

目录

naios 简介（摘录网络）	2
什么是 Nagios ?	2
nagios 安装.....	3
安装依赖软件.....	3
新增 nagios 帐号.....	3
下载 Nagios 和插件程序包编译安装.....	3
验证程序是否被正确安装.....	5
修改 nagios 页面文件.....	5
启动 Nagios.....	6
中文版 nagios 安装.....	7
常见报错解决.....	7
nagios 配置文件解析.....	8
nagios 默认配置文件介绍.....	8
配置文件之间的关系.....	9
templates.cfg 文件.....	10
关于报警级别.....	12
resource.cfg 文件.....	12
commands.cfg 文件.....	13
hosts.cfg 文件.....	14
services.cfg 文件.....	14
contacts.cfg 文件.....	15
timeperiods.cfg 文件.....	16
监控 web 主机配置举例.....	16
创建 host 和 service.....	16
修改 nagios.cfg.....	19
修改联系人信息.....	19
修改 templates.cfg.....	19
测试配置文件.....	20
Cacti+nagios 学习记录 v1	
监控服务、流量、性能	
V2 加入了 cacti 的使用，模版及脚本的导入，创建监 事图，插件的安装使用，及与 nagios 的结合使用， 当然也加入 nagios 的常见服务监控举例	
liujmsunits@gmail.com	
ok.来看一下界面效果.....	21
感知和处理状态抖动.....	22
利用 NRPE 扩展 Nagios 功能.....	26
NRPE 介绍.....	26
在 Linux 客户端安装 NRPE....	28
在服务器端安装 NRPE.....	30
遇到过的报错.....	31
cacti 安装与配置.....	33
cacti 简介.....	33
nagios、cacti 整合必要.....	34

cacti 环境依赖包安装.....	34
配置 PHP.....	35
配置 MySQL.....	35
安装 cacti.....	36
sping 安装配置.....	38
客户端 snmpd 的安装配置.....	39
cacti 界面.....	40
cacti 使用.....	43
cacti 脚本与模版.....	43
cacti 插件.....	43
cacti+nagios 结合使用.....	43
线上环境的系统及流量监控需求.....	43

naios 简介

nagios 安装

nagios 配置文件解析

nagios 常见监控举例

nagios 使用

cacti 安装与配置

cacti 使用

cacti 脚本与模版

cacti 插件

cacti+nagios 结合使用

线上环境的系统及流量监控需求

naios 简介（摘录网络）

什么是 Nagios？

Nagios 是一款用于系统和网络监控的应用程序。它可以在你设定的条件下对主机和服务进行监控，在状态变差和变好的时候给出告警信息。

Nagios 最初被设计为在 Linux 系统之上运行，然而它同样可以在类 Unix 的系统之上运行。

Nagios 更进一步的特征包括：

- 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）；
具备定义事件句柄功能，它可以在主机或服务的事件发生时获取更多问题定位；
自动的日志回滚；
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；

nagios 安装

安装依赖软件

```
yum install glibc glibc-common httpd gcc php libpng libpng-devel libjpeg libjpeg-devel  
zlib zlib-devel libXpm libXpm-devel
```

gd-devel 在 epel 源里面安装如果出错，则手动安装

```
wget  
wget
```

先删除系统默认安装的 gd
`rpm -e gd --nodeps`

然后 rpm 安装：
`rpm -ivh gd-devel-2.0.35-11.el6.x86_64.rpm gd-2.0.35-11.el6.x86_64.rpm`

新增 nagios 帐号

创建一个名为 **nagios** 的帐号并给定登录口令

```
/usr/sbin/useradd nagios
```

创建一个用户组名为 **nagcmd** 用于从 Web 接口执行外部命令。将 nagios 用户和 apache 用户都加到这个组中。

```
/usr/sbin/groupadd nagcmd  
/usr/sbin/usermod -G nagcmd nagios  
/usr/sbin/usermod -G nagcmd apache
```

下载 Nagios 和插件程序包编译安装

wget

wget

编译安装：

```
tar xzf nagios-3.0rc1.tar.gz
```

```
cd nagios-3.0rc1
```

运行 Nagios 配置脚本并使用先前开设的用户及用户组：

```
./configure --with-command-group=nagcmd
```

编译 Nagios 程序包源码

```
make all
```

安装二进制运行程序、初始化脚本、配置文件样本并设置运行目录权限

```
*** Compile finished ***

If the main program and CGIs compiled without any errors, you
can continue with installing Nagios as follows (type 'make'
without any arguments for a list of all possible options):

make install
- This installs the main program, CGIs, and HTML files

make install-init
- This installs the init script in /etc/rc.d/init.d

make install-commandmode
- This installs and configures permissions on the
  directory for holding the external command file

make install-config
- This installs *SAMPLE* config files in /usr/local/nagios/etc
  You'll have to modify these sample files before you can
  use Nagios. Read the HTML documentation for more info
  on doing this. Pay particular attention to the docs on
  object configuration files, as they determine what/how
  things get monitored!

make install-webconf
- This installs the Apache config file for the Nagios
  web interface

make install-exfoliation
- This installs the Exfoliation theme for the Nagios
  web interface

make install-classicui
- This installs the classic theme for the Nagios
```

```
make install
```

```
make install-init
```

```
make install-commandmode
```

```
make install-config
```

```
make install-webconf
```

```
make install-exfoliation
```

```
make install-classicui
```

安装 nagios 插件

```
tar -zxvf nagios-plugins-1.5.tar.gz
cd nagios-plugins-1.5
./configure --with-nagios-user=nagios --with-nagios-
group=nagios
make
make install
```

验证程序是否被正确安装

切换目录到安装路径（这里是/usr/local/nagios），看是否存在 etc、bin、sbin、share、var 这五个目录，如果存在则可以表明程序被正确的安装到系统了。后表是五个目录功能的

简要说明：

bin Nagios 执行程序所在目录，nagios 文件即为主程序

etc Nagios 配置文件位置，初始安装完后，只有几个*.cfg-sample 文件

sbin Nagios Cgi 文件所在目录，也就是执行外部命令所需文件所在的目录

Share Nagios 网页文件所在的目录

Var Nagios 日志文件、spid 等文件所在的目录

修改 nagios 页面文件

```
[root@ngios-cacti nagios]# cat
/etc/httpd/conf.d/nagios.conf
# SAMPLE CONFIG SNIPPETS FOR APACHE WEB SERVER
#
# This file contains examples of entries that need
# to be incorporated into your Apache web server
# configuration file. Customize the paths, etc. as
# needed to fit your system.

ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"

<Directory "/usr/local/nagios/sbin">
# SSLRequireSSL
Options ExecCGI
AllowOverride None
Order allow,deny
```

```
    Allow from all
#   Order deny,allow
#   Deny from all
#   Allow from 127.0.0.1
    AuthName "Nagios Access"
    AuthType Basic
    AuthUserFile /usr/local/nagios/etc/htpasswd.users
    Require valid-user
</Directory>
```

```
Alias /nagios "/usr/local/nagios/share"
```

```
<Directory "/usr/local/nagios/share">
#   SSLRequireSSL
    Options None
    AllowOverride None
    Order allow,deny
    Allow from all
#   Order deny,allow
#   Deny from all
#   Allow from 127.0.0.1
    AuthName "Nagios Access"
    AuthType Basic
    AuthUserFile /usr/local/nagios/etc/htpasswd.users
    Require valid-user
</Directory>
```

这个是安装完 nagios 默认添加的，可以不用修改

添加一个 web 登录帐号就行，默认是有一个帐号，如果单纯的添加去掉 -c

```
[root@ngios-cacti nagios]# htpasswd -c /usr/local/nagios/etc/htpasswd.users nagios
New password:
Re-type new password:
Adding password for user nagios
```

并将 httpd 的运行用户和组改为 nagios , nagcmd

启动 Nagios

把 Nagios 加入到服务列表中以使之在系统启动时自动启动

```
chkconfig --add nagios
chkconfig nagios on
```

验证 Nagios 的样例配置文件

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

如果没有报错，可以启动 Nagios 服务

```
service nagios start
```

中文版 nagios 安装

wget

安装过程和上面是一样的，只是中文版本相对低点，没有最新功能，而且会出现莫名其妙的错误，不推荐使用

常见报错解决



Forbidden

You don't have permission to access /nagios/ on this server.

Apache/2.2.15 (Red Hat) Server at 211.211.211.200 Port 80

原因是：

`DirectoryIndex index.php index.html index.html.var` 加入 index.php，前提是安装了 php，正常情况是不需要添加的



原因是 php 未安装，yum install php -y

Not Found

The requested URL /nagios/cgi-bin/statusmap.cgi was not found on this server.

Apache/2.2.15 (Red Hat) Server at 211.211.211.200 Port 80

原因是 gd-devel 没有安装，确保这些 yum install gd gd-devel libpng libpng-devel libjpeg libjpeg-devel zlib zlib-devel 都安装了 gd-devel 这个软件报在 epel 源安装如果出错，可以手动安装，参照前面

nagios 配置文件解析

nagios 默认配置文件介绍

nagios 安装完毕后，默认的配置文件在/usr/local/nagios/etc 目录下，每个文件或目录含义如下表所示：

文件名	用途
cgi.cfg	控制 cgi 访问的配置文件
nagios.cfg	Nagios 主配置文件
resource.cfg	变量定义文件，或者叫资源文件，通过在此文件中定义变量，以便让其它配置文件引用，如\$USER1\$
objects	objects 是一个目录，在此目录下有很多配置文件模板，用于定义 Nagios 对象
objects/commands.cfg	命令定义配置文件，里面定义的命令可以被其它配置文件引用
objects/contacts.cfg	定义联系人和联系人组的配置文件
objects/localhost.cfg	定义监控本地主机的配置文件
objects/printer.cfg	定义监控打印机的一个配置文件模板，默认没有启用此文件。
objects/switch.cfg	监控路由器的一个配置文件模板，默认没有启用此文件
objects/templates.cfg	定义主机、服务的一个模板配置文件，可以在其它配置文件中引用
objects/timeperiods.cfg	定义 nagios 监控时间段的配置文件
objects/windows.cfg	控 Windows 主机的一个配置文件模板，默认没有启用此文件

配置文件之间的关系

在 nagios 的配置过程中涉及到的几个定义有：主机、主机组，服务、服务组，联系人、联系人组，监控时间，监控命令等，从这些定义可以看出，nagios 各个配置文件之间是互为关联，彼此引用的。

成功配置出一台 nagios 监控系统，必须要弄清楚每个配置文件之间依赖与被依赖的关系，最重要的有四点：

- 第一：定义监控哪些主机、主机组、服务和组
- 第二：定义这个监控要使用什么命令实现，
- 第三：定义监控的时间段，

第四：定义主机或服务出现问题时要通知的联系人和联系人组。

为了能更清楚的说明问题，同时也为了维护方便，建议将 nagios 各个定义对象创建独立的配置文件：

即为：

创建 hosts.cfg 文件来定义主机和主机组

创建 services.cfg 文件来定义服务

用默认的 contacts.cfg 文件来定义联系人和联系人组

用默认的 commands.cfg 文件来定义命令

用默认的 timeperiods.cfg 来定义监控时间段

用默认的 templates.cfg 文件作为资源引用文件

templates.cfg 文件

nagios 主要用于监控主机资源以及服务，在 nagios 配置中称为对象，为了不必重复定义一些监控对象，Nagios 引入了一个模板配置文件，将一些共性的属性定义成模板，以便于多次引用。这就是 templates.cfg 的作用。

下面详细介绍下 templates.cfg 文件中每个参数的含义：

```
define contact{
    name          generic-contac #联系人名称，
    service_notification_period 24x7 #当服务出现异常时，发送通知的时间段，
    这个时间段“7x24”在 timeperiods.cfg 文件中定义
    host_notification_period 24x7 #当主机出现异常时，发送通知的时间段，
    这个时间段“7x24”在 timeperiods.cfg 文件中定义
    service_notification_options w,u,c,r #这个定义的是“通知可以被发出的情况”。w 即 warn，表示警告状态，u 即 unknown，表示不明状态，c 即 criticle，表示紧急状态，r 即 recover，表示恢复状态。也就是在服务出现警告状态、未知状态、紧急状态和重新恢复状态时都发送通知给使用者。
    host_notification_options d,u,r #定义主机在什么状态下需要发送通知给使用者，d 即 down，表示宕机状态，u 即 unreachable，表示不可到达状态，r 即 recovery，表示重新恢复状态。
    service_notification_commands notify-service-by-email #服务故障时，发送通知的方式，可以是邮件和短信，这里发送的方式是邮件，其中“notify-service-by-email”在 commands.cfg 文件中定义。
    host_notification_commands notify-host-by-email #主机故障时，发送通知的方式，可以是邮件和短信，这里发送的方式是邮件，其中“notify-host-by-email”在 commands.cfg 文件中定义。
    register      0
}
```

```

    }

    define host{
        name generic-host #主机名称，这里的主机名，并不是直接对应到真正机
器的主机名，乃是对应到在主机配置文件里所设定的主机名。
        notifications_enabled      1
        event_handler_enabled      1
        flap_detection_enabled     1
        failure_prediction_enabled  1
        process_perf_data          1
        retain_status_information   1
        retain_nonstatus_information 1
        notification_period 24x7    #指定“发送通知”的时间段，也就是可以在什
么时候发送通知给使用者。
        register                    0
    }
define host{
    name linux-server #主机名称
    use generic-host #use 表示引用，也就是将主机 generic-host 的所有属性
引用到 linux-server 中来，在 nagios 配置中，很多情况下会用到引用。
    check_period 24x7 #这里的 check_period 告诉 nagios 检查主机的时间段
    check_interval 5 #nagios 对主机的检查时间间隔，这里是 5 分钟。
    retry_interval 1 #重试检查时间间隔，单位是分钟。
    max_check_attempts 10 #nagios 对主机的最大检查次数，也就是 nagios 在检
查发现某主机异常时，并不马上判断为异常状况，而是多试几次，因为有可能
只是一时网络太拥挤，或是一些其他原因，让主机受到了一点影响，这里的 10
就是至少试 10 次的意思。
    check_command check-host-alive #指定检查主机状态的命令，其中“check-
host-alive”在 commands.cfg 文件中定义。
    notification_period workhours #主机故障时，发送通知的时间范围，其中
“workhours”在 timeperiods.cfg 中进行了定义，下面会陆续讲到。
    notification_interval 120 #在主机出现异常后，故障一直没有解决，nagios
再次对使用者发出通知的时间。单位是分钟。如果你觉得，所有的事件只需要
一次通知就够了，可以把这里的选项设为 0
    notification_options d,u,r #定义主机在什么状态下可以发送通知给使用者，
d 即 down，表示宕机状态，u 即 unreachable，表示不可到达状态，r 即
recovery，表示重新恢复状态。
    contact_groups admins #指定联系人组，这个“admins”在 contacts.cfg 文件
中定义。
    register 0
}
define service{
    name generic-service #定义一个服务名称

```

```
active_checks_enabled      1
passive_checks_enabled     1
parallelize_check          1
obsess_over_service        1
check_freshness            0
notifications_enabled      1
event_handler_enabled      1
flap_detection_enabled     1
failure_prediction_enabled  1
process_perf_data          1
retain_status_information   1
retain_nonstatus_information 1
is_volatile                0
check_period 24x7 #这里的 check_period 告诉 nagios 检查服务的时间段。
max_check_attempts 3 #nagios 对服务的最大检查次数。
normal_check_interval 10 #此选项是用来设置服务检查时间间隔，也就是说，
nagios 这一次检查和下一次检查之间所隔的时间，这里是 10 分钟。
retry_check_interval 2 #重试检查时间间隔，单位是分钟。
contact_groups admins #指定联系人组，同上。
notification_options w,u,c,r #这个定义的是“通知可以被发出的情况”。w 即
warn，表示警告状态，u 即 unknown，表示不明状态，c 即 criticle，表示紧急状
态，r 即 recover，表示恢复状态。也就是在服务出现警告状态、未知状态、紧
急状态和重新恢复后都发送通知给使用者。
notification_interval 60 #在服务出现异常后，故障一直没有解决，nagios 再
次对使用者发出通知的时间。单位是分钟。如果你认为，所有的事件只需要一
次通知就够了，可以把这里的选项设为 0。
notification_period 24x7 #指定“发送通知”的时间段，也就是可以在什么
时候发送通知给使用者。
register 0
}
```

关于报警级别

- (1) w : WARNING , 警告
- (2) u : UNKNOWN , 未知
- (3) c : CRITICAL , 危险 (已达临界值)
- (4) d : DOWN , 已宕机
- (5) r : RECOVERY , 状态已恢复至 OK
- (6) f : FLAPPING , (未看懂这个状态的意思 , 也许是状态波动很大)
- (7) n : NONE , 不发送告警通知邮件

resource.cfg 文件

resource.cfg 是 nagios 的变量定义文件，文件内容只有一行：

\$USER1\$=/usr/local/nagios/libexec

其中，变量\$USER1\$指定了安装 nagios 插件的路径，如果把插件安装在了其它路径，只需在这里进行修改即可。需要注意的是，变量必须先定义，然后才能在其它配置文件中进行引用。

Nagios 可用的全部的宏

主机宏

\$HOSTNAME\$ 主机简称(如"web")，取自于主机定义里的 host_name 域

\$HOSTADDRESS\$ 主机地址。取自于主机定义里的 address 域

服务宏

\$SERVICESTATE\$ 服务状态描述，有 w , u , c

\$SERVICEDESC\$ 对当前服务的描述

联系人宏

\$CONTACTNAME\$ 表示联系人，在联系人文件中定义

通知宏

\$NOTIFICATIONTYPE\$ 返回下面信息：("PROBLEM", "RECOVERY", "ACKNOWLEDGEMENT", "FLAPPINGSTART", "FLAPPINGSTOP", "FLAPPINGDISABLED", "DOWNTIMESTART", "DOWNTIMEEND", or "DOWNTIMECANCELLED").

日期/时间宏

\$LONGDATETIME\$ 当前的日期/时间戳

文件宏

\$LOGFILE\$ 日志文件的保存位置。

\$MAINCONFIGFILE\$ 主配置文件的保存位置。

其他宏

\$ADMINEMAIL\$ 全局的管理员 EMail 地址

\$ARGn\$ 指向第 n 个命令传递参数(通知、事件处理、服务检测等)。Nagios 支持最多 32 个参数宏

commands.cfg 文件

此文件默认是存在的，无需修改即可使用，当然如果有新的命令需要加入时，在此文件进行添加即可。这里并未列出文件的所有内容，仅仅介绍了配置中用到的一些命令。

#下面是 notify-host-by-email 命令的定义

```
define command{
```

```
command_name check-host-alive    #命令名称，用来检测主机状态。
command_line $USER1$/check_ping -H $HOSTADDRESS$ -w 3000.0,80% -c
5000.0,100% -p 5    #这里的变量$USER1$在 resource.cfg 文件中进行定义，即
$USER1$=/usr/local/nagios/libexec，那么 check_ping 的完整路径
为/usr/local/nagios/libexec/check_ping。“-w 3000.0,80%”中“-w”说明后面的一对
值对应的是“WARNING”状态，“80%”是其临界值。“-c 5000.0,100%”中“-c”说明后
面的一对值对应的是“CRITICAL”，“100%”是其临界值。“-p 1”说明每次探测发送
一个包。
}
#下面是 notify-host-by-email 命令的定义
define command{
command_name check_ftp
command_line $USER1$/check_ftp -H $HOSTADDRESS$ $ARG1$    # $ARG1$ 是
指在调用这个命令的时候，命令后面的第一个参数。
}
#下面是 check_http 命令的定义
define command{
command_name check_http
command_line $USER1$/check_http -I $HOSTADDRESS$ $ARG1$
}
#下面是 check_ssh 命令的定义
define command{
command_name check_ssh
command_line $USER1$/check_ssh $ARG1$ $HOSTADDRESS$
}
.....
```

hosts.cfg 文件

此文件默认不存在，需要手动创建，hosts.cfg 主要用来指定被监控的主机地址以及相关属性信息，一个配置好的实例如下：

```
define host{
use linux-server    #引用主机 linux-server 的属性信息，linux-server 主
机在 templates.cfg 文件中进行了定义。
host_name web    #主机名
alias ixdba-web    #主机别名
address 192.168.12.251    #被监控的主机地址，这个地址可以是 ip，
```

也可以是域名。

```
}

define host{
    use          linux-server
    host_name     mysql
    alias         ixdba-mysql
    address       192.168.12.237
}
```

```
define hostgroup{          #定义一个主机组
    hostgroup_name sa-servers #主机组名称，可以随意指定。
    alias          sa servers  #主机组别名
    members        web,mysql   #主机组成员，其中“web”、“mysql”就是上面定
义的 两个主机。
}
```

services.cfg 文件

此文件默认也不存在，需要手动创建，services.cfg 文件主要用于定义监控的服务和主机资源，例如监控 http 服务、ftp 服务、主机磁盘空间、主机系统负载等等。

```
define service{
    use          local-service #引用 local-service 服务的属性值，local-service 在
templates.cfg 文件中进行了定义。
    host_name    web          #指定要监控哪个主机上的服务，“web”在 hosts.cfg 文
件中进行了定义。
    service_description PING#对监控服务内容的描述，以供维护人员参考。
    check_command check_ping!100.0,20%!500.0,60% #指定检查的命令，
check_ping 命令在 commands.cfg 中定义，后跟两个参数，命令与参数间用!分割。

}
```

```
define service{
    use          local-service
    host_name     web
    service_description SSH
    check_command check_ssh # check_ssh 命令也在 commands.cfg 中定义。
}
```

```
define service{
use                local-service
host_name          web
service_description SSHD
check_command       check_tcp!22
}
```

contacts.cfg 文件

contacts.cfg 是一个定义联系人和联系人组的配置文件，当监控的主机或者服务出现故障，nagios 会通过指定的通知方式（邮件或者短信）将信息发给这里指定的联系人或者使用者。

```
define contact{
    contact_name nagiossystem    #联系人名称
    use          generic-contact # 引用 generic-contact 的属性信息，其中
    “generic-contact”在 templates.cfg 文件中进行定义
    alias        sa-system      #联系人别名
    email        #联系人的邮件地址
}
define contactgroup {
    contactgroup_name admins #联系人组名称
    alias            system administrator group #联系人组描述
    members          nagiossystem    #联系人组成员，其中“sasystem”就是上面
    定义的联系人
}
```

timeperiods.cfg 文件

此文件只要用于定义监控的时间段，下面是一个配置好的实例：

#下面是定义一个名为 24x7 的时间段，即监控所有时间段

```
define timeperiod{
    timeperiod_name 24x7
    alias           24 Hours A Day, 7 Days A Week
    sunday          00:00-24:00
}
```

```
monday    00:00-24:00
tuesday   00:00-24:00
wednesday 00:00-24:00
thursday  00:00-24:00
friday     00:00-24:00
saturday   00:00-24:00
}
```

#下面是定义一个名为 workhours 的时间段，即工作时间段。

```
define timeperiod{
    timeperiod_name workhours
    alias           Normal Work Hours
    monday          09:00-17:00
    tuesday          09:00-17:00
    wednesday        09:00-17:00
    thursday         09:00-17:00
    friday           09:00-17:00
}
```

监控 web 主机配置举例

创建 host 和 service

我这里=直接新建一个配置文件 liujm.cfg，内容如下：

```
[root@nagios objects]# cat liujm.cfg
##### liujm host #####
define host{
    use          linux-server
    host_name     route
    alias         route
    address       211.211.211.247
    icon_image    switch.gif
    statusmap_image switch.gd2
    2d_coords     100,200
    3d_coords     100,200,100
}
define host{
    use          linux-server
```

```

    host_name    apache-web
    alias        apache web
    address      www.liujm.com
    parents      route
    icon_image    web.gif
    statusmap_image    web.gd2
    2d_coords    100,300
    3d_coords    100,300,100
}
define host{
    use          linux-server
    host_name    apache-sshweb
    alias        apache ssh web
    address      www.liumm.com
    parents      route
    icon_image    web.gif
    statusmap_image    web.gd2
    2d_coords    200,300
    3d_coords    200,300,100
}
define host{
    use          linux-server
    host_name    mysql
    alias        apache mysql
    address      mysql.liujm.com
    parents      route
    icon_image    server.gif
    statusmap_image    server.gd2
    2d_coords    300,300
    3d_coords    300,300,100
}
##### liujm hostgroup ###
define hostgroup{
    hostgroup_name    liujm.com
    alias            liujm.com
    members          route,apache-web,apache-sshweb,mysql
}
##### liujm web service #####
define service{
    use          liujm-service    ; Name of service template to use
    host_name    apache-web,route
    service_description    PINGLIUJM

```

```

        check_command      check_ping!100.0,20%!500.0,60%
    }
define service{
    use      liujm-service ; Name of service template to use
    host_name      apache-web
    service_description      HTTPD
        check_command      check_http
        notifications_enabled      0
    }
define service{
    use      liujm-service ; Name of service template to use
    host_name      apache-sshweb
    service_description      SSHHTTPD
        check_command      check_http
        notifications_enabled      0
    }

define service{
    use      liujm-service ; Name of service template to use
    host_name      mysql
    service_description      MYSQLD
        check_command      check_tcp!3306
        notifications_enabled      0
    }

##### liujm servicegroup #####
define servicegroup{
    servicegroup_name apache-webservice
    alias apache-webservice
    members      apache-web,PINGLIUJM,apache-
web,HTTPD,apache-sshweb,SSHHTTPD,mysql,MYSQLD
}

```

我这里把 host 和 service 写到一个文件里面了，我觉得这样也比较好理解

修改 nagios.cfg

注释 localhost 这一行：

#cfg_file=/usr/local/nagios/etc/objects/localhost.cfg

添加 liujm 这一行：

cfg_file=/usr/local/nagios/etc/objects/liujm.cfg

修改联系人信息

```
define contact{
    contact_name    nagiosadmin
    use             generic-contact
    alias           Nagios Admin
    email           liujm8936@163.com
}
```

改为你自己的 email，这里只有 163 的能收到邮件，其他的像 qq、gmail、hotmail 测试过都无法收到邮件，能通过 mail 发邮件的前提是启动了邮件服务器，不需要做多少配置，直接启动就行，邮件服务器自行研究接着就是检查配置文件

修改 templates.cfg

这里就根据 linux-server 及 linux-service 来创建 liujm-server 及 liujm-service

```
define host{
    name                liujm-server      ; The name of this host
    use                 generic-host      ; This is a generic host
    check_period        24x7              ; By default, the host is checked 24x7
    check_interval      2                  ; Actively check the host every 2 minutes
    retry_interval      1                  ; Schedule a retry 1 minute after a failed check
    max_check_attempts  1                  ; Check the host 1 time before declaring it unreachable
    check_command        check-host-alive ; Define the check command to use
    notification_period  24x7              ; Lin
    ; Note
    ; the
    notification_interval 5                ; Rese
    notification_options  d,u,r            ; Only
    contact_groups        admins            ; Noti
    register              0                 ; DON
    #
    action_url            /pnp4nagios/index.php
}
```

```
define service{
    name                liujm-service           ; The name of the service
    use                  generic-service         ; Inherit from generic-service
    max_check_attempts  4                       ; Re-check attempts
    normal_check_interval 5                     ; Check interval
    retry_check_interval 1                       ; Re-check interval
    register             0                       ; DONT register
    #action_url           /pnp4nagios/index.php/graph?host=
}
```

在这里可以根据自己的需求来修改通知级别，通知人及检查频率等

测试配置文件

```
[root@nagiosobjects]#
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
.....
Total Warnings: 0
Total Errors: 0
```

Things look okay - No serious problems were detected during the pre-flight check

如果有报错就更加报错来修改配置文件

最后就是重启 nagios：service nagios restart

ok.来看一下界面效果

至于 map 的三维拓扑图，比较难理解，可以根据田朝阳老师那个拓扑图来划自己的拓扑图

211.211.211.200/nagios/

公开课 安全 文学杂志 网购团购 酒店车票 电脑技术 linux技术 英语考试 旅游 自学考试 开源权威 大家网-全球顶级教育... Google 翻译 Google 百度技术沙龙 安全平台 云计算

Nagios®

常用

- 主页
- 文档(英文)
- 中文说明文档

当前状态

- 总览
- 概览图
- 主机
- 服务
- 主机组
- 汇总
- 表格
- 服务组
- 汇总
- 表格
- 问题故障
- 服务 (未设置)
- 主机 (未设置)
- 网络整体

快速查找:

全部状态概要

最近更新: 2014年1月13日(周一) 16:53:56
更新间隔90秒一次
Nagios® Core™ 3.2.3 - www.nagios.org
登陆帐户: nagiosadmin
Nagios® - www.nagios.org

网络故障

网络故障次数:

主机的状态

宕机状态: 0 (DOWN)	不可达状态: 0 (UNREACHABLE)	正常状态: 4 (UP)	未决状态: 0 (PENDING)
----------------	------------------------	--------------	-------------------

服务的状态

紧急状态: 0 (CRITICAL)	告警状态: 0 (WARNING)	未知状态: 0 (UNKNOWN)	正常状态: 5 (OK)	未决状态: 0 (PENDING)
--------------------	-------------------	-------------------	--------------	-------------------

性能状态

服务检查执行时间	0.01 / 4.25 / 2.097 秒
服务检查反应时间	0.63 / 1.55 / 1.030 秒
主机检查执行时间	4.01 / 5.38 / 4.463 秒
主机检查反应时间	1.06 / 1.89 / 1.373 秒
已检查的活动[主机/服务]	4 / 5
被动[主机/服务]检查	0 / 0

网络健康状态

主机的健康状态:

服务的健康状态:

报告

- 可用性
- 趋势
- 报警
- 历史

主机	别名	状态	最近检查时间	持续时间	状态信息
apache-sshweb	apache ssh web	运行	2014-01-13 16:37:56	0日 2时 1分 58秒	PING OK - Packet loss = 0%, RTA = 0.07 ms
apache-web	apache web	运行	2014-01-13 16:39:36	0日 1时 58分 43秒	PING OK - Packet loss = 0%, RTA = 0.08 ms
mysql	apache mysql	运行	2014-01-13 16:35:16	0日 1时 36分 39秒	PING OK - Packet loss = 0%, RTA = 0.06 ms
route	route	运行	2014-01-13 16:37:26	0日 1时 33分 42秒	PING OK - Packet loss = 0%, RTA = 0.29 ms

当前的网络状态

最近更新: 2014年1月13日(周一) 16:56:34
更新间隔90秒一次
Nagios® Core™ 3.2.3 - www.nagios.org
登陆帐户: nagiosadmin
Nagios® - www.nagios.org

[所有主机的历史信息](#)
[所有主机的通知历史信息](#)
[所有主机的主机详细状态](#)

主机状态汇总

运行	宕机	不可达	未决
4	0	0	0
所有故障		所有类型	
0		4	

服务状态汇总



正常(OK)	告警	未知	紧急	未决
5	0	0	0	0
所有故障		所有类型		
0		5		

所有主机的正常状态

主机名	别名	服务	状态	最近检查时间	持续时间	尝试次数	状态信息
apache-sshweb	apache ssh web	SSHHTTPD	正常(OK)	2014-01-13 15:34:01	0日 2时 2分 34秒	1/3	HTTP OK: HTTP/1.1 301 Moved Permanently - 436 bytes in 0.005 second res
apache-web	apache web	HTTPD	正常(OK)	2014-01-13 16:42:57	0日 0时 18分 38秒	1/3	HTTP OK: HTTP/1.0 200 OK - 46773 bytes in 2.172 second response time
		PINGUJIM	正常(OK)	2014-01-13 16:52:17	0日 1时 59分 19秒	1/3	PING OK - Packet loss = 0%, RTA = 0.08 ms
mysql	apache mysql	MYSQLD	正常(OK)	2014-01-13 16:49:21	0日 0时 17分 14秒	1/3	TCP OK - 0.001 second response time on port 3306
route	route	PINGUJIM	正常(OK)	2014-01-13 16:53:17	0日 1时 34分 18秒	1/3	PING OK - Packet loss = 0%, RTA = 0.33 ms

我们关掉一台服务器来测试一下是否能收到邮件
服务关闭后会有报错邮件，恢复后会有 ok 邮件发来

今日(17)

<input type="checkbox"/>	nagios		** PROBLEM Service Alert: apache mysql/MYSQLD is CRITICAL **
<input type="checkbox"/>	nagios		** PROBLEM Service Alert: apache web/HTTPD is CRITICAL **
<input type="checkbox"/>	nagios		** RECOVERY Service Alert: apache web/HTTPD is OK **
 <input type="checkbox"/>	nagios		** RECOVERY Service Alert: apache ssh web/SSHHTTPD is OK **

来看看邮件内容

```
发件人 : nagios<nagios@domain.tld> +
收件人 : liujm8936<liujm8936@163.com> +
时 间 : 2014年01月13日 20:03 (星期一)
```

***** Nagios *****

Notification Type: PROBLEM

Service: MYSQLD

Host: apache mysql

Address: mysql.liujm.com

State: CRITICAL

Date/Time: Mon Jan 13 20:03:29 CST 2014

Additional Info:

Connection refused

好 ok , nagios 配置介绍到此为止

感知和处理状态抖动

摘自田朝阳老师的文档

这个有点不好理解，因为我的虚拟机环境老不稳定，出现抖动的现象，所以这里摘录介绍一下

介绍

Nagios 支持可选的发现主机与服务抖动的功能。当服务与主机状态改变过于频繁时会产生抖动，其结果产生了故障与恢复的通知风暴。抖动可能是由于配置的问题(如门限过低)、有毛病的服务或是真实的网络问题。

感知抖动是如何工作的？

在此之前，我想说的是抖动的感知有点难实现。如何精确地确定网络与主机的什么叫做“过分频繁”？当我第一次考虑对感知抖动的实现时，我试图找到发现抖动本该或应该或是如何做的信息，但是一无所获，所以决定用一种对我言是一种合理的方式来解决它...

每当 Nagios 对主机与服务进行检测，它将查看该主机或服务是否已开始或停止抖动，条件有几条：

1. 保存好的对主机与服务的检测结果至少 21 个；
2. 分析历史检测结果确定状态变换发生了；
3. 用状态转换判定主机与服务状态值改变的百分比；
4. 比较这个百分比是否越过了设定的抖动门限的最低值与最高值；

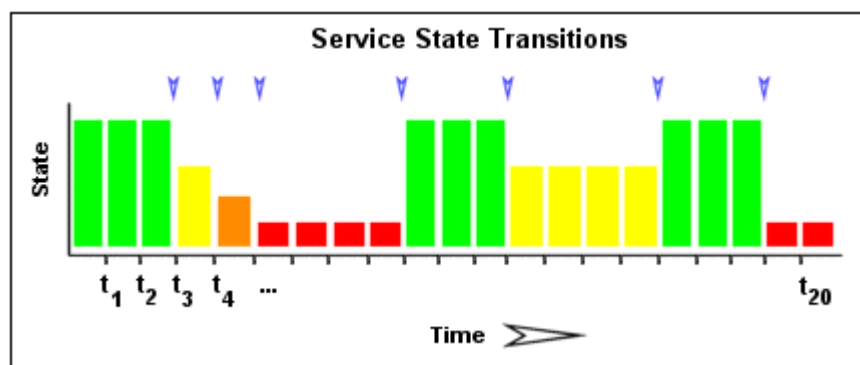
认定主机与服务的抖动开始是它的状态改变率首次高于抖动门限的高限。

认定主机与服务的抖动结束是它的状态改变率低于抖动门限低限(前提是它已经处于抖动状态)。

例子

下面用个服务来更详细地说明如何感知抖动的...

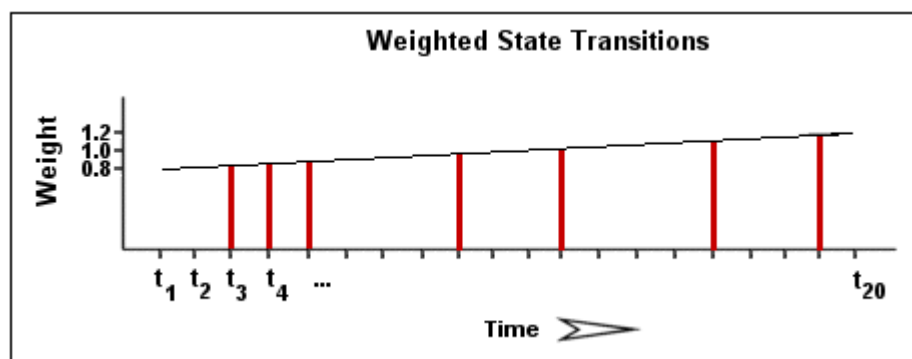
下图给出了最近 21 次检测结果的按时序的历史状态。正常(OK)态标记为绿色，告警(WARNING)态为黄色，紧急(CRITICAL)为红色，未知(UNKNOWN)态为橙色。



对历史检测结果的检查决定了哪个时间里有状态变换发生，状态变换发生于存档状态与其前一次状态不同的时刻。由于用数组保存了最近 21 次检测结果，因而可以知道最多可能会产生 20 次变化。在本例中有 7 次状态变化，在图中上方用蓝色箭头示意出来。

感知状态抖动逻辑使用状态变换来判定整体服务的状态变化率，用于度量服务变化或更改的频度。没有发生过状态变化的变化率为 0%，而每次都变化的状态变化率是 100%。服务的状态变化应该在此之间变化。

当计算服务的状态变化率时，感知抖动的算法将会给对近期变化更多权重，旧的变化权重低。特别地，将近期变化给出 50% 的权重。图中示出对指定服务使用了近期变化有更多权重来计算整体变化率的情况。



利用图示结果，计算一下服务的状态变化率。共有 7 次状态变化(分别位于 t_3 、 t_4 、 t_5 、 t_9 、 t_{12} 、 t_{16} 和 t_{19})。没有任何状态变化权重时结果将会是 35%：

$(7 \text{ 次查出的状态变化} / 20 \text{ 次最大状态变化次数}) * 100\% = 35\%$

因为感知抖动的检测逻辑使用近期变化更大的权重，所以该例中实际计算时变化率会低于 35%。假定这个加权后的变化率是 31%...

使用计算后的服务的状态变化率(31%)来比对抖动门限将会发生：

1. 如果先前没有发生抖动且 31% 等于或超出了抖动门限的高限，Nagios 将判定服务开始抖动；
2. 如果服务先前处于抖动而且 31% 低于抖动门限的低限，Nagios 将判定服务停止抖动；

如果两个都没有发生，感知抖动逻辑将不会对服务做任何动作，因为它既没有变为抖动也或许正在抖动。

服务的抖动感知

每当 Nagios 对服务进行检测时就会来做检查看它是否抖动(不管是自主检测还是强制检测)。

服务的抖动感知机制见上面例子中的描述说明。

主机的抖动感知

主机的抖动感知与服务的相似，只是一个重要的不同：Nagios 将在如下情形时尝试对其进行抖动中的检测：

- 1. 主机检测时(自主检测或强制检测时都会做)
- 2. 有时与主机绑定的服务被检测时。更特殊地，当至少 x 次的抖动感知做过时，此处的 x 等于全部与主机绑定服务的平均检测间隔时间。

为什么要这样？由于最少的两次抖动检查次数间的时间最少是等于服务检测间隔时间。然而可能对主机的监控并非基于规格化的间隔，所以对主机的抖动检测可能对它的抖动感知的检查不是主机检测的间隔时间。同样地，要知道对服务的检查会叠加到主机的抖动感知检测上。毕竟服务是主机上的属性而不是别的...在种种检查速率相比之下，这个是最好的方式来多次地对主机进行抖动检查，所以你也得如此。

抖动检测门限

Nagios 在抖动感知逻辑中用若干个值来判定状态变化率。既有主机的也有服务的，配置里面有全局的门限高限和低限也有专门针对主机的或是服务的门限。Nagios 将在没有指定专门主机的或服务的门限时使用全局的门限值。

下表给出了全局的、专给主机的和专给服务的的门限值的控制变量。

表 8.24.

对象类型	全局变量	对象专属的变量
主机	<code>low_host_flap_threshold</code> <code>high_host_flap_threshold</code>	<code>low_flap_threshold</code> <code>high_flap_threshold</code>
服务	<code>low_service_flap_threshold</code> <code>high_service_flap_threshold</code>	<code>low_flap_threshold</code> <code>high_flap_threshold</code>

给抖动检测所用的状态

通常 Nagios 将记录下针对主机和服务的最后 21 次检测结果用于抖动感知逻辑，而不管全部的检查结果。

提示



在抖动感知逻辑中可以排除主机或服务的某种状态，在主机或服务定义中使用 `flap_detection_options` 域来指明哪些状态(如运行(UP)、宕机(DOWN)、正常(OK)、紧急(CRITICAL)等)要进入抖动检查。如果没有设置它，全部的主机与服务的状态都会被用于抖动

感知逻辑之中。

抖动处理

当服务或主机首次发现处于抖动时，Nagios 将会：

1. 记录下服务与主机正在抖动的信息；
2. 给主机与服务增加一个非持续性的注释以说明它正在抖动；
3. 给服务与主机相关的联系人发送一个"开始抖动"的通知；
4. 压制主机与服务其他通知(这个在通知逻辑中有一个过滤)；

当服务或主机停止抖动时，Nagios 将会：

1. 记录下主机与服务停止了抖动；
2. 删除最初的给主机与服务增加的开始抖动的注释；
3. 给主机与服务相关的联系人送出一个"抖动停止"的通知；
4. 停止阻塞该主机与服务的通知(通知转回到正常的通知逻辑)。

使能抖动感知功能

在 Nagios 打开抖动感知功能，需要如下设置：

1. 将 `enable_flap_detection` 域设置为 1；
2. 在主机与服务对象定义中的 `flap_detection_enabled` 域设置为 1；

如果想关闭全局的抖动感知功能，将 `enable_flap_detection` 域设置为 0；

如果只想关闭一部分主机与服务的抖动检查，使用在主机与服务对象定义里 **`flap_detection_enabled`** 域来控制它；

利用 NRPE 扩展 Nagios 功能

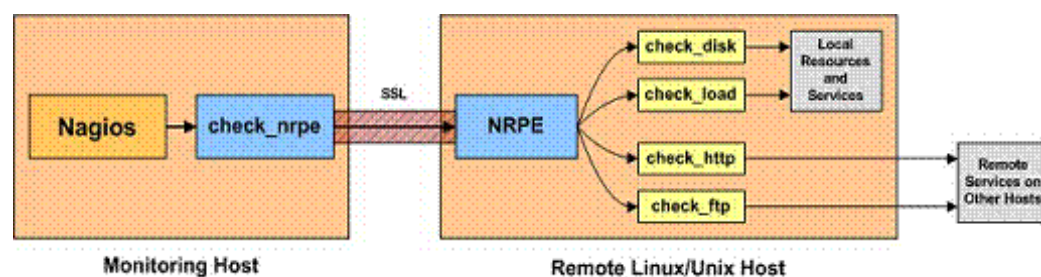
这里介绍安装直接参照官方文档，在 nrpe 压缩包里面的 docs 目录下有

NRPE 介绍

NRPE 是 Nagios 的一个功能扩展，它可在远程 Linux 和 UNIX 主机上执行插件程序。通过在远程服务器上安装 NRPE 构件及 Nagios 插件程序来向 Nagios 监控平台提供该服务器的一些本地情况，如 CPU 负载、内存使用、硬盘使用，服务等。这里将 Nagios 监控平台称为 Nagios 服务器端，而将远程被监控的服务器称为 Nagios 客户端。

下图为 NRPE 构件监控远程主机本地信息的运行原理：

监控远程主机原理图



NRPE 组成部分与检测类型

NRPE 总共由两部分组成：

check_nrpe 插件，位于监控主机上

NRPE daemon, 运行在远程被监控的 Linux 主机上

当监控远程 Linux/UNIX 主机服务或资源时，工作流程如下：

- nagios 会运行 check_nrpe 这个插件，并且会告诉它需要检查什么；
- check_nrpe 插件会连接到远程的 NRPE daemon，所用的方式是 SSL；
- NRPE daemon 会运行相应的 Nagios 插件来执行检查动作；
- NPRE daemon 将检查的结果返回给 check_nrpe 插件，插件将其递交给 Nagios 做处理。

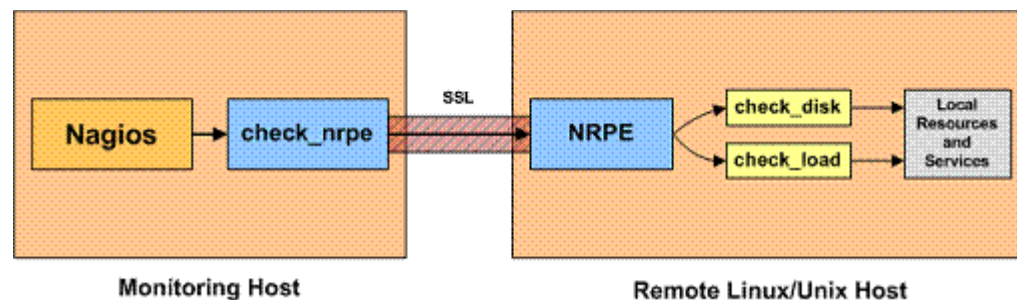
NRPE daemon 需要 Nagios 插件安装在远程的 Linux 主机上，否则 daemon 不能做任何监控。

NRPE 的检测类型分为两种：

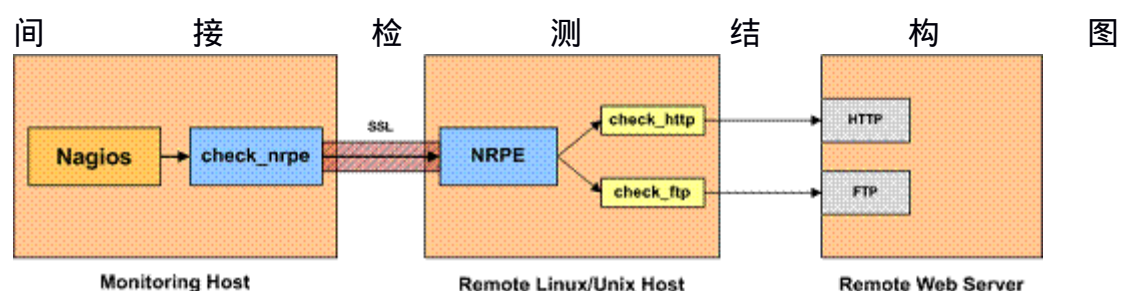
直接检测：检测的对象是运行 NRPE 的那台 Linux 主机的本地资源，原理如下：

直接使用 NRPE 插件监控远程 Linux/UNIX 主机的本地或者私有资源；如 CPU 负载、内存使用、SWAP 空间使用、硬盘等运行状况。

直接检测结构图



间接检测：当运行 Nagios 的监控主机无法访问到某台被监控主机，但是运行 NRPE 的机器可以访问得到的时候，运行 NRPE 的主机就充当一个中间代理，将监控请求发送到被监控对象上。



在 Linux 客户端安装 NRPE

安装 Nagios 插件 nagios-plugin

添加 nagios 用户名，且不允许 nagios 用户登录，此用户用于与 Nagios 服务器通信所用。

客户端安装 **nagios-plugin**

```
wget
wget
```

```
# useradd -s /sbin/nologin nagios
```

```
# tar -zxvf nagios-plugins-1.4.14.tar.gz
# cd nagios-plugins-1.4.14
# ./configure
# make && make install
```

安装 NRPE

在 Linux 客户端安装 nrpe 程序包，根据编译提示向导完成安装操作。在安装的过程中会看到 NRPE 的端口为 5666，且可通过 Xinetd 服务来控制 nrpe 进程，具体实现步骤如下：

客户端安装 **NRPE**

```
#tar zxvf tar zxvf nrpe-2.12.tar.gz
# cd nrpe-2.12
# ./configure
#make all
#make install-plugin
#make install-daemon
#make install-daemon-config
#make install-xinetd
#chown -R nagios:nagios /usr/local/nagios/
```

配置 NRPE

定义被监控的 Linux 主机的对象，监控此主机的 CPU 负载、登录用户数、磁盘分区、进程、swap 使用情况等，编辑/usr/local/nagios/etc/nrpe.cfg 文件，内容如下示例：

NRPE 配置与设定

```
command[check_users]=/usr/local/nagios/libexec/check_users -w 5 -c 10
command[check_load]=/usr/local/nagios/libexec/check_load -w 15,10,5 -c 30,25,20
command[check_sda2]=/usr/local/nagios/libexec/check_disk -w 20% -c 10% -p /dev/sda2
command[check_zombie_procs]=/usr/local/nagios/libexec/check_procs -w 5 -c 10 -s Z
command[check_total_procs]=/usr/local/nagios/libexec/check_procs -w 150 -c 200
command[check_swap_1]=/usr/local/nagios/libexec/check_swap -w 20 -c 10
```

定义 Xinetd 服务支持 nrpe

这里只需要修改 `only_from` 项，增加 Nagios 服务的地址即可，这样一来服务器与客户端就可进行 nrpe 会话，监控到 Linux 客户端相关信息，被监控端也更加容易维护管理，见下图：

```
#vim /etc/xinetd.d/nrpe
```

定义 Xinetd 服务支持 nrpe

```
[root@liujm nrpe-2.15]# cat /etc/xinetd.d/nrpe
# default: on
# description: NRPE (Nagios Remote Plugin Executor)
service nrpe
{
    flags             = REUSE
    socket_type       = stream
    port              = 5666
    wait              = no
    user              = nagios
    group             = nagios
    server             = /usr/local/nagios/bin/nrpe
    server_args       = -c /usr/local/nagios/etc/nrpe.cfg --inetd
    log_on_failure    += USERID
    disable           = no
    only_from         = 127.0.0.1 211.211.211.200
}
```

定义服务端口

在 Linux 客户端"/etc/services" 文件增加一行

```
nrpe          5666/tcp          #Naigos_Client
```

测试 NRPE

由于 NRPE 相应的插件已经安装成功，这里使用 `check_nrpe` 命令来验证是否 nrpe 是否正常运行，如果执行以下命令能够显示 NRPE 的具体版本信息，则表示 nrpe 运行正常，加载重启 xinetd 服务即可。

NRPE 功能测试

```
#/usr/local/nagios/libexec/check_nrpe -H localhost
```

```
NRPE v2.12
```

```
#/etc/init.d/xinetd restart
```

```
[root@liujm nrpe-2.15]# /usr/local/nagios/libexec/check_nrpe -H localhost
CHECK_NRPE: Error - Could not complete SSL handshake.
```

原因是 xinetd 的 nrpe 没有加入本机 ip

在服务器端安装 NRPE

服务器安装 NRPE

```
# tar zxvf nrpe-2.12.tar.gz
# cd nrpe-2.12
# ./configure && make all
# make install-plugin
```

测试是否安装正确

```
[root@nagios nrpe-2.15]# /usr/local/nagios/libexec/check_nrpe -H 211.211.211.180
NRPE v2.15
[root@nagios nrpe-2.15]#
```

由于在 Nagios 命令定义文件 `commands.cfg` 没有 `check_nrpe` 命令，因此需要对此文件进行修改与定义，配置细节如下图：

在 **`commands.cfg`** 文件中增加 **NRPE** 配置

```
define command{
    command_name check_nrpe
    command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
}
```

定义被监控主机

在被监控或远程主机上增加 `check_nrpe` 的相关配置，由于 `hosts.cfg` 已定义了相应的主机，所以这里编辑文件 Nagios 服务器上的 `services.cfg` 文件即可

增加 NRPE 指令

```
define service{
    use                liujm-service
    host_name          apache-web
    service_description LOAD
    check_command       check_nrpe!check_load
}
define service{
    use                liujm-service
    host_name          apache-web
    service_description LOGIN_USER
    check_command       check_nrpe!check_users
}
define service{
    use                liujm-service
    host_name          apache-web
    service_description DISK
    check_command       check_nrpe!check_sda2
}
define service{
    use                liujm-service
    host_name          apache-web
    service_description TOTAL_PROCS
    check_command       check_nrpe!check_total_procs
}
```

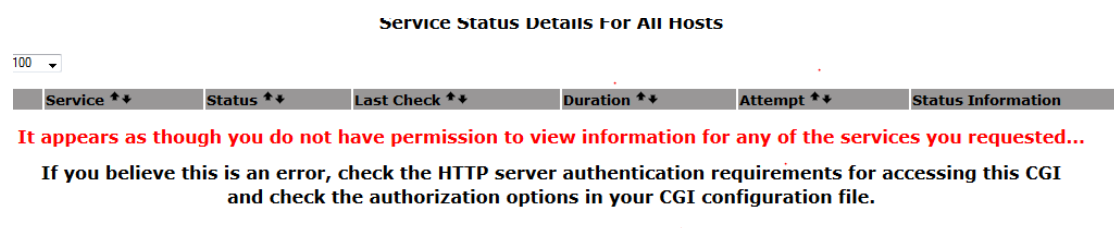
查看配置文件是否正确

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

重新加载配置文件

```
# /etc/init.d/nagios reload
```

遇到过的报错



当时还出现一个奇怪的错误，有时能显示 liujm.cfg 中定义的主机，有时不能显示

原因是 cfg 配置文件有错误，可能是 host_name alias 两者意义理解错误，在下面的 members 中引用的都是 alias 的名字，当然还有下面的原因

新增加的机器有时能显示状态，刷新一会儿又不能显示，只显示本机的状态，why?

原因是开启了两个 nagios 进程

```
[root@nagios etc]# ps aux |grep nagios
nagios 6682 0.0 1.9 22932 4712 ? Ss 18:58 0:03 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios 6688 0.0 0.2 22292 568 ? S 18:58 0:01 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios 21684 0.0 0.3 11468 824 ? S 23:00 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 21685 0.0 0.3 11468 820 ? S 23:00 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 21686 0.0 0.3 11468 820 ? S 23:00 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 21687 0.0 0.3 11468 820 ? S 23:00 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 21798 0.0 0.4 22904 1048 ? Ss 08:35 0:04 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios 21804 0.0 0.0 22292 132 ? S 08:35 0:05 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
root 21941 0.0 0.3 103300 836 pts/0 S+ 23:02 0:00 grep nagios
nagios 29076 0.0 0.1 11468 444 ? S 09:07 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 29077 0.0 0.1 11468 452 ? S 09:07 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 29078 0.0 0.1 11468 444 ? S 09:07 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 29079 0.0 0.1 11468 448 ? S 09:07 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
```

解决：

```
[root@nagios etc]# killall nagios
[root@nagios etc]#
[root@nagios etc]#
[root@nagios etc]# ps aux |grep nagios
root 21950 0.0 0.3 103300 840 pts/0 S+ 23:02 0:00 grep nagios
[root@nagios etc]#
[root@nagios etc]#
[root@nagios etc]#
[root@nagios etc]# service nagios start
nagios dead but subsys locked
Starting nagios:
[ OK ]
[root@nagios etc]#
[root@nagios etc]#
[root@nagios etc]#
[root@nagios etc]# ps aux |grep nagios
nagios 22001 0.2 0.6 22932 1648 ? Ss 23:02 0:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios 22003 0.0 0.2 10692 712 ? S 23:02 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 22004 0.0 0.2 10692 716 ? S 23:02 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 22005 0.0 0.2 10692 708 ? S 23:02 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 22006 0.0 0.3 11468 756 ? S 23:02 0:00 /usr/local/nagios/bin/nagios --worker /usr/local/nagios/var/rw/nagios.qh
nagios 22007 0.0 0.2 22292 568 ? S 23:02 0:00 /usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios 22008 0.0 0.4 113692 1068 ? S 23:02 0:00 /usr/local/nagios/libexec/check_ping -H www.liumm.com -W 3000.0,80% -C 5000.0,100% -
root 22010 0.0 0.3 103300 836 pts/0 S+ 23:02 0:00 grep nagios
```

nagios dead but subsys locked!

出现这个问题是修改了 nagios 配置文件，然后 reload nagios 服务，就发现 nagios 服务没有启动，也无法查询到状态，也无法启动或者关闭 nagios 服务

解决办法：

原因是配置文件错误，导致 nagios 锁定文件未删除,执行如下命令：

rm -rf /var/lock/subsys/nagios

/var/lock/subsys 目录的作用的说明:

很多程序需要判断是否当前已经有一个实例在运行，这个目录就是让程序判断是否有实例运行的标志，比如说 xinetd，如果存在这个文件，表示已经有 xinetd 在运行了，否则就是没有，当然程序里面还要有相应的判断措施来真正确定是否有实例在运行。通常与该目录配套的还有/var/run 目录，用来存放对应实例的 PID，如果你写脚本的话，会发现这 2 个目录结合起来可以很方便的判断出许多服务是否在运行，运行的相关信息等等。

实际上，判断是否上锁就是判断这个文件，所以文件存在与否也就隐含了是否上锁。而这个目录的内容并不能表示一定上锁了，因为很多服务在启动脚本里用 touch 来创建这个加锁文件，在系统结束时该脚本负责清除锁，这本身就不可靠（比如意外失败导致锁文件仍然存在），我在脚本里一般是结合 PID 文件（如果有 PID 文件的话），从 PID 文件里得到该实例的 PID，然后用 ps

测试是否存在该 PID，从而判断是否真正有这个实例在运行，更加稳妥的方法是用进程 通讯了，不过这样的话单单靠脚本就做不到。

nagios 启动和关闭会创建和删除锁

```
prog="nagios"
config="${prefix}/etc/nagios.cfg"
pidfile="${prefix}/var/nagios.lock"
user="nagios"
group="nagios"
checkconfig="true"
ramdiskdir="/var/nagios/ramcache"
use_precached_objects="false"

test -e /etc/sysconfig/$prog && . /etc/sysconfig/$prog

lockfile=/var/lock/subsys/$prog
USE_RAMDISK=${USE_RAMDISK:-0}
```

```
echo
test $retval -eq 0 && touch $lockfile
return $retval
}

stop() {
echo -n "Stopping $prog: "
killproc -p ${pidfile} -d 10 $exec
retval=$?
echo
test $retval -eq 0 && rm -f $lockfile
return $retval
}
```

所以修改了配置文件避免出现这样的问题可以直接 restart

cacti 安装与配置

cacti 简介

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。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。

cacti 是用 php 语言实现的一个软件，它的主要功能是用 snmp 服务获取数据，然后用 rrdtool 储存和更新数据，当用户需要查看数据的时候用 rrdtool 生成图表

呈现给用户。因此，snmp 和 rrdtool 是 cacti 的关键。Snmp 关系着数据的收集，rrdtool 关系着数据存储和图表的生成。

Mysql 配合 PHP 程序存储一些变量数据并对变量数据进行调用，如：主机名、主机 ip、snmp 团体名、端口号、模板信息等变量。

snmp 抓到数据不是存储在 mysql 中，而是存在 rrdtool 生成的 rrd 文件中（在 cacti 根目录的 rra 文件夹下）。rrdtool 对数据的更新和存储就是对 rrd 文件的处理，rrd 文件是大小固定的档案文件（Round Robin Archive），它能够存储的数据笔数在创建时就已经定义。关于 RRDTool 的知识请参阅 RRDTool 教学。

nagios、cacti 整合必要

Nagios 监控服务器的状态很强大，并且报警功能也很不错，但对像流量这样的持续数据的展现能力却比较弱，虽然有类似 PNP 这样的插件可以对数据进行存储展现，但在看习惯 Cacti 的图后，PNP 这样枯燥的图就很难接受了，因此以下大体描述了一下如何整合 Nagios 和 Cacti，将点状态和线状态都清晰的展现。

cacti 本身的 thold 的插件也可以提供报警功能，并且也可以支持 msn,fetion,email,并且能够自己创建模版，例如可以定义流量比如在 10M-100M 之间是正常，超过这个区间就进行报警等等，这个是非常不错的。那么为什么要引进 nagios 呢，可以这么讲 cacti 所获取的信息都是通过 snmp 协议进行的，我们通过 snmp 可以获取什么呢，磁盘信息，流量信息，负载信息等等，那么我们需要探测一个 http 服务是否正常怎么办，Nagios 就是最好的解决办法，并且 nagios 的检测插件非常丰富，可以直接拿来简单配置一下 command 就可以用了。因此在监控报警方面，cacti 和 nagios 结合起来是比较好的选择。说结合其实只是表面的，cacti 和 nagios 还是以他们原来的方式进行工作，ndo 负责将 nagios 收集的数据存在数据库中，cacti 的 npc 插件会从数据库中取数据在 cacti 的 npc 标签中来展现。是否需要结合其实按照自己的需求来吧，对于不同的系统管理员可能关心的东西不一样。

cacti 环境依赖包安装

安装都参考官方文档

Required Packages for RPM-based Operating Systems

- httpd
- php
- php-mysql
- php-snmp

-
- php-ldap (when using [LDAP authentication](#))
 - php-xml
 - mysql
 - mysql-server
 - net-snmp (depending on the distro, net-snmp-utils may be required)
 - crond (cron, cronie or the like)

For installation of official patches, you will require some utilities

- wget
- patch

安装上面所需的软件

```
yum install httpd php php-snmp php-mysql mysql mysql-server net-snmp net-snmp-devel
```

配置 PHP

确保以下被模块加载：安装完后默认都已加载

- mysql (For configuration, see note below)
- SNMP (For configuration, see note below)
- XML
- Session
- Sockets
- LDAP (Required only when using LDAP authentication)
- GD (Required only for some Plugins)

You may run the following command to get the list of all available PHP modules

```
php -m
```

Please verify, that the modules are installed and configured correctly. There are several ways to do so, please consult

PHP configuration instructions¹ for a complete description.

We will continue using the most recommended way of configuring php extension modules. Please find the file

/etc/php.ini and make the following changes to it:

extension_dir = /etc/php.d

This will enable PHP to find more configuration directives in that very directory.

Other distros point to

/usr/lib/php/modules instead. In each case, you should locate e.g. mysql.so in that directory.

Activate the MySQL extension via /etc/php.d/mysql.ini

; Enable mysql extension module

extension=mysql.so

Activate the SNMP extension via /etc/php.d/snmp.ini

; Enable snmp extension module

extension=snmp.so

If using PHP 4.3.5 or less include the following line. If using 4.3.6 or greater, you should remove this line if present.

session.save_path=/tmp

If you want to allow template importing, uncomment the following line:

file_uploads = On

配置 MySQL

Set a password for the root user

mysqladmin --user=root password somepassword

mysqladmin --user=root --password reload

修改 MySQL 的最大连接数

vi /etc/my.cnf

添加以下行

[mysqld]

set-variable=max_connections=1000

set-variable=max_user_connections=500

set-variable=wait_timeout=200

max_connections 设置最大连接数为 1000

max_user_connections 设置每用户最大连接数为 500

wait_timeout 表示 200 秒后将关闭空闲 (IDLE) 的连接, 但是对正在工作的连接不影响。

保存退出, 并重新启动 MySQL

重新启动 MySQL 后使用下面的命令查看修改是否成功

mysqladmin -uroot -p variables

Password:

//可以看到以下项说明修改成功

```
| max_connections      | 1000
| max_user_connections | 500
| wait_timeout         | 200
```

安装 cacti

其实 cacti 可以理解成是 php 语言写的网站代码，下面的安装就是配置数据库，php，httpd 网站环境，详细可参考官网手册

```
wget
```

```
tar xvf cacti-0.8.8b.tar.gz
```

```
cd cacti-0.8.8b
```

Create the MySQL database:

```
mysqladmin --user=root create cacti
```

Import the default cacti database:

```
mysql -uroot -p cacti < cacti.sql
```

Optional: Create a MySQL username and password for Cacti.

```
mysql --user=root -p
```

```
mysql> GRANT ALL ON cacti.* TO cactiuser@localhost IDENTIFIED BY
'somepassword';创建用户
```

```
mysql> flush privileges;
```

Edit include/config.php and specify the database type, name, host, user and password for your Cacti configuration.

```
$database_type = "mysql";
```

```
$database_default = "cacti";
```

```
$database_hostname = "localhost";
```

```
$database_username = "cactiuser";
```

```
$database_password = "password";
```

```
vim include/global.php
```

```
$database_type = "mysql";
```

```
$database_default = "cacti";
```

```
$database_hostname = "localhost";
```

```
$database_username = "cactiuser";
```

```
$database_password = "password";
```

添加定时任务：crontab -e

```
*/5 * * * * cactiuser php /var/www/html/cacti/poller.php > /dev/null 2>&1
```

拷贝 **cacti** 代码到 **/var/www/html**

```
mv /cacti-0.8.8b /var/www/html/cacti
```

修改权限：`chown -R nagios:nagcmd /var/www/html/cacti`

```
chmod 775 /var/www/html/rrd /var/www/html/resource
```

```
usermod -g nagcmd cactiuser
```

如果权限不对，cactiuser 没有写入权限，会导致无法出图

配置 httpd，在 **/etc/httpd/conf.d/** 中加入 **cacti.conf**

```
[root@nagios-cacti cacti]# cat /etc/httpd/conf.d/cacti.conf
```

```
Alias /cacti "/var/www/html/cacti"
```

```
<Directory "/var/www/html/cacti">
```

```
Options ExecCGI
```

```
AllowOverride None
```

```
Order allow,deny
```

```
Allow from all
```

```
AuthName "Nagios Access"
```

```
AuthType Basic
```

```
AuthUserFile /usr/local/nagios/etc/htpasswd.users
```

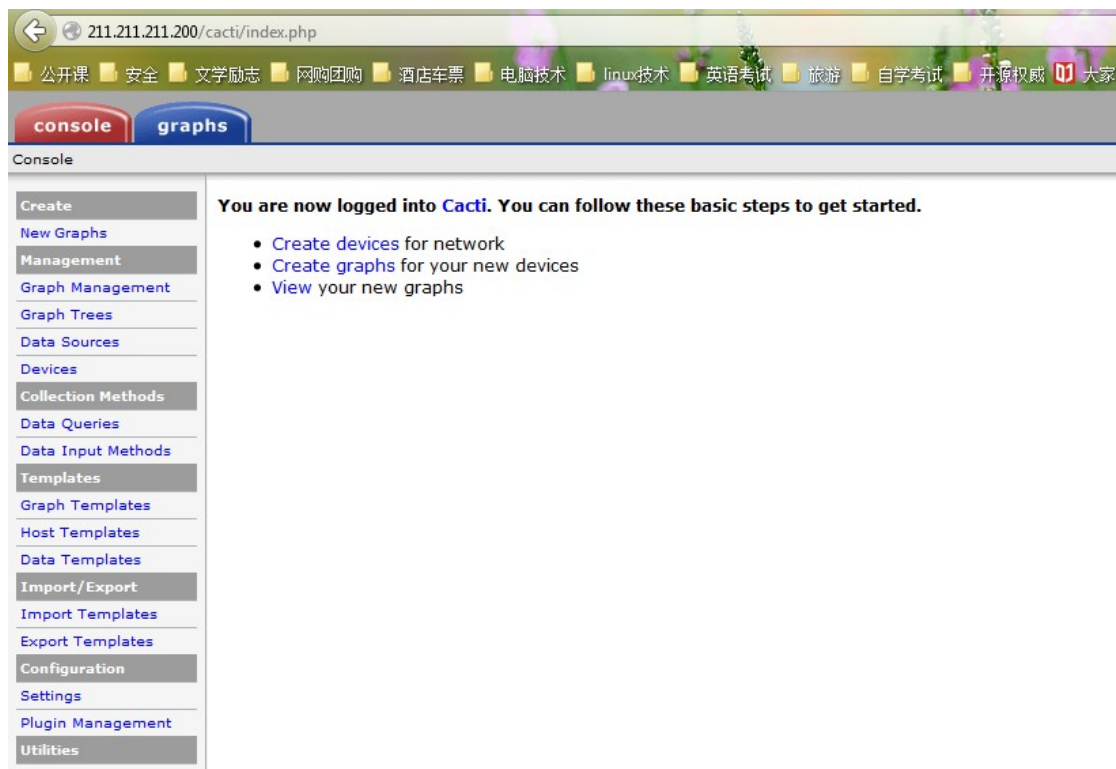
```
Require valid-user
```

```
</Directory>
```

重启 httpd 服务，通过网页访问 211.211.211.200/cacti

网页安装默认提示安装就好了

网页帐号密码都是 admin，登录会要求修改密码



到此 cacti 安装完成

sping 安装配置

spine 是一个基于 C 语言的，非常快速的轮询引擎。它是默认的 cmd.php 的可选替代，主要是为了加快 SNMP 轮训。如果你发现你的 cmd.php 运行超过 300 秒的话，推荐使用 Spine。这里讲讲如何安装配置 spine 替代 cmd.php。

安装 net-snmp-lib net-snmp net-snmp-utils 还需要安装 net-snmp-devel 不然会报 configure: error: Cannot find SNMP headers. 错误

```
wget cacti-spine-0.8.8b.tar.gz
```

```
cd cacti-spine-0.8.8b.tar.gz
```

```
shell>aclocal
```

```
shell>libtoolize --force (glibtoolize --force on Max OS)
```

```
shell>autoheader
```

```
shell>autoconf
```

```
shell>automake
```

```
shell>./configure
```

```
shell>make
```

```
shell>make install
```

修改配置文件/usr/local/spine/etc/Spine.conf.

```
DB_Host      127.0.0.1
DB_Database  cacti
DB_User      cactiuser
DB_Pass      password
DB_Port      3306
DB_PreG      0
```

```
cp /usr/local/spine/etc/spine.conf.dist /etc/spine.conf
```

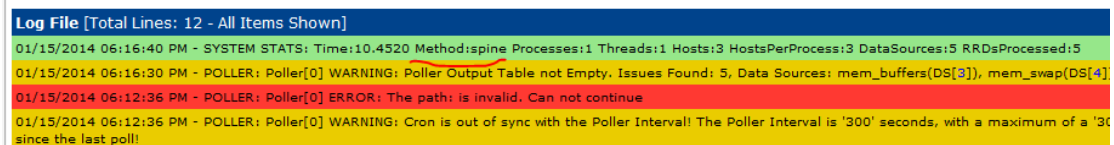
```
[root@nagios-cacti ~]# /usr/local/spine/bin/spine
SPINE: Using spine config file [/etc/spine.conf]
SPINE: Version 0.8.8b starting
SPINE: Time: 2.5959 s, Threads: 5, Hosts: 3
```

如果是其他输出如 FATAL: Unable to read configuration file! (Spine init)可能配置文件有错误。

使用方法：

- ① 以 admin 用户登录到 Cacti。
- ② 从 目录 菜单 中 选择 设置 " Settings" ， 选择 " Paths" 选项 卡 。
- ③ 在 " Spine Poller File Path" 中 输入 Spine 可 执行 程序 的 完整 路径 ， 我 的 是 /usr/local/spine/bin/spine ， 保 存 设 置 。
- ④ 选择 "Poller" 选项 卡 。 在 "Poller Type" 下 拉 框 设 置 中 ， 选 择 "spine" 。
- ⑤ 选择 "Paths" 选项 卡 。 Spine Poller File Path 后 面 加 入 /usr/local/spine/bin/spine

log 日志：



The screenshot shows a log file with the following content:

```
Log File [Total Lines: 12 - All Items Shown]
01/15/2014 06:16:40 PM - SYSTEM STATS: Time:10.4520 Method:spine Processes:1 Threads:1 Hosts:3 HostsPerProcess:3 DataSources:5 RRDsProcessed:5
01/15/2014 06:16:30 PM - POLLER: Poller[0] WARNING: Poller Output Table not Empty. Issues Found: 5, Data Sources: mem_buffers(DS[3]), mem_swap(DS[4])
01/15/2014 06:12:36 PM - POLLER: Poller[0] ERROR: The path: is invalid. Can not continue
01/15/2014 06:12:36 PM - POLLER: Poller[0] WARNING: Cron is out of sync with the Poller Interval! The Poller Interval is '300' seconds, with a maximum of a '30' since the last poll!
```

客户端 snmpd 的安装配置

```
#vi /etc/snmp/snmpd.conf
```

1, 更改 com2sec local localhost public 为
com2sec mynetwork 211.211.211.200 public

注：211.211.211.200 为 cacti 监控服务器的 IP 地址，public 为 snmp 在进行 community 时用的名字

2, 更改

```
access notConfigGroup "" any noauth exact systemview none none
```

改为

```
access notConfigGroup "" any noauth exact all none none
```

```
3, #view all included .1 80
```

将前面的 # 注释 去掉。 保存退出

如果还是无法抓到 snmp 数据包

用 `ps aux | grep snmp` 查看是否启动，如没有看到进程，`service snmpd restart`

测试 snmp 是否正常

```
#snmpwalk -c public -v 2c localhost
```

```
#snmpwalk -v 1 -c public localhost IP-MIB::ipAdEntIfIndex
```

```
IP-MIB::ipAdEntIfIndex.61.xxx.xxx.xxx = INTEGER: 2 IP-MIB::ipAdEntIfIndex.127.0.0.1  
= INTEGER: 1 IPMIB::
```

```
ipAdEntIfIndex.172.xxx.xxx.xxx = INTEGER: 3
```

该命令表示，使用 SNMPV2c 版本和 community 名称 “public”，取服务器（localhost）的所有 interface（网卡）的信息。

如果 localhost 的 SNMP 服务正常，则会返回服务器或交换设备的网卡信息。如果返回错误信息，请检查你的 SNMP 配置或网络是否正常。

也可以把 localhost 该为远端主机

```
[root@nagios-cacti ~]# snmpwalk -v 1 -c public 211.211.211.180 IP-MIB::ipAdEntIfIndex  
IP-MIB::ipAdEntIfIndex.127.0.0.1 = INTEGER: 1  
IP-MIB::ipAdEntIfIndex.211.211.211.180 = INTEGER: 2  
IP-MIB::ipAdEntIfIndex.211.211.211.181 = INTEGER: 2
```

安全

或许有人会说开启了 SNMP 服务后，服务器和设备会变得不安全了。其实这样的言论可能还停留在多年前对 SNMP 的看法。其实 SNMP 最容易被利用的地方不是它的缺陷，而是它默认的查询 community——“public”。不负责任的管理人员总会留下后门给黑客，这并非软件的错。当前 SNMP 协议的版本是 SNMP3，该版本改变的之前只使用 community 进行查询的方式而是采用了用户名和密码验证，大大改善了 SNMP 的安全性。即使不使用 SNMP3 版本，也可以在配置文件中限制能访问 SNMP 服务的网段。

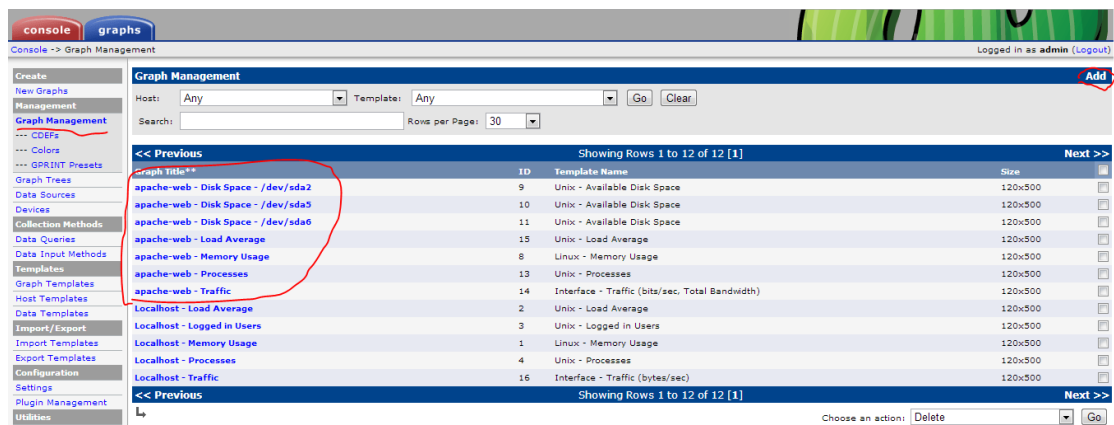
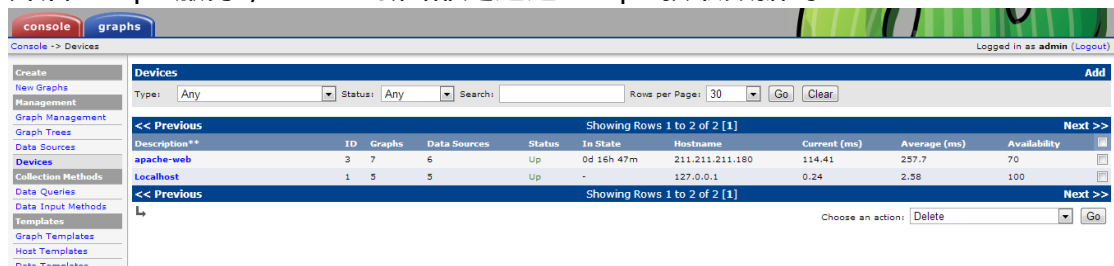
当然，如果你对 SNMP 还是不放心的，可以将 SNMP 服务 bind 在内网 IP 上。没有内网 IP？防火墙、ACL 总会用吧？

cacti 界面

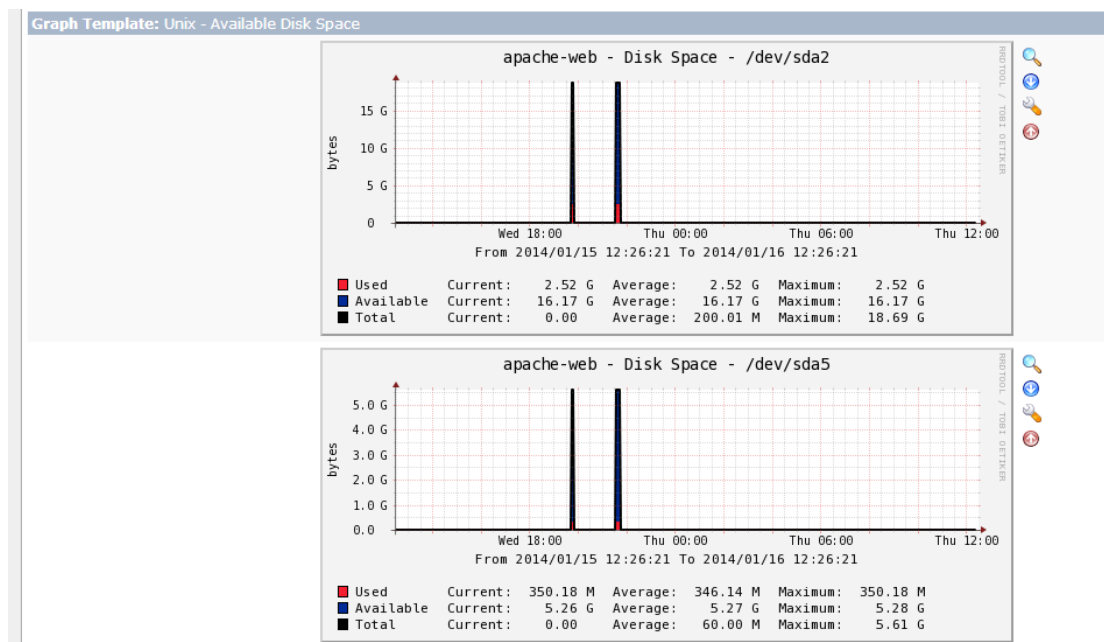
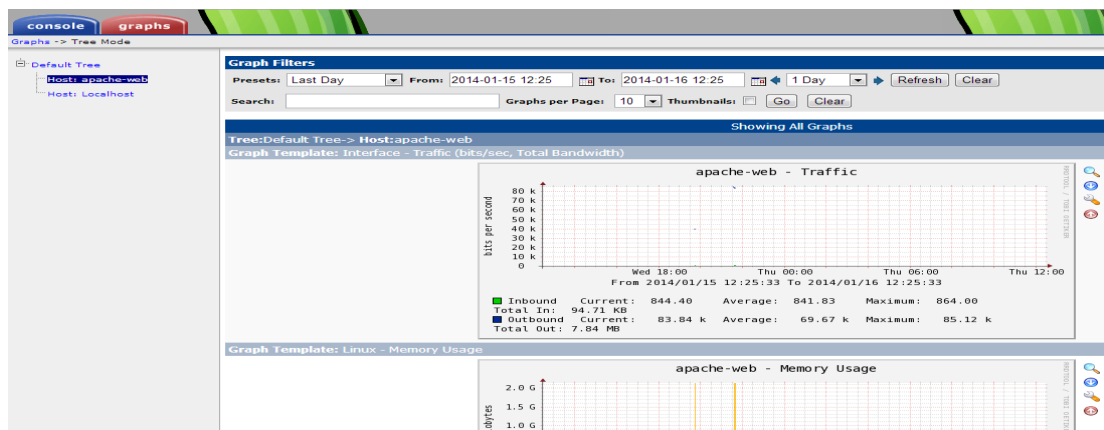
登录后的界面，这里可以添加监控设备创建监控图像，及一些其他的管理



我添加的 web 服务器的监控，可以点击右上角的 add 加入 device，前提是开启 snmpd 服务，rrdtool 绘图都是通过 snmpd 获取数据的



看看刚才创建的监控图像，点击 graphs



对于 cacti 监控界面的操作还不熟悉，这些只是简单的监控，深入监控需要进一步学习，特别是脚本与模版

cacti 报错解决

无法生成图片

修改权限：`chown -R nagios:nagcmd /var/www/html/cacti`

`chmod 775 /var/www/html/rrd /var/www/html/resource`

`usermod -g nagcmd cactiuser`

cacti 时区报错

```
PHP Warning: date(): It is not safe to rely on the system's timezone settings. You are *required* to use the date.timezone setting or the date_default_timezone_set() function. In case you used any of those methods and you are still getting this warning, you most likely misspelled the timezone identifier. We selected 'Asia/Chongqing' for 'CST/8.0/no DST' instead in /var/www/html/cacti/include/global_arrays.php on line 676
PHP Warning: strtotime(): It is not safe to rely on the system's timezone settings. You are *required* to use the date.timezone setting or the date_default_timezone_set() function. In case you used any of those methods and you are still getting this warning, you most likely misspelled the timezone identifier. We selected 'Asia/Chongqing' for 'CST/8.0/no DST' instead in /var/www/html/cacti/include/global_arrays.php on line 677
PHP Warning: date(): It is not safe to rely on the system's timezone settings. You are *required* to use the date.timezone setting or the date_default_timezone_set() function. In case you used any of those methods and you are still getting this warning, you most likely misspelled the timezone identifier. We selected 'Asia/Chongqing' for 'CST/8.0/no DST' instead in /var/www/html/cacti/include/global_arrays.php on line 677
```

解决：

首先将系统时区改为

```
[root@nagios-cacti cli]# cat /etc/sysconfig/clock
ZONE="Asia/Chongqing"
```

然后在 include/global.php 中加入一行

```
$database_port = "3306";
$database_ssl = false;

date_default_timezone_set('Asia/Chongqing');
/* Default session name - Session name must contain alpha
```

cacti 使用

1，cacti 界面介绍

登陆 Cacti 后，可以看到左上角是两个选项卡，“console”和“graphs”。console 表示控制台，在此进行所有的配置等操作；而 graphs 则是用来查看所有服务器的性能图像的界面。



2 . console 菜单

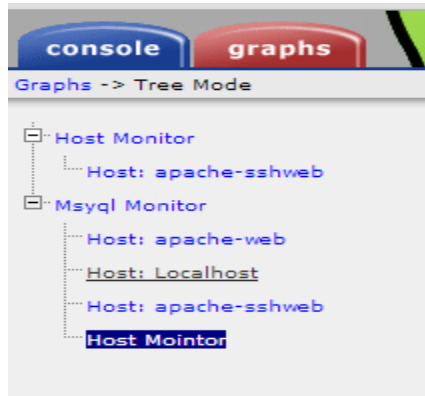
Create：

New Graphs——创建新图像的快捷方式；

Management：

Graph Management——图像管理。可以在此删除、复制图像，Cacti 会自动创建图像。不过如果我们有特殊的需要，比如将几张图上的数据合并在一张图像上的话也可以在此手工新建图像；

Graph Trees——图像树。在 graphs 界面里，图像或 devices 是树状结构显示的，可以在此设置树的结构；



Data Sources——管理 rrd 文件。这包括我们创建的主机图像，cacti 自动创建，不需要修改

Devices——设备管理。这是我们最经常需要修改的地方，可以在此创建新的设备或修改其名称等信息。

Collection Methods

Data Queries 和 Data Input Methods 是采集数据的方式，一般我们无需对这两项进行修改；

Templates

Graph Templates、Host Templates 和 Data Templates 分别是图像模板、主机类型模板和数据模板。这些模板可以导出、导入也可以自己编写，一般无需修改。

Import/Export

Import Templates 和 Export Templates，对上述模板的导入、导出。我们可以在 Cacti 的官方网站上找到这些模板，不过需要注意模板对于的 Cacti 的版本。

Configuration

Settings ——Cacti 的主要配置菜单；

可以在此重新设置对应的程序的路径、版本等信息。也可以设置图像的输出方式（允许 ftp）、显示效果、登陆方式（允许使用 LDAP）等。

Utilities

System Utilities ——显示 Cacti 系统的一些 cache 和 log 信息，如果 log 文件太大建议直接到后台查看；

User Management ——用户管理。可以在此添加、删除用户，并对每个用户设置

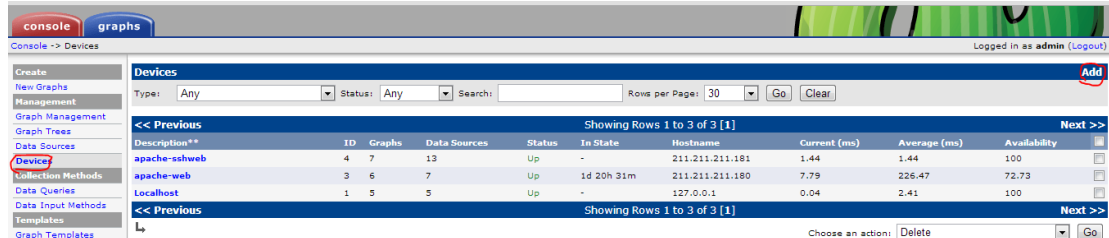
详细的权限；

Logout User ——注销用户。

3，创建监测点

进入 Cacti 的 console 面板 - >

点击“Devices”进入设备面板 - >



点击“Add”添加新设备 - >

Form for adding a new device in Cacti. Fields include:

- Description: haproxy
- Hostname: 211.211.211.170
- Host Template: General Linux Machine
- Number of Collection Threads: 1 Thread (default)
- Disable Host: ☐
- Availability/Reachability Options: SNMP Uptime
- Ping Timeout Value: 400
- Ping Retry Count: 1
- SNMP Version: Version 1
- SNMP Community: public
- SNMP Port: 161
- SNMP Timeout: 500
- Maximum OID's Per Get Request: 10
- Additional Options: (empty)
- Notes: (empty)

Buttons: Cancel, Create

填写要监测服务器的各种信息，其中 Host Template 请选择“Local Linux Machine”或“ucd/net SNMP Host”（选择一个合适的主机模板） - >

点击“Create”保存信息，如果 SNMP 连接没有问题，左上角会出现该服务器的信息，否则会出现“SNMP error”的红色字样 - >

Save Successful.

haproxy (211.211.211.170)

SNMP Information

System: Linux haproxy 2.6.32-220.17.1.el6.x86_64 #1 SMP Fri Jun 8 13:49:13 CST 2012 x86_64
Uptime: 12740 (0 days, 0 hours, 2 minutes)
Hostname: haproxy
Location: Unknown (edit /etc/snmp/snmpd.conf)
Contact: Root root@localhost (configure /etc/snmp/snmp.local.conf)

Devices [edit: haproxy]

General Host Options

Description: haproxy

Hostname: 211.211.211.170

Host Template: General Linux Machine

Number of Collection Threads: 1 Thread (default)

Disable Host: ☐

点击上部的“Create Graphs for this Host ”为该设备创建需监测的内容。监测的内容分两种，“Graph Templates”和“Data Query”,区别在于“Data Query”能根据SNMP 信息列出监测项目的信息。例如 Data Query 里的“Interface Statistics”可以看到该主机所有网卡的信息，这样我们可以选择需要监测的网卡。点击右侧的正方形选择框勾选上要监测的项目 - >

haproxy (211.211.211.170) General Linux Machine

Host: haproxy (211.211.211.170) Graph Types: All

*Edit this Host
*Create New Host

Graph Templates

Graph Template Name	
Creates ucd/net - Available Disk Space	<input checked="" type="checkbox"/>
Creates ucd/net - CPU Usage	<input checked="" type="checkbox"/>
Creates ucd/net - Load Average	<input checked="" type="checkbox"/>
Creates ucd/net - Memory Usage	<input checked="" type="checkbox"/>
Create: (Select a graph type to create)	

Data Query [SNMP - Get Processor Information]

This data query returned 0 rows, perhaps there was a problem executing this data query. You can run this data query in debug mode to get more information.

Data Query [SNMP - Interface Statistics]

This data query returned 0 rows, perhaps there was a problem executing this data query. You can run this data query in debug mode to get more information.

Select a graph type: In/Out Bits

Data Query [Unix - Get Mounted Partitions]

Device Name	Mount Point	
/dev/sda1	/boot	<input type="checkbox"/>
/dev/sda2	/	<input checked="" type="checkbox"/>
/dev/sda5	/var	<input type="checkbox"/>
/dev/sda6	/home	<input checked="" type="checkbox"/>
/dev/sda0	/opt/iso	<input type="checkbox"/>

console graphs

Create New Graphs

Test host (192.168.100.110) Local Linux Machine

Create new graphs for the following host: Test host (192.168.100.110)

*Edit this Host
*Create New Host

Graph Templates

Graph Template Name	
Creates Linux - Memory Usage	<input checked="" type="checkbox"/>
Creates Unix - Load Average	<input checked="" type="checkbox"/>
Creates Unix - Logged in Users	<input checked="" type="checkbox"/>
Creates Unix - Processes	<input checked="" type="checkbox"/>
Create: (Select a graph type to create)	

Data Query [Unix - Get Mounted Partitions]

Device Name	Mount Point	
/dev/mapper/voldgroup00-Logvol000	/	<input checked="" type="checkbox"/>
/dev/sda1	/boot	<input type="checkbox"/>

cancel create

点击“Create”创建选择的监测内容，已经选择创建的内容会自动变成灰色并且不能再点选。Cacti 会自动创建该监测点的 rrd 文件（在 rra 文件夹中）、“Data Source”和“graph”条目。

haproxy (211.211.211.170) General Linux Machine

Host: haproxy (211.211.211.170) Graph Types: All

*Edit this Host
*Create New Host


Graph Templates

Graph Template Name	
Creates ucd/net - Available Disk Space	<input type="checkbox"/>
Creates ucd/net - CPU Usage	<input type="checkbox"/>
Creates ucd/net - Load Average	<input type="checkbox"/>
Creates ucd/net - Memory Usage	<input type="checkbox"/>
Create: (Select a graph type to create)	

创建监测点完毕。

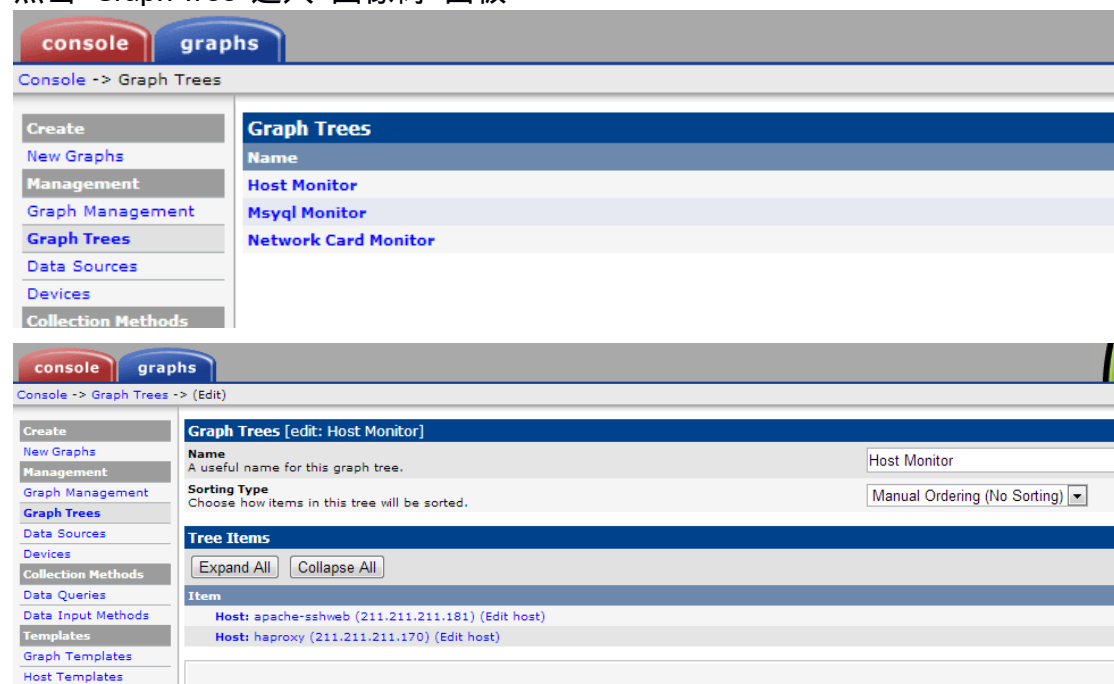
4. 查看监测点

点击“Graph Management”可以看到刚才创建的监测点对应的图像，注意由于Cacti默认每5分钟到监测服务器上取一次数据，所以刚创建的监测点会出现图像不能显示的现象，我们手动执行一下 `php /var/www/html/cacti/poller.php`

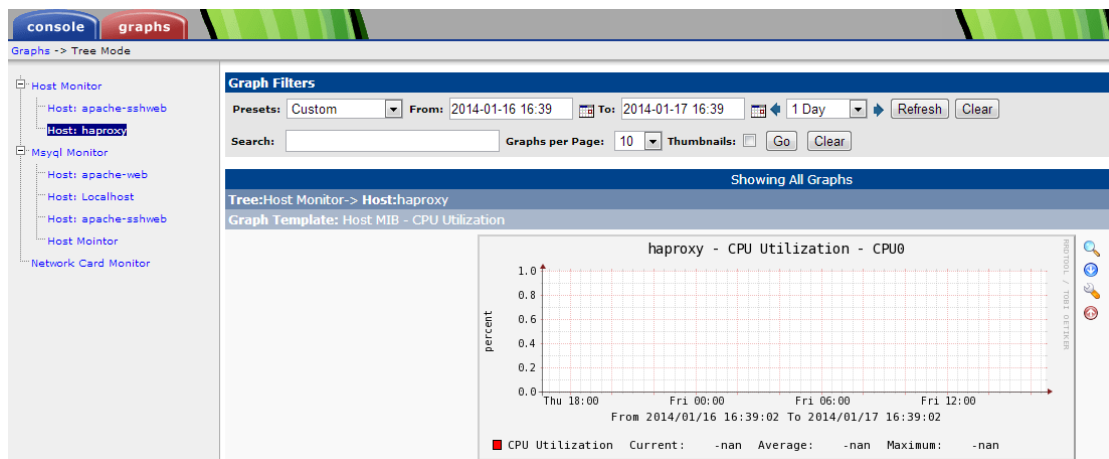


Graph Title**	ID	Template Name
haproxy - CPU Usage	27	ucd/net - CPU Usage
haproxy - CPU Utilization - CPU0	33	Host MIB - CPU Utilization
haproxy - Disk Space - /dev/sda2	30	Unix - Available Disk Space
haproxy - Disk Space - /dev/sda6	31	Unix - Available Disk Space
haproxy - Hard Drive Space	26	ucd/net - Available Disk Space
haproxy - Load Average	28	ucd/net - Load Average
haproxy - Memory Usage	29	ucd/net - Memory Usage
haproxy - Traffic - eth0	32	Interface - Traffic (bits/sec)
haproxy - Traffic - eth0	34	Interface - Traffic (bits/sec)

为了方便查看，可以将刚才新创建的设备或图像加入到“图像树”上：
点击“Graph Tree”进入“图像树”面板 - >



在这里可以根据自己的需求创建目录树，来看看我创建的目录树



ok，需要等一段时间才能有流量图，每隔 5 分钟才收集一次数据

cacti 脚本与模版

Cacti 脚本及模板论坛：

完整的 cacti 脚本及模板列表：

我打算监控的东西包括：系统类：cpu、memory、disk、load

服 务 类：apache、nginx、mysql master
slave、lvs、squid

流量类：网卡流量、网站总流量、PV、交换机流量

添加 mysql 数据库的监控脚本与模版

wget <http://mysql-cacti-templates.googlecode.com/files/better-cacti-templates-1.1.8.tar.gz>

下载模版解压，将 scripts 目录中文件拷贝到 cacti/scripts 目录中，将 templates 目录中文件同页面 import 进去

修改 ss_get_mysql_stats.php 文件

```
$mysql_user = 'cactiuser';
```

```
$mysql_pass = 'password';
```

```
$cache_dir = '/tmp';
```

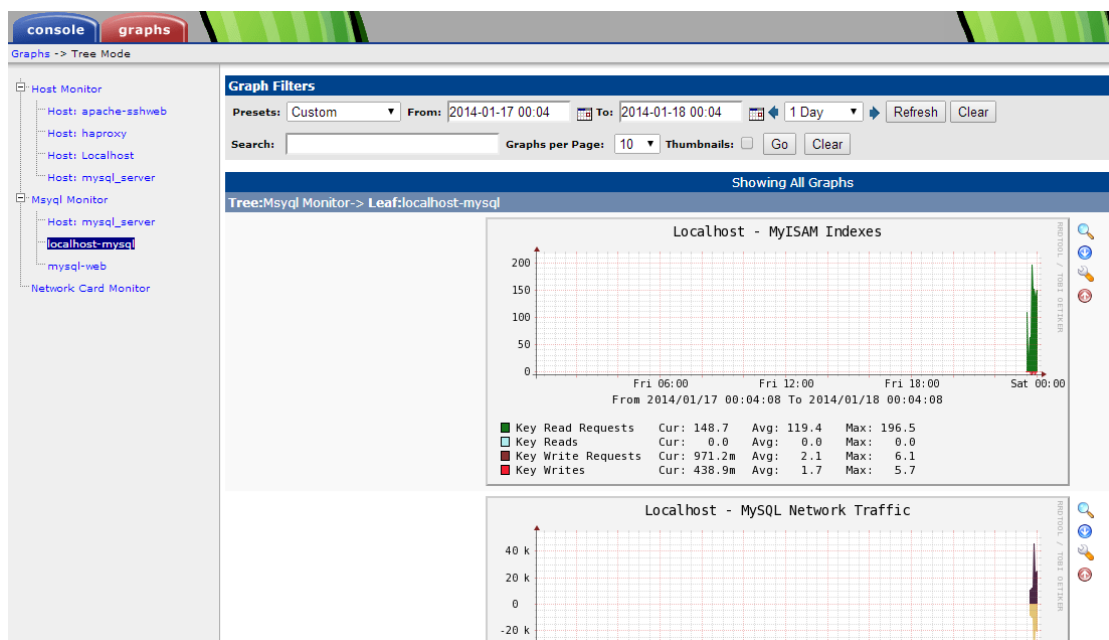
 注意该目录的权限，系统中的 cactiuser 用户要能够读写。

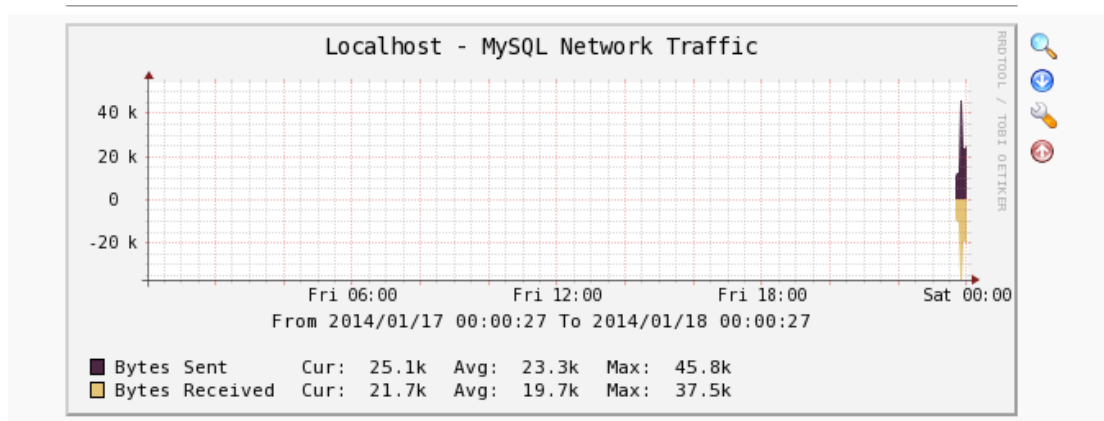
配置 MySQL 服务器，让 cacti 所在机器能够访问 MySQL 服务器的状态信息，必须拥有“process”权限。如果要监控 InnoDB 状态，还必须有“SUPER”权限。

mysql> grant process,super on *.* to 'cactiuser'@'%' identified by 'password';
%可以改为自定义的 ip 或者网段，单加入这一条似乎无法获取一些图像数据？
不知道为什么？对数据库不怎么熟悉
下面这条命令是给予所以权限：
mysql> grant all privileges on cacti.* to 'cactiuser'@'%' identified by "password";



刚刚导入进去的绘图模版，然后依据前面介绍的在 new graphs 那里加入 mysql 服务的监控





添加 **nginx/apache** 监控模版

wget <http://mysql-cacti-templates.googlecode.com/files/better-cacti-templates-1.1.8.tar.gz>

mysql 的监控模版里面包含了 nginx/apache 模版，直接在此页面 import 就可以了，当然也包含其他的一些模版，如果需要也可以一起导入，使用模版和上面一样

添加 **squid/memcache** 监控模版

直接到官网下载，导入

wget
http://docs.cacti.net/_media/usertemplate:graph:host_resources_mib:cacti_graph_template_squid_-_request_rate.xml.zip

wget
http://docs.cacti.net/_media/usertemplate:graph:squid:cacti_graph_template_squid_-_http_requests.xml.zip

wget
http://docs.cacti.net/_media/usertemplate:graph:host_resources_mib:cacti_graph_template_squid_-_hit_ratio.xml.zip

wget
http://docs.cacti.net/_media/usertemplate:graph:squid:cacti_graph_template_squid_-_http_service_time.xml.zip

squid 配置：

配置 squid 的配置文件并设置 squid 支持 snmp 协议,.当然如果要是编译的 squid 时,需要在编译的时候启用 snmp 协议，并重启 squid 服务。

```
acl cactiserver src 211.211.211.200 #该处设置的是 cactiserver 的 IP 地址
```

```
acl SNMP snmp_community snmppublic #snmppublic 是 snmp 探测时候的 community 的名字，当然在此如果是为了安全起见需要改成自己所需要的 community 名字，应为在模板中的 community 的名字是 snmppublic，为了省事没有修改
```

```
snmp_port 161 （默认是 161）
```

```
snmp_access allow SNMP cactiserver
```

```
snmp_access deny all 这里基于 acl 设置了只允许相应的主机来监控 squid 服务
```

cacti 插件

```
wget http://docs.cacti.net/_media/plugin:monitor-v1.3-1.tgz
```

cacti+nagios 结合使用

PHP 支持 JSON 扩展

PHP 必须安装 PDO 和 JSON 扩展。由于 NPC 使用了 Ext JS，如果没有 JSON 扩展，NPC 的界面不会出来，只能看到一个空白页面。初始化 PHP 环境 需要用 phpize 命令，所以也要安装 php-devel。

建议用 yum 安装，只要安装 php-common 就行了

ndoutils 安装

ndoutils 是将 Nagios 的配置及监控信息存储到数据库里，NPC 通过调用 ndo 所存储的数据来展现 Nagios 的信息。ndoutils 需要用到 mysql 的 mysql-lib 及 mysql-inc，因此需要安装 mysql-devel

需要详细了解 ndoutils 的可以看看源代码 docs 目录下的官方文档

```
wget http://downloads.sourceforge.net/project/nagios/ndoutils-1.x/ndoutils-1.5.2/ndoutils-1.5.2.tar.gz?r=http%3A%2F%2Fsourceforge.net%2Fprojects%2Fnagios%2Ffiles%2Fndoutils-1.x%2Fndoutils-1.5.2%2F&ts=1390125251&use_mirror=nchc
tar xvf ndoutils-1.5.2.tgz
cd ndoutils-1.5.2
./configure --with-ndo2db-user=nagios --with-ndo2db-group=nagcmd
make
```

这里不需要执行 make install。只需要将一下文件

ndomod-3x.o ndo2db-3x file2sock log2ndo 拷贝到 /usr/local/nagios/bin 目录

拷贝文件

```
cd src
cp ndomod-3x ndo2db-3x log2ndo file2sock /usr/local/nagios/bin
```

初始化数据库

1，创建 nagios 数据库，并创建用户 cactiuser 给予 SELECT, INSERT, UPDATE, and DELETE privileges

2 执行脚本，

```
cd db
./installdb
```

创建 ndoutils 配置文件

```
config
cp ndo2db.cfg-sample /usr/local/nagios/etc/ndo2db.cfg
cp ndomod.cfg-sample /usr/local/nagios/etc/ndomod.cfg
```

修改 ndo2db.cfg

```
[root@nagios-cacti etc]# cat ndo2db.cfg | grep ^[^#]
lock_file=/usr/local/nagios/var/ndo2db.lock
ndo2db_user=nagios
ndo2db_group=nagcmd
```

```
socket_name=/usr/local/nagios/var/ndo.sock
db_servertype=mysql
db_host=localhost
db_port=3306
db_name=cacti
db_prefix=np_
db_user=cactiuser
db_pass=roottoor
debug_level=1
debug_file=/usr/local/nagios/var/ndo2db.debug
```

确保 nagios.cfg 中
event_broker_options=-1

启动 NDO2DB daemon.

```
/usr/local/nagios/bin/ndo2db-3x -c /usr/local/nagios/etc/ndo2db.cfg
```

service nagios restart

npc 安装

NPC(Nagios Plugin for Cacti)是一个 Cacti 插件，安装后可以在 Cacti 界面里使用 Nagios 的功能

```
wget http://dl.cactifans.org/plugins/npc-2.0.4.tar.gz
tar -zxvf npc-2.0.4.tar.gz
mv npc/ /var/www/html/cacti/plugins/
```

在 console 下的 plugin 插件下面安装及 enable

在 setting 里面的 npc 项做如下配置，前提是先启动 nagios 才有 nagios.cmd



npc 界面一直显示 off，经查看是 ndomod 模块没有加载成功，报错如下：

```
Jan 19 23:16:05 nagios nagios: wproc: Registry request: name=Core Worker 28464;pid=28464
Jan 19 23:16:05 nagios nagios: Error: Could not load module '/usr/local/nagios/bin/ndomod.o' -> /usr/
vicedependency_list
Jan 19 23:16:05 nagios nagios: Error: Failed to load module '/usr/local/nagios/bin/ndomod.o'.
Jan 19 23:16:05 nagios nagios: Error: Module loading failed. Aborting.
```

找过很多原因，折腾了半天，原来是 ndo2db 版本不适合 nagios4.0 的版，只适用于 nagios3.x

线上环境的系统及流量监控需求