**6\_Operation02LNMPnginx+fastcgiNGINX高级技术**

**一 案例：部署LNMP环境(动态网页环境)**

静态网页:pdf,doc,mp3,mp4,txt,html...

动态网页:shell,php,java,python,perl,ruby...

静态网页处理流程:用户->nginx[/usr/.../html/静态文档]

动态网页处理流程:用户->nginx[/usr/.../html/编程语言编写的执行文档]

**LNMP**:主流的企业网站平台,Linux+Nginx+MySQL\MariaDB+PHP\Perl\Python

**1.1 问题**

安装部署Nginx、MariaDB、PHP环境

安装部署Nginx、MariaDB、PHP、PHP-FPM；

启动Nginx、MariaDB、FPM服务；

并测试LNMP是否工作正常。

**1.2 方案**

LNMP（Linux、Nginx、MySQL、PHP）

在系统中，源码安装Nginx，使用RPM包安装MariaDB、PHP、PHP-FPM软件。

操作过程中需要安装的软件列表如下：

nginx

mariadb、mariadb-server、mariadb-devel #客户端,服务端,依赖包

php、php-fpm、php-mysql #解释器,服务,扩展包(用于和mysql连接)

备注：mariadb（数据库客户端软件）、mariadb-server（数据库服务器软件）、mariadb-devel（其他客户端软件的依赖包）、php（解释器）、php-fpm（进程管理器服务）、php-mysql（PHP的数据库扩展包）。

**1.3 步骤**

**1.3.1安装软件**(proxy主机)

1）使用yum安装基础依赖包

~]# yum -y install gcc openssl-devel pcre-devel

2）源码安装Nginx（如果前面课程中已经安装Nginx，则忽略这一步）

~]# useradd -s /sbin/nologin nginx

~]# tar -xvf nginx-1.12.2.tar.gz

~]# cd nginx-1.12.2

nginx-1.12.2]# ./configure \

> --user=nginx --group=nginx \

> --with-http\_ssl\_module

~]# make && make install

3）安装MariaDB

Mariadb在新版RHEL7光盘中包含有该软件，配置yum源后可以直接使用yum安装：

~]# yum -y install mariadb mariadb-server mariadb-devel

4）php和php-fpm

~]# yum -y install php php-mysql

~]# yum -y install php-fpm

**1.3.2启动服务**

1）启动Nginx服务:80端口（如果已经启动nginx，则可以忽略这一步）

这里需要注意的是，如果服务器上已经启动了其他监听80端口的服务软件（如httpd），则需要先关闭该服务，否则会出现冲突。

~]# systemctl stop httpd #如果该服务存在则关闭该服务

~]# /usr/local/nginx/sbin/nginx #启动Nginx服务

~]# netstat -utnlp | grep :80 #端口检查

2）启动MySQL服务:3306端口

~]# systemctl start mariadb #启动服务器

~]# systemctl status mariadb #查看服务状态

~]# systemctl enable mariadb #设置开机启动

~]# netstat -utnlp | grep :3306 #端口检查

3）启动PHP-FPM服务:9000端口

~]# systemctl start php-fpm #启动服务

~]# systemctl status php-fpm #查看服务状态

~]# systemctl enable php-fpm #设置开机启动

~]# netstat -utnlp | grep :9000 #端口检查

**1.3.3 client主机测试**firefox 192.168.4.5,失败,L N M P之间无关联,需要修改配置文件进行关联,实现对PHP页面的支持.

**二 案例2：构建LNMP平台**

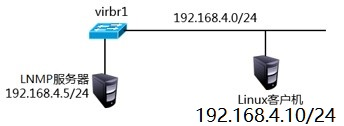
**2.1 问题**

沿用练习一，通过调整Nginx服务端配置，实现以下目标(对动态页面的支持)：

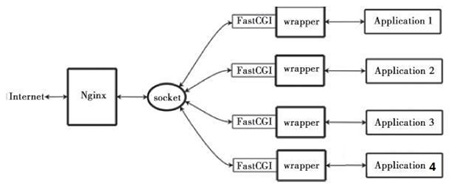
配置Fast-CGI支持PHP网页\创建PHP测试页面，测试使用PHP连接数据库的效果

**2.2 方案**

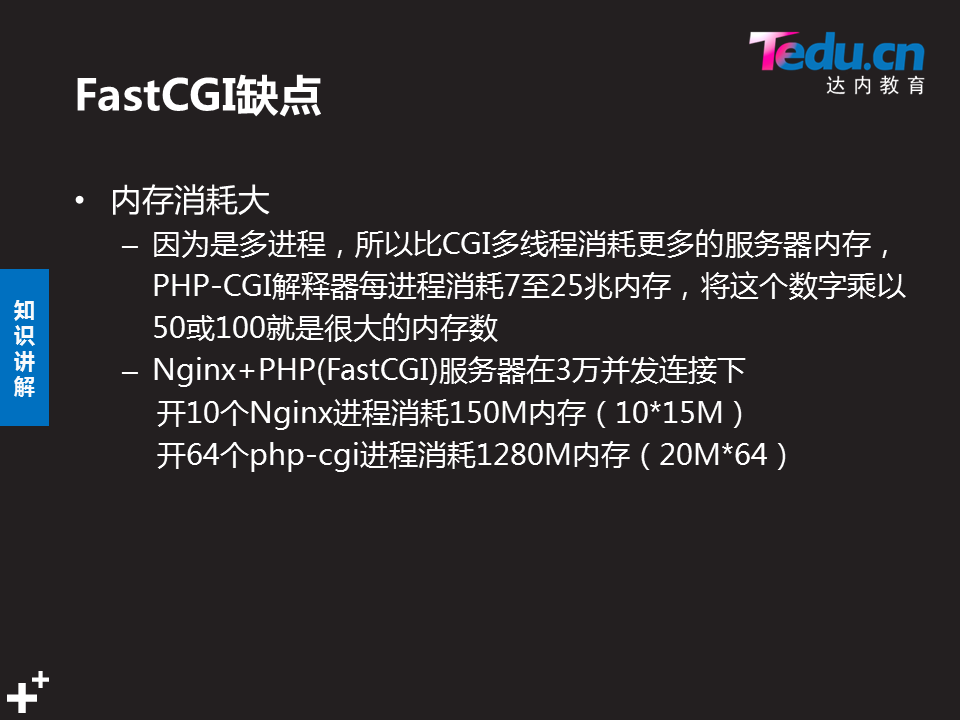
使用2台RHEL7虚拟机，其中一台作为LNMP服务器（192.168.4.5）、另外一台作为测试用的Linux客户机（192.168.4.10），如图所示。



Nginx结合FastCGI技术即可支持PHP页面架构，如图所示。







因此本案例，需要延续练习一的实验内容，通过修改Nginx及php-fpm配置文件实现对PHP页面的支持。

FastCGI支持PHP,C/C++,Java,Perl,Python,Ruby等

注意，FastCGI的内存消耗问题，一个PHP-FPM解释器将消耗约25M的内存。

**2.3 步骤**

**2.3.1 php-fpm配置文件**

1）查看php-fpm配置文件（实验中不需要修改该文件）

etc]# vim **/etc/php-fpm.d/www.conf**

[www]

listen = 127.0.0.1:9000 #PHP端口号

pm.max\_children = 32  #最大进程数量

pm.start\_servers = 15   #最小进程数量

pm.min\_spare\_servers = 5    #最少需要几个空闲着的进程

pm.max\_spare\_servers = 32    #最多允许几个进程处于空闲状态

**2.3.2 修改Nginx配置文件并启动服务**

**nginx**

**如果用户访问静态,则直接返回**

**如果用户访问动态,则转发给端口9000,执行脚本后返回**

**location可以匹配用户的地址栏,从/(网页根目录)开始**

**location / 能匹配一切,最后匹配;location可理解为”匹配或if”**

详细流程:

用户地址栏http://192.168.4.5/a.jpg->找文件->返回

用户地址栏http://192.168.4.5/b.php->匹配->找文件->转发->执行->返回

~]# **vim /usr/local/nginx/conf/nginx.conf**

location / {

root html; #网页根目录,表示网页在目录html下

index index.php index.html index.htm;

#设置默认首页，当用户在浏览器地址栏中只写域名或IP，不说访问什么页面时，服务器会把三个页面从左到右按顺序返回给用户

}

location ~ \.php$ { #~号代表正则匹配包含,匹配以.php结尾的,\表示转义

root html; #声明.php文件在网页根目录下

fastcgi\_pass 127.0.0.1:**9000**;

**#将请求转发给本机9000端口，调用PHP解释器**

fastcgi\_index index.php;

#fastcgi\_param...\_script\_name; #此行注释掉

include **fastcgi.conf**; #加载其他配置文件

}

~]# /usr/local/nginx/sbin/nginx -s reload #nginx启动状态下重载

**2.3.3 创建PHP页面，测试LNMP架构能否解析PHP页面**

1）创建PHP测试页面1，可以参考lnmp\_soft/php\_scripts/test.php：

~]# vim /usr/local/nginx/html/test.php

<?php

$i="This is a test Page";

echo $i;

?>

2）创建PHP测试页面,连接并查询MariaDB数据库。

可以参考lnmp\_soft/php\_scripts/mysql.php:

~]# vim /usr/local/nginx/html/mysql.php

<?php

$mysqli = new mysqli('localhost','root','密码','mysql');

#注意：root为mysql数据库的账户名称，密码需要修改为实际mysql密码，无密码则留空即可

#localhost是数据库的域名或IP，mysql是数据库的名称

if (mysqli\_connect\_errno()){

    die('Unable to connect!'). mysqli\_connect\_error();

} #底色表示1段代码,下同

$sql = "select \* from user";

$result = $mysqli->query($sql);

while($row = $result->fetch\_array()){

    printf("Host:%s",$row[0]);

    printf("</br>");

    printf("Name:%s",$row[1]);

    printf("</br>");

}

?>

3）客户端使用浏览器访问服务器PHP首页文档，检验是否成功：

[root@client ~]# firefox http://192.168.4.5/test.php

[root@client ~]# firefox http://192.168.4.5/mysql.php

4）LNMP常见问题日志

Nginx的默认访问日志文件为/usr/local/nginx/logs/access.log

Nginx的默认错误日志文件为/usr/local/nginx/logs/error.log

PHP的默认错误日志文件为/var/log/php-fpm/www-error.log

如果动态网站访问失败，可用参考错误日志，查找错误信息。

**三 案例3：地址重写**

**3.1 问题**

沿用练习二，通过调整Nginx服务端配置，实现以下目标：

所有访问a.html的请求，重定向到b.html;

所有访问192.168.4.5的请求重定向至www.tmooc.cn；

所有访问192.168.4.5/下面子页面，重定向至www.tmooc.cn/下相同的页面；

实现firefox与curl访问相同页面文件，返回不同的内容。

**3.2 方案**

关于Nginx服务器的地址重写，主要用到的配置参数是rewrite：

rewrite regex replacement flag

**rewrite 旧地址 新地址 [选项]** **# /表示网页根目录,旧地址支持正则表达式**

**3.3 步骤**

**3.3.1修改配置文件(访问a.html重定向到b.html)**

1）修改Nginx配置文件：

~]# vim /usr/local/nginx/conf/nginx.conf

server {

listen 80;

server\_name localhost;  **#测试时要注意添加地址重写时的域名**

**rewrite /a.html /b.html;**   **#在server\_name下添加**

location / {

root html;

index index.html index.htm;

}

}

~]# echo "BB" > /usr/local/nginx/html/b.html

2）重新加载配置文件

~]# /usr/local/nginx/sbin/nginx -s reload

3）客户端测试

[root@client ~]# firefox http://192.168.4.5/a.html

**3.3.2访问a.html重定向到b.html（跳转地址栏）**

1）修改Nginx服务配置：

~]# vim /usr/local/nginx/conf/nginx.conf

server {

listen 80;

server\_name localhost;

**rewrite /a.html /b.html redirect;**

#选项redirect的功能是在浏览器地址栏显示重写后的地址

location / {

     root html;

index index.html index.htm;

}

}

2）重新加载配置文件

~]# /usr/local/nginx/sbin/nginx -s reload

3）客户端测试（仔细观察浏览器地址栏的变化）

[root@client ~]# firefox http://192.168.4.5/a.html

**3.3.3修改配置文件(访问192.168.4.5的请求重定向至www.tmooc.cn)**

1) 修改Nginx配置文件

~]# vim /usr/local/nginx/conf/nginx.conf

server {

listen 80;

server\_name localhost;

**rewrite ^/ http://www.tmooc.cn/;**

**# ^/,正则表达式,将网页根目录下的所有页面全部重写为www.tmooc.cn**

**# rewrite /a.html /b.html redirect;** #此行可注释掉

location / {

root html;

index index.html index.htm;

}

}

2）重新加载配置文件

~]# /usr/local/nginx/sbin/nginx -s reload

#请先确保nginx是启动状态，否则运行该命令会报错,报错信息如下：

3）客户端测试（真实机测试，真实机才可以连接tmooc）

[root@room9pc01 ~]# firefox http://192.168.4.5

**3.3.4修改配置文件(访问192.168.4.5/下面子页面，重定向至www.tmooc.cn/下相同的页面)**

1) 修改Nginx配置文件

~]# vim /usr/local/nginx/conf/nginx.conf

server {

listen 80;

server\_name localhost;

**rewrite ^/(.\*)$ http://www.tmooc.cn/$1;**

# $1表示粘贴前面的(.\*)

location / {

root html;

index index.html index.htm;

}

}

2）重新加载配置文件

~]# /usr/local/nginx/sbin/nginx -s reload

3）客户端测试（真实机测试，真实机才可以连接tmooc）

[root@room9pc01 ~]# firefox http://192.168.4.5

[root@room9pc01 ~]# firefox http://192.168.4.5/test

**3.3.5修改配置文件(实现curl和火狐访问相同链接返回的页面不同),类似手机和电脑访问同一个网站同一个页面,显示的页面效果不一样**

1) 创建网页目录以及对应的页面文件：

~]# echo "I am Normal page" > /usr/local/nginx/html/test.html

~]# mkdir **-p** /usr/local/nginx/html/firefox/

~]# echo "firefox page" > /usr/local/nginx/html/firefox/test.html

2) 修改Nginx服务配置

~]# vim /usr/local/nginx/conf/nginx.conf

server {

listen 80;

server\_name localhost;

location / {

root html;

index index.html index.htm;

}

#这里，~(包含)符号代表正则匹配，\*符号代表不区分大小写

if ($http\_user\_agent ~\* firefox) {  #识别客户端firefox浏览器

rewrite ^/(.\*)$ /firefox/$1;

}

}

3）重新加载配置文件

~]# /usr/local/nginx/sbin/nginx -s reload

4）客户端测试

[root@client ~]# firefox http://192.168.4.5/test.html

[root@client ~]# curl -u tom:123456 http://192.168.4.5/test.html

5）地址重写格式【总结】

rewrite 旧地址 新地址 [选项];旧地址支持正则表达式

选项

last 不再读其他rewrite

break 不再读其他语句，结束请求

redirect 临时重定向

permament 永久重定向