/smokeping  /usr/bin/gmake install |

现在smokeping安装完成

4、配置smokeping

a.创建cache、data、var目录

|  |  |
| --- | --- |
| 1  2 | cd /usr/local/smokeping  mkdir cache data var |

b.在创建日志

|  |  |
| --- | --- |
| 1 | touch /var/log/smokeping.log |

c.授权

|  |  |
| --- | --- |
| 1  2 | chown apache:apache cache data var  chown apache:apache /var/log/smokeping.log |

d.修改配置文件

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | cd /usr/local/smokeping/htdocs/  mv smokeping.fcgi.dist smokeping.fcgi  cd /usr/local/smokeping/etc  mv config.dist config  vim config  cgiurl   = http://some.url/smokeping.cgi |

主要是修改这行，把some.url修改为你的ip或者域名

|  |  |
| --- | --- |
| 1  2 | \*\*\* Database \*\*\*  step     = 300 |

然后修改step，从300改为60，这是检测的时间。

e.配置完成之后修改密码文件权限

|  |  |
| --- | --- |
| 1 | chmod 600 /usr/local/smokeping/etc/smokeping\_secrets.dist |

f.修改apache的配置

|  |  |
| --- | --- |
| 1 | vim /etc/httpd/conf/httpd.conf |

添加如下

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | Alias /cache "/usr/local/smokeping/cache/"  Alias /cropper "/usr/local/smokeping/htdocs/cropper/"  Alias /smokeping "/usr/local/smokeping/htdocs/smokeping.fcgi"  <Directory "/usr/local/smokeping">  AllowOverride None  Options All  AddHandler cgi-script .fcgi .cgi  Order allow,deny  Allow from all  DirectoryIndex smokeping.fcgi  </Directory> |

f.设置开机启动

|  |  |
| --- | --- |
| 1 | echo "/usr/local/smokeping/bin/smokeping --logfile=/var/log/smokeping.log 2>&1 &" >> /etc/rc.local |

g.启动http与smokeping

|  |  |
| --- | --- |
| 1  2 | /etc/init.d/httpd start  /usr/local/smokeping/bin/smokeping --logfile=/var/log/smokeping.log 2>&1 & |

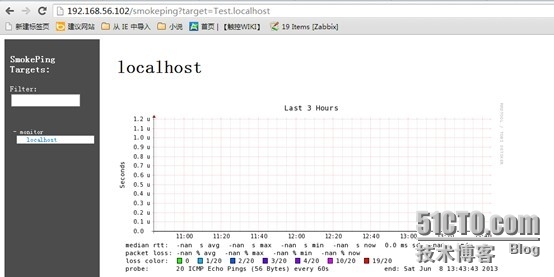
然后访问你在config里设置的cgiurl= <http://some.url/smokeping>

注意如果出现500错误就是因为你没有关闭selinux的问题，关闭了就能正常打开。

剩下就该你自己修改你要监控的内容了

注意+是第一层，++是第二层

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | + Test  menu= monitor  #parents = owner:/Test/James location:/  title=monitor all  ++ localhost  menu = localhost  title =localhost  alerts = someloss  slaves = boomer slave2  host = 192.168.56.102 |

然后smokeping的web界面如下图[](http://img1.51cto.com/attachment/201308/102203668.jpg)

如果各位就这样放到网上的话，是十分的不安全，所以需要加入访问控制。

h.修改httpd.conf里的内容

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | <Directory "/usr/local/smokeping">  AllowOverride None  Options All  AddHandler cgi-script .fcgi .cgi  AllowOverride AuthConfig  Order allow,deny  Allow from all  AuthName "Smokeping"  AuthType Basic  AuthUserFile /usr/local/smokeping/htdocs/htpasswd  Require valid-user  DirectoryIndex smokeping.fcgi  </Directory> |

把之前修改的httpd.conf里的这一段直接覆盖就行

然后进入cd /usr/local/smokeping/htdocs目录

设置登录账户与密码

|  |  |
| --- | --- |
| 1 | htpasswd -c /usr/local/smokeping/htdocs/htpasswd admin |

这个是设置登录账户为admin，密码在后面输入

然后重启httpd就可以实现密码验证登录

[](http://img1.51cto.com/attachment/201308/102305866.jpg)

需要注意，一定要同步好时间，可以使用cp /usr/share/zoneinfo/Asia/Shanghai /etc/localtime或者ntpdate 210.72.145.44。

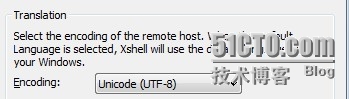
如果需要在网页里展示中文，修改/usr/local/smokeping/etc的config文件

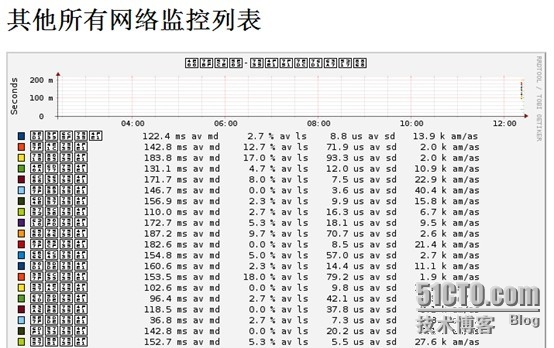
|  |  |
| --- | --- |
| 1  2 | \*\*\* Presentation \*\*\*  charset = utf-8 //注：在这里添加 |

然后在menu与titile里修改中文，重启即可

有一个要注意的地方就是，你输入的中文必须在utf-8的字符编码下输入的中文字符，不然会出现乱码。

如果在xshel下，选择file-properities-terminal

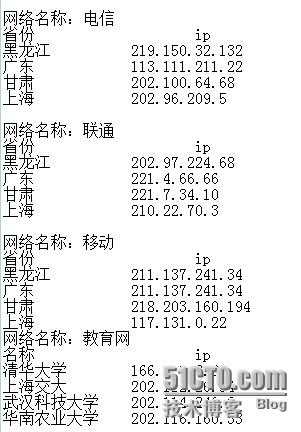
[](http://img1.51cto.com/attachment/201308/102459527.jpg)

如果还是不显示就看看你系统里是否安装了中文字体，或者在安装一个[](http://img1.51cto.com/attachment/201308/102549966.jpg)

debian上可以先apt-cache search wqy(文泉译), centos可以yum search wqy试试。

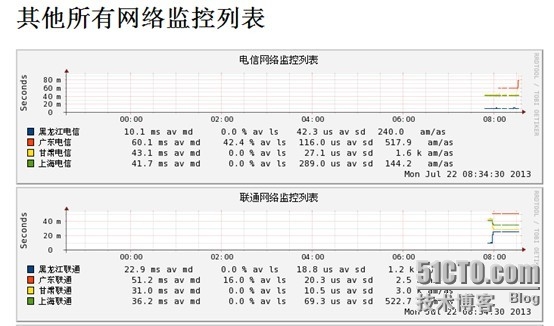
然后yum install后在重启smokeping

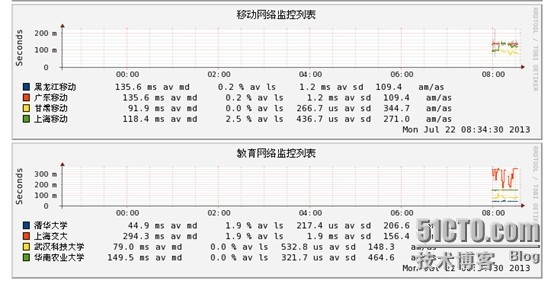
如果想监控本机到电信、联通、移动、教育网的网络情况监控，可以使用下面内容

[](http://img1.51cto.com/attachment/201308/102648507.jpg)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140 | + Other  menu = 其他网络监控  title = 其他所有网络监控列表  ++ dianxin  menu = 电信网络监控  title = 电信网络监控列表  host = /Other/dianxin/dianxin-hlj /Other/dianxin/dianxin-gd /Other/dianxin/dianxin-gs /Other/dianxin/dianxin-sh  +++ dianxin-hlj  menu = 黑龙江电信  title = 黑龙江电信  alerts = someloss  #slaves = boomer slave2  host = 219.150.32.132  +++ dianxin-gd  menu = 广东电信  title = 广东电信  alerts = someloss  #slaves = boomer slave2  host = 113.111.211.22  +++ dianxin-gs  menu = 甘肃电信  title = 甘肃电信  alerts = someloss  #slaves = boomer slave2  host = 202.100.64.68  +++ dianxin-sh  menu = 上海电信  title = 上海电信  alerts = someloss  #slaves = boomer slave2  host = 202.96.209.5  #+++ dianxin-multi  #menu = 多个电信网络监控列表  #title = 多个电信网络监控列表  #alerts = someloss  #slaves = boomer slave2  #host = /Other/dianxin/dianxin-hlj /Other/dianxin/dianxin-gd /Other/dianxin/dianxin-gs /Other/dianxin/dianxin-sh  #  ++ liantong  menu = 联通网络监控  title = 联通网络监控列表  host = /Other/liantong/liantong-hlj /Other/liantong/liantong-gd /Other/liantong/liantong-gs /Other/liantong/liantong-sh  +++ liantong-hlj  menu = 黑龙江联通  title = 黑龙江联通  alerts = someloss  #slaves = boomer slave2  host = 202.97.224.68  +++ liantong-gd  menu = 广东联通  title = 广东联通  alerts = someloss  #slaves = boomer slave2  host = 221.4.66.66  +++ liantong-gs  menu = 甘肃联通  title = 甘肃联通  alerts = someloss  #slaves = boomer slave2  host = 221.7.34.10  +++ liantong-sh  menu = 上海联通  title = 上海联通  alerts = someloss  #slaves = boomer slave2  host = 210.22.70.3  #+++ liantong-multi  #menu = 多个联通网络监控列表  #title = 多个联通网络监控列表  #alerts = someloss  #slaves = boomer slave2  #host = /Other/liantong/liantong-hlj /Other/liantong/liantong-gd /Other/liantong/liantong-gs /Other/liantong/liantong-sh  ++ yidong  menu = 移动网络监控  title = 移动网络监控列表  host = /Other/yidong/yidong-hlj /Other/yidong/yidong-gd /Other/yidong/yidong-gs /Other/yidong/yidong-sh  +++ yidong-hlj  menu = 黑龙江移动  title = 黑龙江移动  alerts = someloss  #slaves = boomer slave2  host = 211.137.241.34  +++ yidong-gd  menu = 广东移动  title = 广东移动  alerts = someloss  #slaves = boomer slave2  host = 211.137.241.34  +++ yidong-gs  menu = 甘肃移动  title = 甘肃移动  alerts = someloss  #slaves = boomer slave2  host = 218.203.160.194  +++ yidong-sh  menu = 上海移动  title = 上海移动  alerts = someloss  #slaves = boomer slave2  host = 117.131.0.22  #+++ yidong-multi  #menu = 多个移动网络监控列表  #title = 多个移动网络监控列表  #alerts = someloss  #slaves = boomer slave2  #host = /Other/yidong/yidong-hlj /Other/yidong/yidong-gd /Other/yidong/yidong-gs /Other/yidong/yidong-sh  ++ jiaoyu  menu = 教育网络监控  title = 教育网络监控列表  host = /Other/jiaoyu/jiaoyu-qh /Other/jiaoyu/jiaoyu-sh /Other/jiaoyu/jiaoyu-wh /Other/jiaoyu/jiaoyu-hn  +++ jiaoyu-qh  menu = 清华大学  title = 清华大学  alerts = someloss  #slaves = boomer slave2  host = 166.111.8.28  +++ jiaoyu-sh  menu = 上海交大  title = 上海交大  alerts = someloss  #slaves = boomer slave2  host = 202.112.26.34  +++ jiaoyu-wh  menu = 武汉科技大学  title = 武汉科技大学  alerts = someloss  #slaves = boomer slave2  host = 202.114.240.6  +++ jiaoyu-hn  menu = 华南农业大学  title = 华南农业大学  alerts = someloss  #slaves = boomer slave2  host = 202.116.160.33  #+++ jiaoyu-multi  #menu = 多个教育网络监控列表  #title = 多个教育网络监控列表  #alerts = someloss  #slaves = boomer slave2  #host = /Other/jiaoyu/jiaoyu-qh /Other/jiaoyu/jiaoyu-sh /Other/jiaoyu/jiaoyu-wh /Other/jiaoyu/jiaoyu-hn |

展示效果为

[](http://img1.51cto.com/attachment/201308/102750921.jpg)

[](http://img1.51cto.com/attachment/201308/102813218.jpg)

5、关于smokeping的master与slave配置

slave主机会自己去检查监测点的情况（loss and rtt），并将数值提交给master主机（通过smokeping.cgi)。  
值得注意的是，slave并不需要config文件，每次slave提交完数据以后，会询问master它自己的配置文件是否有修改，如果有修改的话slave会进行更新。

配置

在master的config里修改

|  |  |
| --- | --- |
| 1  2  3  4 | \*\*\* Slaves \*\*\*  secrets=/usr/local/smokeping/etc/smokeping\_secrets.dist #密码文件  +slave1#名称  display\_name=slave1 |

要注意的是"slave1"这个名字需要设置为slave主机的hostname，并且写在"slavesecrets.con"文件里面，如果不想用slave主机的hostname，那么在slave主机启动smokeping服务的时候需要使用"--slave-name"来指定自己的名字。在section

\*\*\* Targets \*\*\*

中指定那个host需要使用哪个slave帮忙进行检测。下层会继承上层的配置，如果下层不像继承的话，可以配置"slaves ="来避免使用slave。

具体安装与配置不介绍。

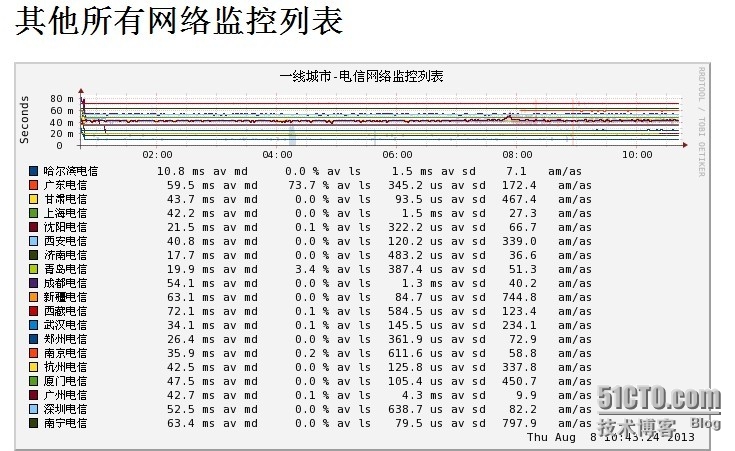
master/slave模式中，master和slave的图片都会显示在同一个页面中，如果不希望显示master的图片，可以使用

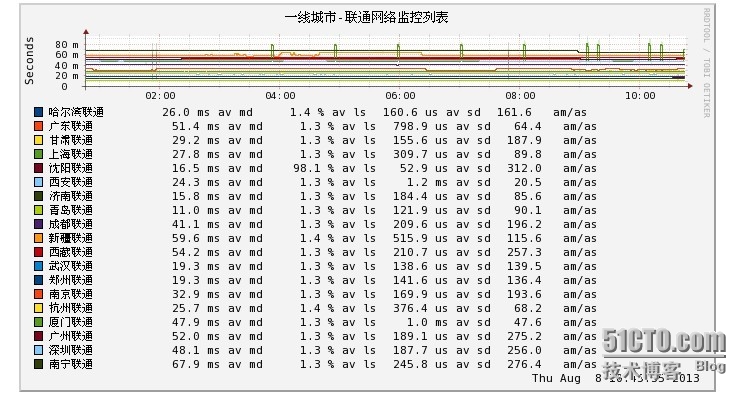
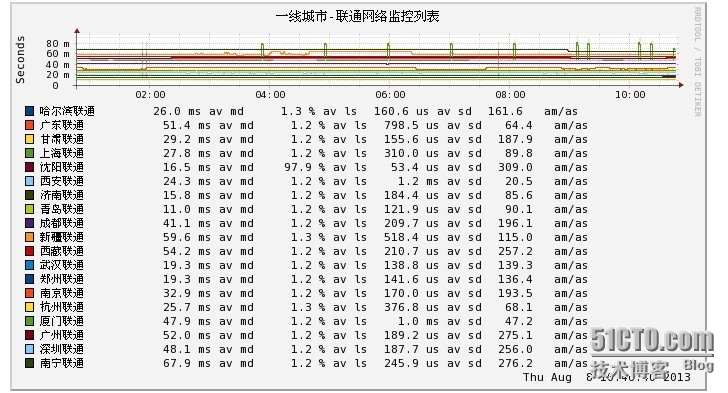
nomasterpoll=yes

参数。  
另外要注意的一点是，master上面启动"smokeping"进程和apache进程的用户必须要一致。因为rrd文件是由"smokeping"进程创建的，而数据的update却是由"smokeping.cgi"进行的，也就是apache用户。如果不一致的话会出现rrd文件无法被更新的情况，这个情况即使在slave主机上面进行debug也是看不出来的。具体的现象是在页面上面看到master的图像出现数据，但是slave的图像一直是空白的，数据显示是"nan"。

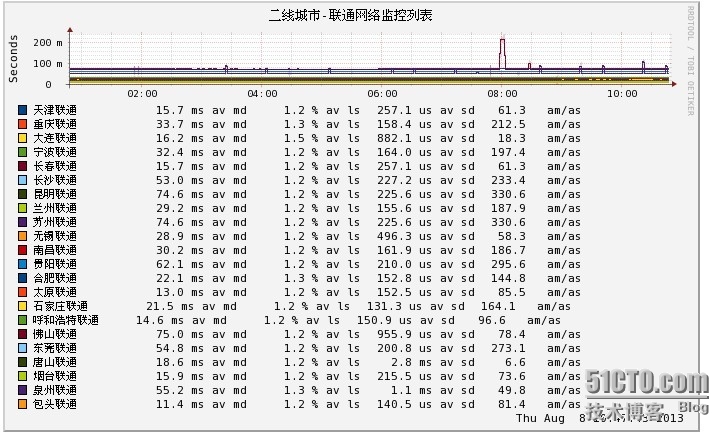
如果大家还想细化的监控，比如按照3大运行商（电信、联通、移动）到1、2、3线城市的网络延时与丢包情况，如下图显示

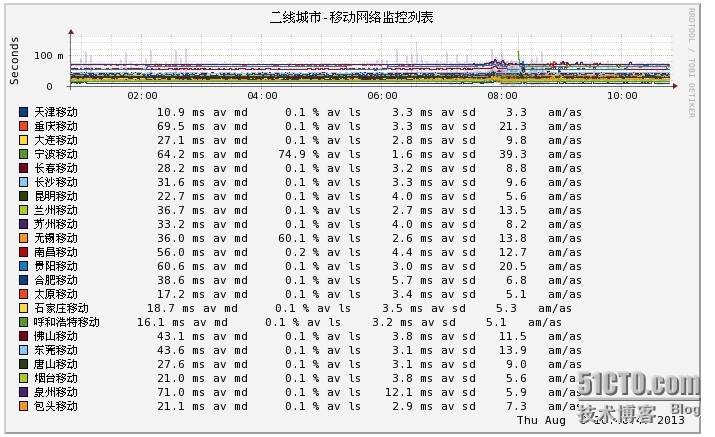
1、本机房到电信运行商的1线城市网络监控情况

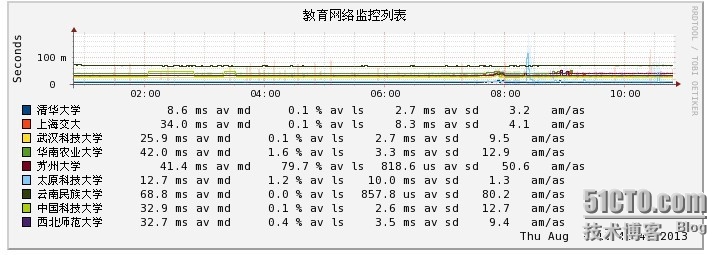
[](http://img1.51cto.com/attachment/201308/104353283.jpg)

2、本机房到联通运行商的1线城市网络监控情况3、本机房到移动运行商的1线城市网络监控情况[](http://img1.51cto.com/attachment/201308/104702477.jpg)

4、本机房到电信运行商的2线城市网络监控情况[](http://img1.51cto.com/attachment/201308/104750338.jpg)

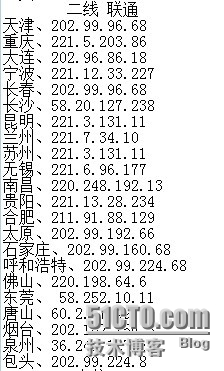
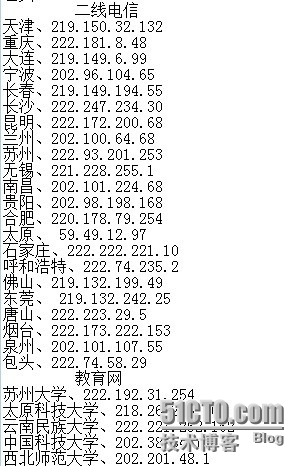
5、本机房到联通运行商的2线城市网络监控情况[](http://img1.51cto.com/attachment/201308/104831285.jpg)

6、本机房到移动运行商的2线城市网络监控情况[](http://img1.51cto.com/attachment/201308/104913560.jpg)

7、本机房到教育网的网络监控情况[](http://img1.51cto.com/attachment/201308/104959883.jpg)

如果达到这样的监控效果，你就能很清楚的了解你当前idc机房的网络情况，从图上的延时与丢包能分辨出你机房的网络是否稳定，是否为多线，是否为BGP机房，到各城市的3个运行商网络各是什么情况，如果出现问题，如果有针对的解决。而且如果选择新机房的时候，你可以根据smokeping的监控结果来判断这个机房是否适合。

下面是上面监控里ip的信息（如果大家想实现上面监控，可以使用下面的ip）

[](http://img1.51cto.com/attachment/201308/105602701.jpg)[](http://img1.51cto.com/attachment/201308/105631911.jpg)[](http://img1.51cto.com/attachment/201308/105707942.jpg)[](http://img1.51cto.com/attachment/201308/105745402.jpg)[](http://img1.51cto.com/attachment/201308/105828253.jpg)

本文出自 “[吟—技术交流](http://dl528888.blog.51cto.com/)” 博客，请务必保留此出处<http://dl528888.blog.51cto.com/2382721/1266598>

# 5、smokeping使用 - 梦~从这里起航 - 51CTO技术博客

http://mayulin.blog.51cto.com/blog/1628315/514397

5.1图表参数说明

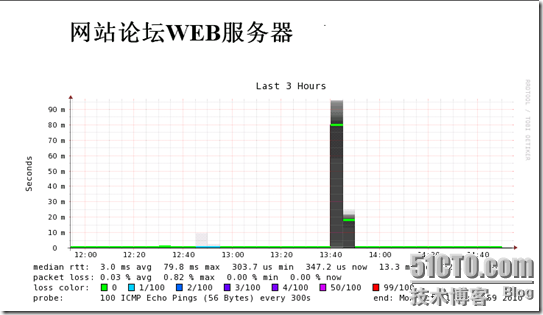
（1） RTT(Round-Trip Time)：简单说它是一个数据报在网络上两点中间往返一次的时间。是影响TCP性能和表征网络运行状况的重要参数。在网络中实时、准确地测量大量TCP设备和系统的RTT参数是网络管的重要环节之一。Smokeping就是这样的自动测试系统，它向目标设备和系统发送各种类型的测试数据包，测量、记录和展示RTT。

（2） Median RTT：它是中间数并不是平均值。Smokeping有多种类型的探针，探针在默认的设置下，每300秒向目标设备发送20测探测数据包。假如这20个数据包都返回的话，它就记录下了20个RTT，那么Median RTT就是第十个包的RTT；如果有5个包丢失的话，那么Median RTT就是第八个返回的包的RTT值。

（3） Avg RTT：它是每一个测试回合中所有RTT的算术评价值。

（4） Avg pkt loss：丢包率。

（5） Probe:100ICMP Echo Pings(56 Bytes)every 300s每一个绿色的短横线都是一个测试回合300秒内用ping测试100次。绿色画出的是中间数的位置，一个回合中的其它值都在它附近被以灰度的形式被刻画；

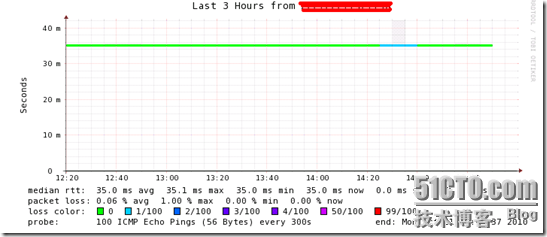
[](http://img1.51cto.com/attachment/201103/13/1628315_1300005201uvgr.png)

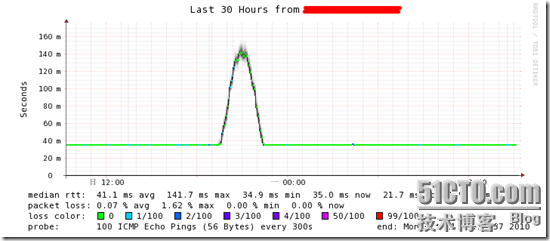
5.2 查看网络状况

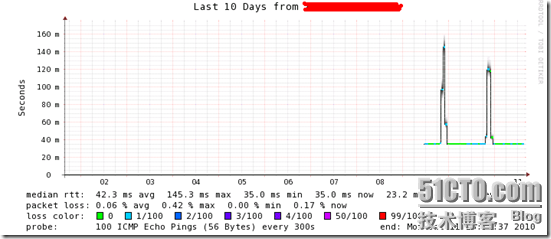
Smokeping可以方便地查看所监控的网络状况：

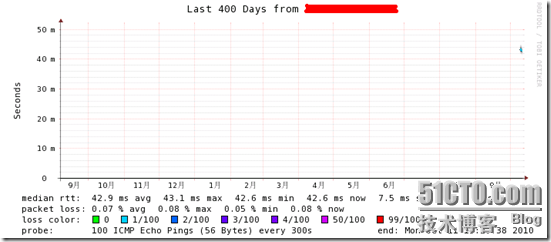
（1）可以很具体在图上看出网络的延迟及丢包等状况；

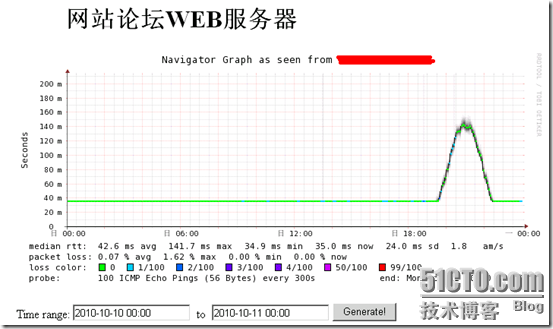
（2）具有Last 3 Hours，Last 30 Hours，Last 10 Days，Last 400 Days图像，还可以通过输入具体的时段来查看图像，如下所示:

[](http://img1.51cto.com/attachment/201103/13/1628315_1300005206gptT.png)

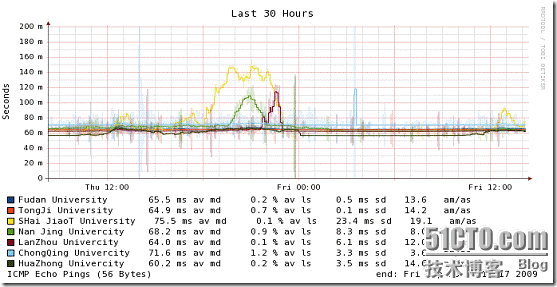
[](http://img1.51cto.com/attachment/201103/13/1628315_1300005212EwB0.png)

[](http://img1.51cto.com/attachment/201103/13/1628315_13000052175nIk.png)

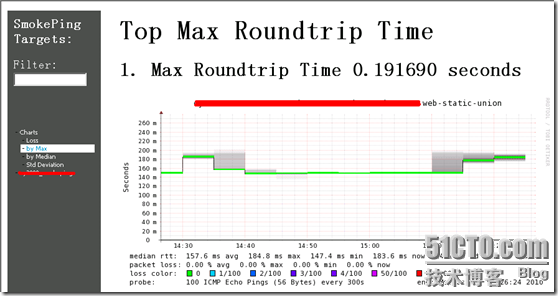
[](http://img1.51cto.com/attachment/201103/13/1628315_1300005221PgEt.png)

[](http://img1.51cto.com/attachment/201103/13/1628315_1300005226U1KE.png)

（3） multi-graph在一张图里面画上多个监测点的数值，方便对照分析以及整体状况统计，如下所示：

[](http://img1.51cto.com/attachment/201103/13/1628315_1300005232MR77.gif)

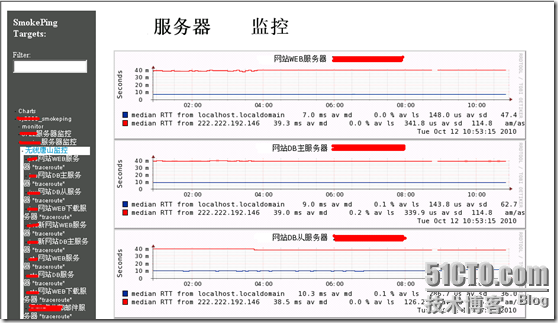
（4）在“Charts”下可以查看 “Top Packet Loss”，“Top Max Roundtrip Time”，“Top Median Roundtrip Time”，“Top Standard Deviation”情况，这样可以相当方便地判断哪里的网络有问题，从而可以快速地作出处理。如下图所示：

[](http://img1.51cto.com/attachment/201103/13/1628315_1300005238pOT5.png)

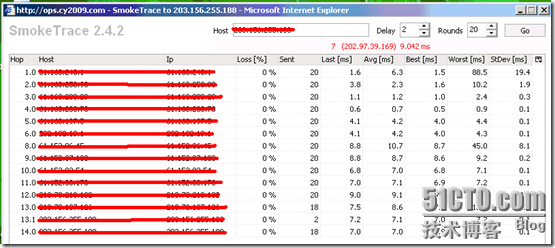
（5）traceroute功能

在smokeping的2.4.2版本开始加入了traceroute功能，使用方法如下：

在左边栏目中选择需要进行traceroute的主机，点击“\*traceroute\*”进行测试

[](http://img1.51cto.com/attachment/201103/13/1628315_13000052436R2W.png)

测试结果如图所示：

[](http://img1.51cto.com/attachment/201103/13/1628315_1300005254awIS.png)