作者 wujunze 2015.11.02 23:16:00
写了41篇文章,回复382人,

如何正确配置Nginx+PHP

Nginx ("engine x") 是一个高性能的 HTTP 和 反向代理 服务器，也是一个 IMAP/POP3/SMTP 代理服务器。

对很多人而言，配置Nginx+PHP无外乎就是搜索一篇教程，然后拷贝粘贴。听上去似乎也没什么问题，可惜实际上网络上很多资料本身年久失修，漏洞百出，如果大家不求甚解，一味的拷贝粘贴，早晚有一天会为此付出代价。

假设我们用PHP实现了一个前端控制器，或者直白点说就是统一入口：把PHP请求都发送到同一个文件上，然后在此文件里通过解析「REQUEST_URI」实现路由。

此时很多教程会教大家这样配置Nginx+PHP:

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

    root /path;

    location / {
        index index.html index.htm index.php;

        if (!-e $request_filename) {
            rewrite . /index.php last;
        }
    }

    location ~ \.php$ {
        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME /path$fastcgi_script_name;
        fastcgi_pass 127.0.0.1:9000;
        fastcgi_index index.php;
    }
}
```

这里面有很多错误，或者说至少是坏味道的地方，大家看看能发现几个。

…

我们有必要先了解一下Nginx配置文件里指令的继承关系：Nginx配置文件分为好多块，常见的从外到内依次是「http」、「server」、「location」等等，缺省的继承关系是从外到内，也就是说内层块会自动获取外层块的值作为缺省值（有例外，详见参考）。

参考：UNDERSTANDING THE NGINX CONFIGURATION INHERITANCE M

让我们先从「index」指令入手吧，在问题配置中它是在「location」中定义的：

```
location / {
    index index.html index.htm index.php;
}
```

一旦未来需要加入新的「location」，必然会出现重复定义的「index」指令，这是因为多个「location」是平级的关系，不存在继承，此时应该在「server」里定义「index」，借助继承关系，「index」指令在所有的「location」中都能生效。

参考：Nginx Pitfalls

…

接下来看看「if」指令，说它是大家误解最深的Nginx指令毫不为过：

```
if (!-e $request_filename) {
    rewrite . /index.php last;
}
```

很多人喜欢用「if」指令做一系列的检查，不过这实际上是「try_files」指令的职责：

try_files $uri $uri/ /index.php;

除此以外，初学者往往会认为「if」指令是内核级的指令，但是实际上它是rewrite模块的一部分，加上Nginx配置实际上是声明式的，而非过程式的，所以当其和非rewrite模块的指令混用时，结果可能会非你所愿。

参考：IfIsEvil and How nginx “location if” works

…

下面看看「fastcgi_params」配置文件：

include fastcgi_params;

Nginx有两份fastcgi配置文件，分别是「fastcgi_params」和「fastcgi.conf」，它们没有太大的差异，唯一的区别是后者比前者多了一行「SCRIPT_FILENAME」的定义：

```
fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
```

注意：$document_root 和 $fastcgi_script_name 之间没有 /。

原本Nginx只有「fastcgi_params」，后来发现很多人在定义「SCRIPT_FILENAME」时使用了硬编码的方式，于是为了规范用法便引入了「fastcgi.conf」。

不过这样的话就产生一个疑问：为什么一定要引入一个新的配置文件，而不是修改旧的配置文件？这是因为「fastcgi_param」指令是数组型的，和普通指令相同的是：内层替换外层；和普通指令不同的是：当在同级多次使用的时候，是新增而不是替换。换句话说，如果在同级定义两次「SCRIPT_FILENAME」，那么它们都会被发送到后端，这可能会导致一些潜在的问题，为了避免此类情况，便引入了一个新的配置文件。

参考：FASTCGI_PARAMS VERSUS FASTCGI.CONF - NGINX CONFIG HISTORY

…

files」指令做一次过滤：

try_files $uri =404;

参考：Nginx文件类型错误解析漏洞

…

依照前面的分析，给出一份改良后的版本，是不是比开始的版本清爽了很多：

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

    root /path;
    index index.html index.htm index.php;

    location / {
        try_files $uri $uri/ /index.php;
    }

    location ~ \.php$ {
        try_files $uri =404;

        include fastcgi.conf;
        fastcgi_pass 127.0.0.1:9000;
    }
}
```

实际上还有一些瑕疵，主要是「try_files」和「fastcgi_split_path_info」不够兼容，虽然能够解决，但方案比较丑陋，具体就不多说了，有兴趣的可以参考问题描述。

补充：因为「location」已经做了限定，所以「fastcgi_index」其实也没有必要。

标签： nginx 服务器