

第 9 章

Nginx 的 Web 缓存服务与新浪网的开源 NCACHE 模块

9.1 什么是 Web 缓存

Web 缓存位于内容源 Web 服务器和客户端之间，当用户访问一个 URL 时，Web 缓存服务器会去后端 Web 源服务器取回要输出的内容，然后，当下一个请求到来时，如果访问的是相同的 URL，Web 缓存服务器直接输出内容给客户端，而不是向源服务器再次发送请求。Web 缓存降低了内容源 Web 服务器、数据库的负载，减少了网络延迟，提高了用户访问的响应速度，增强了用户体验。

Web 缓存服务器中，最著名的要数 Squid Cache（简称为 Squid），已经在大多数网站中使用。Squid 是一个流行的自由软件（GNU 通用公共许可证）的代理服务器和 Web 缓存服务器。Squid 有广泛的用途，从作为网页服务器的前置 cache 服务器缓存相关请求来提高 Web 服务器的速度，到为一组人共享网络资源而缓存万维网、域名系统和其他网络搜索，到通过过滤流量帮助网络安全，到局域网通过代理上网。Squid 主要设计用于在 Unix 一类系统运行。

现在，新版本 Nginx 的 `proxy_cache` 指令开始支持 Web 缓存服务，另外，新浪网为 Nginx 开发的 NCACHE 模块，也能够支持 Web 缓存服务，解决了 Squid 不能充分利用多核 CPU 的局限，速度比 Squid 更快。

9.2 Nginx 的 Web 缓存服务

Nginx 从 0.7.48 版开始, 支持了类似 Squid 的缓存功能。这个缓存是把 URL 及相关组合当作 Key, 用 md5 算法对 Key 进行哈希, 得到硬盘上对应的哈希目录路径, 从而将缓存内容保存在该目录内。它可以支持任意 URL 链接, 同时也支持 404/301/302 这样的非 200 状态码。虽然目前官方的 Nginx Web 缓存服务只能为指定 URL 或状态码设置过期时间, 不支持类似 Squid 的 PURGE 指令, 手动清除指定缓存页面, 但是, 通过一个第三方的 ngx_cache_purge 模块, 可以清除指定 URL 的缓存。

金山逍遥网已经在生产环境使用 Nginx 的 proxy_cache 缓存功能多月, 十分稳定, 速度不逊于 Squid。在功能上, Nginx 已经具备 Squid 所拥有的 Web 缓存加速功能、清除指定 URL 缓存的功能。而在性能上, Nginx 对多核 CPU 的利用, 胜过 Squid 不少。另外, 在反向代理、负载均衡、健康检查、后端服务器故障转移、重写、易用性上, Nginx 也比 Squid 强大很多。这使得一台 Nginx 可以同时作为“负载均衡服务器”与“Web 缓存服务器”来使用。

Nginx 的 Web 缓存服务主要由 proxy_cache 相关指令集和 fastcgi 相关指令集构成, 前者用于反向代理时, 对后端内容源服务器进行缓存, 后者主要用于对 FastCGI 的动态程序进行缓存。两者的功能基本上一致。

9.2.1 proxy_cache 相关指令集

1. proxy_cache 指令

语法: proxy_cache zone_name;

默认值: None

使用环境: http, server, location

该指令用于设置哪个缓存区将被使用, zone_name 的值为 proxy_cache_path 指令创建的缓存区名称。

2. proxy_cache_path 指令

语法: proxy_cache_path path [levels=number] keys_zone=zone_name:zone_size [inactive=time] [max_size=size];

默认值: None

使用环境: http

该指令用于设置缓存文件的存放路径。示例如下:


```
proxy_cache_path /data0/proxy_cache_dir levels=1:2 keys_zone=cache_one:
500m inactive=1d max_size=30g;
```

注意该指令只能在 http 标签内配置, levels 指定该缓存空间有两层 hash 目录, 第一层目录为 1 个字母, 第二层为 2 个字母, 保存的文件名会类似/data0/proxy_cache_dir/c/29/b7f54b2df7773722d-382f4809d65029c; keys_zone 参数用来为这个缓存区起名, 500m 指内存缓存空间大小为 500MB; inactive 的 1d 指如果缓存数据在 1 天内没有被访问, 将被删除; max_size 的 30g 是指硬盘缓存空间为 30GB。

3. proxy_cache_methods 指令

语法: proxy_cache_methods [GET HEAD POST];

默认值: proxy_cache_methods GET HEAD;

使用环境: http, server, location

该指令用于设置缓存哪些 HTTP 方法, 默认缓存 HTTP GET/HEAD 方法, 不缓存 HTTP POST 方法。

4. proxy_cache_min_uses 指令

语法: proxy_cache_min_uses the_number;

默认值: proxy_cache_min_uses 1;

使用环境: http, server, location

该指令用于设置缓存的最小使用次数, 默认值为 1。

5. proxy_cache_valid 指令

语法: proxy_cache_valid reply_code [reply_code ...] time;

默认值: None

使用环境: http, server, location

该指令用于对不同返回状态码的 URL 设置不同的缓存时间, 例如:

```
proxy_cache_valid 200 302 10m;
proxy_cache_valid 404 1m;
```

设置 200、302 状态的 URL 缓存 10 分钟, 404 状态的 URL 缓存 1 分钟。

```
proxy_cache_valid 5m;
```

如果不指定状态码, 直接指定缓存时间, 则只有 200、301、302 状态的 URL 缓存 5 分钟。

```
proxy_cache_valid 200 302 10m;
proxy_cache_valid 301 1h;
```



```
proxy_cache_valid any 1m;
```

对没有单独设置的状态码，全部设置缓存时间为 1 分钟。

6. proxy_cache_key 指令

语法: proxy_cache_key line;

默认值: None

使用环境: http, server, location

该指令用来设置 Web 缓存的 Key 值，Nginx 根据 Key 值 md5 哈希存储缓存。一般根据\$host（域名）、\$request_uri（请求的路径）等变量组合成 proxy_cache_key。例如：

```
proxy_cache_key "$host:$server_port$uri$is_args$args";
```

9.2.2 proxy_cache 完整示例

上一节中，我们已经介绍了 proxy_cache 的相关指令集，现在来看一个 proxy_cache 的完整示例。

(1) 首先，我们要按照以下步骤，把第三方的 ngx_cache_purge 模块编译安装到 Nginx 中，用来清除指定 URL 的缓存，示例如代码 9-1 所示。

代码 9-1

```
ulimit -SHn 65535
wget ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/pcre-8.00.tar.gz
tar zxvf pcre-8.00.tar.gz
cd pcre-8.00/
./configure
make && make install
cd ../

wget http://labs.frickle.com/files/nginx_cache_purge-1.0.tar.gz
tar zxvf ngx_cache_purge-1.0.tar.gz

wget http://nginx.org/download/nginx-0.8.32.tar.gz
tar zxvf nginx-0.8.32.tar.gz
cd nginx-0.8.32/
./configure --user=www --group=www --add-module=../ngx_cache_purge-1.0
--prefix=/usr/local/webserver/nginx --with-http_stub_status_module
--with-http_ssl_module
make && make install
cd ../
```

(2) 然后，在同一分区下创建两个缓存目录，分别供 proxy_temp_path、proxy_cache_path 指令设置缓存路径。

注：两个指定设置的缓存路径必须为同一磁盘分区，不能跨分区。

```
mkdir -p /data0/proxy_temp_path
mkdir -p /data0/proxy_cache_path
```

(3) Nginx 配置文件 (nginx.conf)：对扩展名为 gif、jpg、jpeg、png、bmp、swf、js、css 的图片、Flash、JavaScript、CSS 文件开启 Web 缓存，其他文件不缓存，示例如代码 9-2 所示。

代码 9-2

```
user www www;

worker_processes 8;

error_log /data1/logs/nginx_error.log crit;

pid /usr/local/webserver/nginx/nginx.pid;

#Specifies the value for maximum file descriptors that can be opened by this process.
worker_rlimit_nofile 51200;

events
{
    use epoll;
    worker_connections 51200;
}

http
{
    include mime.types;
    default_type application/octet-stream;

    #charset utf-8;

    server_names_hash_bucket_size 128;
    client_header_buffer_size 32k;
    large_client_header_buffers 4 32k;

    sendfile on;
    #tcp_nopush on;

    keepalive_timeout 30;

    tcp_nodelay on;

    #注：proxy_temp_path 和 proxy_cache_path 指定的路径必须在同一分区
    proxy_temp_path /data0/proxy_temp_path;
    #设置 Web 缓存区名称为 cache_one，内存缓存空间大小为 500MB，自动清除超过 1 天没有被访问的缓存数据，
    #硬盘缓存空间大小为 30GB。
    proxy_cache_path /data0/proxy_cache_path levels=1:2 keys_zone=cache_one:200m
    inactive=1d max_size=30g;
```



```
upstream my_server_pool {
    server 192.168.1.2:80 weight=1 max_fails=2 fail_timeout=30s;
    server 192.168.1.3:80 weight=1 max_fails=2 fail_timeout=30s;
    server 192.168.1.4:80 weight=1 max_fails=2 fail_timeout=30s;
}

server
{
    listen 80;
    server_name my.domain.com;

    location /
    {
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_pass http://my_server_pool;
    }

    location ~ .*\. (gif|jpg|jpeg|png|bmp|swf|js|css)$
    {
        #使用 Web 缓存区 cache_one
        proxy_cache cache_one;

        #对不同 HTTP 状态码缓存设置不同的缓存时间
        proxy_cache_valid 200 304 12h;
        proxy_cache_valid 301 302 1m;
        proxy_cache_valid any 1m;

        #设置 Web 缓存的 Key 值, Nginx 根据 Key 值 md5 哈希存储缓存, 这里根据“域名、URI、参数”组合成 Key。
        proxy_cache_key $host$uri$sis_args$args;

        #反向代理, 访问后端内容源服务器
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_pass http://my_server_pool;
    }

    #用于清除缓存, 假设一个 URL 为 http:// my.domain.com/test.gif, 通过访问
    http://my.domain.com/purge/test.gif 可以清除该 URL 的缓存。
    location ~ /purge(/.*)
    {
        #设置只允许指定的 IP 或 IP 段才可以清除 URL 缓存。
        allow 127.0.0.1;
        allow 192.168.0.0/16;
        deny all;
        proxy_cache_purge cache_one $host$1$sis_args$args;
    }

    access_log off;
}
}
```


9.2.3 fastcgi_cache 相关指令集

1. fastcgi_cache 指令

语法: `fastcgi_cache zone_name;`

默认值: `off`

使用环境: `http, server, location`

该指令用于设置哪个缓存区将被使用, `zone_name` 的值为 `fastcgi_cache_path` 指令创建的缓存区名称。

2. fastcgi_cache_path 指令

语法: `fastcgi_cache_path path [levels=number] keys_zone=zone_name:zone_size [inactive=time] [max_size=size];`

默认值: `None`

使用环境: `http`

该指令用于设置缓存文件的存放路径。示例如下:

```
fastcgi_cache_path /data0/fastcgi_cache_dir levels=1:2 keys_zone=cache_one:500m
inactive=1d max_size=30g;
```

注意该指令只能在 `http` 标签内配置, `levels` 指定该缓存空间有两层 `hash` 目录, 第一层目录为 1 个字母, 第二层为 2 个字母, 保存的文件名会类似 `/data0/fastcgi_cache_dir/c/29/b7f54b2df-7773722d382f4809d65029c`; `keys_zone` 参数用来为这个缓存区起名, `500m` 指内存缓存空间大小为 500MB; `inactive` 的 `1d` 指如果缓存数据在 1 天内没有被访问, 将被删除; `max_size` 的 `30g` 是指硬盘缓存空间为 30GB。

3. fastcgi_cache_methods 指令

语法: `fastcgi_cache_methods [GET HEAD POST];`

默认值: `fastcgi_cache_methods GET HEAD;`

使用环境: `http, server, location`

该指令用于设置缓存哪些 HTTP 方法, 默认缓存 HTTP GET/HEAD 方法, 不缓存 HTTP POST 方法。

4. fastcgi_cache_min_uses 指令

语法: `fastcgi_cache_min_uses the_number;`

默认值: `fastcgi_cache_min_uses 1;`

使用环境: `http, server, location`

该指令用于设置缓存的最小使用次数, 默认值为 1。

5. `fastcgi_cache_valid` 指令

语法: `fastcgi_cache_valid reply_code [reply_code ...] time;`

默认值: `None`

使用环境: `http, server, location`

该指令用于对不同返回状态码的 URL 设置不同的缓存时间, 例如:

```
fastcgi_cache_valid 200 302 10m;  
fastcgi_cache_valid 404 1m;
```

设置 200、302 状态的 URL 缓存 10 分钟, 404 状态的 URL 缓存 1 分钟。

```
fastcgi_cache_valid 5m;
```

如果不指定状态码, 直接指定缓存时间, 则只有 200、301、302 状态的 URL 缓存 5 分钟。

```
fastcgi_cache_valid 200 302 10m;  
fastcgi_cache_valid 301 1h;  
fastcgi_cache_valid any 1m;
```

对没有单独设置的状态码, 全部设置缓存时间为 1 分钟。

6. `fastcgi_cache_key` 指令

语法: `fastcgi_cache_key line;`

默认值: `None`

使用环境: `http, server, location`

该指令用来设置 Web 缓存的 Key 值, Nginx 根据 Key 值 md5 哈希存储缓存。一般根据 FastCGI 服务器的地址和端口、`$request_uri` (请求的路径) 等变量组合成 `fastcgi_cache_key`。例如:

```
fastcgi_cache_key 127.0.0.1:9000$request_uri;
```

9.2.4 `fastcgi_cache` 完整示例

上一节中, 我们已经介绍了 `fastcgi_cache` 的相关指令集, 现在来看一个 `fastcgi_cache` 的完整示例:

(1) 首先, 在同一分区下创建两个缓存目录, 分别供 `fastcgi_temp_path`、`fastcgi_cache_path` 指令设置缓存路径。

注：两个指定设置的缓存路径必须为同一磁盘分区，不能跨分区。

```
mkdir -p /data0/fastcgi_temp_path
mkdir -p /data0/fastcgi_cache_path
```

(2) Nginx 配置文件 (nginx.conf)：对扩展名为 gif、jpg、jpeg、png、bmp、swf、js、css 的图片、Flash、JavaScript、CSS 文件开启 Web 缓存，其他文件不缓存，示例如代码 9-3 所示。

代码 9-3

```
user www www;

worker_processes 8;

error_log /data1/logs/nginx_error.log crit;

pid /usr/local/webserver/nginx/nginx.pid;

#Specifies the value for maximum file descriptors that can be opened by this process.
worker_rlimit_nofile 51200;

events
{
    use epoll;
    worker_connections 51200;
}

http
{
    include mime.types;
    default_type application/octet-stream;

    #charset utf-8;

    server_names_hash_bucket_size 128;
    client_header_buffer_size 32k;
    large_client_header_buffers 4 32k;

    sendfile on;
    #tcp_nopush on;

    keepalive_timeout 30;

    tcp_nodelay on;

    #注：fastcgi_temp_path 和 fastcgi_cache_path 指定的路径必须在同一分区
    fastcgi_temp_path /data0/fastcgi_temp_path;
    #设置 Web 缓存区名称为 cache_one，内存缓存空间大小为 500MB，自动清除超过 1 天没有被访问的缓存数据，
    #硬盘缓存空间大小为 30GB。
    fastcgi_cache_path /data0/fastcgi_cache_path levels=1:2
    keys_zone=cache_one:200m inactive=1d max_size=30g;
```



```
upstream my_server_pool {
    server 192.168.1.2:80 weight=1 max_fails=2 fail_timeout=30s;
    server 192.168.1.3:80 weight=1 max_fails=2 fail_timeout=30s;
    server 192.168.1.4:80 weight=1 max_fails=2 fail_timeout=30s;
}

server
{
    listen      80;
    server_name my.domain.com;
    root /data0/htdocs;

    location ~ .*\. (php|php5)$
    {
        #使用 Web 缓存区 cache_one
        fastcgi_cache cache_one;

        #对不同 HTTP 状态码缓存设置不同的缓存时间
        fastcgi_cache_valid 200 10m;
        fastcgi_cache_valid 301 302 1h;
        fastcgi_cache_valid any 1m;

        #设置 Web 缓存的 Key 值, Nginx 根据 Key 值 md5 哈希存储缓存, 这里根据 “FastCGI 服务器的 IP、
        端口、请求的 URI” 组合成 Key。
        fastcgi_cache_key 127.0.0.1:9000$request_uri;

        #FastCGI 服务器
        fastcgi_pass 127.0.0.1:9000;
        fastcgi_index index.php;
        include fcgi.conf;
    }

    access_log off;
}
}
```

9.3 新浪网开源软件项目——基于 Nginx 的 NCACHE 网页缓存系统

NCACHE 是基于 Nginx 的 Web 服务器模型构建起来的缓存系统, 是新浪公司的开源产品。NCACHE 最早的时候是作为 Nginx 的一个 HTTP 模块进行开发的, 因为当时希望获得更好的兼容性和可扩展性, 作为独立模块, 可以被更好地推广和使用, 安装也会很方便。但后来发现随着代码量的增加、功能的扩充, Nginx 的原有模块框架已经不能很好地满足要求了, 因此, 我们提取了 Nginx 的内核代码, 并把 CACHE 部分嵌入其中, 形成了今天的 NCACHE。

NCACHE 本身功能并不强大，且不具备像 SQUID 般完善的功能和开发框架，甚至不能支持 RFC 中关于 CACHE 部分的描述。NCACHE 完全是一套定制化的产品，可以满足要快速部署、简单易用、大并发量、大存储量的需求，它不需要复杂的配置，不需要冗余的复杂代码，并使用最先进的技术组合。

NCACHE 2.0 版本，是作为一个完整的 Nginx 模块进行发布和使用的，从原有的 NCACHE 内核中进行了剥离，更方便开发者的安装和配置。

NCACHE 3.0 版本，相对于 2.0 版本有了很大的改进，对文件的缓存不再使用传统的目录模式，而是通过 MMAP 一个大文件，在其中以页分配的形式存储缓存数据，由操作系统来负责决定哪些数据应该留在内存里，这与 VARNISH 缓存的原理是一致的，大大提高了 IO 性能，目前该版本只支持 64 位 Linux 和 FREEBSD 系统。

9.3.1 NCACHE 模块的安装

NCACHE 需要在编译 Nginx 时，将自己编译进 Nginx。NCACHE 模块最新的版本为 ncache-3.1.64，目前该版本只支持 64 位 Linux 和 FREEBSD 操作系统，ncache-3.1.64 能够兼容 nginx-0.6.x 系列版本，目前还不兼容 nginx-0.7.x、nginx-0.8.x 版本。

下面，我们将 ncache-3.1.64 版本编译进 nginx-0.6.39，如代码 9-4 所示：

代码 9-4

```
mkdir -p /data0/software
cd /data0/software
wget http://ncache.googlecode.com/files/ncache-3.1_64_linux.tar.gz
tar zxvf ncache-3.1_64_linux.tar.gz
wget http://sysoev.ru/nginx/nginx-0.6.39.tar.gz
tar zxvf nginx-0.6.39.tar.gz
cd nginx-0.6.39/
./configure --prefix=/usr/local/webserver/ncache --add-module=../ncache --user=www
--group=www --with-http_stub_status_module --with-http_ssl_module
make && make install
```

9.3.2 NCACHE 配置文件编写

创建配置文件/usr/local/webserver/ncache/conf/nginx.Conf，如代码 9-5 所示：

代码 9-5

```
user www www;

worker_processes 4;
```




```
worker_rlimit_nofile 20480;

error_log logs/error.log;

events
{
    use epoll;

    worker_connections 81920;
}

http
{
    keepalive_timeout 10;

    ncache_max_size 24;

    proxy_buffering off;

    ncache_dir /data1/cache_file 60;#the file for storage,60G
    ncache_dir /data2/cache_file 60;
    hash_index_dir /data1/ncache_index;#v3.1,you need to define the index file
    position
    auto_delete_file on;#enable the auto delete file
    ncache_ignore_client_no_cache on;

    upstream backend
    {
        server 10.0.0.1;
    }

    sendfile on;

    send_timeout 10;

    client_header_timeout 10;

    tcp_nodelay on;

    log_format main '$proxy_add_x_forwarded_for - $remote_user [$time_local]'
        '$request' $status $bytes_sent '
        '$http_referer' '$http_user_agent' $remote_addr';

    include "mime.types";

    default_type text/html;

    server
    {
        server_name .blog.sina.com.cn;
```



```
listen *:80;

set $xvia "blog.sina.com.cn";

set $proxy_add_agent "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1;
  NginxCache)";
access_log logs/blog.sina.com.cn-access_log main;

location /
{
    if ($request_method ~ "PURGE")
    {
        rewrite (.*?) /PURGE$1 last;
    }

    ncache_http_cache;

    error_page 404 = /fetch$request_uri;

    add_header    Sina-Cache $xvia;
}

location /ncache_state
{
    ncache_state;
}

location /fetch
{
    internal;
    proxy_pass http://backend;
    add_header    Sina-Cache $xvia;
    proxy_hide_header User-Agent;
    proxy_set_header User-Agent $proxy_add_agent;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}

location /PURGE/
{
    access_log logs/purge.blog.sina.com.cn-access_log main;
    internal;
    allow 10.55.37.0/24;
    allow 10.69.3.0/24;
    allow 10.49.10.0/24;
    deny all;
    ncache_purge;
}
```



```
}  
}
```

9.3.3 NCACHE 的管理维护

1. 运行 NCACHE

```
/usr/local/webserver/ncache/sbin/nginx
```

2. 停止 NCACHE

```
killall -9 nginx
```

3. 清除缓存的 Shell 脚本

```
#!/bin/sh  
kill -9 `ps auwx|grep nginx|awk '{print $2}`  
killall -9 nginx  
rm ../logs/ncache_index  
echo "" > ../logs/error.log  
rm /data0/ncache_file  
rm /data1/ncache_file  
rm /data2/ncache_file  
rm /data3/ncache_file
```

4. 查看 NCACHE 状态

```
curl "http://127.0.0.1/ncache_state"
```

9.3.4 NCACHE 后端内容源服务器设置

(1) 必须用 HTTP Header 头 “Cache-Control: max-age=秒数” 来控制缓存时间，如果不指定将不缓存。

(2) 后端内容源服务器发送 NCACHE 的 HTTP 数据，必须带有 “Content-Length” Header 头。

(3) NCACHE 的缓存时间以分钟为单位，会将所有 max-age=的秒数值转换为分钟，如果 max-age 小于 1 分钟，NCACHE 会将缓存时间设置成 1 分钟。

(4) 自动删除缓存文件进程，会在每天的凌晨 2 点删除大约 20% 的不活动缓存数据。