### **PHP7中php.ini、php-fpm和www.conf 配置**

php-fpm.conf是PHP-FPM特有的配置文件

php.ini是所以php模式中必须的配置文件

两者的区别是，php-fpm.conf是PHP-FPM进程管理器的配置文件，php.ini是PHP解析器的配置文件

#### **php.ini是php运行核心配置文件,下面是一些常用配置**

extension\_dir=""  
● 设置PHP的扩展库路径  
  
log\_errors = On  
● 在关闭display\_errors后开启PHP错误日志（路径在php-fpm.conf中配置）  
  
zend\_extension=opcache.so extension=mysqli.so extension=pdo\_mysql.so  
● 设置PHP的opcache和mysql动态库  
  
date.timezone = PRC  
● 设置PHP的时区  
  
opcache.enable=1  
● 开启opcache  
  
open\_basedir = /usr/share/nginx/html;  
● 设置PHP脚本允许访问的目录（需要根据实际情况配置）

#### **php-fpm.conf是php-fpm进程服务的配置文件,下面是一些常用配置**

error\_log = /usr/local/php/logs/php-fpm.log  
● 设置错误日志的路径  
  
include=/usr/local/php7/etc/php-fpm.d/\*.conf  
● 引入www.conf文件中的配置（默认已设置）

#### **php-fpm.conf 以及 www.conf的主要配置信息**

pid = run/php-fpm.pid  
● pid设置，默认在安装目录中的var/run/php-fpm.pid，建议开启  
  
error\_log = log/php-fpm.log  
● 错误日志，默认在安装目录中的var/log/php-fpm.log  
  
log\_level = notice  
● 错误级别. 可用级别为: alert（必须立即处理）, error（错误情况）, warning（警告情况）, notice（一般重要信息）, debug（调试信息）. 默认: notice.  
  
emergency\_restart\_threshold = 60  
emergency\_restart\_interval = 60s  
● 表示在emergency\_restart\_interval所设值内出现SIGSEGV或者SIGBUS错误的php-cgi进程数如果超过 emergency\_restart\_threshold个，php-fpm就会优雅重启。这两个选项一般保持默认值。  
  
process\_control\_timeout = 0  
● 设置子进程接受主进程复用信号的超时时间. 可用单位: s(秒), m(分), h(小时), 或者 d(天) 默认单位: s(秒). 默认值: 0.  
  
daemonize = yes  
● 后台执行fpm,默认值为yes，如果为了调试可以改为no。在FPM中，可以使用不同的设置来运行多个进程池。 这些设置可以针对每个进程池单独设置。  
  
listen = 127.0.0.1:9000  
● 监听端口，即nginx中php处理的地址，一般默认值即可。可用格式为: 'ip:port', 'port', '/path/to/unix/socket'. 每个进程池都需要设置.  
  
listen.backlog = -1  
● backlog数，-1表示无限制，由操作系统决定，此行注释掉就行。  
  
listen.allowed\_clients = 127.0.0.1  
● 允许访问FastCGI进程的IP，设置any为不限制IP，如果要设置其他主机的nginx也能访问这台FPM进程，listen处要设置成本地可被访问的IP。默认值是any。每个地址是用逗号分隔. 如果没有设置或者为空，则允许任何服务器请求连接  
  
listen.owner = www listen.group = www listen.mode = 0666  
● unix socket设置选项，如果使用tcp方式访问，这里注释即可。  
  
user = www group = www  
● 启动进程的帐户和组

#### **php-fpm 进程池优化方法**

pm = dynamic  
● 对于专用服务器，pm可以设置为static。  
  
如何控制子进程，选项有static和dynamic。如果选择static，则由pm.max\_children指定固定的子进程数。如果选择dynamic，则由下开参数决定：  
  
pm.max\_children  
● 静态方式下开启的php-fpm进程数量，在动态方式下他限定php-fpm的最大进程数（这里要注意pm.max\_spare\_servers的值只能小于等于pm.max\_children）  
  
pm.start\_servers  
● 动态方式下的起始php-fpm进程数量。  
  
pm.min\_spare\_servers  
● 保证空闲进程数最小值，如果空闲进程小于此值，则创建新的子进程  
  
pm.max\_spare\_servers  
● 保证空闲进程数最大值，如果空闲进程大于此值，此进行清理  
  
● 如果dm设置为static，那么其实只有pm.max\_children这个参数生效。系统会开启参数设置数量的php-fpm进程。php-fpm一个进程大概会占20m-40m的内存，所以他的数字大小的设置要根据你的物理内存的大小来设置，还要注意到其他的内存占用，如数据库，系统进程等，来确定以上4个参数的设定值！  
  
● 如果dm设置为dynamic，4个参数都生效。系统会在php-fpm运行开始时启动pm.start\_servers个php-fpm进程，然后根据系统的需求动态在pm.min\_spare\_servers和pm.max\_spare\_servers之间调整php-fpm进程数。参数要求pm.start\_servers的值在pm.min\_spare\_servers和pm.max\_spare\_servers之间。  
  
pm.max\_requests = 1000  
● 设置每个子进程重生之前服务的请求数.  
  
● 最大处理请求数是指一个php-fpm的worker进程在处理多少个请求后就终止掉，master进程会重新respawn一个新的。  
  
● 这个配置的主要目的是避免php解释器或程序引用的第三方库造成的内存泄露。  
  
对于可能存在内存泄漏的第三方模块来说是非常有用的. 如果设置为 '0' 则一直接受请求. 等同于 PHP\_FCGI\_MAX\_REQUESTS 环境变量. 默认值: 0.  
  
pm.status\_path = /status  
● FPM状态页面的网址. 如果没有设置, 则无法访问状态页面. 默认值: none. munin监控会使用到  
  
ping.path = /ping  
● FPM监控页面的ping网址. 如果没有设置, 则无法访问ping页面. 该页面用于外部检测FPM是否存活并且可以响应请求. 请注意必须以斜线开头 (/)。  
  
ping.response = pong  
● 用于定义ping请求的返回相应. 返回为 HTTP 200 的 text/plain 格式文本. 默认值: pong.  
  
request\_terminate\_timeout = 0  
● 设置单个请求的超时中止时间. 该选项可能会对php.ini设置中的'max\_execution\_time'因为某些特殊原因没有中止运行的脚本有用. 设置为 '0' 表示 'Off'.当经常出现502错误时可以尝试更改此选项。  
  
request\_slowlog\_timeout = 10s  
● 当一个请求该设置的超时时间后，就会将对应的PHP调用堆栈信息完整写入到慢日志中. 设置为 '0' 表示 'Off'  
  
slowlog = log/$pool.log.slow  
● 慢请求的记录日志,配合request\_slowlog\_timeout使用  
  
rlimit\_files = 1024  
● 设置文件打开描述符的rlimit限制. 默认值: 系统定义值默认可打开句柄是1024，可使用 ulimit -n查看，ulimit -n 2048修改。  
  
rlimit\_core = 0  
● 设置核心rlimit最大限制值. 可用值: 'unlimited' 、0或者正整数. 默认值: 系统定义值.  
  
chroot =  
● 启动时的Chroot目录. 所定义的目录需要是绝对路径. 如果没有设置, 则chroot不被使用.  
  
chdir =  
● 设置启动目录，启动时会自动Chdir到该目录. 所定义的目录需要是绝对路径. 默认值: 当前目录，或者/目录（chroot时）  
  
catch\_workers\_output = yes  
● 重定向运行过程中的stdout和stderr到主要的错误日志文件中. 如果没有设置, stdout 和 stderr 将会根据FastCGI的规则被重定向到 /dev/null . 默认值: 空.`  
  
clear\_env = no            
● 清理环境  
  
variables\_order  
variables\_order参数详解在另一篇文章https://www.php.cn/php-weizijiaocheng-429277.html

#### **常见错误及解决办法整理**

请求的超时中止时间未设置  
  
● request\_terminate\_timeout的值如果设置为0或者过长的时间，可能会引起PHP 脚本会一直执行下去。这样，当所有的 php-cgi 进程都卡在 file\_get\_contents() 函数时，这台 Nginx+PHP 的 WebServer 已经无法再处理新的 PHP 请求了，Nginx 将给用户返回“502 Bad Gateway”。设置一个  
  
PHP脚本最大执行时间是必要的，但是，治标不治本。例如改成 30s，如果发生 file\_get\_contents() 获取网页内容较慢的情况，这就意味着 150 个 php-cgi 进程，每秒钟只能处理 5 个请求，WebServer 同样很难避免"502 Bad Gateway"。解决办法是request\_terminate\_timeout设置为10s或者一个合理的值，或者给file\_get\_contents加一个超时参数！  
  
max\_requests参数配置不当  
● max\_requests参数配置不当，可能会引起间歇性502错误：  
  
pm.max\_requests = 1000  
● 设置每个子进程重生之前服务的请求数. 对于可能存在内存泄漏的第三方模块来说是非常有用的. 如果设置为 '0' 则一直接受请求. 等同于 PHP\_FCGI\_MAX\_REQUESTS 环境变量. 默认值: 0.  
  
这段配置的意思是，当一个 PHP-CGI 进程处理的请求数累积到 500 个后，自动重启该进程。

#### 但是为什么要重启进程呢？

● 一般在项目中，我们多多少少都会用到一些 PHP 的第三方库，这些第三方库经常存在内存泄漏问题，如果不定期重启 PHP-CGI 进程，势必造成内存使用量不断增长。因此 PHP-FPM 作为 PHP-CGI 的管理器，提供了这么一项监控功能，对请求达到指定次数的 PHP-CGI 进程进行重启，保证内存使用量不增长。

#### php-fpm的慢日志，debug及异常排查神器 ● request\_slowlog\_timeout设置一个超时的参数，slowlog设置慢日志的存放位置，tail -f /var/log/www.slow.log即可看到执行过慢的php过程。 大家可以看到经常出现的网络读取超过、Mysql查询过慢的问题，根据提示信息再排查问题就有很明确的方向了。

我们可以使用php-fpm.conf配置慢日志

我们是可以开启慢日志功能的。

1. slowlog = /usr/local/var/log/php-fpm.log.slow
2. request\_slowlog\_timeout = 5s

当某个请求的时间超过了5秒，就会在慢日志中记录相应的记录，注意上面的时间5s，不能忽略了单位，相应的还有其他单位，m分，h时(当然，这这样做太扯淡了)

php-fpm慢日志会记录下进程号，脚本名称，具体哪个文件哪行代码的哪个函数执行时间过长：

1. [21-Nov-2016 10:30:38] [pool www] pid 11877
2. script\_filename = /var/www/ceshi/c.php
3. [0xb70fb88c] sleep() /var/www/ceshi/c.php:2

通过日志，我们就可以知道第2行的sleep 函数有点问题，这样我们就能追踪问题了。