Python 脚本编程及系统大规模自动化运维-Aapache配置

实验要求：用python脚本实现自动替换虚拟主机的DocumentRoot，脚本名为apache_conf_docroot_replace.py
例如，将www.uplooking.com的网站根目录修改为/var/www/baidu.com

思路：

1. 建立三个编译后的正则表达式对象：
   A. 匹配VirtualHost的开始
   B. 匹配VirtualHost的结尾
   C. 匹配DocumentRoot这一行
2. 创建一个函数来完成 replace_docroot()

/etc/httpd/conf.d/baidu.conf 原先的配置

```
[root@masterb0 ~]# cat /etc/httpd/conf.d/baidu.conf
<VirtualHost *:80>
    ServerAdmin webmaster@www.baidu.com
    ServerName www.baidu.com
     DocumentRoot /var/www/html
    ErrorLog "/var/log/httpd/www.baidu.com-error_log"
    CustomLog "/var/log/httpd/www.baidu.com-access_log" common
</VirtualHost>
```

```
#!/usr/bin/env python

from cStringIO import StringIO
import re

vhost_start = re.compile(r'<VirtualHost\s+>')
vhost_end = re.compile(r'</VirtualHost')
name_re = re.compile(r'(ServerName\s+)(\S+)')
docroot_re = re.compile(r'(DocumentRoot\s+)(\S+)')

def replace_docroot(conf_string,vhost,new_docroot):
    '''yield new lines of an httpd.conf file where docroot lines matching
        the specified vhost are replaced with the new_docroot
    '''
    conf_file = StringIO(conf_string)
    in_vhost = False
    curr_vhost = None
    for line in conf_file:
        vhost_start_match = vhost_start.search(line)
        if vhost_start_match:
            in_vhost = True
        name_match = name_re.search(line)
        if name_match:
            curr_vhost = name_match.groups()[1]
        if in_vhost and (curr_vhost == vhost):
            docroot_match = docroot_re.search(line)
            if docroot_match:
                sub_line = docroot_re.sub(r'\1%s' %docroot,line)
                line = sub_line
        vhost_end_match = vhost_end.search(line)
        if vhost_end_match:
            in_vhost = False
        yield line

if __name__ == '__main__':
    import sys
    conf_file = sys.argv[1]
    vhost = sys.argv[2]
    docroot = sys.argv[3]
    conf_string = open(conf_file).read()
    for line in replace_docroot(conf_string,vhost,docroot):
        print line
```

小知识点：

1. cStringIO模块的StringIO()函数可以将文本中的行转化为列表，就类似于bash中的while read line
2. re.compile()编译后的正则表达式规则
3. def 定义函数
4. for 循环
5. vhost_start.search(line)做正则匹配，全文搜索
6. name_match.groups()[1]截取列表的第二个元素
7. docroot_re.sub(r'\1%s' %docroot,line)正则表达式做替换，将docroot_re规则中的第二个括号中的内容替换为%

后的docroot的内容

8. sys模块的argv[]方法可以调用位置变量，argv[]1类似于bash中$1
9. open()函数打开文件
10. read()函数读取文件中的内容，保存成字符串

除了使用StringIO()函数可以将文本中的行转化为列表之外，我们还可以使用字符串内置的函数splitlines()

```
In [1]: import sys

In [3]: conf_string=open('/home/kiosk/Desktop/pp').read()

In [4]: print conf_string
1 ok
2 bb
3 cc


In [5]: conf_file = conf_string.splitlines()

In [6]: print conf_file
['1 ok', '2 bb', '3 cc']

In [7]: from cStringIO import StringIO

In [8]: conf_file=StringIO(conf_string)

In [9]: print conf_file
<cStringIO.StringI object at 0xf6ebe0>

In [10]: for i in conf_file:
   ....:     print i
   ....:
1 ok

2 bb

3 cc
```

执行脚本apache_conf_docroot_replace.py后的结果

```
[root@masterb0 ~]# python apache_conf_docroot_replace.py /etc/httpd/conf.d/baidu.conf
www.baidu.com /var/www/baidu.com
<VirtualHost *:80>

    ServerAdmin webmaster@www.baidu.com

    ServerName www.baidu.com

    DocumentRoot /var/www/baidu.com

    ErrorLog "/var/log/httpd/www.baidu.com-error_log"

    CustomLog "/var/log/httpd/www.baidu.com-access_log" common

</VirtualHost>
```

该脚本并没有将原配置文件修改，只是将修改后的结果显示到屏幕上；注意这个脚本必须要求DocumentRoot放在ServerName后面

日志解析

课堂实践：读取一个apache访问日志，查看每一个独立客户端连接获得的字节数。

apache访问日志格式如下所示：

```
172.25.254.250 - - [14/Oct/2016:12:53:58 +0800] "GET /rhel7.2/x86_64/vms/db100-masterb-vdb.qcow2
HTTP/1.1" 200 197120 "-" "curl/7.29.0"
```

ElementTree（元素树）

课堂实践：使用ElementTree解析Tomcat用户文件

总结

本章讲述了python操作文本的一些基本技术。我们使用了

- 来自标准库的内建string类型
- 正则表达式
- 标准输入输出
- StringIO
- urllib模块

将以上知识进行联合，应用到以下案例中：

- 解析apache日志文件
- 解析Tomcat用户文件

当复杂的文本处理已经超过了使用grep，sed，awk所能应对的程度时，我们可以选择python。