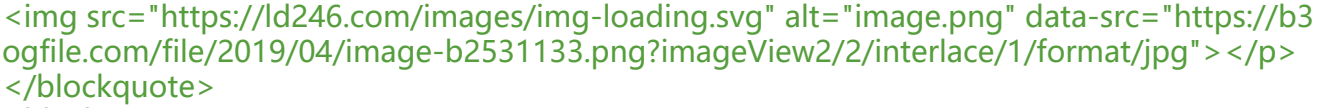
Nginx 简述 _ 路径描述 _location 配置查找顺序

作者：guichun68
原文链接：https://ld246.com/article/1554563994529
来源网站：链滴

<h2 id="Nginx-简述">Nginx 简述</h2>
Nginx 是一个开源的且高性能、可靠的 HTTP 中间件、代理服务。
<h2 id="选择Nginx原因">选择 Nginx 原因</h2>
<h3 id="IO多路复用epoll">IO 多路复用 epoll</h3>
多个描述符的 I/O 操作都能在一个线程内并发交替地顺序完成，这就叫 I/O 多路复用，这里的复用”指的是复用同一个线程。
<blockquote>
什么是 epoll
IO 多路复用的实现方式 select、poll、epoll
</blockquote>

select 缺点
能够监视文件描述符的数量存在最大限制
线性扫描效率低下
epoll 模型
1）每当 FD(文件描述符)就绪，采用系统的回调函数之间将 FD 放入，效率更高。
2）最大连接数无限制

<h3 id="轻量级">轻量级</h3>
功能模块少
代码模块化
<h3 id="CPU亲和-affinity-">CPU 亲和（affinity）</h3>
什么是 CPU 亲和
<blockquote>
是一种把 CPU 核心和 Nginx 工作进程绑定方式，把每个 worker 进程固定在一个 cpu 上执行减少切换 cpu 的 cache miss，获得更好的性能。


</blockquote>
为什么需要 CPU 亲和
<blockquote>
Nginx 正是利用了 cpu 的亲和来提升并发处理能力和减少不必要的额外性能损耗。当今服务器般都是多 cpu 多核心的，nginx 作为接入层的中间件，他对于 cpu 的亲和就尤为重要，nginx 是由个不同的 worker 的进程进行处理，假设有一台双 cpu 的服务器，每个 cpu4 个核心，把 8 个 worker 进程分别绑定到不同 cpu 核心上，这样均匀的分配，就能降低因 cpu 自动切换时的性能损失，
</blockquote>
###sendfile 零拷贝
<blockquote>
nginx 在处理静态文件方面非常有优势，因为其采用 senfile 运作方式
对比原来的 HttpServer 方式，当我们请求一个静态文件时，它会经过内核空间和用户空间，最终到达 socket，通过 socket 响应给用户。对于一台服务器操作系统会发生多次切换(内核空间到用户空间之)。


</blockquote>
<blockquote>
因为静态文件不需要过多的用户空间的逻辑处理，可以直接传输。 Linux2.1 引入 sendfile 技术 sendfile 正是运用这种零拷贝传输模式，即文件的传输只通过内核空间传递给 socket 响应给用户。以在 cdn 服务或动静分离的静态文件处理方面 nginx 会比其他服务性能高很多。


</blockquote>
##安装目录简介
对于 yum 这种基于 RPM 包管理的软件包管理器安装的软件，可以使用 rpm -ql nginx</st

ong> 来查看指定软件(如 nginx)的所有文件目录，如下：
<img src="https://ld246.com/images/img-loading.svg" alt="image.png" data-src="https://b3ogfile.com/file/2019/04/image-ab4c8755.png?imageView2/2/interlace/1/format/jpg">
主要目录介绍：
<table>
<thead>
<tr>
<th align="left">路径</th>
<th align="right">类型</th>
<th align="center">作用</th>
</tr>
</thead>
<tbody>
<tr>
<td align="left">/etc/logrotate.d/nginx</td>
<td align="right">配置文件</td>
<td align="center">Nginx 日志轮转，用于 logrotate 服务的日志切割</td>
</tr>
<tr>
<td align="left">/etc/nginx
/etc/nginx/nginx.conf
/etc/nginx/conf.d
/etc/nginx/onf.d/default.conf</td>
<td align="right">目录、配置文件</td>
<td align="center">Nginx 主配置文件</td>
</tr>
<tr>
<td align="left">/etc/nginx/fastcgi_params
/etc/nginx/uwsgi_params
/etc/nginx/scg_params</td>
<td align="right">配置文件</td>
<td align="center">cgi 相关，fastcgi 配置</td>
</tr>
<tr>
<td align="left">/etc/nginx/koi-utf
/etc/nginx/koi-win
/etc/nginx/win-utf</td>
<td align="right">配置文件</td>
<td align="center">编码转换映射转化文件</td>
</tr>
<tr>
<td align="left">/etc/nginx/mime.types</td>
<td align="right">配置文件</td>
<td align="center">设置 http 协议的 Content-Type 与扩展名对应关系</td>
</tr>
<tr>
<td align="left">/usr/lib/systemd/system/nginx-debug.service
/usr/lib/system/nginx.service
/etc/sysconfig/nginx
/etc/sysconfig/nginx-debug</td>
<td align="right">配置文件</td>
<td align="center">用于配置出系统守护进程管理器管理方式</td>
</tr>
<tr>
<td align="left">/usr/lib64/nginx/modules
/etc/nginx/modules</td>
<td align="right">目录</td>
<td align="center">Nginx 模块目录</td>
</tr>
<tr>
<td align="left">/usr/sbin/nginx
/usr/sbin/nginx-debug</td>
<td align="right">命令</td>

<td align="center">Nginx 服务的启动管理的终端命令</td>
</tr>
<tr>
<td align="left">/usr/share/doc/nginx-1.12.0
/usr/share/doc/nginx-1.12.0/COPYRIGHT<
r>/usr/share/man/man8/nginx.8.gz</td>
<td align="right">文件、目录</td>
<td align="center">Nginx 的手册和帮助文件</td>
</tr>
<tr>
<td align="left">/var/cache/nginx</td>
<td align="right">目录</td>
<td align="center">Nginx 的缓存目录</td>
</tr>
<tr>
<td align="left">/var/log/nginx</td>
<td align="right">目录</td>
<td align="center">Nginx 的日志目录</td>
</tr>
</tbody>
</table>
<h2 id="location查找顺序">location 查找顺序</h2>
<img src="https://ld246.com/images/img-loading.svg" alt="image.png" data-src="https:
/b3logfile.com/file/2019/04/image-ba54b781.png?imageView2/2/interlace/1/format/jpg"><
p>
<blockquote>
<code>总结：</code>
location 的命中过程是这样的
1：先判断精准命中，如果命中，则立即返回结果并结束解析过程。
2： 判断普通命中，如果有多个命中，“记录”下来“最长”的命中结果。
（注意：记录但不结束，最长的为准）
3：继续判断正则表达式的解析结果，按配置里的正则表达式顺序为准，由上到下开始匹配，一旦匹
成功 1 个，则立即返回结果，并结束解析过程。
<code>延伸分析</code>：
a:普通命中 顺序无所谓，是因为按命中的长短来确定的。
b: 正则命中，顺序有所谓，因为是从前往后命中的。
</blockquote>