## 案例二：监控Nginx服务器状态码是否有502

相信运维过Nginx+php-fpm+MySQL网站的朋友一定遇到过502问题，发生502问题的原因有很多种，而最常见的是由于php-fpm资源耗完导致。而本案例要监控的这台服务器就是这种情况，平时一直都很好，但若网站访问量很高，就会有502的状态码出现。发生502的问题时，需要及时分析造成php-fpm资源耗尽的原因，所以要做一个监控脚本，当有502状态码第一时间告警通知我们。

具体要求如下：

1）脚本一分钟执行一次

2）监控502状态可以通过分析网站的访问日志，也可以通过用curl工具发起http请求，获得状态码，建议通过分析访问日志，假如访问日志路径为/data/logs/access.log，日志片段如下：

54.36.149.38 - [16/Sep/2018:21:38:41 +0800] www.lishiming.net "/thread-5360-1-1.html" 301 "GET /thread-5360-1-1.html HTTP/1.1""-" "Mozilla/5.0 (compatible; AhrefsBot/5.2; +http://ahrefs.com/robot/)"

42.236.10.91 - [16/Sep/2018:21:38:41 +0800] a.lishiming.net "/" 403 "GET / HTTP/1.1""http://a.lishiming.net/" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/50.0.2661.102 Safari/537.36; 360Spider" "

111.147.182.161 - [16/Sep/2018:21:38:41 +0800] www.lishiming.net "/data/attachment/forum/month\_1211/epel-release-6-7.noarch.rpm" 301 "GET /data/attachment/forum/month\_1211/epel-release-6-7.noarch.rpm HTTP/1.1""-" "curl/7.29.0"

3）一分钟内出现502的次数超过50次则需要告警

4）告警需要发邮件通知，通知邮箱为aming@aminglinux.com

知识点一：计划任务cron

本案例要求我们一分钟执行一次脚本，所以要用到cron。执行命令：

crontab -e

这样就进入到了编辑页面，计划任务的格式是这样的：

# Example of job definition:

# .---------------- 分钟 (范围0 - 59)

# | .------------- 小时 (范围0 - 23)

# | | .---------- 日期 (范围1 - 31)

# | | | .------- 月 (范围1 - 12) 也可以用英文的简写形式： jan,feb,mar,apr ...

# | | | | .---- 周 (范围0 - 6) (周日可以用0或者7表示) 也可以写成英文简写形式：sun,mon,tue,wed,thu,fri,sat

# | | | | |

# \* \* \* \* \* 用户名（可以省略） 要执行的命令

示例1：

每周1,3,5凌晨4点20分执行脚本/usr/local/sbin/123.sh

20 4 \* \* 1,3,5 /bin/bash /usr/local/sbin/123.sh

示例2：

每隔3天清空文件/data/log/tmp.log

0 0 \*/3 \* \* true>/data/log/tmp.log

示例3：

每天9点到12点，14点到16点执行脚本/usr/local/sbin/xxx.sh

0 9-12,14-16 \* \* \* /bin/bash /usr/local/sbin/xxx.sh

本案例每分钟执行一次，则写成这样：

\* \* \* \* \* /bin/bash /usr/local/sbin/mon\_502.sh

知识点二：过滤关键词

shell脚本中，过滤某个关键词使用grep命令，比如在本例中需要把日志中包含502的日志过滤出来，命令为：

grep '502' /data/log/access.log

这个是最基本的用法，平时脚本中使用grep的机会很多，且用法也比较复杂，通常会和正则表达式配合在一起使用。下面为几个常见的用法，（以下全部示例文件为1.txt）。

过滤出所有包含数字的行：

grep '[0-9]' 1.txt

过滤出以英文字母开头的行：

grep '^[a-zA-Z]' 1.txt

去除所有空行：

grep -v '^$' 1.txt

过滤出包含以abc开头且以123结尾的字符串的行

grep 'abc.\*123' 1.txt

过滤出至少有两个连续数字的行

grep -E '[0-9][0-9]+' 1.txt

或者

grep -E '[0-9]{2,}' 1.txt

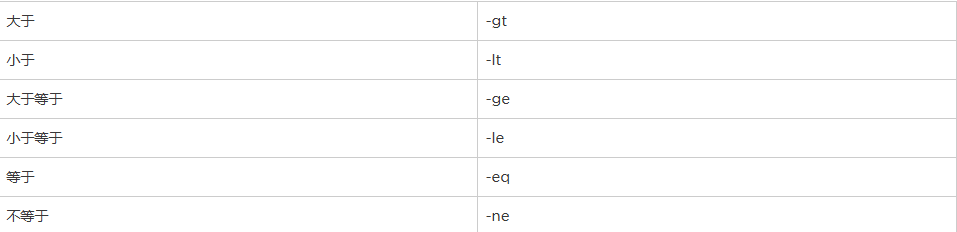
过滤出含有aming或linux的行有几行

grep -En 'aming|linux' 1.txt

再回过头来看本案例的需求，因为是一分钟检测一次，所以过滤的关键词是上一分钟的时间，通过分析日志，可以发现上一分钟可以用date -d &quot;-1 min&quot; "%Y:%H:%M:[0-5][0-9]"这个关键词来表示。

知识点三：shell中的大小比较

shell脚本的语法比较特殊，其中数字比较大小并不能直接使用>,<,>=,<=等字符，而是使用如下方式：



结合if语句，写法如下：

if [ $a -gt 10 ] //当变量a的值大于10的时候

if [ $x -le 100 ] //当变量x的值小于等于100的时候

如果要比较的对象并不是数字，而是一个字符串的时候，可以这样用：

if [ $str == "aminglinux" ] //当变量str的值是aminglinux的时候

知识点四：shell中的逻辑判断

在所有的编程语言中，逻辑判断和循环遍历可以说是必不可少的组成部分，shell中的逻辑判断以if为主，其格式主要有三种。

1）if ... then ... fi 示例代码：

if [ $a -gt 5 ]

then

command1

command2

fi</code></pre>

2）if ... then ... else ... fi示例代码：

if command

then

command1

else

command2

fi

说明：if后面的条件可以是一条命令，命令执行成功则条件成立，执行command1，否则条件不成立，执行command2

3）if ... then ... elif ... then ... else ... fi 示例代码：

if [ $b -ge 10] || [ $b -lt 6 ]

then

command1

elif [ $b -eq 8 ]

then

command2

else

command3

fi

说明：if后面的判断条件可以有多个，用||表示或者，用&amp;&amp;表示并且

知识点五：在命令行下发邮件

从09年踏入运维这个领域，我写过无数个监控脚本，用的最多的告警方式就是发邮件，早期使用系统自带的sendmail或者postfix发邮件给139邮箱，因为139邮箱是带短信提醒的。但是用这种方法发出来的邮件通常是直接拒收或者给放到垃圾邮箱里，最终效果并不好。后来，又通过php调用第三方的邮件（如，QQ邮箱）给139邮箱发邮件，但因为发的多了，QQ邮箱也不好用了。最近又研究了一个python发邮件的脚本，同样调用第三方邮件，不过不用QQ的了，而是使用163的邮箱。因为可以在手机上安装邮件客户端，所以不用担心提醒的问题了，收到邮件直接在app上提醒，比短信还方便。同时，接收邮件人也可以是自己，也就是说自己给自己发邮件，这样也不会有垃圾邮件的烦恼。

下面是发邮件的python代码：

#!/usr/bin/python

#coding:utf-8

import smtplib

from email.mime.text import MIMEText

import sys

mail\_host = 'smtp.163.com'

mail\_user = 'test@163.com'

mail\_pass = 'your\_mail\_password'

mail\_postfix = '163.com'

def send\_mail(to\_list,subject,content):

me = "zabbix 监控告警平台"+"<"+mail\_user+"@"+mail\_postfix+">"

msg = MIMEText(content, 'plain', 'utf-8')

msg['Subject'] = subject

msg['From'] = me

msg['to'] = to\_list

try:

s = smtplib.SMTP()

s.connect(mail\_host)

s.login(mail\_user,mail\_pass)

s.sendmail(me,to\_list,msg.as\_string())

s.close()

return True

except Exception,e:

print str(e)

return False

if \_\_name\_\_ == "\_\_main\_\_":

send\_mail(sys.argv[1], sys.argv[2], sys.argv[3])

说明：该脚本会调用第三方的邮箱账户，需要你填写正确的mail\_host, mail\_user以及mail\_pass。假如脚本名字为mail.py，则发邮件的命令为：

python mail.py aming@aminglinux.com "邮件主题" "邮件内容"

本案例参考脚本

有了以上四个储备知识，下面就该写本案例的脚本了，如下是我提供的一个参考脚本：

vim /usr/local/sbin/mon\_502.sh //内容如下

#!/bin/bash

##该脚本用来监控网站的502问题

##作者：

##日期：2018-09-16

##版本：v0.1

t=`date -d "-1 min" +"%Y:%H:%M:[0-5][0-9]"`

log="/data/logs/access.log"

#假设mail.py已经写好，并放在/usr/local/sbin/下

mail\_script="/usr/local/sbin/mail.py"

mail\_user=aming@aminglinux.com

n=`grep $t $log|grep -n " 502 "`

if [ $n -gt 50 ]

then

python $mail\_script $mail\_user "网站有502" "1分钟内出现了$n次"

fi

增加任务计划：

\* \* \* \* \* /bin/bash /usr/local/sbin/mon\_502.sh 2>/tmp/mon\_502.err

说明：需要在该cron最后面定义一个错误日志输出，如果脚本执行过程中有报错，我们可以到/tmp/mon\_502.err文件中查看错误信息。