Nagios 安装配置详解

Hadoop 集群监控系列

2015/04

河南省电力公司电力科学研究院智能电网所

ShangBingBing

版本	作者	备注
v1.0	商兵兵	2015.04.10，编写完成 v1.0 版本。

目录

1. 前言

最近在研究云监控的相关工具，包括 Nagios、Ganlis 等监控软件，本文记录下 Nagios 的安装配置步骤。

本文不讲解相关原理,若想了解请参考其他资料。

目的：即使之前未触过 nagios,也能按照文中步骤搭建自己的 nagios 监控集群。

作者：商兵兵

部门：河南省电力科学研究院智能电网所

2. nagios 简介

Nagios 是一个可运行在 Linux/Unix 平台之上的开源监视系统，可以用来监视系统运行状态和网络信息。Nagios 可以监视所指定的本地或远程主机以及服务，同时提供异常通知功能。在系统或服务状态异常时发出邮件或短信报警第一时间通知网站运维人员，在状态恢复后发出正常的邮件或短信通知。

Nagios 是一个功能非常强大的开源的系统网络监测程序，通过访问 http://www.nagios.org 可以了解其基本特性。Nagios 不但能够实现对系统 CPU，磁盘、网络等方面参数的基本系统监测，而且还能够监测包括 SMTP，POP3，HTTP，NNTP 等各种基本的服务类型。另外通过一些插件的安装和监测脚本自定义用户可以针对自己的应用程序实现监测，并针对大量的监测主机和多个对象部署层次化的监测架构。而且在监测信息统计方面，Nagios 也能够和例如 Cacti 等程序结合来提供动态统计图表。除此之外 Nagios 拥有强大的日志管理系统，可以实现详细的日志记录以及回卷。针对架构的扩展和服务器数量的增加可以方便地实现监测区域扩展。最难能可贵的是 Nagios 提供了优秀的事件报警功能，能够将一些突发的事件以电子邮件的形式通知管理员并能够针对出现的问题提供一些主动的解决建议和方案，并支持冗余监视。

相对于 Mrtg 以及 RRDtool + Cacti 而言 Nagios 最大的特点之一是其设计者将 Nagios 设计成监测的管理中心尽管其功能是监测服务和主机，但是他自身并不包括这部分功能的代码，所有的监测、监测功能都是由相关插件来完成的，包括报警功能。Nagios 自身也没有报警部分的代码和插件，而是交给用户或者其他相关开源项目组去完成。对于 Nagios 这个监测中心来说，细致的工作必然是交给其他的软件来实现。

注意，Nagios 只能安装在 Linux 或 Unix 平台，不支持 Windows 平台，当然，它可以监控 Windows 平台。

3. 安装规划

3.1 Hadoop 集群概况

主机名称	IP 地址	操作系统	作用
master.hadoop	192.168.195.10	CenterOS 6.5-32bit	Namenode
slave1.hadoop	192.168.195.11	CenterOS 6.5-32bit	Datanode
slave2.hadoop	192.168.195.12	CenterOS 6.5-32bit	Datanode
slave3.hadoop	192.168.195.13	CenterOS 6.5-32bit	Datanode
slave4.hadoop	192.168.195.14	CenterOS 6.5-32bit	Datanode
slave5.hadoop	192.168.195.15	CenterOS 6.5-32bit	Datanode

3.2 部署规划

监控服务主节点(Master)	
项	值
安装主机	master.hadoop
Nagios 主节点需要安装	■ nagios ■ nagios-plugin ■ nrpe ■ php ■ apache

被监控从节点(Slave)	
项	值
安装主机	■ slave1.hadoop ■ slave2.hadoop

	■ slave3.hadoop ■ slave4.hadoop ■ slave5.hadoop
Nagios 从节点需要安装	■ nagios-plugin ■ nrpe

3.3 安装路径规划

项	值
apr	/usr/local/apr
apr-util	/usr/local/apr-util
pcre	/usr/local/pcre
nagios	/usr/local/nagios
nrpe	/usr/local/nrpe
apache	/usr/local/apache
php	/usr/local/php

4. 安装组件包

4.1 检查组件包

Apache 和 PHP 不是安装 Nagios 所必须的，其实也可以说是必须的，你总不能去分析 Nagios 日志吧，因为 Nagios 提供了 Web 监控界面，通过 Web 监控界面可以清晰的看到被监控主机、服务、资源的运行状态，因此，安装配置一个 Web 服务环境是很必要的。

注意：根据官方要求，Apache 必须安装 APR、APR-Util、PCRE，gcc-c++等包。

在安装 Nagios 之前，首先需要保证系统中有以下这些软件包：apache、gcc、gd、gd-devel、glibc、glibc-devel，如下表所示。

组件包	描述
gcc	gcc，为 GNU 编译器套件（GNU Compiler Collection）包括 C、C++、Objective-C、Fortran、Java、Ada 和 Go 语言的前端，也包括了这

	些语言的库（如 libstdc++、libgcj 等等）。
gcc-c++	Linux 环境中的 C++编译器。
glibc	
glibc-common	
glibc-devel	
gd	
gd-devel	
apr	
apr-util	
pcre	
xinetd	
openssl-devel	

在安装某个组件包之前，可以用 rpm 的方式检查以下是否已经安装过。如下图所示，使用此命令，系统会列出已经安装软件包的名称及版本号，未安装的系统会提示“is not installed”。

```
[root@master ~]# rpm -q gcc gcc-c++ glibc glibc-common glibc-devel gd gd-deve
l apr apr-util pcre
gcc-4.4.7-4.el6.i686
gcc-c++-4.4.7-4.el6.i686
glibc-2.12-1.132.el6.i686
glibc-common-2.12-1.132.el6.i686
glibc-devel-2.12-1.132.el6.i686
gd-2.0.35-11.el6.i686
gd-devel-2.0.35-11.el6.i686
apr-1.3.9-5.el6_2.i686
apr-util-1.3.9-3.el6_0.1.i686
pcre-7.8-6.el6.i686
[root@master ~]#
```

对于已经安装的软件包，该如何查看其安装位置等详细信息呢？仍然可以使用 rpm 命令，加个参数-ql 即可。如下图所示查看 apr 的安装详细信息。

```
[root@master ~]# rpm -ql apr
/usr/lib/libapr-1.so.0
/usr/lib/libapr-1.so.0.3.9
/usr/share/doc/apr-1.3.9
/usr/share/doc/apr-1.3.9/CHANGES
/usr/share/doc/apr-1.3.9/LICENSE
/usr/share/doc/apr-1.3.9/NOTICE
[root@master ~]#
```

4.2 删除组件包

有些 linux 操作系统，譬如 CentOS 6.5 等，默认已经安装有 apr、apr-util、pcre、gcc 等包，但是没有安装 gcc-c++包，遇到这种情况时，建议从系统安装光盘中拷贝安装，因为这样可以保证和已经安装的其他软件包在版本上不会冲突，如果从网络上下载其他的版本，则可能会存在版本冲突而导致安装失败。另外，CentOS 默认安装的组件包可能存在版本冲突的现象，如果确定是版本冲突的问题,则可以先删除系统默认安装的组件包。举个例子要删除 apache，可以使用如下命令：rpm -e httpd-2.2.3
后面的版本号不全也没关系，如果有依赖关系就不能卸载，再加上--nodeps 参数即可。不检查依赖强制删除，这个结果就是只删除了 httpd，跟他有依赖关系的其它软件是不会删除的，但是这些软件因为系统里没有了 httpd 也会不能运行，这是所谓的没有删除干净。

4.3 安装组件包

我们在安装 Nagios 或者 apache 所需的组件包时，发现需要安装很多底层的组件包，这个安装过程是非常繁琐的，你会发现，当安装某个组件包时，系统会提示您安装错误，并告知你安装此组件包之前，您必须先安装另外某个组件包，等等。

当我们面对一个集群时，安装组件包将是一个非常耗费精力的工作，而且极容易出现错误，因此，我们在安装过程中总结了一些经验教训，以求起到事半功倍的效果。

在第一台主机上配置安装组件包时，请顺便记录下每个组件的安装顺序，并将安装命令组织成为一个安装脚本，然后在其他主机上安装组件包时，执行此安装脚本即可。
这里我们不再详细讲解各个组件包的安装过程。

5. 安装 apache

Apache HTTP Server（简称 Apache）是 Apache 软件基金会的一个开放源码的网页服务器，可以在大多数计算机操作系统中运行，由于其多平台和安全性被广泛使用，是最流行的 Web 服务器端软件之一。它快速、可靠并且可通过简单的 API 扩展，将 Perl/Python 等解释器编译到服务器中。关于更多 Apache 的介绍，可以参考百科“http://baike.baidu.com/view/28283.htm 或官网 http://www.apache.org/ 或 http://httpd.apache.org/的介绍

5.1 apache 软件下载

Apache 的相关软件包下载地址 http://apr.apache.org/projects.html

1. Apache HTTP Server

截止目前为止，Apache HTTP Server 目前最新的版本是 Apache httpd 2.4.6 Released， 下载地址：http://httpd.apache.org/download.cgi#apache24

2. APR and APR-Util 包

截止目前为止，APR and APR-Util 的最新版本如下，下载地址：http://apr.apache.org/download.cgi

APR 1.4.8, released June 21, 2013

APR-util 1.5.2, released April 4, 2013

APR-iconv 1.2.1, released November 26, 2007

3. PCRE 包

截止目前为止，PCRE 最新的包为 8.33，下载地址如下

https://sourceforge.net/projects/pcre/files/pcre/

5.2 apache 安装过程

5.2.1 安装 gcc 或 gcc-c++

如果系统中没有安装 gcc 等组件，则使用下面的命令进行安装。

```
[root@master tools]# rpm -ivh gcc-c4.4.7-11.el6.i686.rpm
[root@master tools]# rpm -ivh gcc-c++-4.4.7-4.el6.i686.rpm
```

5.2.2 安装 APR

新建目录/usr/local/apr，用作安装目录，将 apr 安装包解压缩到此目录下即可。

```
[root@master apr]# tar -zxf apr-1.5.2.tar.gz
[root@master apr]# ./configure --prefix=/usr/local/apr
[root@master apr]# make
```

```
[root@master apr]# make install
```

5.2.3 安装 APR-Util

新建目录/usr/local/apr-util，用作安装目录，将 apr-util 安装包解压缩到此目录下即可。

```
[root@master apr-util]# tar -zxf apr-util-1.5.2.tar.gz
[root@master apr-util]
# ./configure --prefix=/usr/local/apr-util    --with-apr=/usr/local/apr/bin/apr-1-config
[root@master apr-util]# make
[root@master apr-util]# make install
```

5.2.4 安装 PCRE

新建目录/usr/local/pcre，用作安装目录，将 pcre 安装包解压缩到此目录下即可。

```
[root@master pcre]# tar -zxf pcre-8.33.tar.gz
[root@master pcre]
# ./configure --prefix= /usr/local/pcre --with-apr=/usr/local/apr/bin/apr-1-config
[root@master pcre]# make
[root@master pcre]# make install
```

5.2.5 安装 Apache Http Server

新建目录/usr/local/apache，用作安装目录，将 apache 安装包解压缩到此目录下即可。

```
[root@master apache]# tar zxvf httpd-2.4.6.tar.gz
[root@master apache]
#./configure    --prefix=/usr/local/apache    --with-pcre=/usr/local/pcre    --with-apr=/usr/local/apr
--with-apr-util=/usr/local/apr-util
[root@master apache]#make
[root@master apache]#make install
```

5.2.6 启动 Apache 服务

打开/usr/local/apache/conf/httpd.conf 文件，找到 ServerName 选项，在下面增加服务器 IP 地址，如下所示

```
#
# ServerAdmin: Your address, where problems with the server should be
# e-mailed.  This address appears on some server-generated pages, such
# as error documents.  e.g. admin@your-domain.com
#
ServerAdmin you@example.com

#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
#
#ServerName www.example.com:80
ServerName 192.168.195.10
```

启动 Apache 服务，验证是否安装成功，如果在浏览器里面输入 http://localhost/显示"it works!"，那说明 Apache 已经安装成功。

```
[root@master apache]# /usr/local/apache/bin/apachectl start
httpd (pid 11025) already running
```

6. 安装 php

6.1 php 简介

PHP 于 1994 年由 Rasmus Lerdorf 创建，刚刚开始是 Rasmus Lerdorf 为了要维护个人网页而制作的一个简单的用 Perl 语言编写的程序。这些工具程序用来显示 Rasmus Lerdorf 的个人履历，以及统计网页流量。后来又用 C 语言重新编写，包括可以访问数据库。他将这些程序和一些表单直译器整合起来，称为 PHP/FI。PHP/FI 可以和数据库连接，产生简单的动态网页程序。

6.2 php 安装

在/usr/local/php 下面创建目录 php，在此目录中安装 php。

```
[root@master php]# tar zxvf php-5.3.26.tar.gz
[root@master php]#
./configure --prefix=/usr/local/php --with-apxs2=/usr/local/apache/bin/apxs
[root@master php]# make
[root@master php]# make install
```

配置过程中可能会报错，并提示您缺少某个组件包，譬如缺少 libxml2-devel 包，需要先安装 libxml2-devel 包等，根据提示安装对应的组件包即可。

6.3 php 配置

6.3.1 apache 环境配置

首先找到 Apache 的配置文件配置文件/usr/local/apache/conf/httpd.conf，找到

#User daemon

#Group daemon

修改为

User nagios

Group nagios

另外找到 ServerName www.example.com:80 ,在下面增加 ServerName 添加服务器的 IP 地址

```
#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
#
#ServerName www.example.com:80
ServerName 192.168.195.10
```

另外找到下面配置，增加 index.php

<IfModule dir_module>

DirectoryIndex index.html

</IfModule>

```
#
# DirectoryIndex: sets the file that Apache will serve if a directory
# is requested.
#
<IfModule dir_module>
    DirectoryIndex index.html index.php
</IfModule>
```

接下来找到 <IfModule mime_module> ，在后面增加配置信息 AddType application/x-httpd-php .php，如下所示

```
<IfModule mime_module>
    #
    # TypesConfig points to the file containing the list of mappings from
```

```
# filename extension to MIME-type.
#
TypesConfig conf/mime.types
#
# AddType allows you to add to or override the MIME configuration
# file specified in TypesConfig for specific file types.
#
#AddType application/x-gzip .tgz
#
# AddEncoding allows you to have certain browsers uncompress
# information on the fly. Note: Not all browsers support this.
#
#AddEncoding x-compress .Z
#AddEncoding x-gzip .gz .tgz
#
# If the AddEncoding directives above are commented-out, then you
# probably should define those extensions to indicate media types:
#
AddType application/x-compress .Z
AddType application/x-gzip .gz .tgz
#
# AddHandler allows you to map certain file extensions to "handlers":
# actions unrelated to filetype. These can be either built into the server
# or added with the Action directive (see below)
#
# To use CGI scripts outside of ScriptAliased directories:
# (You will also need to add "ExecCGI" to the "Options" directive.)
#
#AddHandler cgi-script .cgi
```

```
    # For type maps (negotiated resources):
    #AddHandler type-map var
    #
    # Filters allow you to process content before it is sent to the client.
    #
    # To parse .shtml files for server-side includes (SSI):
    # (You will also need to add "Includes" to the "Options" directive.)
    #
    #AddType text/html .shtml
    #AddOutputFilter INCLUDES .shtml
    AddType application/x-httpd-php .php
</IfModule>
```

出于安全考虑，一般访问 nagios 的安全监控界面必须经过授权才能访问（另外，安装配置 PHP 是安装配置 nagios 的过程，如果大家只是为了安装配置 PHP 环境，完全可以跳过下面），这需要增加验证配置，即在 httpd.conf 文件最后添加如下信息：

```
#setting for nagios
ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"
<Directory "/usr/local/nagios/sbin">
    AuthType Basic
    Options ExecCGI
    AllowOverride None
    order allow,deny
    Allow from all
    AuthName "Nagios Access"
    AuthUserFile /usr/local/nagios/etc/htpasswd
    Require valid-user
</Directory>
```

```
Alias /nagios "/usr/local/nagios/share"
<Directory "/usr/local/nagios/share">
    AuthType Basic
    Options None
    AllowOverride None
    order allow,deny
    Allow from all
    AuthName "nagios Access"
    AuthUserFile /usr/local/nagios/etc/htpasswd
    Require valid-user
</Directory>
```

6.3.2 创建 apache 目录验证文件

```
[root@master ~]# /usr/local/apache/bin/htpasswd -c /usr/local/nagios/etc/htpassw
d nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
```

这样就在/usr/local/nagios/etc 目录下创建了一个 htpasswd 验证文件，当通过 http://192.168.195.10/nagios/ 访问时就需要输入用户名和密码了。当然验证文件的密码是加密过的，如下所示：

```
[root@master conf]# cat /usr/local/nagios/etc/htpasswd
nagiosadmin:$apr1$5TnIpV7C$nv091F02UG1b0SvDv0MeL/
```

6.3.3 重启 apache 服务

```
[root@master conf]# /usr/local/apache/bin/apachectl restart
AH00526: Syntax error on line 515 of /usr/local/apache/conf/httpd.conf:
AuthUserFile takes one argument, text file containing user IDs and passwords
[root@master conf]#
```

用 vi 打开/usr/local/apache/conf/httpd.conf 文件，定位到 515 行（下面红色的哪一行），后面的哪一行汉字注释变成导致，如下图所示：

Code Snippet

```
#setting for nagios
ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"
<Directory "/usr/local/nagios/sbin">
AuthType Basic
Options ExecCGI
AllowOverride None
Order allow,deny
Allow from all
AuthName "Nagios Access"
AuthUserFile /usr/local/nagios/etc/htpasswd //用于此目录访问身份验证的文件
Require valid-user
</Directory>
Alias /nagios "/usr/local/nagios/share"
<Directory "/usr/local/nagios/share">
AuthType Basic
Options None
AllowOverride None
Order allow,deny
Allow from all
AuthName "nagios Access"
AuthUserFile /usr/local/nagios/etc/htpasswd
```

Require valid-user

</Directory>

```
504
505
506 etting for nagios
507 ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"
508 <Directory "/usr/local/nagios/sbin">
509     AuthType Basic
510     Options ExecCGI
511     AllowOverride None
512     Order allow,deny
513     Allow from all
514     AuthName "Nagios Access"
515     AuthUserFile /usr/local/nagios/etc/htpasswd          //脱脱薇眉驴录
516     Require valid-user
517 </Directory>
518 Alias /nagios "/usr/local/nagios/share"
519 <Directory "/usr/local/nagios/share">
520     AuthType Basic
521     Options None
522     AllowOverride None
523     Order allow,deny
524     Allow from all
```

将这一行注释清除后，重启 Apache 服务，OK，问题解决，打开 http://192.168.195.10/nagios/ 进入网站，点击其它页面发现乱码，这是因为 Apache 没有开启 cgi 脚本的缘故。进入 apache 的主配置文件 httpd.conf 将#LoadModule cgid_module modules/mod_cgid.so 前的注释符号去掉，重启 Apache 服务即可解决问题。

```
#LoadModule negotiation_module modules/mod_negotiation.
LoadModule dir_module modules/mod_dir.so
LoadModule actions_module modules/mod_actions.so
#LoadModule speling_module modules/mod_speling.so
#LoadModule userdir_module modules/mod_userdir.so
LoadModule alias_module modules/mod_alias.so
#LoadModule rewrite_module modules/mod_rewrite.so
LoadModule php5_module        modules/libphp5.so
```

7. 安装 nagios

Nagios 组件简述

由于 nagios 配置较为繁锁，且里面组件也较多，这里将几个关键的组件列举一下，且做一下简单的介绍及其和其它组件间的关系的描述。

1. nagios-3.2.3.tar.gz 是 nagios 的主要组件，里面包括了各种配置文件；

2. nagios-plugins-1.4.15.tar.gz 是 nagios 的插件，里面提供了各种监控模板及监控命令，如 check_tcp 等等有很多常用的监控对象都可以使用这些模式，当然也可以自己编写脚本来实现，

这一点上 nagios 是非常灵活的；

3． ndoutils-1.4b7.tar.gz，利用它将 nagios 的监控信息存入 mysql 数据库；

4． nrpe-2.12.tar.gz 是一款用来监控被控端主机资源的工具，没有它，nagios 将无法对被控端服务器的主机资源进行监控！

以上是一些主要的组件，还有一些比较重要的组件，如：NSClient-0.3.8-Win32.zip(被控端为 win 操作系统时要安装)，npc （主要用于 cacti 与 nagios 整合时，可用于将 nagios 的监控数据导给 cacti）。

7.1 检查依赖包是否安装

安装 nagios 之前，先用 rpm 命令查看所需组件包是否已经安装，因为 Nagios 需要依赖这些包提供服务，如果没有安装，可参考“第四章节 安装组件包”进行安装。

7.2 建立 nagios 账号

【注意】无论监控服务主节点，或是被监控节点上，都需要先创建 nagios 账号。

```
[root@master ~]# useradd nagios
[root@master ~]# mkdir /usr/local/nagios
[root@master ~]# chown -R nagios.nagios /usr/local/nagios/
[root@master ~]#
```

7.3 下载 nagios 相关包

- **Nagios Core**

Nagios Core 顾名思义，它是 Nagios 系统的核心部分，包含核心监测引擎和一个基本的 web 界面。下载地址 http://www.nagios.org/download/core，目前最新的版本是 4.0.0 beta1，一般我们下载稳定一点的 Nagios Core 3.5.0 版本。

- **Nagios Plugins**

Nagios Plugins 是 Nagios 插件包，通过它来监视服务，应用程序，指标，等等。像 NRPE、

NSClient++等，你可以通过下面链接地址 http://www.nagios.org/download/plugins/ 下载各种插件。

- **Nagios Frontends**

Nagios Frontends 是 Nagios 的皮肤（前端？），你可以通过它得到一些比较个性化、绚丽的 Web 监控界面

- **Nagios Addons**

Nagios Addons 是 Nagios 扩展项目，http://www.nagios.org/download/addons/

7.4 安装 nagios Core

【注意】仅需在监控服务主节点上安装 nagios，而被监控节点上仅需安装 nagios 插件即可。

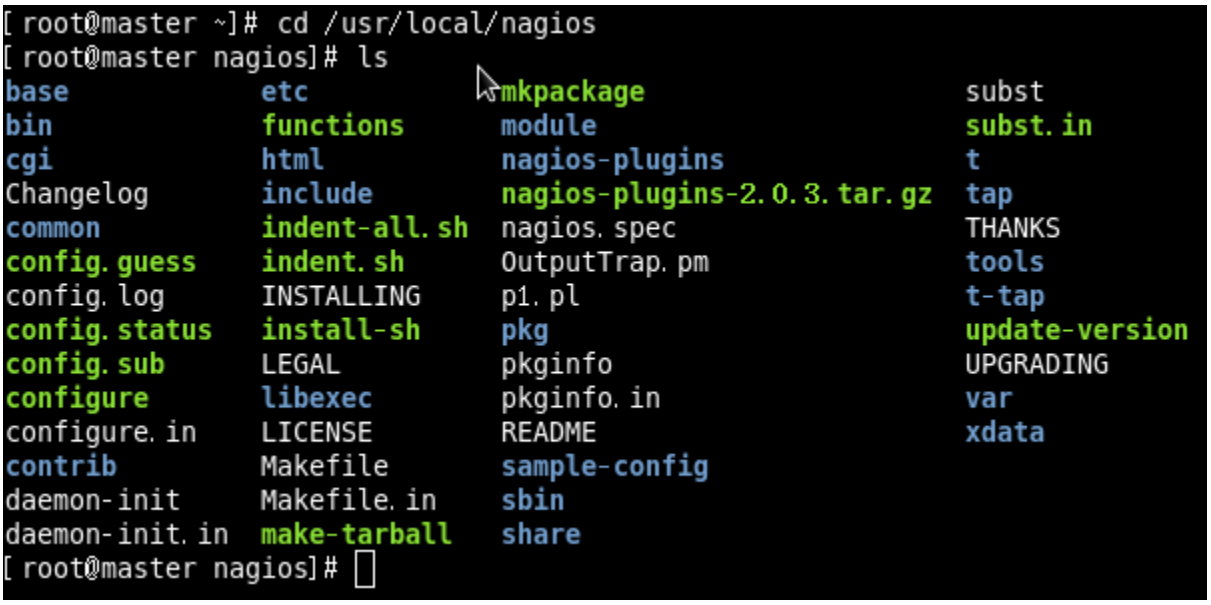
在/usr/local 目录下创建 nagios 目录，在此目录中安装 nagios。

```
[root@master nagios]# tar zxvf nagios-3.5.0.tar.gz
[root@master nagios]# ./configure --prefix=/usr/local/nagios/
[root@master nagios]# make all
[root@master nagios]# make install
[root@master nagios]# make install-init
[root@master nagios]# make install-commandmode
[root@master nagios]# make install-config
[root@master nagios]# make install-webconf
```

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：

目录	描述
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
var/archives	Empty directory for the archived logs
var/rw	Empty directory for the external command file



7.5 安装 nagios 插件

在/usr/local/nagios 目录下创建 nagios 插件目录 nagios-plugins，在此目录中安装 nagios 插件。

```
[root@master nagios]# tar zxvf nagios-plugins-1.4.16.tar.gz
[root@master nagios]# cd nagios-plugins-1.4.16
[root@master nagios-plugins-1.4.16]# ./configure --prefix=/usr/local/nagios
[root@master nagios]#make
[root@master nagios]#make install
```

7.6 apache 基本配置

如果在“第六章 PHP 配置”小节中已经配置过 Apache，则这里可以跳过。

7.7 启动相关服务

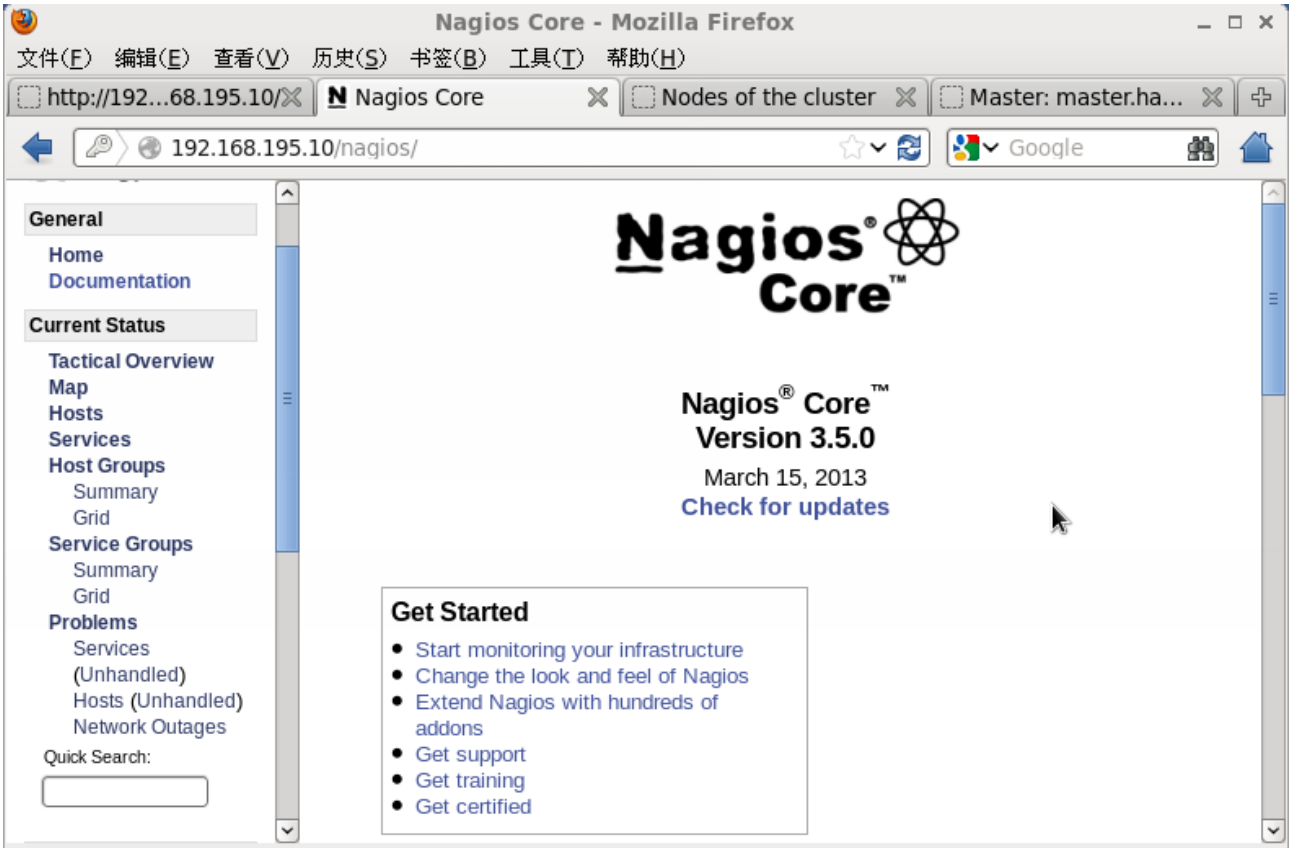
- **启动 Apache 服务**

```
[root@master ~]# /usr/local/apache/bin/apachectl start
```

- 启动 Nagios 服务

```
[root@master ~]# service nagios start
```

http://192.168.195.10/nagios/就可以进入 nagios 监控页面了。



8. 安装 nrpe

8.1 编译安装 nrpe

在/usr/local 目录下创建目录 nrpe，在此目录中安装 nrpe。

```
[root@master ~]# tar -zxf nrpe-2.15.tar.gz
[root@master ~]#./configure --enable-command-args
[root@master ~]# make all
```

```
[root@master ~]# make install-plugin
【注意】下面步骤只需要在被监控节点执行
[root@master ~]#
make install-daemon && make install-daemon-config && make install-xinetd
```

8.2 被监控节点配置

1、如果是被监控节点，需要配置 NRPE 已守护进程运行(通过 xinetd 来运行)。

更改/etc/xinetd.d/nrpe 文件，设置允许 nagios 主节点服务器连接。

```
[root@master ~]# vi /etc/xinetd.d/nrpe
only_from = 127.0.0.1 192.168.56.10
```

2、在/etc/services 结尾增加：

```
nrpe    5666/tcp        # NRPE
```

3、增加对参数的支持

```
[root@master ~]# vi /usr/local/nagios/etc/nrpe.cfg
dont_blame_nrpe=1
```

4、启动 xinetd

```
[root@master ~]# service xinetd restart
```

5、验证 nrpe 是否监听

```
[root@master ~]# netstat -at | grep nrpe
```

6、测试 nrpe 是否正常运行

```
[root@master ~]# /usr/local/nagios/libexec/check_nrpe -H localhost
NRPE v2.15
```

8.3 主节点配置

如果是监控服务主节点，在全部被监控节点 NRPE 配置完成后，可以依次做下检测。

```
[root@master ~]# /usr/local/nagios/libexec/check_nrpe -H 192.168.195.11
NRPE v2.15
[root@master ~]#/usr/local/nagios/libexec/check_nrpe -H 192.168.195.12
```

NRPE v2.15

9. 配置 nagios

9.1 配置远程被监控节点

9.1.1 修改配置文件

```
[root@master ~]# su - nagios
[root@master ~]# vi /usr/local/nagios/etc/nrpe.cfg
```

修改为如下配置内容：

```
command[check_users]=/usr/local/nagios/libexec/check_users -w $ARG1$ -c $ARG2$
command[check_load]=/usr/local/nagios/libexec/check_load -w $ARG1$ -c $ARG2$
command[check_disk]=/usr/local/nagios/libexec/check_disk -w $ARG1$ -c $ARG2$ -p $ARG3$
command[check_procs]=/usr/local/nagios/libexec/check_procs -w $ARG1$ -c $ARG2$ -s $ARG3$
command[check_procs_args]=/usr/local/nagios/libexec/check_procs  $ARG1$
command[check_swap]=/usr/local/nagios/libexec/check_swap -w $ARG1$ -c $ARG2$
```

以上监控命令功能：

check_users	监控登陆用户数
check_load	监控 CPU 负载
check_disk	监控磁盘的使用
check_procs	监控进程数量,状态包括 RSZDT
check_swap	监控 SWAP 分区使用

9.1.2 重启 xinetd 服务

配置完上述命令后,重启 xinetd 服务。

```
[root@master ~]# service xinetd restart
```

9.1.3 校验配置

检查监控命令配置是否 ok。

```
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_users    -a 5 10
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_load     -a 15,10,5 30,25,20
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_disk      -a 20% 10% /
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_procs -a 200 400 RSZDT
/usr/local/nagios/libexec/check_nrpe -H localhost -c check_swap    -a 20% 10%
```

9.2 配置监控服务主节点

9.2.1 cgi.cfg(控制 CGI 访问的配置文件)

(使用 nagios 用户)

```
[root@master ~]# vi /usr/local/nagios/etc/cgi.cfg
```

修改如下内容，为 admin 用户增加权限：

```
default_user_name=admin
authorized_for_system_information=nagiosadmin,admin
authorized_for_configuration_information=nagiosadmin,admin
authorized_for_system_commands=nagiosadmin,admin
authorized_for_all_services=nagiosadmin,admin
authorized_for_all_hosts=nagiosadmin,admin
authorized_for_all_service_commands=nagiosadmin,admin
authorized_for_all_host_commands=nagiosadmin,admin
```

9.2.2 nagios.cfg(nagios 主配置文件)

(使用 nagios 用户)

```
[root@master ~]# vi /usr/local/nagios/etc/nagios.cfg
```

```
#cfg_file=/export/home/nagios/etc/objects/localhost.cfg        (注释掉)
cfg_dir=/export/home/nagios/etc/servers
```

主配置文件声明了监控脚本的存储路径为 ./servers, 默认没有此目录,需要手工创建。

nagios 会读取 servers 目录下面后缀为.cfg 的全部文件作为配置文件。

```
[root@master ~]# cd /usr/local/nagios/etc
[root@master ~]# mkdir servers
[root@master ~]# cd servers
```

9.2.3 定义监控的主机组

声明一个监控的主机组,将主机环境中提到的三台主机全部加入监控。

```
[root@master ~]# vi /export/home/nagios/etc/servers/group.cfg
```

新文件内容如下：

```
define hostgroup{
    hostgroup_name          hadoop-server
    alias                   hadoop Server
    members                 master.hadoop,slave1.hadoop,slave2.hadoop
}
```

解释下上面的配置：

```
hostgroup_name:         主机组的名称,可随意指定
alias:                  主机组别名,可随意指定
members:                主机组成员,多个主机名称之前使用逗号分隔.另外主机名称
```

必须与 define host 中 host_name 一致。主机的定义,后面会说到。

9.2.4 定义监控的主机

下面开始定义具体的主机。

9.2.4.1 本地主机监控配置

先定义本地主机 master.hadoop。

```
[root@master ~]# vi /export/home/nagios/etc/servers/master.hadoop.cfg
```

新文件内容如下：

```
define host{
        use                     linux-server
        host_name               master.hadoop
        alias                   master.hadoop
        address                 192.168.195.10
}

define service{
        use                     local-service
        host_name               master.hadoop
        service_description     Host Alive
        check_command           check-host-alive
}

define service{
        use                     local-service
        host_name               master.hadoop
        service_description     Users
        check_command           check_local_users!20!50
}

define service{
        use                     local-service
        host_name               master.hadoop
        service_description     CPU
        check_command           check_local_load!5.0,4.0,3.0!10.0,6.0,4.0
}

define service{
        use                     local-service
        host_name               master.hadoop
        service_description     Disk Root
        check_command           check_local_disk!20%!10%!/
}

define service{
        use                     local-service
        host_name               master.hadoop
        service_description     Zombie Procs
        check_command           check_local_procs!5!10!Z
}
```

```
define service{
        use                     local-service
        host_name               master.hadoop
        service_description     Total Procs
        check_command           check_local_procs!250!400!RSZDT
}

define service{
        use                     local-service
        host_name               master.hadoop
        service_description     Swap Usage
        check_command           check_local_swap!20!10
}
```

说明下，由于是此主机也是监控服务主节点所在主机,因此可以使用 check_local_* 的相关命令来进行监控。

这个文件中已经将常用的监控项配置进去。

9.2.4.2 远程主机监控配置

再定义远程主机 slave1.hadoop 和 slave2.hadoop。

定义远程主机的监控之前，需要先定义 check_nrpe 命令。

```
[root@master ~]# vi /usr/local/nagios/etc/objects/commands.cfg
```

在文件的最后面添加如下内容：

```
# 'check_nrpe' command definition
define command{
        command_name check_nrpe
        command_line    $USER1$/check_nrpe -H $HOSTADDRESS$ -t 30 -c $ARG1$
}
define command{
        command_name check_nrpe_args
        command_line    $USER1$/check_nrpe -H $HOSTADDRESS$ -t 30 -c $ARG1$ -a $ARG2$
}
```

定义 slave1.hadoop 主机的监控配置。

```
[root@master ~]# vi /usr/local/nagios/etc/servers/slave1.hadoop.cfg
```

新文件内容如下：

```
define host{
    use                 linux-server
    host_name           slave1.hadoop
    alias               slave1.hadoop
    address             192.168.195.11
}

define service{
    use                 local-service
    host_name           slave1.hadoop
    service_description Host Alive
    check_command       check-host-alive
}

define service{
    use                 local-service
    host_name           slave1.hadoop
    service_description Ping Remote Host
    check_command       check_ping!100.0,20%!500.0,60%
}

define service{
    use                 local-service
    host_name           slave1.hadoop
    service_description Users
    check_command       check_nrpe_args!check_users!5 10
}

define service{
    use                 local-service
    host_name           slave1.hadoop
    service_description CPU
    check_command       check_nrpe_args!check_load!15,10,5 30,25,20
}

define service{
    use                 local-service
    host_name           slave1.hadoop
```

```
	service_description	Disk Root
	check_command		check_nrpe_args!check_disk!20% 10% /
}

define service{
	use			local-service
	host_name		slave1.hadoop
	service_description	Zombie Procs
	check_command		check_nrpe_args!check_procs!5!10!Z
}

define service{
	use			local-service
	host_name		slave1.hadoop
	service_description	Total Procs
	check_command		check_nrpe_args!check_procs_args!250!400!RSZDT
}

define service{
	use			local-service
	host_name		slave1.hadoop
	service_description	Swap Usage
	check_command		check_nrpe_args!check_swap!20% 10%
}
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;下面是一些常用进程的监控,主要是云平台相关进程
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; 监控 crond 进程
define service{
		use				local-service
		host_name			slave1.hadoop
		service_description		PS: crond
		check_command			check_nrpe_args!check_procs_args!"-c1:1 -Ccrond"
}
;; 监控 zookeeper 进程
define service{
		use					local-service
		host_name				slave1.hadoop
		service_description			PS: QuorumPeerMain
		check_command				check_nrpe_args!check_procs_args!"-c1:1   -Cjava
-aserver.quorum.QuorumPeerMain"
}
```

```
;;监控 storm 的从节点进程
define service{
        use                             local-service
        host_name                       slave1.hadoop
        service_description             PS: supervisor
        check_command                   check_nrpe_args!check_procs_args!"-c1:1 -Cjava
-adaemon.supervisor"
}
;; 监控 storm 的主节点进程
define service{
        use                             local-service
        host_name                       slave1.hadoop
        service_description             PS: nimbus
        check_command                   check_nrpe_args!check_procs_args!"-c1:1 -Cjava
-adaemon.nimbus"
}
;; 监控 MetaQ 进程
define service{
        use                             local-service
        host_name                       slave1.hadoop
        service_description             PS: MetaQ
        check_command                   check_nrpe_args!check_procs_args!"-c1:1 -Cjava
-ametamorphosis-server-w"
}
;; 监控 Redis 进程
define service{
        use                             local-service
        host_name                       slave1.hadoop
        service_description             PS: redis-server
        check_command                   check_nrpe_args!check_procs_args!"-c1:1
-Credis-server"
}
;; 监控 hadoop 主节点 NameNode 进程
define service{
        use                             local-service
        host_name                       slave1.hadoop
        service_description             PS: NameNode
        check_command                   check_nrpe_args!check_procs_args!"-c1:1 -Cjava
-aserver.namenode.NameNode"
}
;; 监控 hadoop 主节点 SecondaryNameNode 进程
define service{
        use                             local-service
        host_name                       slave1.hadoop
```

```
        service_description                     PS: SecondaryNameNode
        check_command                           check_nrpe_args!check_procs_args!"-c1:1  -Cjava
-aserver.namenode.SecondaryNameNode"
}
;; 监控 hadoop 主节点 ResourceManager 进程
define service{
        use                                     local-service
        host_name                               slave1.hadoop
        service_description                     PS: ResourceManager
        check_command                           check_nrpe_args!check_procs_args!"-c1:1  -Cjava
-aserver.resourcemanager.ResourceManager"
}
;; 监控 hadoop 从节点 DataNode 进程
define service{
        use                                     local-service
        host_name                               slave1.hadoop
        service_description                     PS: DataNode
        check_command                           check_nrpe_args!check_procs_args!"-c1:1  -Cjava
-aserver.datanode.DataNode"
}
;;监控 hadoop 从节点 NodeManager 进程
define service{
        use                                     local-service
        host_name                               slave1.hadoop
        service_description                     PS: NodeManager
        check_command                           check_nrpe_args!check_procs_args!"-c1:1  -Cjava
-aserver.nodemanager.NodeManager"
}
```

说明下，由于 duangr-2 是远程主机,因此使用 check_nrpe_args 命令来监控。

这个文件中已经将常用的监控项配置进去，同时还包含了 hadoop、storm、zookeeper、metaq、redis 的相关进程监控，主要的监控思路是判断进程是否存在。

定义 slave2.hadoop 主机的监控配置。内容与 slave1.hadoop.cfg 类似,只需要修改 host_name 、alias、 address 即可。

9.2.4.3 邮件监控

定义监控人邮件地址。

```
[root@master ~]# vi /usr/local/nagios/etc/objects/contacts.cfg
```

```
define contact{
        contact_name    nagiosadmin             ; Short name of user
        use             generic-contact
        alias           Nagios Admin            ; Full name of user
        email           yourname@domain.com
}
```

除了配置监控邮件的接收人外，还要确保：

本主机与邮件服务器互通。

本主机 SendMail 可以使用外部 SMTP 服务发送邮件。

9.2.4.4 校验配置

```
[root@master ~]# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

9.2.4.5 启动

```
/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
nagios 已经是一个服务,也可以执行如下操作：
service nagios start/stop/restart/status
```

10. nagios 常用命令

nagios 常用命令示例

监控对象		监控阀值
主机资源	主机存活： check_ping	-w 3000.0,80% -c 5000.0,100% -p 5(3000 毫秒响应时间内， 丢包率超过 80%报警告，5000 毫秒响应时间内，丢包率超过 100%报危急，一共发送 5 个包）

	登录用户： check_user	-w 5 -c 10(w 为警告，c 为危急)
	系统负载： check_load	-w 15,10,5 -c 30,25,20(1 分钟，5 分钟，15 分钟大于对应的等待进程数则警告或危急)
	磁盘占用率： check_disk	-w 20% -c 10% -p /（根分区剩余空间为总大小的 20%警告， 10%危急，-p 后是根分区）
	脚本检测磁盘 I/O： check_iostat	-w 5 –c 10 (磁盘 I/O 的 iowait 超过 5%报警告,超过 10%报危急)
	检测僵尸进程： check_zombie _procs	-w 5 -c 10 -s Z（有 5 个僵尸进程报警告，10 个报危急）
	检测总进程数： check_total_procs	-w 150 -c 200（总进程到 150 个警告，200 个报危急）
	脚本检测内存剩余： check_mem	-w 90% -c 95%(内存空闲率 90%以上报警告，95%以上报危急)
	检测交换分区使用率： check_swap	-w 20% -c 10%（交换分区剩余空间为总大小的 20%警告，10%危急）
应用服务监控	监控服务端口： check_tcp	-H localhost2 -p 80(主机与对应的端口号)
	监控页面响应时间： check_http	-H localhost2 -u http:\/\/localhost2/test.jsp –w 5 –c 10(检查页面，超过 5s 报警告，超过 10s 报危急)
	脚本检测 IP 连接数： check_ips	-w 200 –c 250(IP 连接数超过 200 报警告，超过 250 报危急)
流量监控	监控 server 流量: Check_traffic	-V 2c -C public -H localhost2 -I 2 -w 12,30 -c 15,35 -M –b(snmp 版本,用户,主机,对应网卡,警告阀值,危急阀值)

check_ssh

Usage:check_ssh [-46] [-t <timeout>] [-r <remote version>] [-p <port>] <host>

界面拼装参数格式如下共 3 个元素命令!端口!连接超时时间

check_ssh!22!10

■ **参数列表**

参数	描述
-h, --help	帮助

-V, --version	列出版本信息
-H, --hostname=ADDRESS	主机名称,IP 地址,或者 UNIX 套接字(必须有绝对路径)
-p, --port=INTEGER	端口号(默认:22)
-4, --use-ipv4	使用 IPV4 协议连接
-6, --use-ipv6	使用 IPV6 协议连接
-t, --timeout=INTEGER	连接超时秒数(默认:10 秒)
-r, --remote-version=STRING	不匹配服务器版时警告字符串,如对方的版本为 OpenSSH_3.9p1
-V, --verbose	列出详细的命令调试行

■ 举例

./check_ssh -H 192.168.2.220 -p 22 -t 10 -r OpenSSH_3.0pl

SSH WARNING - OpenSSH_3.8.1p1 Debian-8.sarge.6 (protocol 2.0) version mismatch, expected 'OpenSSH_3.0pl'

./check_ssh -H 192.168.2.220 -p 22 -t 10

SSH OK - OpenSSH_3.8.1p1 Debian-8.sarge.6 (protocol 2.0)

check_ssh -H $HOSTADDRESS$ -p 22 -t 10

check_http

Usage: check_http -H <vhost> | -I <IP-address> [-u <uri>] [-p <port>]

[-w <warn time>] [-c <critical time>] [-t <timeout>] [-L]

[-a auth] [-f <ok | warn | critcal | follow>] [-e <expect>]

[-s string] [-l] [-r <regex> | -R <case-insensitive regex>] [-P string]

[-m <min_pg_size>:<max_pg_size>] [-4|-6] [-N] [-M <age>] [-A string] [-k string]

界面拼装参数格式如下共 4 个元素命令!告警时延!严重告警时延!连接超时时间

check_http!0.0020!0.0050!10

■ 参数

参数	描述
-h, --help	帮助
-V, --version	列出版本信息
-H, --hostname=ADDRESS	虚拟主机名加端口(如:excamle.com:5000)
-I, --IP-address=ADDRESS	IP 地址或名称(如果无需 DNS 的查找,使用十进制的地址)
-p, --port=INTEGER	端口数(默认: 80)
-4, --use-ipv4	使用 IPV4 连接
-6, --use-ipv6	使用 IPV6 连接
-e, --expect=STRING	把服务器反馈的第一行(状态)转换成指定的字符串(默认是:HTTP/1. 如果指定跳跃了所有其它逻辑状态行)
-s, --expect=STRING	指定内容
-u, --url=PATH	获取或发送的 URL(默认:/)
-P, --post= STRING	URL 进行 POST 的 HTTP 数据
-N, --no-body	不等待文档正文:获取报头后停止读取。(注意,这是一个 HTTP 的获取和发送,而不是报头)
-M, --max-age=SECONDS	如果文档超过生存期则警告。数据是如下形式的:分数是"10m",小时数是"10h",天数是 "10d"
-T, --content-type=STRING	在传输的时候指定容器类型媒体类型
-l, --linespan	允许正则表达式跨越新行(必须在前面使用 -R 或 -r)
-r, --regex, ,--ereg=STRING	用正则表达式字符串搜索页
-R, --eregi=STRING	用正则表达式字符串搜索页,允许模糊查找
--invert-regex	如果找到返回 CRITICAL,找不到返回 OK
-a, --authorization=AUTH_PAIR	用户名:在站点最基本的密码认证

-A, --useragent=STRING	转换成字符串放在 HTTP 报头里发送,像"用户代理"
-k, --header=STRING	任何其它的标签被放在 HTTP 报头里发送。可以被附加的报头使用多次。
-L, --link=URL	在 HTML 链接里隐藏发送包
-f, --onredirect	怎样解决重定向页
-m, --pagesize=INTEGER<:INTERGER>	最小最大页大小要求(BYTES)
-w, ----warning=DOUBLE	告警状态的返回时间(秒)
-c,--critical=DOUBLE	严重状态的返回时间(秒)
-t, --timeout=INTEGER	指定超时前的时间(默认 10 秒)
-v , --verbose	列出详细的命令调试行

■ 举例

./check_http -H 192.168.2.220 -p 80

HTTP OK HTTP/1.1 200 OK - 5553 bytes in 0.057 seconds |time=0.057428s;;;0.000000 size=5553B;;;0

./check_http -H 192.168.2.220 -p 80 -w 0.0020 -c 0.0060

HTTP WARNING: HTTP/1.1 200 OK - 0.003 second response time |time=0.003068s;0.002000;0.006000;0.000000 size=5553B;;;0

./check_http -H 192.168.2.220 -p 80 -w 0.0030 -c 0.0040

HTTP OK HTTP/1.1 200 OK - 5553 bytes in 0.003 seconds |time=0.002673s;0.003000;0.004000;0.000000 size=5553B;;;0

./check_http -H 192.168.2.220 -p 80 -w 0.0009 -c 0.0040 -t 10

HTTP WARNING: HTTP/1.1 200 OK - 0.002 second response time |time=0.002102s;0.000900;0.004000;0.000000 size=5553B;;;0

check_tcp

关联命令：

check_clamd

check_imap

check_ftp

check_nntp

check_pop

check_udp

check_tcp

Usage:check_tcp -H host -p port [-w <warning time>] [-c <critical time>] [-s <send string>]

[-e <expect string>] [-q <quit string>][-m <maximum bytes>] [-d <delay>]

[-t <timeout seconds>] [-r <refuse state>] [-M <mismatch state>] [-v] [-4|-6] [-j]

[-D <days to cert expiry>] [-S <use SSL>] [-E]

界面拼装参数格式如下共 4 个元素：

命令!端口!告警时延!严重告警时延!连接超时时间

check_tcp!23!0.0020!0.0050!10

■ 参数

参数	描述
-h, --help	帮助
-V, --version	列出版本信息
-H, --hostname=ADDRESS	主机名,IP 地址,或则 UNIX 套接字 (必须是绝对路径)
-p, --port=INTEGER	端口数 (默认: 无)
-4, --use-ipv4	使用 IPV4 连接
-6, --use-ipv6	使用 IPV6 连接
-E, --escape	可以用\n,\r,\t or \发送或跳出字符串，默认情况下不加东西,\r\n 加在退出的时候。
-s, --send=STRING	发送服务器的字符串

-e, --expect = STRING	转换为服务器返回的字符串
-q, --quit= STRING	关闭的连接时发送给服务器的字符串
-r, --refuse=OK|warn|crit	允许 TCP 拒绝的状态 k,warn,crit (默认:warn)
-M, --mismatch= OK|warn|crit	允许预期的字符串,当发现不匹配状态 k,warn,crit (默认:warn)
-j, --jail	隐藏 TCP 套接字的输出
-m, --maxbytes=INTEGER	当接收数据包大于指定的大小时,关闭连接。
-d, --delay	支持在发送数据流和轮询反馈间等待的延迟
-w, ----warning=DOUBLE	告警状态的返回时间(秒)
-c,--critical=DOUBLE	严重状态的返回时间(秒)
-t, --timeout=INTEGER	指定超时前的时间(默认 10 秒)
-v , --verbose	列出详细的命令调试行

■ 举例

./check_tcp -H 192.168.2.220 -p 22 -w 0.0023 -c 0.0067 -t 10

TCP OK - 0.002 second response time on port 22|time=0.002289s;0.002300;0.006700;0.000000;10.000000

./check_tcp -H 192.168.2.220 -p 22 -w 0.0003 -c 0.0006 -t 10

TCP WARNING - 0.000 second response time on port 22|time=0.000318s;0.000300;0.000600;0.000000;10.000000

■ 备注

check_imap=check_tcp!143

check_ftp=check_tcp!21

check_nntp=check_tcp!119

check_pop=check_udp!110

check_udp=check_tcp

check_telnet=check_tcp!23

11. nagios 常见问题

- **通过 VMWare 共享 windows 上的文件夹，通过终端进行 rpm 安装软件包时，提示 can't create transaction lock on /var/lib/rpm/.rpm.lock 错误。**

错误原因：权限问题

解决方法：在终端输入“su”和密码，登录 root，以 root 身份登录就不会有提示

- **nagios 监控失败报错 It appears as though you do not have**

分类： LINUX

安装 nagios 时,在页面上出现监错误信息

It appears as though you do not have permission to view information you requested.....

点击任何选项都是这个错误,郁闷了。

百度了下,好多文章基本都是一样,改 cgi.cfg 文件里的一个参数 use_authentication=1 将 1 改为 0,重启 nagios 即可。

不过，这一项其实是 nagios 保证安全性的一个参数,我觉得改了之后,有可能会带来安全上的隐患，于是我仔细看了下这个文件里的内容，终于让我发现问题所在啦!呵呵...

原来之所以出现这样的提示，主要还是身份验证的问题!!

还记得我们装 nagios 时生成的一个管理员帐号吗？

htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

nagiosadmin 是我们用来登录 nagios 的管理员，而 cgi.cfg 就是用来验证管理员身份的！

如果你生成的管理员用户不是 nagiosadmin,验证的时候就会失败，也就会出现开始我们描述的那个错误了！

所以，我们现在只要把配置文件里的用户名改为实际的管理员用户名,保存,再重新启动 nagios 即可！

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

将以下的几项中的 nagiosadmin 改为你实际的用户名即可,其实也可以在后面加上你的用户名，与前面用逗号隔开

authorized_for_system_information=nagiosadmin

authorized_for_configuration_information=nagiosadmin

authorized_for_system_commands=nagiosadmin

authorized_for_all_services=nagiosadmin

authorized_for_all_hosts=nagiosadmin

authorized_for_all_service_commands=nagiosadmin

authorized_for_all_host_commands=nagiosadmin