LNMP环境相关配置Nginx

原创 kindy520 2018-03-13 23:59:12 评论(0) 947人阅读

LNMP环境搭建已经在上一篇文章记录了。本篇主要记录相关的配置，如Nginx、php

Nginx配置：

1. 默认虚拟主机

Nginx的默认虚拟主机的概念和httpd的类似，第一个被nginx加载的虚拟主机就睡默认虚拟主机。不同的是，nginx可以标记默认虚拟主机，如果没有标记则第一个就是默认的；

修改主配置文件nginx.conf，在结束符号 } 上面加入一行配置，如下：

```
include vhost/*.conf;
}
```

```
    }
    include vhost/*.conf;
}
```
@51CTO博客

意思是/usr/local/nginx/conf/vhost/下面的所有以.conf结尾的文件都会加载，这样我们只需把虚拟主机文件放在vhost目录下就行了

```
[root@centos7-1:~]# mkdir /usr/local/nginx/conf/vhost
[root@centos7-1:~]# cd /usr/local/nginx/conf/vhost/
[root@centos7-1:vhost]# vim default.conf
```
@51CTO博客

```
server
{
    listen 80 default_server;
    server_name default.com;
    index index.htm index.html index.php;
    root /data/nginx/default;
}
```
标记为默认虚拟主机！

@51CTO博客

```
[root@centos7-1:vhost]# /usr/local/nginx/sbin/nginx -t
nginx: the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /usr/local/nginx/conf/nginx.conf test is successful
[root@centos7-1:vhost]# /usr/local/nginx/sbin/nginx -s reload
```
@51CTO博客

创建测试文件：

```
echo "Nginx默认虚拟主机" > /data/nginx/default.com/index.html
```

```
[root@centos7-1:vhost]# curl -x127.0.0.1:80 default.com
Nginx默认虚拟主机
[root@centos7-1:vhost]#
```
@51CTO博客

2. 用户认证

新建一个虚拟主机：

```
cd /usr/local/nginx/conf/vhost
vim test.com.conf
```

```
[root@centos7-1:vhost]# cat test.com.conf
server
{
    listen 80;
    server_name test.com;
    index index.htm index.html index.php;
    root /data/nginx/test.com;

    location /
    {
        auth_basic  "Auth";
        auth_basic_user_file    /usr/local/nginx/conf/htpasswd;
    }
}
```


使用httpd的htpasswd命令：

```
[root@centos7-1:vhost]# htpasswd -c /usr/local/nginx/conf/htpasswd test
New password:
Re-type new password:
Adding password for user test
```



```
[root@centos7-1:vhost]# usr/local/nginx/sbin/nginx -t
nginx: the configuration file /usr/local/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /usr/local/nginx/conf/nginx.conf test is successful
[root@centos7-1:vhost]# usr/local/nginx/sbin/nginx -s reload
[root@centos7-1:vhost]# mkdir /data/nginx/test.com
[root@centos7-1:vhost]# echo "test.com !" > /data/nginx/test.com/index.html
echo "test.com " 选中状态下 Ctrl+c 复制 > /data/nginx/test.com/index.html
[root@centos7-1:vhost]# curl -x127.0.0.1:80 test.com
<html>
<head><title>401 Authorization Required</title></head>
<body bgcolor="white">
<center><h1>401 Authorization Required</h1></center>
<hr><center>nginx/1.8.0</center>
</body>
</html>
```


windows下测试的效果：

对目录进行用户认证：

```
location /admin/
{
    auth_basic  "Auth";
    anth_basic_user_file    /usr/local/nginx/conf/htpasswd;
}
```
@51CTO博客

3. 域名重定向

```
server
{
    listen 80;
    server_name test1.com test2.com;
    index index.htm index.html index.php;
    root /data/nginx/test1.com;
    # 域名重定向
    if ($host != 'test1.com'){
        rewrite ^/(.*)$ http://test1.com/$1 permanent;
    }
}
```

```
[root@centos7-1:vhost]# curl -x127.0.0.1:80 test2.com/123.txt -I
HTTP/1.1 301 Moved Permanently
Server: nginx/1.8.0
Date: Tue, 13 Mar 2018 13:32:34 GMT
Content-Type: text/html
Content-Length: 184
Connection: keep-alive
Location: http://test1.com/123.txt
```
@51CTO博客

4. Nginx的访问日志

Nginx日志格式定义在配置文件里：

```
[root@centos7-1:vhost]# grep -A2 log_format /usr/local/nginx/conf/nginx.conf
    log_format combined_realip '$remote_addr $http_x_forwarded_for [$time_local]'
    ' $host "$request_uri" $status'
    ' "$http_referer" "$http_user_agent"';
[root@centos7-1:vhost]#
```
@51CTO博客

vim test.com.conf，在server里面添加一行内容即可！

```
    # 访问日志
    access_log /tmp/nginx_default.log combined_realip;
}
```
@51CTO博客

```
[root@centos7-1:vhost]# curl -x127.0.0.1:80 test1.com/12
<html>
<head><title>404 Not Found</title></head>
<body bgcolor="white">
<center><h1>404 Not Found</h1></center>
<hr><center>nginx/1.8.0</center>
</body>
</html>
[root@centos7-1:vhost]# tail /tmp/nginx_default.log
127.0.0.1 - [13/Mar/2018:21:40:51 +0800] test2.com "/123.txt" 301 "-" "curl/7.29.0"
127.0.0.1 - [13/Mar/2018:21:40:59 +0800] test2.com "/12" 301 "-" "curl/7.29.0"
127.0.0.1 - [13/Mar/2018:21:41:04 +0800] test1.com "/12" 404 "-" "curl/7.29.0"
```
@51CTO博客

nginx日志切割需要编写脚本实现：

```
vim /usr/local/sbin/nginx_log_rorate.log.sh
#!/bin/bash
d=`date -d "-1 day" +%Y%m%d`
logdir="/data/logs"
nginx_pid="/usr/local/nginx/logs/nginx.pid"
cd $logdir
for log in `ls *.log`
do
    mv $log $log-$d
done
/bin/kill -HUP `cat $nginx_pid`
```

写完脚本，还需要增加任务计划：

0 0 * * * /bin/bash /usr/local/sbin/nginx_log_rorate.log.sh

5. 配置静态文件不记录日志并添加过期时间

vim test.com.conf

```
# 过滤静态文件并设置过期时间
location ~ .*\.(js|css)$ → 指定过滤的文件类型
{
    expires     12h;  → 过期时间
    access_log  off;
}
```
@51CTO博客

```
[root@centos7-1:vhost]# echo "dadnaksld" > /data/nginx/test1.com/test.js
[root@centos7-1:vhost]# echo "da21541ld" > /data/nginx/test1.com/test.css
[root@centos7-1:vhost]# curl -x127.0.0.1:80 -I test1.com/test.js
HTTP/1.1 200 OK
Server: nginx/1.8.0
Date: Tue, 13 Mar 2018 13:54:33 GMT
Content-Type: application/javascript
Content-Length: 10
Last-Modified: Tue, 13 Mar 2018 13:53:48 GMT
Connection: keep-alive
ETag: "5aa7d7ec-a"
Expires: Wed, 14 Mar 2018 01:54:33 GMT
Cache-Control: max-age=43200
Accept-Ranges: bytes
                                                    @51CTO博客
```

查看日志文件：（没有记录！）

```
[root@centos7-1:vhost]# tail /tmp/nginx_default.log
127.0.0.1 - [13/Mar/2018:21:40:51 +0800] test2.com "/123.txt" 301 "-" "curl/7.29.0"
127.0.0.1 - [13/Mar/2018:21:40:59 +0800] test2.com "/12" 301 "-" "curl/7.29.0"
127.0.0.1 - [13/Mar/2018:21:41:04 +0800] test1.com "/12" 404 "-" "curl/7.29.0"
[root@centos7-1:vhost]#                                    @51CTO博客
```

6. Nginx防盗链

vim test.com.conf

```
# 防盗链
location ~* ^.+\.(gif|jpg|png|swf|flv|zip|doc|pdf|gz|bz2|jpeg|bmp|xls)$
{
    expires     7d;
    valid_referers none blocked server_names *.test1.com;
    if ($invalid_referer){
        return 403;
    }
    access_log off;
}
                                                    @51CTO博客
```

测试：

```
[root@centos7-1:vhost]# touch /data/nginx/test1.com/2.png
[root@centos7-1:vhost]# curl -x127.0.0.1:80 -I -e "http://123.com/" test1.com/2.png
HTTP/1.1 403 Forbidden
Server: nginx/1.8.0
Date: Tue, 13 Mar 2018 14:02:29 GMT
Content-Type: text/html
Content-Length: 168
Connection: keep-alive

[root@centos7-1:vhost]# curl -x127.0.0.1:80 -I -e "http://test1.com/" test1.com/2.png
HTTP/1.1 200 OK
Server: nginx/1.8.0
Date: Tue, 13 Mar 2018 14:02:41 GMT
Content-Type: image/png
Content-Length: 0
Last-Modified: Tue, 13 Mar 2018 14:01:53 GMT
Connection: keep-alive
ETag: "5aa7d9d1-0"
Expires: Tue, 20 Mar 2018 14:02:41 GMT
Cache-Control: max-age=604800
Accept-Ranges: bytes
```


7. 访问控制（主允许192.168.159.131和127.0.0.1 访问等！）

```
# 访问控制
location /admin2/
{
    allow 192.168.159.131;
    allow 127.0.0.1;
    deny all;
}
# 对文件进行访问控制
location ~ .*(abc|image)/.*\.php$
{
    deny all;
}
# 针对user-agent限制
if ($http_user_agent ~ 'Spider/3.0|YoudaoBot|baidu')
{
    return 403;
}
```


8. Nginx解析php（经常出现502错误！查看fastcgi_passwd是否和php-fpm服务监听一致）

```
# 解析php
location ~ \.php$
{
    include fastcgi_params;
    fastcgi_pass unix:/tmp/php.fcgi.sock;
    fastcgi_index index.php;
    fastcgi_param SCRIPT_FILENAME /data/nginx/test1.com$fastcgi_script_name;
}
```


9. Nginx代理
   vim /usr/local/nginx/conf/vhost/proxy.conf

```
server
{
    listen 80;
    server_name test1.com;

    location /
    {
        proxy_pass  http://121.201.9.155/;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP  $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```


或者这样：

```
upstream qq_com
{
    ip_hash;
    server 61.135.157.156:80;
    server 125.39.240.113:80;
}
server
{
    listen 80;
    server_name www.qq.com;

    location /
    {
        proxy_pass      http://qq_com;
        proxy_set_header Host   $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```


10. Nginx配置SSL

SSL工作流程:

浏览器发送一个https的请求给服务器;

服务器要有一套数字证书，可以自己制作（后面的操作就是阿铭自己制作的证书），也可以向组织申请，区别就是自己颁发的证书需要客户端验证通过，才可以继续访问，而使用受信任的公司申请的证书则不会弹出>提示页面，这套证书其实就是一对公钥和私钥;

服务器会把公钥传输给客户端;

客户端（浏览器）收到公钥后，会验证其是否合法有效，无效会有警告提醒，有效则会生成一串随机数，并用收到的公钥加密;

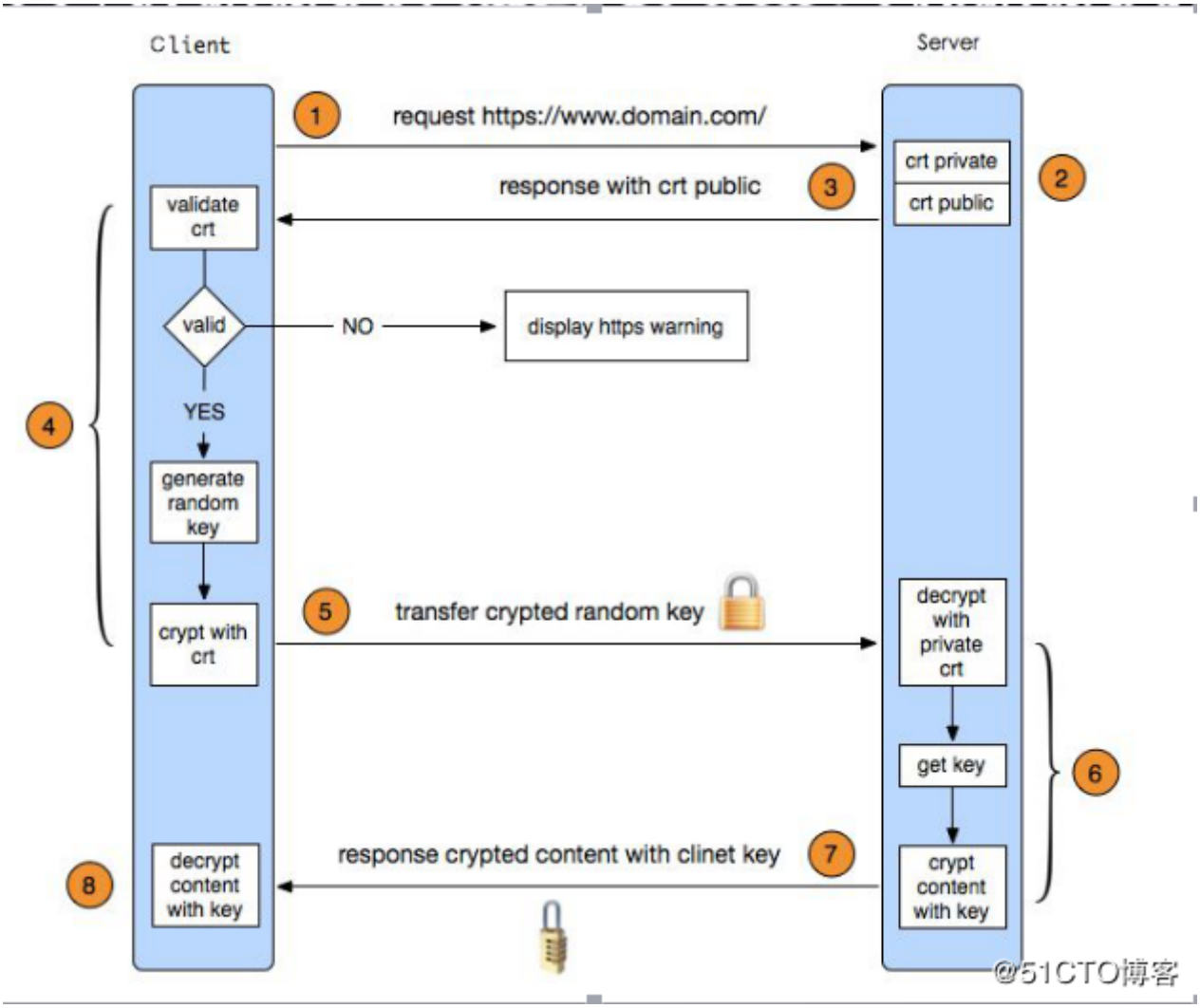
客户端把加密后的随机字符串传输给服务器;

服务器收到加密随机字符串后，先用私钥解密（公钥加密，私钥解密），获取到这一串随机数后，再用这串随机字符串加密传输的数据（该加密为对称加密，所谓对称加密，就是将数据和私钥也就是这个随机字符串>通过某种算法混合在一起，这样除非知道私钥，否则无法获取数据内容）;

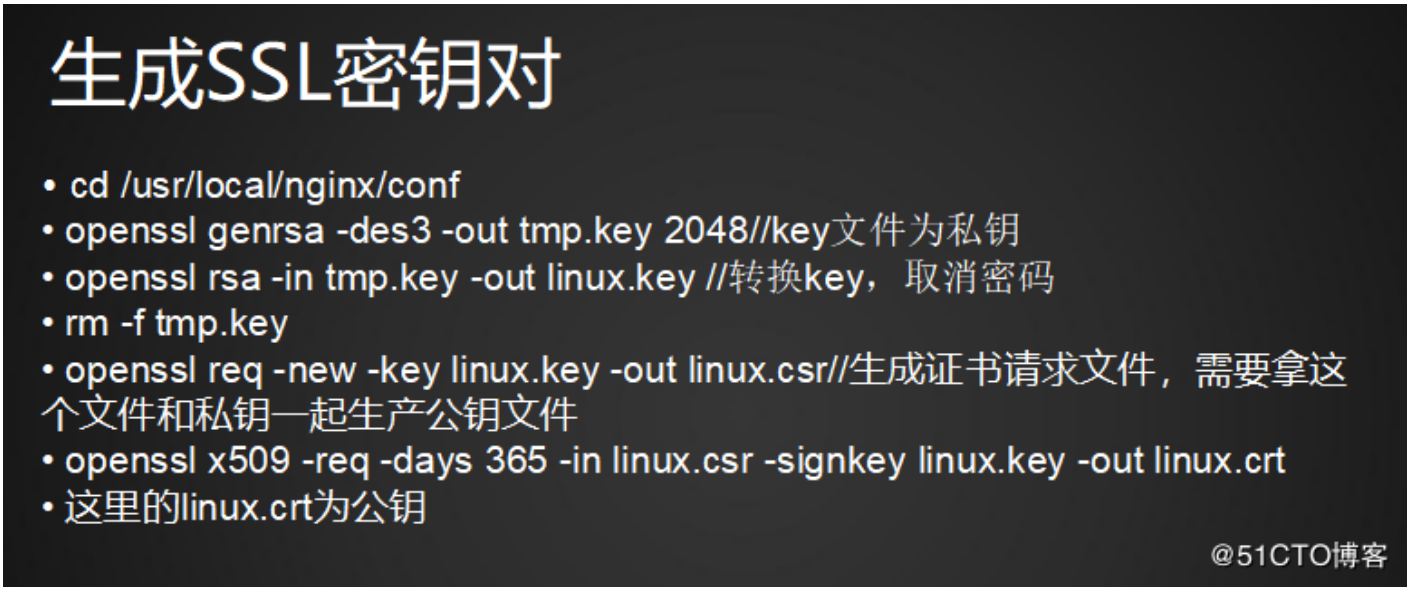
服务器把加密后的数据传输给客户端;

客户端收到数据后，再用自己的私钥也就是那个随机字符串解密;

HTTPS通信过程:


Client
Server
1
request https://www.domain.com/
crt private
crt public
2
response with crt public
3
validate crt
valid
NO
display https warning
YES
4
generate random key
crypt with crt
5
transfer crypted random key
decrypt with private crt
get key
6
response crypted content with clinet key
7
decrypt content with key
8
crypt content with key
@51CTO博客

生成SSL密钥对：


生成SSL密钥对
• cd /usr/local/nginx/conf
• openssl genrsa -des3 -out tmp.key 2048//key文件为私钥
• openssl rsa -in tmp.key -out linux.key //转换key，取消密码
• rm -f tmp.key
• openssl req -new -key linux.key -out linux.csr//生成证书请求文件，需要拿这个文件和私钥一起生产公钥文件
• openssl x509 -req -days 365 -in linux.csr -signkey linux.key -out linux.crt
• 这里的linux.crt为公钥
@51CTO博客

Nginx配置SSL

- vim /usr/local/nginx/conf/vhost/ssl.conf//加入如下内容

```
server
{
  listen 443;
  server_name test1.com;
  index index.html index.php;
  root /data/nginx/test1.com;
  ssl on;
  ssl_certificate linux.crt;
  ssl_certificate_key linux.key;
  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
}
```

- -t && -s reload //若报错unknown directive “ssl”， 需要重新编译nginx，加上--with-http_ssl_module
- mkdir /data/nginx/test1.com
- echo “ssl test page.” >/data/nginx/test1.com/index.html
- 编辑hosts，增加127.0.0.1 test1.com
- curl https://test1.com/

@51CTO博客

php-fpm配置：

1. php-fpm的pool
   vim /usr/local/php-fpm/etc/php-fpm.conf

```
[global]
pid = /usr/local/php-fpm/var/run/php-fpm.pid
error_log = /usr/local/php-fpm/var/log/php-fpm.log
include = etc/php-fpm.d/*.conf
[www]
```

@51CTO博客

```
[root@centos7-1:~]# mkdir /usr/local/php-fpm/etc/php-fpm.d
[root@centos7-1:~]# cd /usr/local/php-fpm/etc/php-fpm.d
[root@centos7-1:php-fpm.d]# vim www.conf
[root@centos7-1:php-fpm.d]# cat !$
cat www.conf
[www]
listen = /tmp/www.sock
listen.mode=666
user = php-fpm
group = php-fpm
pm = dynamic
pm.max_children = 50
pm.start_servers = 20
pm.min_spare_servers = 5
pm.max_spare_servers = 35
pm.max_requests = 500
rlimit_files = 1024
```

@51CTO博客

```
[root@centos7-1:php-fpm.d]# vim wap.conf
[root@centos7-1:php-fpm.d]# cat !$
cat wap.conf
[wap]
listen = /tmp/wap.sock
listen.mode=666
user = php-fpm
group = php-fpm
pm = dynamic
pm.max_children = 50
pm.start_servers = 20
pm.min_spare_servers = 5
pm.max_spare_servers = 35
pm.max_requests = 500
rlimit_files = 1024
[root@centos7-1:php-fpm.d]#
```

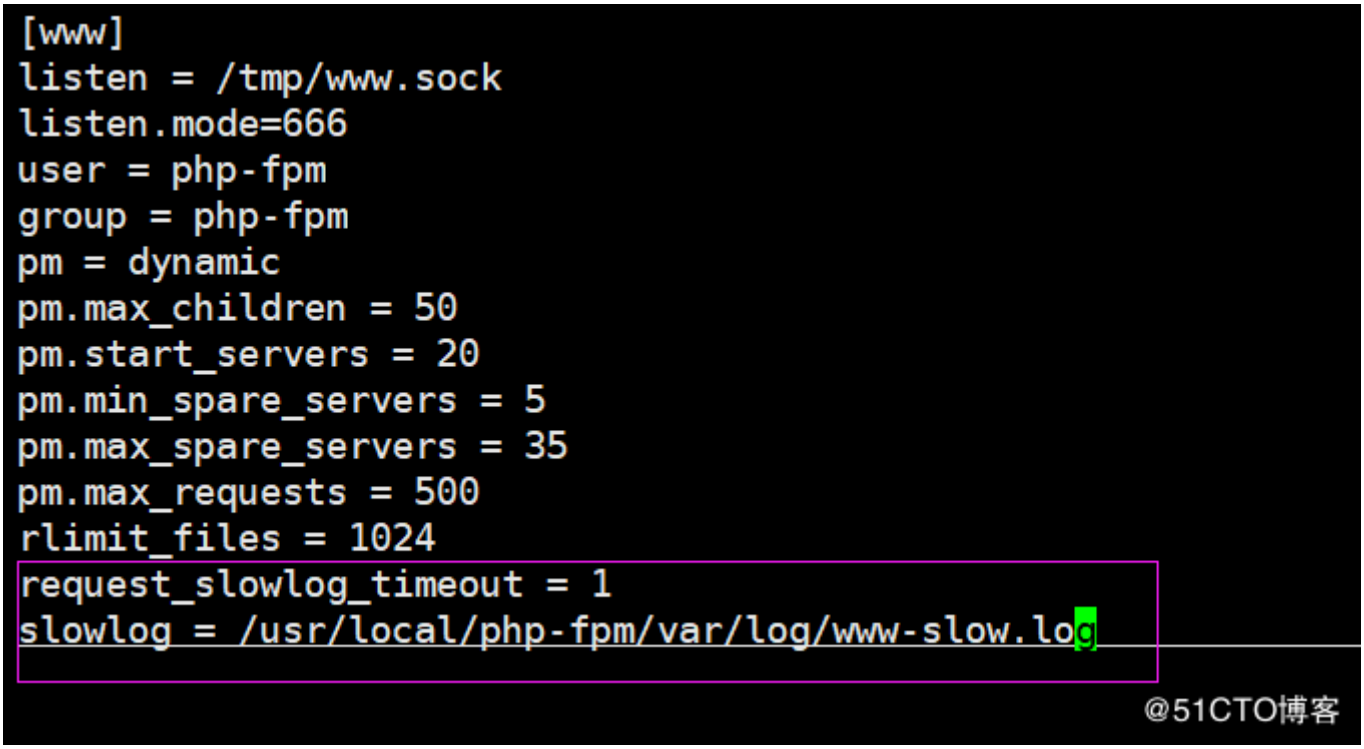


```
[root@centos7-1:php-fpm.d]# /usr/local/php-fpm/sbin/php-fpm -t
[13-Mar-2018 23:54:25] NOTICE: configuration file /usr/local/php-fpm/etc/php-fpm.conf test is succe
ssful
```



```
[root@centos7-1:php-fpm.d]# service php-fpm restart
Gracefully shutting down php-fpm . done
Starting php-fpm  done
[root@centos7-1:php-fpm.d]# ls /tmp/*.sock
/tmp/mysql.sock  /tmp/php-fcgi.sock  /tmp/wap.sock
[root@centos7-1:php-fpm.d]#
```


2. php-fpm的慢执行日志
   vim /usr/local/php-fpm/etc/php-fpm.d/www.conf



3. php-fpm定义open_basedir
   php_admin_value[open_basedir]=/data/nginx/wap.com:/tmp/
4. php-fpm进程管理
   pm = dynamic //动态进程管理，也可以是static
   pm.max_children = 50 //最大子进程数，ps aux可以查看
   pm.start_servers = 20 //启动服务时会启动的进程数
   pm.min_spare_servers = 5 //定义在空闲时段，子进程数的最少数量，如果达到这个数值时，php-fpm服务会自动派生新的子进程。
   pm.max_spare_servers = 35 //定义在空闲时段，子进程数的最大值，如果高于这个数值就开始清理空闲的子进程。
   pm.max_requests = 500 //定义一个子进程最多处理的请求数，也就是说在一个php-fpm的子进程最多可以处理这么多请求，当达到这个数值时，它会自动退出。