APACHE WEB服务

Web基础

我们平时上网的时候，输入网址的时候都会输入什么，www.baidu.com，对不对，那么这个www是什么呢？这个www代表的是world wide web，WWW可以让Web客户端（常用浏览器）访问浏览Web服务器上的页面。 它是一个由许多互相链接的超文本组成的系统，通过互联网访问，在这个系统中，每个有用的事物，称为一样“资源”；并且由一个全局“统一资源标识符”（URL）标识；这些资源通过超文本传输协议（Hypertext Transfer Protocol）传送给用户，而后者通过点击链接来获得资源。

HTTP是Hypertext Transfer Protocol的缩写，即超文本传输协议。 顾名思义，HTTP提供了访问超文本信息的功能，是WWW浏览器和WWW服务器之间的应用层通信协议。HTTP协议是用于分布式协作超文本信息系统的、通用的、面向对象的协议。通过扩展命令，它可用于类似的任务，如域名服务或分布式面向对象系统。WWW使用HTTP协议传输各种超文本页面和数据。

网页文件是用HTML（标准通用标记语言下的一个应用）编写的，可在WWW上传输，能被浏览器识别显示的文本文件。其扩展名是.htm和.html。

Web的服务器最常见的有windows上的IIS，还有linux上的apache。另外还有例如nginx等的轻量级服务器。都是比较常用的一些服务器。我们基础课主要简单介绍一些apache的一个简单搭建。

Apache的简介

- HTTP (Web) server、开源、1995年
- 官网： http://httpd.apache.org/

Apache 起初由 Illinois 大学 Urbana-Champaign 的国家高级计算程序中心开发。此后，Apache 被开放源代码团体的成员不断的发展和加强。Apache 服务器拥有牢靠可信的美誉，已用在超过半数的因特网站中－特别是几乎所有最热门和访问量最大的网站。

开始，Apache只是Netscape网页服务器（现在是Sun ONE）的之外的开放源代码选择。渐渐的，它开始在功能和速度。超越其他的基于Unix的HTTP服务器。1996年4月以来，Apache一直是Internet上最流行的HTTP服务器: 1999年5月它在 57% 的网页服务器上运行；到了2005年7月这个比例上升到了69%。

作者宣称因为这个名字好记才在最初选择它，但是流传最广的解释是（也是最显而易见的）:这个名字来自这么一个事实:当Apache在1995年初开发的时候，它是由当时最流行的HTTP服务器NCSA HTTPd 1.3 的代码修改而成的，因此是“一个修补的（a patchy）”服务器。然而在服务器官方网站的FAQ中是这么解释的:“‘Apache’这个名字是为了纪念名为Apache(印地语)的美洲印第安人土著的一支，众所周知他们拥有高超的作战策略和无穷的耐性”。

Apache的软件名称就叫做httpd，这里要注意一下，这里我们以el6作为服务器，el7作为客户端。

Apache的软件结构

```
apache
软件            httpd
                      rhel6    httpd-2.2.15-29.el6_4.x86_64
                      rhel7    httpd-2.4.6-17.el7.x86_64
service      httpd
daemon       httpd
端口           80
配置文件     /etc/httpd/conf/httpd.conf
                 /etc/httpd/conf.d/*.conf
数据文件     /var/www/
                 /var/www/uplooking/ www.uplooking.com----网站根目录
日志文件     /var/log/httpd
```

Apache虚拟主机方案

1. 基于端口的虚拟主机

```
www.uplooking.com:80     ---->    /var/www/uplooking.com/
www.uplooking.com:8080   ---->    /var/www/abc.com/
```

1. 基于名称的虚拟主机

```
www.uplooking.com----/var/www/uplooking.com/
www.abc.com----/var/www/abc.com/
```

Apache实践

项目实践**1**：配置基于端口的虚拟主机

```
    8080--->/var/www/8080.com/
****************************************
1)修改主配置文件，打开要监听的端口号
[root@rhel6 ~]# vim /etc/httpd/conf/httpd.conf
Listen 80
Listen 8080
2)为不同的虚拟主机创建配置文件
[root@rhel6 conf.d]# pwd
/etc/httpd/conf.d
[root@rhel6 conf.d]# vim 8080.conf
<VirtualHost *:8080>
    ServerAdmin booboo@8080.com
    DocumentRoot /var/www/8080.com
    ServerName www.8080.com
    ErrorLog logs/8080.com-error_log
    CustomLog logs/8080.com-access_log common
</VirtualHost>

3.创建网站根目录，并创建网站默认首页
[root@rhel6 conf.d]# mkdir /var/www/8080.com
[root@rhel6 conf.d]# echo 8080.com > /var/www/8080.com/index.html
4.重启服务
[root@rhel6 conf.d]# service httpd start
```

拓展8081和8082端口

1. 关闭selinux
2. 配置selinux安全上下文，允许httpd监听8081和8082端口

```
[root@rhel6 conf.d]# semanage port -l|grep http
http_cache_port_t              tcp      3128, 8080, 8118, 8123, 10001-10010
http_cache_port_t              udp      3130
http_port_t                    tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
pegasus_http_port_t            tcp      5988
pegasus_https_port_t           tcp      5989

[root@rhel6 conf.d]# semanage port -a -t http_port_t -p tcp 8081
[root@rhel6 conf.d]# semanage port -a -t http_port_t -p tcp 8082
```

项目实践**2**：配置基于名称的虚拟主机

```
    www.taobao.com       /var/www/taobao.com/
    www.abc.com      /var/www/abc.com/
**************************************************
#####
服务端
#####
1.修改主配置文件
[root@rhel6 conf.d]# vim /etc/httpd/conf/httpd.conf
NameVirtualHost *:80
2.创建拓展配置文件
[root@rhel6 conf.d]# vim taobao.conf
<VirtualHost *:80>
    ServerAdmin booboo@taobao.com
    DocumentRoot /var/www/taobao.com
    ServerName www.taobao.com
    ErrorLog logs/taobao.com-error_log
    CustomLog logs/taobao.com-access_log common
</VirtualHost>

<Directory "/var/www/taobao.com/">
    Options Indexes ==>如果没有对应的index文件，就将该目录下的其他文件罗列出来。
</Directory>

[root@rhel6 conf.d]# vim abc.conf
[root@rhel6 conf.d]# cat abc.conf
<VirtualHost *:80>
    ServerAdmin booboo@abc.com
    DocumentRoot /var/www/abc.com
    ServerName www.abc.com
    ErrorLog logs/abc.com-error_log
    CustomLog logs/abc.com-access_log common
</VirtualHost>

<Directory "/var/www/abc.com/">
    Options Indexes
</Directory>

[root@rhel6 conf.d]# mkdir /var/www/taobao.com
[root@rhel6 conf.d]# mkdir /var/www/abc.com
[root@rhel6 conf.d]# echo www.taobao.com > /var/www/taobao.com/index.html
[root@rhel6 conf.d]# echo www.abc.com > /var/www/abc.com/index.html
[root@rhel6 conf.d]# service httpd restart
==========================================================================================
=
#####
客户端
#####
1.安装elinks软件，bash下的一个网站浏览软件
[root@rhel7 mnt]# yum install -y elinks
2.添加域名解析
[root@rhel7 mnt]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
```

```
172.25.0.11 www.taobao.com
172.25.0.11 www.abc.com
3.通过links命令访问www.taobao.com和www.abc.com
[root@rhel7 mnt]# links www.taobao.com
[root@rhel7 mnt]# links www.abc.com
#####
-X 带图形化界面的远程登录
#####
[kiosk@foundation0 Desktop]$ ssh root@172.25.0.10 -X
Last login: Fri Aug  5 03:14:10 2016
/usr/bin/xauth:  file /root/.Xauthority does not exist
[root@rhel7 ~]# firefox
```

项目实践3：配置基于名称的虚拟主机针对某一个目录做限制

```
    Directory是针对某一个目录做限制的意思。
        Options
            Indexes的意思代表如果没有对应的index文件，就将该目录下的其他文件罗列出来。
            Multiview代表的是多视图
            Follow sysmlink代表允许连接到其他目录。
        AllowOverride None
        最常用的是：
        Order allow,deny
        Allow from All
        deny from 172.25.0.10==>该目录允许所有人，除了172.25.0.11
        或者
        Order deny,allow
        deny from All
        allow from 172.25.0.10==>该目录不允许所有人，除了172.25.0.11
```

项目实践4：配置基于名称的虚拟主机_别名

- 别名　　alias
  - 作用在访问该目录的时候,无论之前的虚拟主机站点名是什么,会统一转到某一个指定的目录。

```
Alias /download/ /"var/www/soft/"
```

访问 download 目录时,无论站点名是什么,都会交给 var/www/soft/ 去处理,注意:代表目录时,最后一个 / 不能省略

项目实践4：配置基于名称的虚拟主机_用户名和密码访问

- 1）配置文件中添加认证　　AllowOverride AuthConfig
  - 2）创建认证文件和用户密码　　htpasswd -cmb /etc/httpd/test booboo uplooking
    *3）新增用户和密码　　htpasswd -bm /etc/httpd/test tom uplooking
    *4）删除用户　　htpasswd -D /etc/httpd/test jack
    *5）修改密码　　先删除再创建

```
设置用户和密码
<Directory "/var/www/booboo.com/football/">
Options Indexes MultiViews FollowSymLinks
AllowOverride AuthConfig
#仅有网页认证 ( 账号密码 ) 可覆写 ;
AuthName "student"
#在要你输入账号与密码的对话窗口中 , 出现的『提示字符』
AuthType basic
# 认证的类型
AuthUserFile "/etc/httpd/test"
# 这个目录所使用的账号密码配置文件
Require valid-user
# 后面接可以使用的账号 , 此处是让该密码文件内的用户都能够登入
Order allow,deny
Allow from all
</Directory>

# 通过htpasswd添加用户和密码
[root@rhel6 conf.d]# htpasswd -cmb /etc/httpd/test booboo uplooking
Adding password for user student
# 再添加一个用户和密码
[root@rhel6 www]# htpasswd -bm /etc/httpd/test tom uplooking
Adding password for user tom
[root@rhel6 www]# htpasswd -bm /etc/httpd/test jack uplooking
Adding password for user jack
# 查看保存apache用户名和密码的文件内容
[root@rhel6 www]# cat /etc/httpd/test
booboo:$apr1$y6g/XYrn$eEOR4WeAPfONSLExj7x2D1
tom:$apr1$epxVgzER$Or7.R0.5s3z8FbPKrOS1e1
jack:$apr1$Alo3.7mf$3CmcCG9mXp7eEALUsa3YY1

# 删除用户
[root@rhel6 www]# htpasswd -D /etc/httpd/test jack
Deleting password for user jack
[root@rhel6 www]# cat /etc/httpd/test
booboo:$apr1$y6g/XYrn$eEOR4WeAPfONSLExj7x2D1
tom:$apr1$epxVgzER$Or7.R0.5s3z8FbPKrOS1e1

# 修改用户密码只能先删除再创建
[root@rhel6 abc.com]# htpasswd --help
Usage:
    htpasswd [-cmdpsD] passwordfile username
    htpasswd -b[cmdpsD] passwordfile username password

    htpasswd -n[mdps] username
    htpasswd -nb[mdps] username password
 -c  Create a new file.
 -n  Don\'t update file; display results on stdout.
 -m  Force MD5 encryption of the password.
 -d  Force CRYPT encryption of the password (default).
 -p  Do not encrypt the password (plaintext).
 -s  Force SHA encryption of the password.

 -b  Use the password from the command line rather than prompting for it.
```

```
 -D  Delete the specified user.
On Windows, NetWare and TPF systems the '-m' flag is used by default.
On all other systems, the '-p' flag will probably not work.
```

项目实践**5**：在**rhel7**中搭建**apahce**，实现以下功能

- 配置基于名称的虚拟主机 www.abc.com 和 www.uplooking.com
- 只允许rhel6 172.25.0.11 能够访问 www.uplooking.com

```
rhel6   rhel7
2.2 2.4
order   require

》eg1:所有请求都被拒绝
order deny,allow //先拒绝，后允许
deny from all   //拒绝所有
------------------------------
require all denied //拒绝所有

》eg2：所有请求都允许
order allow，deny //先允许，后拒绝
allow from all
-------------------------------------
require all granted

》拒绝172.25.0.10/test.uplooking.com 的请求
order allow，deny //先允许，后拒绝
allow from all
deny from 172.25.0.10
--------------------------------------
require  all granted
require no ip 172.25.0.10
require no host test.uplooking.com

》只允许172.25.3.10和172.25.0.0/24 uplooking.com 的请求
order deny，allow //先决绝，后允许
deny from all
allow from 172.25.3.10
allow from 172.25.0.0/24
-------------------------------------
require all denied
require ip 172.25.3.10 172.25.0
require host  uplooking.com
```

Apache配置文件**/etc/httpd/conf/httpd.conf**

1）

ServerRoot “/etc/httpd“

ServerRoot用于指定守护进程httpd的运行目录，httpd在启动之后将自动将进程的当前目录改变为这个目录，因此如果设置文件中指定的文件或目录是相对路径，那么真实路径就位于这个ServerR oot定义的路径之下。

2）

hhhidFile /var/run/httpd.pid

PidFile指定的文件将记录httpd守护进程的进程号，由于httpd能自动复制其自身，因此系统中有多个httpd进程，但只有一个进程为最初启动的进程，它为其他进程的父进程，对这个进程发送信号将影响所有的httpd进程。PidFILE定义的文件中就记录httpd父进程的进程号。

Timeout 300

Timeout定义客户程序和服务器连接的超时间隔，超过这个时间间隔（秒）后服务器将断开与客户机的连接。

KeepAlive On

在HTTP 1.0中，一次连接只能作传输一次HTTP请求，而KeepAlive参数用于支持HTTP 1.1版本的一次连接、多次传输功能，这样就可以在一次连接中传递多个HTTP请求。不过只有较新的浏览器才支持这个功能

MaxKeepAliveRequests 100

MaxKeepAliveRequests为一次连接可以进行的HTTP请求的最大请求次数。将其值设为0将支持在一次连接内进行无限次的传输请求。事实上没有客户程序在一次连接中请求太多的页面，通常达不到这个上限就完成连接了。

KeepAliveTimeout 15

KeepAliveTimeout测试一次连接中的多次请求传输之间的时间，如果服务器已经完成了一次请求，但一直没有接收到客户程序的下一次请求，在间隔超过了这个参数设置的值之后，服务器就断开连接。

ThreadsPerChild 50

设置服务器使用进程的数目。这是以服务器的响应速度为准的，数目太大则会变慢

MaxRequestsPerChild 30

使用子进程的方式提供服务的Web服务，常用的方式是一个子进程为一次连接服务，这样造成的问题就是每次连接都需要生成、退出子进程的系统操作，使得这些额外的处理过程占据了计算机的大量处理能力。因此最好的方式是一个子进程可以为多次连接请求服务，这样就不需要这些生成、退出进程的系统消耗，Apache就采用了这样的方式，一次连接结束后，子进程并不退出，而是停留在系统中等待下一次服务请求，这样就极大的提高了性能。但由于在处理过程中子进程要不断的申请和释放内存，次数多了就会造成一些内存垃圾，就会影响系统的稳定性，并且影响系统资源的有效利用。因此在一个副本处理过一定次数的请求之后，就可以让这个子进程副本退出，再从原始的htt pd进程中重新复制一个干净的副本，这样就能提高系统的稳定性。这样，每个子进程处理服务请求次数由MaxRe questPerChild定义。 缺省的设置值为30，这个值对于具备高稳定性特点的FreeBSD系统来讲是过于保守的设置，可以设置为1000甚至更高，设置为0支持每个副本进行无限次的服务处理。

ServerAdmin root@localhost

配置文件中应该改变的也许只有ServerAdmin， 这一项用于配置WWW服务器的管理员的email地址，这将在HTTP服务出现错误的条件下返回给浏览器，以便让Web使用者和管理员联系，报告错误。习惯上使用服务器上的webmaster作为WWW服务器的管理员，通过邮件服务器的别名机制，将发送到webmaster 的电子邮件发送给真正的Web管理员。

ServerName localhost

缺省情况下，并不需要指定这个ServerName参数，服务器将自动通过名字解析过程来获得自己的名字，但如果服务器的名字解析有问题（通常为反向解析不正确），或者没有正式的DNS名字，也可以在这里指定I P地址。当ServerName设置不正确的时候，服务器不能正常启动。

通常一个Web服务器可以具有多个名字，客户浏览器可以使用所有这些名字或IP地址来访问这台服务器，但在没有定义虚拟主机的情况下，服务器总是以自己的正式名字回应浏览器。ServerName就定义了Web服务器自己承认的正式名字，例如一台服务器名字（在DNS中定义了A类型）为freebsd.exmaple.org.cn，同时为了方便记忆还定义了一个别名（CNAME记录）为www.exmaple.org.cn，那么Apache自动解析得到的名字就为freebsd.example.org.cn，这样不管客户浏览器使用哪个名字发送请求，服务器总是告诉客户程序自己为freebsd.example.org.cn。虽然这一般并不会造成什么问题，但是考虑到某一天服务器可能迁移到其他计算机上，而只想通过更改DNS中的www别名配置就完成迁移任务，所以不想让客户在其书签中使用 freebsd记录下这个服务器的地址，就必须使用ServerName来重新指定服务器的正式名字。

3）

DocumentRoot “/var/www/html“

DocumentRoot定义这个服务器对外发布的超文本文档存放的路径，客户程序请求的URL就被映射为这个目录下的网页文件。这个目录下的子目录，以及使用符号连接指出的文件和目录都能被浏览器访问，只是要在URL上使用同样的相对目录名。注意，符号连接虽然逻辑上位于根文档目录之下，但实际上可以位于计算机上的任意目录中，因此可以使客户程序能访问那些根文档目录之外的目录，这在增加了灵活性的同时但减少了安全性。Apache在目录的访问控制中提供了FollowSymLinks选项来打开或关闭支持符号连接的特性。

4）

```
# <Directory />
    Options FollowSymLinks
    AllowOverride None
   </Directory>
```

Apache服务器可以针对目录进行文档的访问控制，然而访问控制可以通过两种方式来实现，一个是在设置文件httpd.conf（或access.conf）中针对每个目录进行设置，另一个方法是在每个目录下设置访问控制文件，通常访问控制文件名字为.htaccess。虽然使用这两个方式都能用于控制浏览器的访问，然而使用配置文件的方法要求每次改动后重新启动httpd守护进程，比较不灵活，因此主要用于配置服务器系统的整体安全控制策略，而使用每个目录下的.htaccess文件设置具体目录的访问控制更为灵活方便.

由于Apache对一个目录的访问控制设置是能够被下一级目录继承的，因此对根目录的设置将影响到它的下级目录。注意由于AllowOverride None的设置，使得Apache服务器不需要查看根目录下的访问控制文件，也不需要查看以下各级目录下的访问控制文件，直至httpd.conf（或access.conf ）中为某个目录指定了允许Alloworride，即允许查看访问控制文件。由于Apache对目录访问控制是采用的继承方式，如果从根目录就允许查看访问控制文件，那么Apache就必须一级一级的查看访问控制文件，对系统性能会造成影响。而缺省关闭了根目录的这个特性，就使得Apache从httpd.conf中具体指定的目录向下搜寻，减少了搜寻的级数，增加了系统性能。因此对于系统根目录设置AllowOverride None不但对于系统安全有帮助，也有益于系统性能。

```
    <Directory />
   Options Indexes FollowSymLinks
     AllowOverride None
     Order allow,deny
     Allow from all
    </Directory>
```

这里定义的是系统对外发布文档的目录的访问设置，设置不同的AllowOverride选项，以定义配置文件中的目录设置和用户目录下的安全控制文件的关系，而Options选项用于定义该目录的特性。

```
#  ErrorLog /var/log/httpd-error.log
    LogLevel warn
    LogFormat “%h %l %u %t \“%r\“ %>s %b \“%{Referer}i\“ \“%{User-Agent}i\““ combined
    LogFormat “%h %l %u %t \“%r\“ %>s %b“ common
    LogFormat “%{Referer}i -> %U“ referer
    LogFormat “%{User-agent}i“ agent
    #CustomLog /var/log/httpd-access.log common
    #CustomLog /var/log/httpd-referer.log referer
    #CustomLog /var/log/httpd-agent.log agent
    CustomLog /var/log/httpd-access.log combined
```

这里定义了系统日志的形式，对于服务器错误记录，由ErrorLog、LogLevel 来定义不同的错误日志文件及其记录内容。 这是通过在 CustomLog中指定不同的记录类型来完成的。common表示普通的对单页面请求访问记录，referer表示每个页面的引用记录，可以看出一个页面中包含的请求数，agent表示对客户机的类型记录，显然可以将现有的combined 定义的设置行注释掉，并使用common、referer和agent作为CustomLog的参数，来为不同种类的日志分别指定日志记录文件。显然，LogFormat是用于定义不同类型的日志进行记录时使用的格式， 这里使用了以%开头的宏定义，以记录不同的内容。

如果这些参数指定的文件使用的是相对路径，那么就是相对于ServerRoot的路径。

```
# Alias /icons/ “/usr/local/www/icons/“
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
```

Alias参数用于将URL与服务器文件系统中的真实位置进行直接映射，一般的文档将在DocumentRoot 中进行查

询，然而使用Alias定义的路径将直接映射到相应目录下，而不再到DocumentRoot 下面进行查询。因此Alias可以用来映射一些公用文件的路径，例如保存了各种常用图标的icons路径。这样使得除了使用符号连接之外，文档根目录（DocumentRoot）外的目录也可以通过使用了Alias映射，提供给浏览器访问。
定义好映射的路径之后，应该需要使用Directory语句设置访问限制。

```
#NameVirtualHost 12.34.56.78:80
#NameVirtualHost 12.34.56.78
#
# ServerAdmin webmaster@host.some_domain.com
# DocumentRoot /www/docs/host.some_domain.com
# ServerName host.some_domain.com
# ErrorLog logs/host.some_domain.com-error_log
# CustomLog logs/host.some_domain.com-access_log common
#
```

这个字段是虚拟主机的相关内容，虚拟主机是在一台Web服务器上，可以为多个单独域名提供Web服务，并且每个域名都完全独立，包括具有完全独立的文档目录结构及设置，这样域名之间完全独立，不但使用每个域名访问到的内容完全独立，并且使用另一个域名无法访问其他域名提供的网页内容。