企业监控利器Nagios实践

# 监控软件分类

mrtg：老牌的监控系统，网站带宽流量图，历史趋势图，也可以对资源进行监控，现在用的不多

cacti：特色是绘图，历史趋势图，通过插件可以实现报警。故障分析上较差，而且报警机制有待完善。

nagios:特点是专注与报警，也可以配置pnp，cacti，hyperic出图等。特别适合监控大量服务器上面的服务是否正常，通过插件实现页面配置管理。

zabbix：新出的出图报警软件。有页面配置管理，通过php程序绘图。对服务器的资源消耗稍大。

我司目前使用的是nagios监控系统。

# nagios监控工具

## 主机或服务状态监控

nagios是一款开源的监控软件，从它可以监控的设备类型上来看，主要包含网络设备，服务器设备。常见的网络设备如：路由器、交换机、防火墙、F5、打印机等，常见的服务器设备主要分为：UNIX类、Linux类以及Windows类。按我的理解凡是支持snmp协议的设备，包含PC都可以通过nagios进行监控。当然，nagios实现对主机资源及服务的监控并非全依靠snmp协议，它最为主要的监控手段是通过nrpe组件来实现。

## 监控告警通知

nagios对在网络中发现的问题会及时产生告警信息并通过事先定义好的方式，如邮件、短信、微信等方式通知相关人员。随着网络运维工作更加自动化，还可以通过nagios支持的相应API接口，开发相应的程序，实现其自动或人干预去对监控发现的问题进行自动化的处理。

## 监控信息可视化

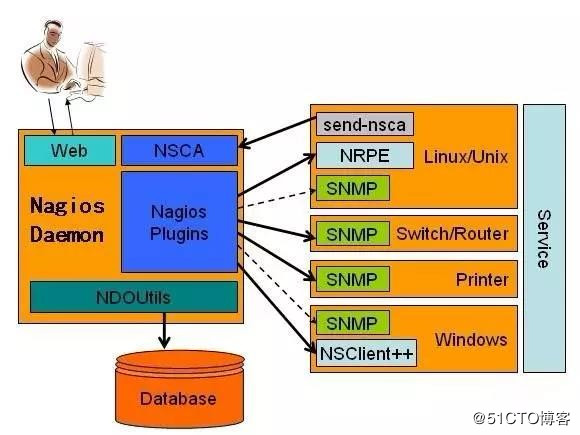
nagios结合web服务器，可以将整个网络所监控的所有信息以web页面的形式展现出来，还可以结合外部软件实现监控数据可视化，以图表的形式展示在web页面中，本文将介绍nagios常用的画图软件包pnp，现在叫pnp4nagios。

## 监控数据存储

nagios监控到的数据会存储下来，可以直接以文件的形式存储也可以通过NDOUtils组件存储到如mysql类的数据库中，从而可以很好支持监控历史数据的查询。

## nagios系统组成

nagios系统主要包含nagios daemon、nagios plugin、nrpe、web三个组件，它还包含NDOUtils、NSCA、NSClinet++组件，它们共同组成一个完整的nagios，组成逻辑图如下所示：



### Nagios Daemon

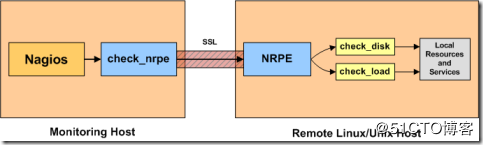
nagios系统的核心组件，它负责组织与管理各组件，将它们协调起来共同完成监控任务，并完成监控信息的组织与展示。

### Nagios Plugins

nagios plugins主要就是nagios核心组件自带以及用户自开发的一些插件，它们是实现各项监控的具体小程序，由它们将采集到相应的数据以后，回送给nagios服务器。

### NRPE

nagios系统要想取得被监控主机的存活状态、http、ftp、ssh服务是否可用，可以通过程序探测的出来，但如果要想取得被监控端上如磁盘容量，cpu负载这类本地信息时，如果没有相应的权限就不行，所以就产生了代理程序，事先在被监控机上安装代理程序(Linux系统是nrpe软件)，然后通过它们来获取监控数据，再回送给nagios服务器。当nrpe启动以后，它会开启5666端口。nrpe的工程原理如下图所示：



NRPE 总共由两部分组成：

check\_nrpe：位于nagios server上。

NRPE daemon：位于被监控的Linux主机上。

当Nagios 需要监控某个远程Linux 主机时：

nagios 会运行check\_nrpe 这个插件，告诉它要检查什么

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；

NRPE 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。

### NSClinet++

NSClient++这一组件是安装在windows主机上，相当于nagios server在windows端的代理程序。

### NSCA

NSAC这一组件适用于部署分布式nagios监控系统时使用，它可以实现让被监控端主动将需要监控的信息发送给nagios服务端。

## nagios安装---服务端

### 配置yum源采

用国内的yum，如阿里云的

### 解决perl编译问题

后面编译的软件有perl程序，提前设置环境变量。

echo "export LC\_ALL=C" >>/etc//profile

source /etc/profile

### 解决系统时间同步

不解决时间同步，会导致nagios配置异常甚至失败

/usr/sbin/ntpdate 0.pool.ntp.org

将时间同步写到定时任务中

echo "\*/10 \* \* \* \* root /usr/sbin/ntpdate 0.pool.ntp.org >/dev/null 2&>1">>/etc/crontab

### 安装nagios所需基础软件包

编译软件 只需要yum安装LAMP环境即可

yum install gcc glibc glibc-common -y

yum install gd gd-devel -y

yum install mysql\* -y

yum install httpd php php-gd -y

启动lamp服务

### 创建nagios用户及组

/usr/sbin/useradd nagios

/usr/sbin/groupadd nagcmd

/usr/sbin/usermod -a -G nagcmd nagios

/usr/sbin/usermod -a -G nagcmd apache

### 安装nagios

解压nagios

cd nagios

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

make all

make install

make install-init

make install-config

make install-commandmode

### 安装nagios web配置及创建登录用户

make install-webconf

htpasswd -c /usr/local/nagios/etc/htpasswd.users tuwei 创建用户并设置密码

也可以htpasswd -cb /usr/local/nagios/etc/htpasswd.users tuwei 123456无需交互

### 创建报警联系人信息

vim /usr/local/nagios/etc/objects/contacts.cfg +35

修改报警接收联系人mail

开启linux默认发邮件sendmail服务

### 安装nagios插件--服务端

#### 安装nagios-plugins

tar zxf nagios-plugins-1.4.16.tar.gz

cd nagios-plugins-1.4.16

./configure --with-nagios-user=nagios --with-nagios-group=nagios --with-mysql=/var/lib/mysql \

--enable-perl-modules

make && make install

检查插件个数：ls /usr/local/nagios/libexec |wc -l

添加nagios服务到开机自启动

chkconfig nagios on

启动nagios服务

查看相关信息/usr/local/nagios/bin/nagios

检查语法

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

简单地检查语法

/etc/init.d/nagios checkconfig

也可以修改nagios启动脚本，使其显示检查结果信息

vim /etc/init.d/nagios +177 删除>/dev/null 2>&1

#### 安装nrpe

服务端会生成/usr/local/nagios/libexec/check\_nrpe 插件

服务端同时也是客户端，需要被监控

./configure

make all

make install-plugin

make install-daemon

make install-daemon-config

## nagios客户端安装

无需安装lamp环境、nagios主程序

进行时间同步，关闭selinux

创建用户：/usr/sbin/useradd -m nagios -s /sbin/nologin

安装nagios-plugins插件

tar zxf nagios-plugins-1.4.16.tar.gz

cd nagios-plugins-1.4.16

./configure --prefix=/usr/local/nagios --enable-perl-modules --enable-redhat-pthread-workaround --如果是redhat则需要

make && make install

安装nrpe插件

tar zxf nrpe-2.12.tar.gz

cd nrpe-2.12

./configure

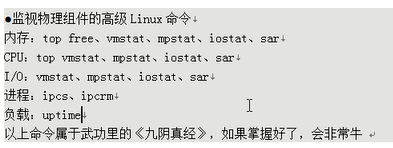
make all

make install-plugin

make install-daemon

make install-daemon-config

安装其他插件（check\_iostat需要的依赖包）



cd /tools/

tar zxvf Params-Validate-0.91.tar.gz

cd Params-Validate-0.91

perl Makefile.PL

make

make install

cd ..

#----------------------------------------

cd /tools/

tar zxvf Class-Accessor-0.31.tar.gz

cd Class-Accessor-0.31

perl Makefile.PL

make

make install

cd ..

#----------------------------------------

cd /tools/

tar zxvf Config-Tiny-2.12.tar.gz

cd Config-Tiny-2.12

perl Makefile.PL

make

make install

cd ..

#----------------------------------------

cd /tools/

tar zxvf Math-Calc-Units-1.07.tar.gz

cd Math-Calc-Units-1.07

perl Makefile.PL

make

make install

cd ..

#----------------------------------------

cd /tools/

tar zxvf Regexp-Common-2010010201.tar.gz

cd Regexp-Common-2010010201

perl Makefile.PL

make

make install

cd ..

配置nrpe

nrpe安装完后

[root@tuwei-blog tools]# cd /usr/local/nagios/etc/

[root@tuwei-blog etc]# ll

total 8

-rw-r--r-- 1 nagios nagios 7207 Sep 29 19:43 nrpe.cfg

[root@tuwei-blog etc]# cp nrpe.cfg nrpe.cfg.ori

vim nrpe.cfg +79

allowed\_hosts=192.168.132.20 --->改为nagios监控服务器，如有多台，以逗号隔开

sed -i '199,203d' /usr/local/nagios/etc/nrpe.cfg

echo "command[check\_load]=/usr/local/nagios/libexec/check\_load -w 15,10,6 -c 30,25,20">>/usr/local/nagios/etc/nrpe.cfg

echo "command[check\_mem]=/usr/local/nagios/libexec/check\_memory.pl -w 6% -c 3%">>/usr/local/nagios/etc/nrpe.cfg

echo "command[check\_disk]=/usr/local/nagios/libexec/check\_disk -w 20% -c 8% -p /">>/usr/local/nagios/etc/nrpe.cfg

echo "command[check\_swap]=/usr/local/nagios/libexec/check\_swap -w 20% -c 10%">>/usr/local/nagios/etc/nrpe.cfg

echo "command[check\_iostat]=/usr/local/nagios/libexec/check\_iostat -w 6 -c 10">>/usr/local/nagios/etc/nrpe.cfg



开启nrpe服务

/usr/local/nagios/bin/nrpe -c /usr/local/nagios/etc/nrpe.cfg -d

echo "/usr/local/nagios/bin/nrpe -c /usr/local/nagios/etc/nrpe.cfg -d

" >>/etc/rc.local

## 配置server端nagios监控服务

nagios目录

─[root@salt-master] - [/usr/local/nagios] - [626]

└─[$] ll [20:51:21]

total 32K

drwxrwxr-x 2 nagios nagios 4.0K Sep 28 22:29 bin

drwxrwxr-x 3 nagios nagios 4.0K Sep 28 22:29 etc

drwxr-xr-x 2 root root 4.0K Sep 28 21:23 include

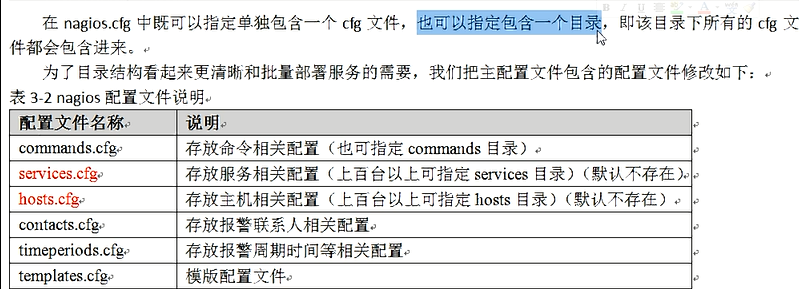
drwxrwxr-x 2 nagios nagios 4.0K Sep 28 22:29 libexec

drwxr-xr-x 5 root root 4.0K Sep 28 21:23 perl

drwxrwxr-x 2 nagios nagios 4.0K Sep 27 21:41 sbin

drwxrwxr-x 11 nagios nagios 4.0K Sep 28 21:23 share

drwxrwxr-x 5 nagios nagios 4.0K Sep 29 20:51 var



配置主配置文件nagios.cfg

找到cfg\_file部分，进行设置

vim /usr/local/nagios/etc/nagios.cfg +34

添加如下内容

cfg\_file=/usr/local/nagios/etc/objects/hosts.cfg

cfg\_file=/usr/local/nagios/etc/objects/services.cfg

cfg\_dir=/usr/local/nagios/etc/objects/services

去掉localhost.cfg配置

添加hosts和services文件及目录文件

cd /usr/local/nagios/etc/objects/

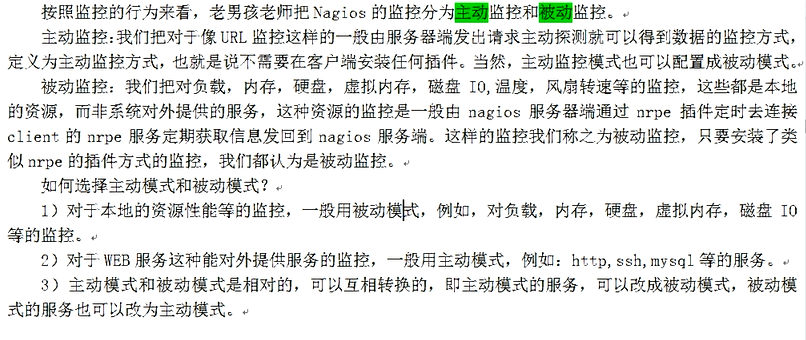
head -51 localhost.cfg >hosts.cfg

touch services.cfg

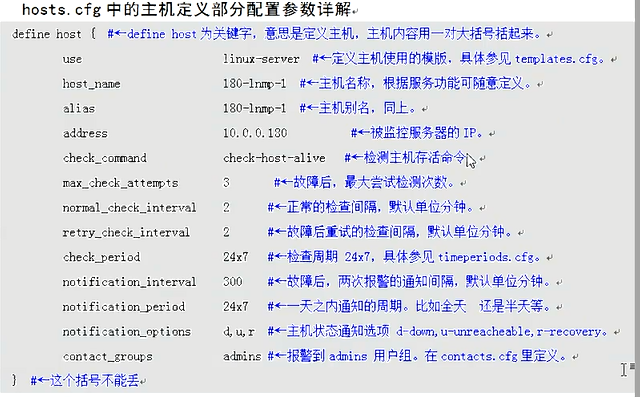
mkdir /usr/local/nagios/etc/objects/services

chown -R nagios.nagios \*

## 监控模式



## hosts.cfg生产配置



本次配置信息如下：共三台主机。

|  |  |  |
| --- | --- | --- |
| 主机名 | IP | 用户 |
| nagios-client1 | 192.168.132.16 | 客户端 |
| nagios-client2 | 192.168.132.14 | 客户端 |
| nagios-server | 192.168.132.20 | nagios服务端 |

vim hosts.cfg

define host{

use linux-server

host\_name 016-nagios-client1

alias 016-nagios-client1

address 192.168.132.16

}

define host{

use linux-server

host\_name 014-nagios-client2

alias 014-nagios-client2

address 192.168.132.14

}

配置文件里加了两台主机的配置。

定义一个主机组

define hostgroup{

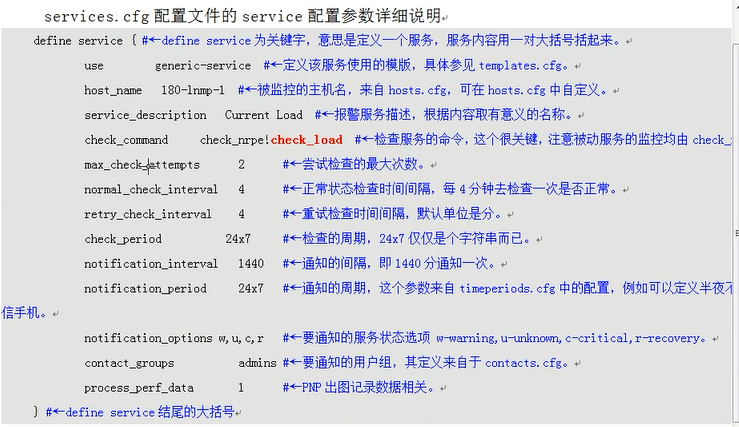
hostgroup\_name linux-servers

alias Linux Servers

members 016-nagios-client1,014-nagios-client2

}

## services.cfg配置



vim services.cfg 以磁盘监控为例

define service{

use generic-tuwei-service

host\_name 016-nagios-client1,014-nagios-client2

service\_description disk

check\_command check\_nrpe!check\_disk

max\_check\_attempts 8

normal\_check\_interval 3

retry\_check\_interval 2

check\_period 24x7

notification\_period 24x7

notification\_interval 360

notification\_options w,u,c,r

contact\_groups admins

process\_perf\_data 1

}也可以将相同的部分放到模板中。

检查语法/etc/init.d/nagios checkconfig

有报错，因为check\_nrpe还没有定义

vim commands.cfg

添加如下内容nrpe定义

# 'check\_nrpe' command definition

define command{

command\_name check\_nrpe

command\_line $USER1$/check\_nrpe -H $HOSTADDRESS$ -c $ARG1$

}

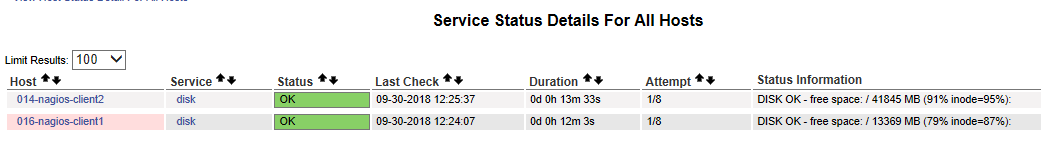
编辑cgi配置文件，进行授权。由于之前创建web登陆用户为tuwei，没有权限查看主机及服务信息。

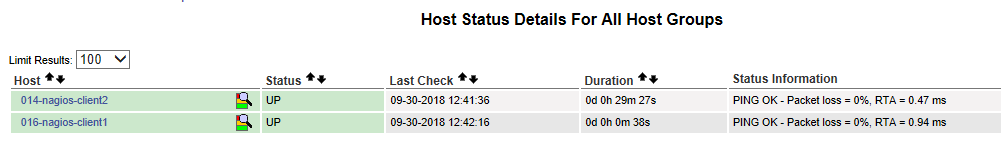
可以将nagiosadmin用户替换为tuwei，或者在后面加上tuwei，用逗号隔开。

:g/nagiosadmin/s//tuwei/g

启动nagios服务

登陆web界面查看services和hosts信息





配置目录生效，之前在nagios主配置文件中加入了目录。

则只要该目录下有\*.cfg的文件都会被nagios加载。比如按照IP来命名配置文件。可以批量生成配置文件。

## nagios模板并开发脚本批量配置模板

把所有服务的配置都简写，然后把相关参数放到模板templates.cfg中

通过脚本批量生成模板，按照监控项，如按照物理服务器资源。

## 服务端口及URL监控

nagios服务端发起监控。

此类服务一般都是开启了对外提供服务的业务。

端口监控，使用的是check\_tcp插件，主动获取客户端信息

[root@nagios-server libexec]# ./check\_tcp -H 192.168.132.14 -p 80

TCP OK - 0.001 second response time on port 80|time=0.000735s;;;0.000000;10.000000

define service{

use generic-service

host\_name 016-nagios-client1,014-nagios-client2

service\_description http\_80

check\_command check\_tcp!80

}

其他参数类似。

如果是监控ping

define service{

use generic-service

host\_name 016-nagios-client1,014-nagios-client2

service\_description ping

check\_command check\_ping!100.0,20%!500.0,60%

}

url监控实质：命令行理解http监控原理。

url监控和端口类似

define service{

use generic-service

host\_name 016-nagios-client1,014-nagios-client2

service\_description http\_url

check\_command check\_weburl! -H blog.tuwei.org -u /test.html

}

在客户端配置域名主机服务，由于是测试环境，在电脑hosts文件中加入ip和对应域名

在command.cfg配置中加入weburl信息。

# 'check\_weburl' command definition

define command{

command\_name check\_weburl

command\_line $USER1$/check\_http -H $ARG1$ -w 10 -c 30

}

特殊字符的url地址进行监控：将url地址添加""即可。

生产环境中可以将以上端口、url都作为模板

## 服务分组

语法如下

define servicegroup {

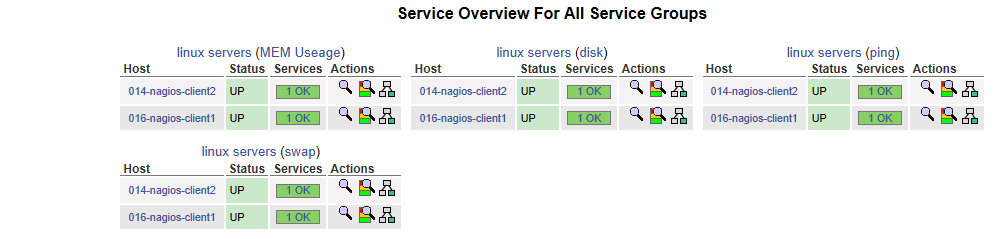
servicegroup\_name MEM Useage

alias linux servers

members 016-nagios-client1, MEM Useage, 014-nagios-client2, MEM Useage

}

修改模板中关于mem，将描述改为MEM Useage。在services.cfg文件中添加服务组信息。



如上，添加了4个服务组。在生产环境中根据业务服务进行查看。

批量生成host信息，如host1,host2,host3，用在service配置中

#!/bin/bash

num=`cat iplist |wc -l`

i=1

>/tmp/tmp.log

cat iplist|while read line

do

if [ $i -eq $num ];then

echo -n "`echo $line|awk '{print $1}'`">>/tmp/tmp.log

else

#echo -n "`echo $line|awk '{print $1}'`,dovecot,">>/tmp/tmp.log

echo -n "`echo $line|awk '{print $1}'`,">>/tmp/tmp.log

fi

let i+=1

done

批量生成host

#!/bin/sh

cat iplist|while read line

do

host=`echo $line|awk '{print $1}'`

ip=`echo $line|awk '{print $2}'`

cat >>hosts.cfg<<EOF

define host{

use linux-server

host\_name $host

alias $host

address $ip

}

EOF

done

## nagios图形监控管理（服务端）

Pnp安装图形监控曲线

pnp网站：[www.pnp4nagios.org](http://www.pnp4nagios.org)

安装pnp软件依赖包

yum -y install cairo pango zlib zlib-devel freetype freetype-devel gd gd-devel

安装libart\_lgpl和rrdtool，这里以编译安装为例。

tar xf libart\_lgpl-2.3.17.tar.gz

cd libart\_lgpl-2.3.17

./configure

make && make install

cp -r /usr/local/include/libart-2.0/ /usr/include/

tar xf rrdtool-1.2.14.tar.gz

cd rrdtool-1.2.14

./configure --prefix=/usr/local/rrdtool --disable-python --disable-tcl

make

make install

[root@salt-master rrdtool-1.2.14]# ls -l /usr/local/rrdtool/bin

total 116

-rwxr-xr-x 1 root root 55641 Oct 2 18:36 rrdcgi

-rwxr-xr-x 1 root root 6727 Oct 2 18:36 rrdtool

-rwxr-xr-x 1 root root 52635 Oct 2 18:36 rrdupdate

安装成功

安装pnp

tar xf pnp-0.4.14.tar.gz

cd pnp-0.4.14

./configure --with-rrdtool=/usr/local/rrdtool/bin/rrdtool \

--with-perfdata-dir=/usr/local/nagios/share/perfdata

make all

make install

make install-config

make install-init

查看。

[root@salt-master pnp-0.4.14]# ll /usr/local/nagios/libexec/|grep process

-rwxr-xr-x 1 nagios nagios 31826 Oct 2 19:33 process\_perfdata.pl

表示安装成功

nagios出图基本配置

编辑nagios配置文件

备份cp nagios.cfg nagios.cfg.ori

vim nagios.cfg +835

将process\_performance\_data=0 改为process\_performance\_data=1

sed -i 's#process\_performance\_data=0#process\_performance\_data=1#' nagios.cfg

将如下两行的#去掉

#host\_perfdata\_command=process-host-perfdata

#service\_perfdata\_command=process-service-perfdata

修改command配置文件，添加收集数据配置。

删除里面关于process-service-perfdata的配置并添加以下内容

# 'process-host-perfdata' command definition

define command{

command\_name process-host-perfdata

command\_line $USER1$/process\_perfdata.pl

}

# 'process-service-perfdata' command definition

define command{

command\_name process-service-perfdata

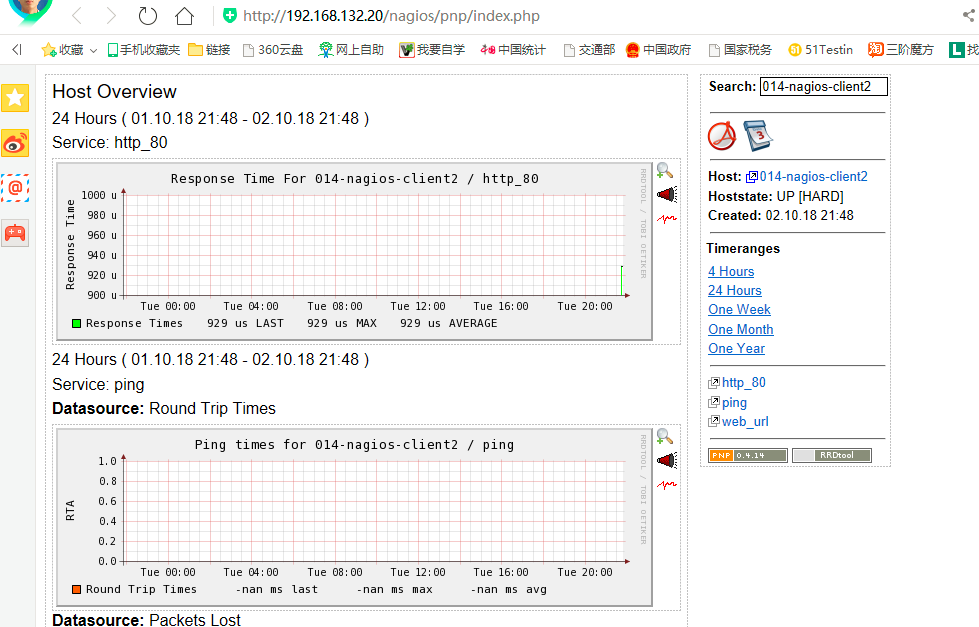
command\_line $USER1$/process\_perfdata.pl

}

模板文件中有process\_perf\_data 1参数

检查语法并reload nagios服务。

查看<http://192.168.132.20/nagios/pnp/index.php>



整合pnp url链接到nagios图形显示界面中

在hosts.cfg中添加或者在模板中添加（主机出图）

action\_url /nagios/pnp/index.php?host=$HOSTNAME$

服务出图，在模板中添加

action\_url /nagios/pnp/index.php?host=$HOSTNAME$&srv=$SERVICEDESC$

重启nagios

出图数据存放的路径

[root@nagios-server tools]# ll /usr/local/nagios/share/perfdata/

total 8

drwxr-xr-x 2 nagios nagios 4096 Oct 2 22:49 014-nagios-client2

drwxr-xr-x 2 nagios nagios 4096 Oct 2 22:49 016-nagios-client1

出图是为了将历史记录保留，可以预判未来的趋势。在生产环境中比较重要。

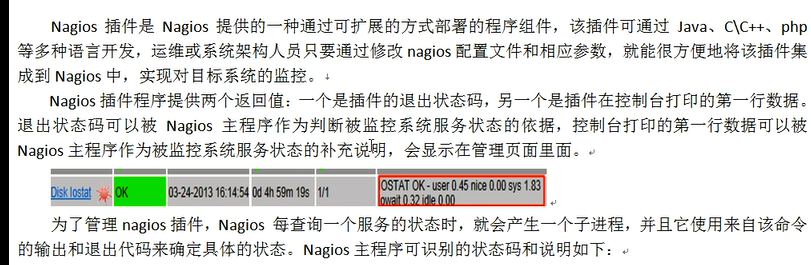
## nagios插件开发

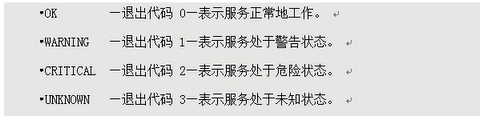
安装nagios-plugins后会生成一些nagios插件，nagios仅仅是一个监控平台。要监控

主机及服务需要配置或调用插件或程序文件才能完成。

虽然有一些插件软件，但生产环境中我们想监控的一些服务。nagios插件没有。，如nfs状态。负载均衡RS的VIP等等。

### 编写nagios插件说明





相关状态可以从如下文件中看到

[root@nagios-server libexec]# head -7 utils.sh

#! /bin/sh

STATE\_OK=0

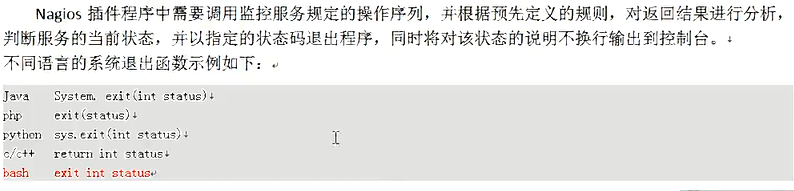
STATE\_WARNING=1

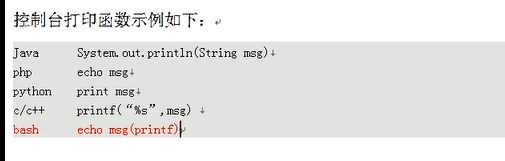
STATE\_CRITICAL=2

STATE\_UNKNOWN=3

STATE\_DEPENDENT=4 --不常用

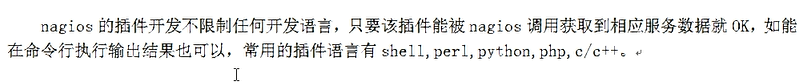
### nagios插件开发原理





以上两点很重要：1，给返回状态码 2，打印状态说明。

### nagios插件开发语言



### 使用shell开发nagios插件

变量测试

[root@nagios-server tools]# basename /etc/inittab ---脚本名称

inittab

[root@nagios-server tools]# dirname /etc/inittab 路径

/etc

开发插件需要用到的：

echo $0

PROGNAME=`basename $0`

PROGPATH=`dirname $0`

echo $PROGNAME

echo $PROGPATH

[root@nagios-server scripts]# sh /server/scripts/a.sh

/server/scripts/a.sh

a.sh

/server/scripts

监控url插件。

[root@nagios-server libexec]# cat check\_url.sh

#!/bin/bash

##############################################################

# File Name: check\_weburl.sh

# Version: V1.0

# Author: tuwei

# Blog Site: http://blog.51cto.com/tuwei

# Created Time : 2018-10-03 11:02:28

# Environment: CentOS 6.9 Kernal 2.6.32

##############################################################

PROGNAME=`basename $0`

PROGPATH=`dirname $0`

. $PROGPATH/utils.sh

if wget -T 10 --spider http://192.168.132.16>/dev/null 2>&1

then

echo 'HTTP OK: HTTP/1.1 200 OK '

exit $STATE\_OK -------简单写法exit 0

else

echo "HTTP is not ok,http\_code:`curl -I -m 10 -o /dev/null -s -w %{http\_code} 192.1132.16`"

exit $STATE\_CRITICAL--------------简单写法exit 2

fi

执行脚本，注意要用全路径。

[root@nagios-server libexec]# sh /usr/local/nagios/libexec/check\_url.sh

HTTP OK: HTTP/1.1 200 OK

将192.168.132.16 http服务关闭。

[root@nagios-server libexec]# sh /usr/local/nagios/libexec/check\_url.sh

HTTP is not ok,http\_code:000

将该脚本修改为通用脚本。将IP修改为变量，并判断参数个数。

### 如何使用插件

chmod 755 check\_url.sh

chown nagios.nagios check\_url.sh

在command配置文件中定义

# 'check\_url' command definition

define command{

command\_name check\_url

command\_line $USER1$/check\_url http://blog.tuwei.org/test.html

}

在services.cfg中添加url服务

如果是要开发如监控密码文件，则需要在监控端开发，不是服务端。

#!/bin/sh

if [ `md5sum -c /mnt/ps|grep OK|wc -l` -eq 1 ];then

echo "/etc/passwd is ok"

exit 0

else

echo "/etc/passwd FAILED"

exit 2

fi

监控mysql，利用自带的mysql插件查询

[root@tuwei-blog nagios]# /usr/local/nagios/libexec/check\_mysql -H 192.168.132.16 -urep -prep99

Uptime: 149369 Threads: 2 Questions: 7 Slow queries: 0 Opens: 16 Flush tables: 1 Open tables: 4 Queries per second avg: 0.0

服务端开发插件简介：

1. 确认插件[root@nagios-server objects]# ll /usr/local/nagios/libexec/check\_http

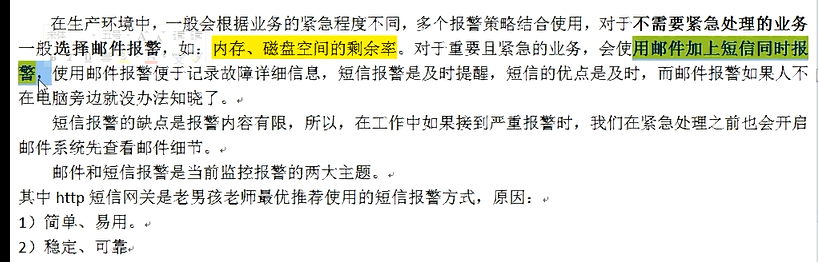
-rwxr-xr-x 1 nagios nagios 520606 Sep 28 21:23 /usr/local/nagios/libexec/check\_http 需要有执行权限

1. 确实command.cfg配置中是否定义了上述插件
2. 编辑服务配置文件，定义相关服务，使用上述已经定义的插件命令
3. 检查语法
4. 加载配置

客户端开发插件：

1. 确认开发插件命令是否存在/usr/local/nagios/libexec，需要有执行权限
2. 确认/usr/local/nagios/etc/nrpe.cfg配置文件中是否定义
3. 重启nrpe服务
4. 到服务器端编辑服务配置文件，定义相关服务，使用上述已经定义的插件命令
5. 检查语法
6. 加载配置

## 报警策略



command.cfg中定义

模板中定义

http短信网关报警

define contact{

contact\_name nagiosadmin

use generic-contact

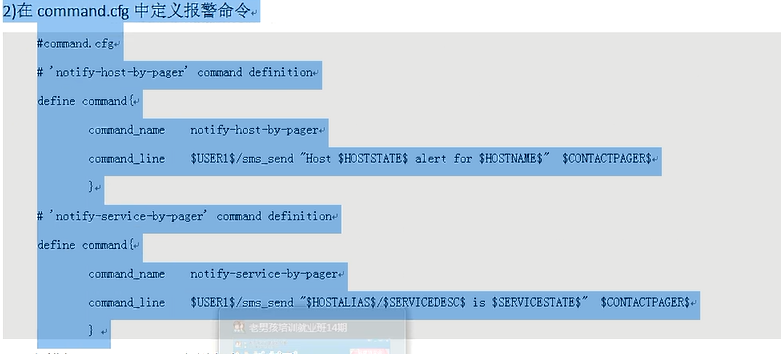
alias Nagios Admin

email 1530213847@qq.com

pager xxxxxxxxx

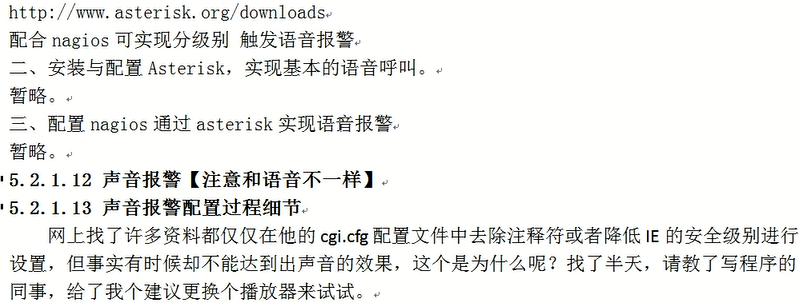
}

手机报警command配置。



模板中定义以上命令。sms\_send为调用http短信网关脚本。

电话语音报警



nagios通过asterisk实现语音报警

<http://gaojing.baidu.com>百度告警

部署

接入步骤：

Step 1：在百度告警中配置服务和升级策略。

Step 2：在Nagios中配置command.cfg，配置百度告警接方式。

Step 3：在Nagios中配置templates.cfg，配置联系人使用百度告警接收告警。

Step 4：测试配置是否成功，使用电话、短信、邮件、微信接收告警！

通过火狐浏览器的nagios checker实现web声音告警，但注意火狐浏览器版本不能过高，建议55及以下。

通过onealert平台http://www.onealert.com/实现微信、短信、电话及钉钉告警

### 设置主机和服务告警次数及频率

在实际工作中可能有的告警不影响业务，但如果没有处理，则会一直收到邮件等告警，造成告警泛滥，这时我们可以设置主机及服务告警频率。

vi /usr/local/nagios/etc/objects/escalations.cfg

define hostescalation{

host\_name test ;被监控主机名称，多个用逗号隔开与Hosts.cfg中一致

first\_notification 3 ; 第3条信息起，改变频率间隔

last\_notification 0 ; 第n条信息起，恢复频率间隔

notification\_interval 10 ; 通知间隔(单位：分)

contact\_groups admin

}

define serviceescalation{

host\_name test ;被监控主机名称，多个用逗号隔开与Hosts.cfg中一致

service\_description PING 被监控服务名称，多个用逗号隔开 与services.cfg中一致

first\_notification 3

last\_notification 0

notification\_interval 10

contact\_groups admin

}

#vi /usr/local/nagios/etc/nagios.cfg

添加：

cfg\_file=/usr/local/nagios/etc/objects/escalations.cfg

### 设置主机和服务依赖

主机和服务依赖是nagios的高级行为。可以根据实际情况设置主机和服务依赖关系，从而进行告警收敛，减少告警次数，而且更有利于问题的排查，从而更快地解决问题，使系统服务恢复正常。参考数据

<https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/4/en/dependencies.html>

如下样例：

define servicedependency {

host\_name 016-nagios-client1

service\_description http\_80

dependent\_host\_name 016-nagios-client1

dependent\_service\_description web\_url

execution\_failure\_criteria n

notification\_failure\_criteria w,u,c

}

常见状态：o: OK; w: WARNING; u: UP; c: CRITICAL; p: PENDDING; n:NONE

测试结果：同一台主机上的web\_url服务依赖http\_80（检测httpd服务的80端口）服务，当http\_80服务状态为w、u、c任意一种时，http\_80状态会发告警通知，由于execution\_failure\_criteria n（none）的设置，就不会再有web\_url的告警信息。另外如果是同一台主机的话dependent\_host\_name 016-nagios-client1可以去掉。

如果是同一主机组中的主机有相同的服务，且也有服务依赖，可以用以下方法进行配置。

define servicedependency{

hostgroup\_name linux-servers

dependent\_service\_description web\_url

service\_description http\_80

execution\_failure\_criteria n

notification\_failure\_criteria w,u,c

}

说明：主机组linux-servers包含两台主机016-nagios-client1和014-nagios-client1，两台主机上都有http\_80和web\_url服务，设置服务依赖后，测试结果同上（只有http\_80的告警通知）。

依赖关系继承：例如服务A和服务D有依赖关系，服务D和服务F也有依赖关系。可以通过以下设置将A和D的依赖关系继承。

define servicedependency{

host\_name HostB

service\_description ServiceD

dependent\_host\_name HostC

dependent\_service\_description ServiceF

execution\_failure\_criteria n

notification\_failure\_criteria w,u,c

inherits\_parent 1

}

因为inherits\_parent指令开启，所以A和D之间的依赖关系测试通过后，才会进行F和D之间的检查。

依赖关系可以被多层继承。

设置主機依赖

define hostdependency {

host\_name Host A

dependent\_host\_name Host C

notification\_failure\_criteria d

}

define hostdependency {

host\_name Host B

dependent\_host\_name Host C

notification\_failure\_criteria d,u

}

主机常见状态：o是UP状态，d是down状态，u是unreache p是pending 未检测状态。主机依赖场景：如一台实体机通过虚拟化后有多台虚拟机，如果设置了虚拟机和实体机的依赖，当实体机宕机时，就不在发出虚拟机的告警通知，从而大大减少了告警次数。

## nagios补充

验证配置文件的正确性

每次修改过你的配置文件，你应该运行一次检测程序来验证配置的正确性。在运行你的Nagios程序之前这是很重要的，否则的话会导致Nagios服务因配置的错误而关闭。

为验证你配置，运行Nagios带命令行参数 -v，象这样：

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

初始化脚本：最简单的启动Nagios守护进程的方式是使用初始化脚本，象这样：

/etc/rc.d/init.d/nagios start

手工方式：你可以手动地启动Nagios守护进程，用命令参数-d，象这样：

/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg

快速启动选项

介绍

只有很少几件事可以减少Nagios的启动或重启总时间。加速启动方法包括有移除些负担还包括加快配置文件处理过程。

利用这些技术在如下一种或几种情况时特别有效：

大型安装配置

复杂地配置(过度地利用模板特性)

需要进行频繁重启动的安装模式

背景

每次Nagios启动和重启时，在它着手进行监控工作之前必须要处理配置文件。启动过程中的配置处理包括如下几步：

读入配置文件

解析模板定义

重粘连("Recombobulating")对象(是我想到的应做各种工作)

复制对象定义

继承对象属性

对象定义排序

验证对象关联关系的完整性

验证回路

和其他...

当有很大的或是很复杂的配置文件要处理时有几步非常消耗时间的。有没有加快这些的办法？当然有！

评估启动时间

在做让启动速度更快的事情之前，需要看看可能性有多少和是否有必要涉足此事。这个比较容易－只是用-s命令行开关启动Nagios以取得计时和调度信息。

下面是个输出样例(做过精减，只是显示了有关部分)，在这个例子中，假定Nagios配置为对25个主机和超过10,000个服务进行监控。

/usr/local/nagios/bin/nagios -s /usr/local/nagios/etc/nagios.cfg

Nagios 3.0-prealpha

Copyright (c) 1999-2007 Ethan Galstad (http://www.nagios.org)

Last Modified: 01-27-2007

License: GPL

Timing information on object configuration processing is listed

below. You can use this information to see if precaching your

object configuration would be useful.

Object Config Source: Config files (uncached)

OBJECT CONFIG PROCESSING TIMES (\* = Potential for precache savings with -u option)

----------------------------------

Read: 0.486780 sec

Resolve: 0.004106 sec \*

Recomb Contactgroups: 0.000077 sec \*

Recomb Hostgroups: 0.000172 sec \*

Dup Services: 0.028801 sec \*

Recomb Servicegroups: 0.010358 sec \*

Duplicate: 5.666932 sec \*

Inherit: 0.003770 sec \*

Recomb Contacts: 0.030085 sec \*

Sort: 2.648863 sec \*

Register: 2.654628 sec

Free: 0.021347 sec

============

TOTAL: 11.555925 sec \* = 8.393170 sec (72.63%) estimated savings

Timing information on configuration verification is listed below.

CONFIG VERIFICATION TIMES (\* = Potential for speedup with -x option)

----------------------------------

Object Relationships: 1.400807 sec

Circular Paths: 54.676622 sec \*

Misc: 0.006924 sec

============

TOTAL: 56.084353 sec \* = 54.676622 sec (97.5%) estimated savings

OK，看看发生了什么。先看汇总信息，大概有11.6秒用于处理配置文件有56秒来验证配置。这意味着每次用这个配置启动或重启Nagios时，它大约会有68秒来做启动事项而不会做任何监控的事情！如果是在定制配置Nagios过程中也是不可容忍的。

那么怎么办？看一下输出内容，如果运用了优化选项，Nagios将可以在配置读取过程节省大约8.4秒而在验证过程可节省63秒。

哇！从68秒到只有5秒？！是的！看看下面是怎么做到的。

预缓存对象配置

Nagios可在解析配置文件过程中做些加速，特别是当配置中使用了模板来做继承等的时候。为降低Nagios解析配置文件的处理时间可用Nagios预处理与预缓存配置文件的功能。

当用-p命令参数来运行Nagios时，Nagios将读入配置文件，处理后将配置结果写入预缓存文件(由主配置文件中precached\_object\_file域指定文件位置)。该预缓存配置文件将包含了预处理后的信息将使Nagios处理配置文件更容易和快捷。必须把-p参数选项与-v或-s命令参数一起使用，如下例。注意要做预缓存配置文件之前配置应是已被验证过的。

/usr/local/nagios/bin/nagios -pv /usr/local/nagios/etc/nagios.cfg

预缓存配置文件有大小明显地比原有配置文件大。这是正常的由设计初衷决定的。

一旦预缓存对象配置文件创建，可以启动Nagios时带上-u命令行选项以让它使用预缓存配置文件而不是配置文件本身。

/usr/local/nagios/bin/nagios -ud /usr/local/nagios/etc/nagios.cfg

如果更改了配置文件，必须在Nagios重启动前要重新验证和重建预缓存配置文件。如果没有重构建预缓存配置文件，Nagios将使用旧配置运行因为是由旧配置生成的预缓存文件，而不是用新的原始配置文件。

跳过回路检测

/usr/local/nagios/bin/nagios -xd /usr/local/nagios/etc/nagios.cfg

联合起来使用

按照下面步骤将会使用预缓存配置文件并且跳过回路检测以充分加速启动。

1、验证配置文件并生成预缓存配置文件，用如下命令：

/usr/local/nagios/bin/nagios -vp /usr/local/nagios/etc/nagios.cfg

2、如果Nagios正在运行，停掉它；

3、启动Nagios，让其使用预缓存配置文件而且跳过回路检测：

/usr/local/nagios/bin/nagios -uxd /usr/local/nagios/etc/nagios.cfg