**Morning：**

1. 部署LNMP环境

**-关于动态、静态网站**

参考阶段1-->ENGINEER-->DAY06-->page03

**-LNMP环境**

**L-->**Linux

**N-->**nginx

**M-->**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)

**P-->**php(解释器)、php-fpm(进程管理器服务)、php-mysql(**模块化：PHP的mysql扩展包**)

1. **安装软件**
   1. 安装基础依赖包

yum -y install gcc openssl-devel pcre-devel zlib-devel

* 1. 源码安装Nginx

参考：阶段2 --> 2.OPERATION --> DAY01 --> Page01

* 1. 安装MariaDB

yum -y install mariadb mariadb-server mariadb-devel

* 1. 安装php与php-fpm（在lnmp\_soft中提供）

yum -y install php php-mysql

cd /root/lnmp-soft

yum -y install php-fpm-5.4.16-42.el7.x86\_64.rpm

#安装本地rpm包

1. **启动服务**
   1. 启动Nginx服务，80端口

/usr/local/nginx/sbin/nginx #手动开启

/etc/rc.local --> /etc/rc.d/rc.local（软链接）

#开机自启脚本，输入nginx到脚本，给/etc/rc.d/rc.local x权限

* 1. 启动MySQL服务，3306端口

systemctl start|status|enable mariadb

#启动服务|查看服务状态|设为开机自启

* 1. 启动PHP-FPM服务，9000端口

systemctl start|status|enable php-fpm

* 1. 设置防火墙与SELinux

firewall-cmd --set-default-zone=trusted

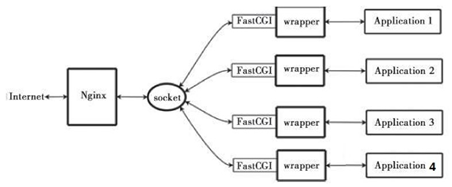
setenforce 0

e) 验证端口状态

netstat -antulp | grep :80 #3306、9000

**Afternoon：**

1、构建LNMP平台



**原理：（Nginx+FastCGI：实现动静分离）**

a）Nginx判断用户请求的页面是静态还是动态。

b）如果是静态，Nginx处理后返回给用户；如果是动态，转发给9000端口，处理后（包括标准/错误输出）返回给Nginx，然后返回到用户浏览器显示内容。

c）如果并发量大，php-fpm会启动更多的php解释进程（每个消耗25M内存）。

步骤：

1. **查看php-fpm配置文件（本实验无需修改）**

vim /etc/php-fpm.d/www.conf

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

1. **修改Nginx配置文件**

server {

root /usr/local/nginx/;

#Mr.Qiu指定server下的根目录，默认为/usr/local/nginx/html

location / {

root html;

index index.php index.html index.htm;

}

location ~ \.php$ { #包含.php结尾

root html;

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

#fastcgi\_paramSCRIPT\_FILENAME /scripts$fastcgi\_script\_name;

#上面配置指令错误，必须注释掉

include fastcgi.conf; #把正确的配置指令引入

}

}

**关于location的知识：（相当于if判断，匹配则无条件执行里面内容）**

-location用于匹配用户地址栏（找不到匹配时，直接匹配location /） -location 支持正则表达式

-location [=|~|~\*|^~] /uri/ {...} 语法介绍：

-配置多个location时，需在server指定root，验证：www.a.com/login/

·= 严格匹配。如果这个查询匹配，那么将停止搜索并立即处理此请求。

·~ 区分大小写匹配(可用正则表达式)

·!~ 区分大小写不匹配

·~\* 区分大小写匹配(可用正则表达式)

·!~\* 不区分大小写不匹配

·^~ 如果把这个前缀用于一个常规字符串,那么告诉nginx 如果路径匹配那 么不测试正则表达式。

·/ 通用匹配, 任何请求都会匹配到

1. **重启Nginx服务**

nginx -s reload

1. **测试能否解析PHP页面**

vim /usr/local/nginx/html/test.php #可创建php脚本

cp mysql.php /usr/local/nginx/html

#拷贝数据库脚本(连接数据库，对数据库作增删改查操作)

firefox http://192.168.4.5/mysql.php #测试PHP解析

1. **日志分析**

Nginx默认访问日志：/usr/local/nginx/logs/access.log

Nginx默认错误日志：/usr/local/nginx/logs/error.log

PHP默认错误日志：/var/log/php-fpm/www-error.log

**关于访问日志access.log配置文件有说明log\_format main**

**(面试常考)**

$remote\_addr - $remote\_user[$time\_local] "$request"

客户端IP - 客户登陆用户(如tom)[访问时间] “请求(哪个页面)协议”

$status $body\_bytes\_sent "$http\_referer"

报错码(正常为200) 网页容量 “从别的网页跳转(一般为-)”

"$http\_user\_agent"

“客户上网的代理工具(操作系统+浏览器)”

Tips：

Nginx的内置变量$http\_user\_agent可以重写手机/电脑的访问页面

“GET /favicon.ico HTTP/1.1” #获取图标文件，显示在浏览器标签旁

1. **动态页面常见错误**

-显示File not found. #找不到test.php

-显示无法连接...An error... #Nginx+php没有启动

-空白页面/显示不正确 #PHP脚本语法错误

-显示下载文件页面 #没有配置location动静分离

1. 地址重写（地址栏跳转）

-地址重写案例：

www.360buy.com --> [www.jd.com](http://www.jd.com) #域名重写

<http://www.baidu.com> --> <https://www.baidu.com> #协议重写

[www.sina.com/pic/图片](http://www.sina.com/pic/图片) --> www.sina.com/picture #目录重写

-配置参数：

rewrite 旧地址(**支持正则**) 新地址 [选项]

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（用于搜索引擎）

permament 永久重定向（用于搜索引擎）

rewrite一般放在location外面

**案例1）访问a.html重定向到b.html（地址栏不变）**

server{

...

server\_name www.a.com;

rewrite /a.html /b.html;

location / { }

}

**案例2）访问a.html重定向到b.html（跳转地址栏）**

server{

...

server\_name www.a.com;

rewrite /a.html /b.html redirect;

location / { }

}

**案例3）访问192.168.4.5的请求重定向至www.tmooc.cn**

server{

...

server\_name www.a.com;

rewrite ^/ http://www.tmooc.cn/; #/开头即跳转（匹配所有）

location / { }

}

Tips：www.a.com/abc/a.jpg --> [www.jd.com](http://www.jd.com)

#弊端：无法实现内容的跳转

rewrite ^/ http://xxx.xx.xx（必须写完整域名）

**案例4）访问192.168.4.5/下面子页面，重定向至www.tmooc.cn/下相同的页面**

server{

...

server\_name www.a.com;

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

location / { }

}

**案例5）实现curl和火狐访问相同链接返回的页面不同**

server{

...

server\_name www.a.com;

location / { }

if ($http\_user\_agent ~\* firefox) {

#识别客户端代理为firefox

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

}

}