



链滴

nginx 详解—日志配置

作者: [rebirth12](#)

原文链接: <https://ld246.com/article/1584004530725>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)



nginx详解—日志配置

一、nginx access日志配置

1.1 access_log日志配置

access_log用来定义日志级别，日志位置。语法如下：

日志级别： `debug > info > notice > warn > error > crit > alert > emerg`

语法格式: `access_log path [format [buffer=size] [gzip[=level]] [flush=time] [if=condition]];`

`access_log off;`

默认值： `access_log logs/access.log combined;`

作用域： `http, server, location, if in location, limit_except`

实例一：

```
access_log /var/logs/nginx-access.log compression buffer=32k;
```

1.2 log_format 定义日志格式

语法格式: `log_format name [escape=default|json] string ...;`

默认值： `log_format combined "...";`

作用域： `http`

实例一：

```
log_format compression '$remote_addr - $remote_user [$time_local] '
    '$request' $status $bytes_sent '
    '$http_referer' '$http_user_agent' '$gzip_ratio';
access_log /var/logs/nginx-access.log compression buffer=32k;
```

1.3 open_log_file_cache

使用open_log_file_cache来设置日志文件缓存(默认是off)。

- max:设置缓存中的最大文件描述符数量, 如果缓存被占满, 采用LRU算法将描述符关闭。
- inactive:设置存活时间, 默认是10s
- min_uses:设置在inactive时间段内, 日志文件最少使用多少次后, 该日志文件描述符记入缓存中, 默认是1次
- valid:设置检查频率, 默认60s
- off: 禁用缓存

语法格式: open_log_file_cache max=N [inactive=time] [min_uses=N] [valid=time];

open_log_file_cache off;

默认值: open_log_file_cache off;

作用域: http, server, location

实例一

```
open_log_file_cache max=1000 inactive=20s valid=1m min_uses=2;
```

1.4 日志中常用的全局变量

- `$remote_addr`, `$http_x_forwarded_for` 记录客户端IP地址
- `$remote_user`记录客户端用户名称
- `$request`记录请求的URL和HTTP协议(GET,POST,DEL,等)
- `$status`记录请求状态
- `$body_bytes_sent`发送给客户端的字节数, 不包括响应头的大小; 该变量与Apache模块mod_log_config里的“%B”参数兼容。
- `$bytes_sent`发送给客户端的总字节数。
- `$connection`连接的序列号。
- `$connection_requests` 当前通过一个连接获得的请求数量。
- `$msec` 日志写入时间。单位为秒, 精度是毫秒。
- `$pipe`如果请求是通过HTTP流水线(pipelined)发送, pipe值为“p”, 否则为“.”。
- `$http_referer` 记录从哪个页面链接访问过来的
- `$http_user_agent`记录客户端浏览器相关信息
- `$request_length`请求的长度(包括请求行, 请求头和请求正文)。
- `$request_time` 请求处理时间, 单位为秒, 精度毫秒; 从读入客户端的第一个字节开始, 直到最后一个字符发送给客户端后进行日志写入为止。

- `$time_iso8601` ISO8601标准格式下的本地时间。
- `$time_local`通用日志格式下的本地时间。

二、nginx日志调试技巧

2.1 仅记录固定 IP 的错误

当你设置日志级别成 debug, 如果你在调试一个在线的高流量网站的话, 你的错误日志可能会记录个请求的很多消息, 这样会变得毫无意义。

在 `events{...}`中配置如下内容, 可以使 Nginx 记录仅仅来自于你的 IP 的错误日志。

```
events {
    debug_connection 1.2.3.4;
}
```

2.2 调试 nginx rewrite 规则

调试rewrite规则时, 如果规则写错只会看见一个404页面, 可以在配置文件中开启nginx rewrite日志进行调试。

```
server {
    error_log /var/logs/nginx/example.com.error.log;
    rewrite_log on;
}
```

`rewrite_log on;` 开启后, 它将发送所有的 rewrite 相关的日志信息到 `error_log` 文件中, 使用 [notice 级别。随后就可以在`error_log` 查看rewrite信息了。

2.3 使用location记录指定URL的日志

```
server {
    error_log /var/logs/nginx/example.com.error.log;
    location /static/ {
        error_log /var/logs/nginx/static-error.log debug;
    }
}
```

配置以上配置后, `/static/` 相关的日志会被单独记录在`static-error.log`文件中。

nginx日志共三个参数

1. `access_log`: 定义日志的路径及格式。
2. `log_format`: 定义日志的模板。
3. `open_log_file_cache`: 定义日志文件缓存。

`proxy_set_header X-Forwarded-For` : 如果后端Web服务器上的程序需要获取用户IP, 从该Header头获取。 `proxy_set_header X-Forwarded-For $remote_addr;`

三、常用例子

3.1 main格式

```
log_format main '$remote_addr - $remote_user [$time_local] "$request" '
                '$status $body_bytes_sent "$http_referer" '
                '"$http_user_agent" "$http_x_forwarded_for" '
                '$upstream_addr $upstream_response_time $request_time ';
access_log logs/access.log main;
```

3.2 json格式

```
log_format logstash_json '{"@timestamp": "$time_iso8601",'
                          '"host": "$server_addr",'
                          '"client": "$remote_addr",'
                          '"size": $body_bytes_sent,'
                          '"responsetime": $request_time,'
                          '"domain": "$host",'
                          '"url": "$request_uri",'
                          '"referer": "$http_referer",'
                          '"agent": "$http_user_agent",'
                          '"status": "$status",'
                          '"x_forwarded_for": "$http_x_forwarded_for"}';
```

坑点:

使用 `$uri` 可以在nginx对URL进行更改或重写，但是用于日志输出可以使用 `$request_uri`代替，如无殊业务需求，完全可以替换。

3.3 压缩格式

日志中增加了压缩的信息。

```
http {
    log_format compression '$remote_addr - $remote_user [$time_local] '
                           '"$request" $status $body_bytes_sent '
                           '"$http_referer" "$http_user_agent" "$gzip_ratio";

    server {
        gzip on;
        access_log /spool/logs/nginx-access.log compression;
        ...
    }
}
```

3.4 upstream格式

增加upstream消耗的时间。

```
http {
    log_format upstream_time '$remote_addr - $remote_user [$time_local] '
                             '"$request" $status $body_bytes_sent '
                             '"$http_referer" "$http_user_agent" '
                             'rt=$request_time uct="$upstream_connect_time" uht="$upstream_header_ime" urt="$upstream_response_time";
```

```
server {  
    access_log /spool/logs/nginx-access.log upstream_time;  
    ...  
}
```