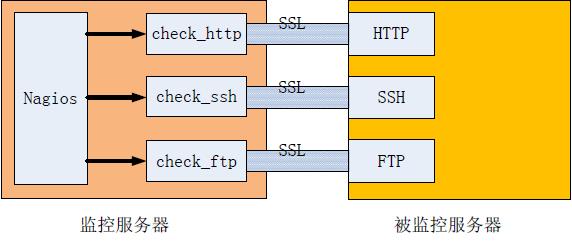
CentOS6 install nagios, pnp2nagios

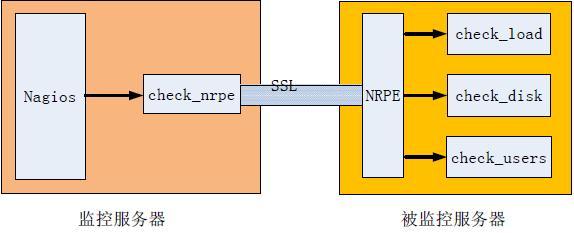
Nagios是一个非常灵活的监控系统，可以监控常见的各种服务以及服务器的运行状态，并在监测到服务器发生故障的情况下，使用各种不同的方式通知维护人员发生了何种故障。Nagios只提供一个监控平台，本身并不提供任何工具来对服务器进行监控或报警，也不提供发送报警信息的代码，所有这些功能都是由插件来实现的。Nagios可以很好地整合所有的插件一起高效的工作。通过分工合作来实现一个完善的监控机制。

Nagios特征包括：

1.监控网络服务（SMTP、POP3、HTTP、SSH、mysql、NNTP、PING等）；  
2.监控主机资源（处理器负荷、磁盘利用率等）；  
3.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；  
4.并行服务检查机制；  
5.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；  
6.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）；  
7.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；  
8.自动的日志回滚；  
9.可以支持并实现对主机的冗余监控；  
10.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；

常用的nagios系统一般组成：nagios、nagios-plugins、nrpe、nsclient++，被动监控时要用到nsca

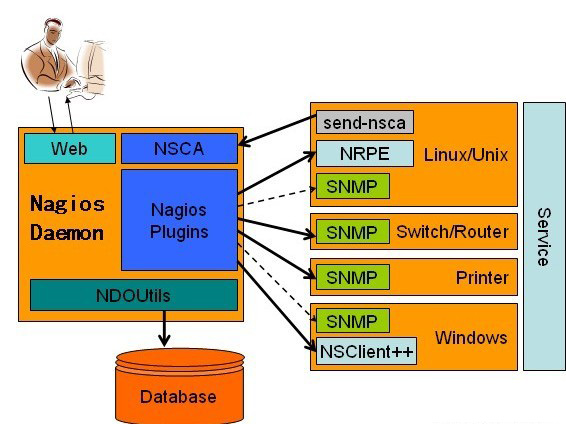
一个典型的架构如下图所示：  


上面架构只能监控到服务器的外部服务，如HTTP、SSH、FTP，要实现对本地信息的监控可以通过在被监控服务器上安装NRPE服务，作为监控代理来实现对本地信息的监控，架构如下：  


NRPE 由两个部分组成：工作在监控机的check\_nrpe 插件、工作在被监控机的NRPE守护进程。

Nagios 服务器执行check\_nrpe 插件并告诉他检查哪个服务，check\_nrpe 插件通过SSL 连接方式联系远程服务器上的NRPE守护进程，NRPE守护进程执行相应的插件完成指定的检查，并返回结果。

完整的Nagios各组件关系图(author:[Marion](http://blog.chinaunix.net/uid-233544-id-2427477.html))



本文主要介绍通过RPM方式安装，需要事先添加两个源：EPEL和RPMFORGE，这也是两个非常稳定、非常常用的YUM源，因为这两个源的安装包时有变化，所以这里只给出发布源的地址，而不是具体的RPM地址：

RPMFORGE：[http://repoforge.org/use/](http://repoforge.org/use/" \t "_blank)  
EPEL：[https://fedoraproject.org/......](https://fedoraproject.org/wiki/EPEL/zh-cn" \t "_blank)

**一、Server端安装：**

**#安装nagios**

yum -y install nagios nagios-nrpe nagios-plugins nagios-plugins-nrpe check\_logfiles

**#添加nagios到系统自动运行**

chkconfig --add nagios

chkconfig nagios on

**nagios相关路径：**

/etc/nagios *#配置文件目录：*

/etc/nagios/objects *#模板相关配置*

/usr/sbin/nagios *# nagios主程序*

/usr/lib(64)/nagios/cgi-bin/ *#nagios CGI文件目录*

/usr/lib(64)/nagios/plugins/ *#插件目录*

/usr/share/nagios/html/ *#网页文件目录*

/**var**/log/nagios/ *#日志，存档目录*

**配置Apache/Nginx密码访问**

# nagiosadmin为用户名，配置这个用户名为省很多事

htpasswd -c /etc/nagios/htpasswd.nagios nagiosadmin

**启动并测试**

service nagios start

service httpd start

#打开浏览器，输入密码，在host里面应该有“localhost”这台机器

http:*//{your\_ip\_address}/nagios*

**二、Nagios的配置文件**

对nagios 进行配置需要了解一个概念object（对象），nagios 使用object 来对各种信息进行组织，nagios 中的包含如下object：

contact（联系人）：何种方式，何种情况下采用何种方式报警给何人

contactgroup（联系人组）：对联系人进行组织

timeperiod（监控时间段）：7×24还是workhours，或者其他自定义的时间段

host（主机）：被监控的服务器

hostgroup（主机组）：对主机进行分类组织

command（命令）：定义Nagios要执行的监控命令

service（服务）：定义Nagios要监控的服务，如web/ssh/ftp等

servicegroup（服务组）：组织多个被监控的服务

配置文件都存放在/etc/nagios/目录或该目录下的conf.d，private，objects目录

cgi.cfg 用于定义浏览器对nagios操作，比如权限控制等

nagios.cfg 用于定义控制nagios行为

nrpe.cfg NRPE配置文件

nsca.cfg 被动监控配置文件

resource.cfg 资源配置文件，用于定义插件路径，一般不需修改

objects目录 监控对象配置文件存放于下面，配置文件可以自己添加，文件名可以自行定义，只要以.cfg结尾就行，但是必须在nagios.cfg加一行代码使nagios读取这个文件，例如你可以新建个linux.cfg来添加linux监控设备，nagios.cfg应该添加一行“cfg\_file=/usr/**local**/nagios/etc/objects/linux.cfg”,一般同一类设备写到一个文件里，当然也可以对默认的文件进行修改来添加监控设备，此目录下默认有下面几个文件

commands.cfg 命令定义配置文件

contacts.cfg 定义联系人列表

timeperiods.cfg 定义时间监控模板

templates.cfg 定义各种模板,可以是主机模板、联系人模板、监控服务模板等

localhost.cfg 用于定义linux监控设备

windows.cfg 用于定义windows监控设备

**switch**.cfg 用于定义交换机，路由器等监控设备

printer.cfg 用于定义打印机监控设备

深入理解Nagios的工作原理和各对象的意义域作用是有效使用Nagios的前提

**三、添加被监控设备**

**Client端安装**

# 更新yum库（略）

**#安装nrpe与nagios-plugins**

yum -y install nagios-nrpe nagios-plugins nagios-plugins-nrpe

**配置（client端）**

vi /etc/nagios/nrpe.cfg

# 下列配置表示允许127.0.0.1, 192.168.121.55（Server）这两台机器访问当前机器的信息

allowed\_hosts=127.0.0.1,192.168.121.55

**测试（client端）**

#启动nrpe (Nagios Remote Plugin Executor)

service nrpe start

**添加到服务并设置为自动启动**

chkconfig --add nrpe

chkconfig nrpe on

**#查看端口是否启动**

netstat -an | grep 5666

**#本机测试服务是否正常**

/usr/lib/nagios/plugins/check\_nrpe -H 127.0.0.1 -c check\_load

/usr/lib64/nagios/plugins/check\_nrpe -H 127.0.0.1 -c check\_load *# 64 bit 命令*

**#服务器端测试访问是否正常。**\*\*在Server上执行\*\*

/usr/lib/nagios/plugins/check\_nrpe -H {client\_ip} -c check\_load

/usr/lib64/nagios/plugins/check\_nrpe -H {client\_ip} -c check\_load *#64bit*

**四、配置（Server端）**

**# 创建client配置文件**（192.168.121.4为client ip）

一个相对比较完整的配置文件格式

1. define host{
2. **use** model\_host *//定义用的模板机*
3. host\_name host\_name(\*) *//定义主机名，可起任意名字*
4. **alias** **alias**(\*) *//定义主机别名*
5. display\_name display\_name *// 定义显示名字*
6. address address(\*) *//主机名对应的ip 地址*
7. parents host\_names *//定义父节点*
8. hostgroups hostgroup\_names *//定义主机组名*
9. check\_command command\_name *//检查主机状态的命令*
10. initial\_state [o,d,u] *//初始化状态*
11. max\_check\_attempts *#(\*) //当检查命令返回值不是“OK”时最大重试次数*
12. check\_interval *# //#分钟进行一次检查*
13. retry\_interval *# //重试间隔时间*
14. active\_checks\_enabled [0/1] *//主动监控开关(1开0闭)*
15. passive\_checks\_enabled [0/1] *//被动监控开关*
16. check\_period timeperiod\_name(\*) *//主机状态检查的时间段*
17. obsess\_over\_host [0/1] *//是否启用主机操作系统探测*
18. check\_freshness [0/1] *//是否启用 freshness 检查。freshness 检查是对于启用被动检查模式的主机而言的，其作用是定期检查主机报告的状态信息，如果该状态信息已经过期，freshness 将会强制做主机检查。freshness\_threshold #*
19. event\_handler command\_name
20. event\_handler\_enabled [0/1] *//主机事件处理是否激活*
21. low\_flap\_threshold *#*
22. high\_flap\_threshold *#*
23. flap\_detection\_enabled [0/1]
24. flap\_detection\_options [o,d,u]
25. process\_perf\_data [0/1]
26. retain\_status\_information [0/1]
27. retain\_nonstatus\_information [0/1]
28. contacts contacts(\*) *//发送报警通知给谁*
29. contact\_groups contact\_groups(\*) *//发送通知给哪个报警组*
30. notification\_interval *#(\*) //主机状态通知功能激活*
31. first\_notification\_delay *#*
32. notification\_period timeperiod\_name(\*) *//发送通知的时间段*
33. notification\_options [d,u,r,f,s] *//定义那些情况下发送通知*
34. notifications\_enabled [0/1] *//发送通告开关*
35. stalking\_options [o,d,u]
36. notes note\_string
37. notes\_url url
38. action\_url url
39. icon\_image image\_file *//定义主机图标*
40. icon\_image\_alt alt\_string
41. vrml\_image image\_file
42. statusmap\_image image\_file
43. 2d\_coords x\_coord,y\_coord *//定义图标在网页中显示的二维坐标*
44. 3d\_coords x\_coord,y\_coord,z\_coord *//定义图标在网页中显示的三维坐标*
45. ...
46. }

一个简单的样例：

1. vi /etc/nagios/objects/192.168.121.4.cfg
3. define host{
4. **use** linux-server
5. host\_name client-1
6. **alias** client-1
7. address 192.168.121.4
8. }
10. *//check\_load*
11. define service{
12. **use** **generic**-service
13. host\_name client-1
14. service\_description Current Load
15. check\_command check\_nrpe!check\_load
16. }
18. *//check\_disk*
19. define service{
20. **use** **generic**-service *//local-service改成generic-service*
21. host\_name client-1 *//localhost改成自定义client-1*
22. service\_description Root Partition
23. check\_command check\_nrpe!check\_disk!20%!10%!/
24. }
26. *//把上面的check\_local\_disk改成check\_nrpe!check\_disk，如果有不带local的，例如：check\_ping改成check\_nrpe!check\_ping*

在上面的配置中：!20%!10%!/表示执行chek\_disk并传递三个参数：20%为警告阀值，10%为严重错误阀值，/表示检查根分区  
但，默认安装的nrpe是不允许传递参数的，需要编译时加上“　--enable-command-args”才可以  
允许传递参数会带来安全隐患，所以在实际应用中，警告和严重错误阀值的参数一般是在被监控端直接写入，而不是通过此处传递 \*\* 重要 \*\*

**#添加check\_nrpe指令**

1. [root@localhost objects]# vim /etc/nagios/objects/commands.cfg *//修改配置文件*
2. *//添加以下内容*
3. define command{
4. command\_name check\_nrpe
5. command\_line $USER1$/check\_nrpe -H $HOSTADDRESS$ -c $ARG1$
6. *//如果客户端使用的不是默认的5667端口，需要在上一行内加上 -p 端口*
7. }

# 将上面配置加到nagios

vi /etc/nagios/nagios.cfg

cfg\_file=/etc/nagios/objects/192.168.121.4.cfg

**运行和测试（Server端）**

/usr/bin/nagios -v /etc/nagios/nagios.cfg *#Server端验证配置*

确认无误，启动nagios

service nagios start

**五、配置（client端）**

**配置nrpe**，添加指令

1. [root@localhost ~]# vi /etc/nagios/nrpe.cfg
3. allowed\_hosts=192.168.1.122,127.0.0.1 *//加上服务端的ip，可以是局域网，或者公网的IP*
5. command[check\_users]=/usr/lib64/nagios/plugins/check\_users -w 5 -c 10
6. command[check\_load]=/usr/lib64/nagios/plugins/check\_load -w 15,10,5 -c 30,25,20
7. command[check\_disk]=/usr/lib64/nagios/plugins/check\_disk -w 20% -c 10% -p /dev/mapper/VolGroup-lv\_root
8. command[check\_zombie\_procs]=/usr/lib64/nagios/plugins/check\_procs -w 5 -c 10 -s Z
9. command[check\_procs]=/usr/lib64/nagios/plugins/check\_procs -w 150 -c 200
10. *#command[check\_http]=/usr/lib64/nagios/plugins/check\_http -H 127.0.0.1 -w 5 -c 10*
11. *#command[check\_ping]=/usr/lib64/nagios/plugins/check\_ping -H 127.0.0.1 -w 3000.0,80% -c 5000.0,100% -p 5*
12. *#command[check\_ssh]=/usr/lib64/nagios/plugins/check\_ssh -4 127.0.0.1*
13. command[check\_swap]=/usr/lib64/nagios/plugins/check\_swap -w 30% -c 10%

**六、被动监测（NSCA）**

当Nagios监控服务器需要同时监控很多台客户机的时候，系统资源占用会比较高，大量机器监控的环境一般会选择被动监控，通过NSCA实现

NSCA可以到http://www.nagios.org上下载，不过我还是喜欢用rpm，添加以上两个源后，可以直接通过yum安装

yum install nsca

由于nsca在client上是主动送结果到服务器,所以在server端只要开启一个nsca服务，由client端向server端发送数据就行了，在server端只需开启nsca服务，然后把server端的send\_nsca脚本和send\_nsca的配置文件拷贝到clent即可。

#nsca服务端可能需要修改的几个位置：

nsca\_user=nagios

nsca\_group=nagios

password=nsgiospassword

decryption\_method=1

此时运行nsca很有可能会出错：

chown nagios.nagios /etc/nagios/nsca.cfg

chmod g+r /etc/nagios/nsca.cfg

另外检查以下两行的路径是否正确，nagios用户是否有权限

chown nagios.nagiocommand\_file=/var/spool/nagios/cmd/nagios.cmd

alternate\_dump\_file=/var/log/nagios/rw/nsca.**dump**

然后重启nsca，检查端口是否已经在监听

netstat -an | grep 5667

输出应类似于：

tcp 0 0 0.0.0.0:5667 0.0.0.0:\* LISTEN

Client端不需配置，直接拷贝Server端的send\_nsca脚本和send\_nsca.cfg即可

共需要传送以下几个参数到server(对应的参数):

A) client hostname,这个必须和监控机上设置的hostname一致!

B) service description,服务定义,同样也必须跟监控机上定义的servicename一致

C) 报警等级,0=ok,1=warning,2=critical,必须是整型

D) 报警内容,这个可以自由发挥,以能够清楚描述报警信息为原则.

使用以下语句可以达到我们的目的:

echo "$NSCACLIENT;$SERVICEDESC;$ALTLEVEL;$MSG"|$SEND\_NSCA -H $NSCA\_SERVER -to 10 -d ";" -c "$NSCA\_CONF"

以上语句里面,$NSCACLIENT=客户端的hostname,必须与监控机上定义的hostname一致  
$SERVICEDESC为服务描述,必须与监控机上定义的servicename一致  
$ALTLEVEL为报警等级,根据脚本执行结果返回  
$MSG 为返回的信息

以上4项内容用分号分隔,然后通过send\_nsca的 –d “;”参数来分别区分并发送给nagios监控机以达到发报警的目的!

以tsar为例：

echo "websrv.name;tsar;0;OK|cpu.util=10.44 mem.util=37.37 swap.swpin=0.00 swap.swpout=0.00 tcp.retran=0.00 traffic.bytin=1385.00 traffic.bytout=915.00 io.sda.util=2.56 partition./.util=15.53 partition./var.util=9.78 load.load1=0.36 load.load5=0.32 load.load15=0.28 "|/usr/sbin/send\_nsca -H nagios.srv -p 7002 -to 10 -d ";" -c /usr/sbin/send\_nsca.cfg

注：websrv.name:是指nagios监控机中的定义的host\_name,也可以是机器IP  
tsar是指nagios监控机中的service\_description  
0是指传过去的状态值，这里为OK (0表示ok，1表示warn，2表示critical)  
OK是指的状态说明,这里只是一个测试  
cpu.util.....部分是具体发送过去的tsar检测到的性能数据

最后就是nagios里面的定义

1. define host{
2. **use** **generic**-host
3. host\_name websrv.name
4. **alias** webserver-wuhan-luxiang
5. address 1.2.3.4
6. }
8. define service{
9. **use** **generic**-service
10. service\_description tsar
11. active\_checks\_enabled 0 *#设置为被动模式*
12. check\_command check\_dummy!2
13. }
15. active\_checks\_enabled 0
16. passive\_checks\_enabled 1

作用就是：定制被动监控，取消主动监控

脚本还需要check\_dummy，这个地方随便定义即可,比如check\_tcp，因为被动模式，服务器端只收取客户端的信息，所以这个地方不关键,当然如果有使用到check\_freshness，默认为0，想让在故障时由服务端执行对应的指令来配合

另外在commands.cfg中加入：

1. define command {
2. command\_name check\_dummy
3. command\_line $USER1$/check\_dummy $ARG1$ "$ARG2$"
4. }

七、Windows客户端  
Windows客户端可以安装NSClient++，过程很简单，到[http://www.nsclient.org/nscp/](http://www.nsclient.org/nscp/" \t "_blank)下载NSClient按提示一步步安装就可以了，不详述

八、报警设置  
邮件报警的设置也非常容易，此文不再记录。

**九、pnp4nagios安装**

以pnp4nagios 0.4为例，rpm安装

1.安装pnp4nagios

yum install pnp4nagios rrdtool

**配置nagios**

2.修改nagios.cfg

1. vi /etc/nagios/nagios.cfg *//修改以下内容*
3. process\_performance\_data=1 *//由0改为1*
4. host\_perfdata\_command=process-host-perfdata *//前面的注释拿掉*
5. service\_perfdata\_command=process-service-perfdata *//注释拿掉*
6. enable\_environment\_macros=1

3.修改commands.cfg

注释掉原有对process-host-perfdata和process-service-perfdata，重新定义

1. vi /etc/nagios/objects/commands.cfg
3. define command {
4. command\_name process-service-perfdata
5. command\_line /usr/bin/perl /usr/libexec/pnp4nagios/process\_perfdata.pl
6. }
8. define command {
9. command\_name process-host-perfdata
10. command\_line /usr/bin/perl /usr/libexec/pnp4nagios/process\_perfdata.pl -d HOSTPERFDATA
11. }

4.修改配置文件templates.cfg,添加以下内容

1. vi /etc/nagios/objects/templates.cfg
3. define host {
4. name hosts-pnp
5. **register** 0
6. action\_url /nagios/pnp4nagios/graph?host=$HOSTNAME$
7. process\_perf\_data 1
8. }
9. define service {
10. name srv-pnp
11. **register** 0
12. action\_url /nagios/pnp4nagios/graph?host=$HOSTNAME$&srv=$SERVICEDESC$
13. process\_perf\_data 1
14. }

5.修改服务器配置文件localhost.cfg

2. vi /etc/nagios/objects/localhost.cfg
4. define host{
5. **use** linux-server,hosts-pnp
6. host\_name localhost
7. **alias** localhost
8. address 127.0.0.1
9. }
11. define service{
12. **use** **local**-service,srv-pnp
13. host\_name localhost
14. service\_description PING
15. check\_command check\_ping!100.0,20%!500.0,60%
16. }

6.重启nagios，启动npcd

service nagios reload

service npcd restart

7.配置apache/nginx  
pnp4nagios4.10版本apache至此不需要做任何配置，直接就可以访问了，升级到0.6后发现必须rewrite了

1. <Directory "/usr/share/nagios/html/pnp4nagios">
2. AllowOverride All
3. Order allow,deny
4. Options FollowSymlinks
5. Allow from all
6. <IfModule mod\_rewrite.c>
7. RewriteEngine On
8. Options FollowSymLinks
9. RewriteBase /nagios/pnp4nagios
10. RewriteRule ^(application|modules|system) - [F,L]
11. RewriteCond %{REQUEST\_FILENAME} !-f
12. RewriteCond %{REQUEST\_FILENAME} !-d
13. RewriteRule .\* index.php/$0 [PT,L,NE]
14. </IfModule>
15. </Directory>

如果websrv是nginx

新建 Nginx 站点配置（未验证）：

1. server {
2. listen 80;
3. server\_name pnp.**local**;
5. location / {
6. root /usr/share/nagios/html/pnp4nagios;
7. index index.html index.htm index.php;
8. **if** (!-e $request\_filename) {
9. rewrite ^/(.+)$ /index.php?kohana\_uri=$1 **last**;
10. **break**;
11. }
12. }
14. location ~ \.php$ {
15. root /usr/share*//nagios/html/pnp4nagios;*
16. fastcgi\_index index.php;
17. fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;
18. include fastcgi\_params;
19. fastcgi\_pass unix:/tmp/php-fpm.sock;
20. }
21. }

小结：  
看起来好像已经很复杂了，其实只要理解了Nagios的监控的原理，理解Nagios的的对象（host/service/command）的意义，Nagios的配置还是非常简单的

附：  
通过前端web查看nagios的时候，在apache下一切正常，在nginx下竟然出现了：

It appears **as** though you **do** **not** have permission to view information **for** any of the hosts you requested

修改/etc/nagios/cgi.cfg里的用户名是无效的，最后无意间看到有一个

*#default\_user\_name=guest*

设置项，取消注释，修改为

default\_user\_name=nagiosadmin

重启nagios，一切正常，如此看来，原因在于nginx的auth\_basic方式传递过来的用户名并没有被nagios识别出来

初步完工，小秀一下成果  