Squid 中文权威指南

（第 5 章）

译者序：

本人在工作中维护着数台 Squid 服务器，多次参阅 Duane Wessels（他也是 Squid 的创始人）的这本书，原书名是"Squid: The Definitive Guide"，由 O'Reilly 出版。我在业余时间把它翻译成中文，希望对中文 Squid 用户有所帮助。对普通的单位上网用户，Squid 可充当代理服务器；而对 Sina,NetEase 这样的大型站点，Squid 又充当 WEB 加速器。这两个角色它都扮演得异常优秀。窗外繁星点点，开源的世界亦如这星空般美丽，而 Squid 是其中耀眼的一颗星。

对本译版有任何问题，请跟我联系，我的Email是：yonghua_peng@yahoo.com.cn

彭勇华

目 录

第 5 章 运行 Squid

5.1 squid 命令行选项

在开始其他事情之前，让我们先看一下 squid 的命令行选项。这里的许多选项你从不会使用，另外有些仅仅在调试问题时有用。

-a port

指定新的 http_port 值。该选项覆盖了来自 squid.conf 的值。然而请注意，你能在 squid.conf 里指定多个值。-a 选项仅仅覆盖配置文件里的第一个值。（该选项使用字母 a 是因为在 Harvest cache 里，HTTP 端口被叫做 ASCII 端口）

-d level

让 squid 将它的调试信息写到标准错误（假如配置了，就是 cache.log 和 syslog）。level 参数指定了显示在标准错误里的消息的最大等级。在多数情况下，d1 工作良好。请见 16.2 章关于调试等级的描述。

-f file

指定另一个配置文件。

-h

显示用法。

-k function

指示 squid 执行不同的管理功能。功能参数是下列之一：reconfigure, rotate, shutdown, interrupt, kill, debug, check, or parse. reconfigure 导致运行中的 squid 重新读取配置文件。rotate 导致 squid 滚动它的日志，这包括了关闭日志，重命名，和再次打开它们。shutdown 发送关闭 squid 进程的信号。interrupt 立刻关闭 squid，不必等待活动会话完成。kill 发送 KILL 信号给 squid，这是关闭 squid 的最后保证。debug 将 squid 设置成完全的调试模式，假如你的 cache 很忙，它能迅速的用完你的磁盘空间。check 简单的检查运行中的 squid 进程，返回的值显示 squid 是否在运行。最后，parse 简单的解析 squid.conf 文件，如果配置文件包含错误，进程返回非零值。

-s

激活将日志记录到 syslog 进程。squid 使用 LOCAL4 syslog 设备。0 级别调试信息以优先级 LOG_WARNING 被记录，1 级别消息以 LOG_NOTICE 被记录。更高级的调试信息不会被发送到 syslogd.你可以在/etc/syslogd.conf 文件里使用如下接口：

local4.warning /var/log/squid.log

-u port

指定另一个 ICP 端口号，覆盖掉 squid.conf 文件里的 icp_port。

-v
打印版本信息。

-z
初始化 cache，或者交换，目录。在首次运行 squid，或者增加新的 cache 目录时，你必须使用该选项。

-C
阻止安装某些信号句柄，它们捕获特定的致命信号例如 SIGBUS 和 SIGSEGV。正常的，这些信号被 squid 捕获，以便它能干净的关闭。然而，捕获这些信号可能让以后调试问题困难。使用该选项，致命的信号导致它们的默认动作，通常是 coredump。

-D
禁止初始化 DNS 测试。正常情况下，squid 直到验证它的 DNS 可用才能启动。该选项阻止了这样的检测。你也能在 squid.conf 文件里改变或删除 dns_testnames 选项。

-F
让 squid 拒绝所有的请求，直到它重新建立起存储元数据。假如你的系统很忙，该选项可以减短重建存储元数据的时间。然而，如果你的 cache 很大，重建过程可能会花费很长的时间。

-N
阻止 squid 变成后台服务进程。

-R
阻止 squid 在绑定 HTTP 端口之前使用 SO_REUSEADDR 选项。

-V
激活虚拟主机加速模式。类似于 squid.conf 文件里的 httpd_accel_host virtual 指令。

-X
强迫完整调试模式，如你在 squid.conf 文件里指定 debug_options ALL,9 一样。

-Y
在重建存储元数据时，返回 ICP_MISS_NOFETCH 代替 ICP_MISS.忙碌的父 cache 在重建时，该选项可以导致最少的负载。请见 10.6.1.2 章。

5.2 对配置文件查错

在开启 squid 之前，你应该谨慎的验证配置文件。这点容易做到，运行如下命令即可：
%squid -k parse
假如你看不到输出，配置文件有效，你能继续后面的步骤。
然而，如果配置文件包含错误，squid 会告诉你：
squid.conf line 62: http_access allow okay2

aclParseAccessLine: ACL name 'okay2' not found.

这里你可以看到，62 行的 http_access 指令指向的 ACL 不存在。有时候错误信息很少：

FATAL: Bungled squid.conf line 76: memory_pools

在这个情形里，我们忘记了在 76 行的 memory_pools 指令后放置 on 或 off.

建议你养成习惯：在每次修改配置文件后，使用 squid -k parse。假如你不愿麻烦，并且你的配置文件有错误，squid 会告诉你关于它们而且拒绝启动。假如你管理着大量的 cache，也许你会编辑脚本来自动启动，停止和重配置 squid。你能在脚本里使用该功能，来确认配置文件是有效的。

5.3 初始化 cache 目录

在初次运行 squid 之前，或者无论何时你增加了新的 cache_dir，你必须初始化 cache 目录。命令很简单：

%squid -z

对 UFS 相关的存储机制（ufs,aufs,and diskd;见第 8 章），该命令在每个 cache_dir 下面创建了所需的子目录。你不必担心 squid 会破坏你的当前 cache 目录（如果有的话）。

在该阶段属主和许可权是通常遇到的问题。squid 在特定的用户 ID 下运行，这在 squid.conf 文件里的 cache_effective_user 里指定。用户 ID 必须对每个 cache_dir 目录有读和写权限。否则，你将看到如下信息：

Creating Swap Directories

FATAL: Failed to make swap directory /usr/local/squid/var/cache/00:

(13) Permission denied

在这样的情形下，你该确认/usr/local/squid/var/cache 目录的所有组成都可被 squid.conf 给定的用户 ID 访问。最终的组件--cache 目录--必须对该用户 ID 可写。

cache 目录初始化可能花费一些时间，依赖于 cache 目录的大小和数量，以及磁盘驱动器的速度。假如你想观察这个过程，请使用-X 选项：

%squid -zX

5.4 在终端窗口里测试 squid

一旦你已经初始化 cache 目录，就可以在终端窗口里运行 squid，将日志记录到标准错误。这样，你能轻易的定位任何错误或问题，并且确认 squid 是否成功启动。使用-N 选项来保持 squid 在前台运行，-d1 选项在标准错误里显示 1 级别的调试信息。

%squid -N -d1

你将看到类似于以下的输出：

2003/09/29 12:57:52| Starting Squid Cache

version 2.5.STABLE4 for i386-unknown-freebsd4.8...

2003/09/29 12:57:52| Process ID 294

2003/09/29 12:57:52| With 1064 file descriptors available

2003/09/29 12:57:52| DNS Socket created on FD 4

2003/09/29 12:57:52| Adding nameserver 206.107.176.2 from /etc/resolv.conf

2003/09/29 12:57:52| Adding nameserver 205.162.184.2 from /etc/resolv.conf

2003/09/29 12:57:52| Unlinkd pipe opened on FD 9
2003/09/29 12:57:52| Swap maxSize 102400 KB, estimated 7876 objects
2003/09/29 12:57:52| Target number of buckets: 393
2003/09/29 12:57:52| Using 8192 Store buckets
2003/09/29 12:57:52| Max Mem size: 8192 KB
2003/09/29 12:57:52| Max Swap size: 102400 KB
2003/09/29 12:57:52| Rebuilding storage in /usr/local/squid/var/cache (DIRTY)
2003/09/29 12:57:52| Using Least Load store dir selection
2003/09/29 12:57:52| Set Current Directory to /usr/local/squid/var/cache
2003/09/29 12:57:52| Loaded Icons.
2003/09/29 12:57:52| Accepting HTTP connections at 0.0.0.0, port 3128, FD 11.
2003/09/29 12:57:52| Accepting ICP messages at 0.0.0.0, port 3130, FD 12.
2003/09/29 12:57:52| WCCP Disabled.
2003/09/29 12:57:52| Ready to serve reques

假如你看到错误消息，你该首先修正它。请检查输出信息的开始几行以发现警告信息。最普通的错误是文件/目录许可问题，和配置文件语法错误。假如你看到一条不引起注意的错误消息，请见 16 章中关于 squid 故障处理的建议和信息。如果还不行，请检查 squid FAQ, 或查找邮件列表来获得解释。

一旦你见到"Ready to serve requests"消息，就可用一些 HTTP 请求来测试 squid。配置你的浏览器使用 squid 作为代理，然后打开某个 web 页面。假如 squid 工作正常，页面被迅速载入，就象没使用 squid 一样。另外，你可以使用 squidclient 程序，它随 squid 发布：

% squidclient http://www.squid-cache.org/

假如它正常工作，squid 的主页 html 文件会在你的终端窗口里滚动。一旦确认 squid 工作正常，你能中断 squid 进程（例如使用 ctrl-c）并且在后台运行 squid。

5.5 将 squid 作为服务进程运行

正常情况下你想将 squid 以后台进程运行（不出现在终端窗口里）。最容易的方法是简单执行如下命令：

%squid -s

-s 选项导致 squid 将重要的状态和警告信息写到 syslogd。squid 使用 LOCAL4 设备，和 LOG_WARNING 和 LOG_NOTICE 优先权。syslog 进程实际可能会或不会记录 squid 的消息，这依赖于它被如何配置。同样的消息被写进 cache.log 文件，所以假如你愿意，忽略-s 选项也是安全的。

当你不使用-N 选项来启动 squid，squid 自动在后台运行并且创建父/子进程对。子进程做所有的实际工作。父进程确认子进程总在运行。这样，假如子进程意外终止，父进程启动另外一个子进程以使 squid 正常工作。通过观察 syslog 消息，你能看到父/子进程交互作用。

Jul 31 14:58:35 zapp squid[294]: Squid Parent: child process 296 started

这里显示的父进程 ID 是 294，子进程是 296。当你查看 ps 的输出，你可以看到子进程以(squid)形式出现：

%ps ax | grep squid
  294 ?? Is 0:00.01 squid -sD
  296 ?? S 0:00.27 (squid) -sD (squid)

假如 squid 进程意外终止，父进程启动另一个。例如：

Jul 31 15:02:53 zapp squid[294]: Squid Parent: child process 296 exited due to signal 6

Jul 31 15:02:56 zapp squid[294]: Squid Parent: child process 359 started

在某些情形下，squid 子进程可能立即终止。为了防止频繁的启动子进程，假如子进程连续 5 次没有运行至少 10 秒钟，父进程会放弃。

Jul 31 15:13:48 zapp squid[455]: Squid Parent: child process 474 exited with status 1

Jul 31 15:13:48 zapp squid[455]: Exiting due to repeated, frequent failures

如果发生这样的事，请检查 syslog 和 squid 的 cache.log 以发现错误。

5.5.1 squid_start 脚本

当 squid 以后台进程运行时，它查找 squid 执行程序目录下的名为 squid_start 的文件。假如发现，该程序在父进程创建子进程之前被执行。你能使用该脚本完成特定的管理任务，例如通知某人 squid 在运行，管理日志文件等。除非 squid_start 程序存在，squid 不会创建子进程。

squid_start 脚本在你使用绝对或相对路径启动 squid 时才开始工作。换句话说，squid 不使用 PATH 环境变量来定位 squid_start.这样，你应该养成习惯这样启动 squid：

% /usr/local/squid/sbin/squid -sD

而不要这样：

%squid -sD

5.6 启动脚本

通常你希望 squid 在每次计算机重启后自动启动。对不同的操作系统，它们的启动脚本如何工作也很不同。我在这里描述一些通用的环境，但对你自己的特殊操作系统，也许该有特殊的处理方法。

5.6.1 /etc/rc.local

最容易的机制之一是/etc/rc.local 脚本。这是个简单的 shell 脚本，在每次系统启动时以 root 运行。使用该脚本来启动 squid 非常容易，增加一行如下：

/usr/local/squid/sbin/squid -s

当然你的安装位置可能不同，还有你可能要使用其他命令行选项。不要在这里使用-N 选项。

假如因为某些理由，你没有使用 cache_effective_user 指令，你可以尝试使用 su 来让 squid 以非 root 用户运行：

/usr/bin/su nobody -c '/usr/local/squid/sbin/squid -s'

5.6.2 init.d 和 rc.d

init.d 和 rc.d 机制使用独立的 shell 脚本来启动不同的服务。这些脚本通常在下列目录之中：/sbin/init.d, /etc/init.d, /usr/local/etc/rc.d.脚本通常获取单一命令行参数，是 start 或 stop。某些系统仅仅使用 start 参数。如下是启动 squid 的基本脚本：

```
#!/bin/sh
# this script starts and stops Squid
case "$1" in
start)
        /usr/local/squid/sbin/squid -s
        echo -n ' Squid'
        ;;
stop)
        /usr/local/squid/sbin/squid -k shutdown
        ;;
esac
```

Linux 用户可能在启动 squid 之前需要设置文件描述符限制。例如：

```
echo 8192 > /proc/sys/fs/file-max
limit -HSn 8192
```

为了使用该脚本，先找到脚本存放的目录。给它一个有意义的名字，类似于其他的系统启动脚本。可以是 S98squid 或 squid.sh。通过重启计算机来测试该脚本，而不要假想它会正常工作。

5.6.3 /etc/inittab

某些操作系统支持另一种机制，是/etc/inittab 文件。在这些系统中，init 进程启动和停止基于运行等级的服务。典型的 inittab 接口类似如此：

```
sq:2345:once:/usr/local/squid/sbin/squid -s
```

使用该接口，init 进程启动 squid 一次并且随后忘记它。squid 确认它驻留在运行状态，象前面描述的一样。或者，你能这样做:

```
sq:2345:respawn:/usr/local/squid/sbin/squid -Ns
```

这里我们使用了 respawn 选项，假如进程不存在 init 会重启 squid。假如使用 respawn，请确认使用-N 选项。

在编辑完 inittab 文件后，使用下面的命令来使 init 重新读取它的配置文件和启动 squid：

```
# init q
```

5.7 chroot 环境

某些人喜欢在 chroot 环境运行 squid。这是 unix 的功能，给予进程新的 root 文件系统目录。在 squid 受安全威胁时，它提供额外等级的安全保护。假如攻击者在某种程度上通过 squid 获取了对操作系统的访问权，她仅仅能访问在 chroot 文件系统中的文件。在 chroot 树之外的系统文件，她不可访问。

最容易在 chroot 环境里运行 squid 的方法是，在 squid.conf 文件里指定新的 root 目录，如下：

```
chroot /new/root/directory
```

chroot()系统调用需要超级用户权限，所以你必须以 root 来启动 squid。

chroot 环境不是为 unix 新手准备的。它有点麻烦，因为你必须在新的 root 目录里重复放置大量的文件。例如，假如默认的配置文件正常在/usr/local/squid/etc/squid.conf，并且你

使用 chroot 指令，那么文件必须位于/new/root/directory/usr/local/squid/etc/squid.conf.你必须将位于$prefix/etc,$prefix/share,$prefix/libexec 下的所有文件拷贝到 chroot 目录。请确认$prefix/var 和 cache 目录在 chroot 目录中存在和可写。

同样的，你的操作系统需要将大量的文件放在 chroot 目录里，例如/etc/resolv.conf 和/dev/null.假如你使用外部辅助程序，例如重定向器（见 11 章）或者验证器（见 12 章），你也需要来自/usr/lib 的某些共享库。你可以使用 ldd 工具来查找给定的程序需要哪些共享库：

```
% ldd /usr/local/squid/libexec/ncsa_auth
/usr/local/squid/libexec/ncsa_auth:
        libcrypt.so.2 => /usr/lib/libcrypt.so.2 (0x28067000)
        libm.so.2 => /usr/lib/libm.so.2 (0x28080000)
        libc.so.4 => /usr/lib/libc.so.4 (0x28098000)
```

你可以使用 chroot 命令来测试辅助程序：

```
# chroot /new/root/directory /usr/local/squid/libexec/ncsa_auth
/usr/libexec/ld-elf.so.1: Shared object "libcrypt.so.2" not found
```

更多的关于 chroot 的信息，请见你系统中 chroot()的 manpage.

5.8 停止 squid

最安全的停止 squid 的方法是使用 squid -k shutdown 命令：

```
%squid -k shutdown
```

该命令发送 TERM 信号到运行中的 squid 进程。在接受到 TERM 信号后，squid 关闭进来的套接字以拒收新请求。然后它等待一段时间，用以完成外出请求。默认时间是 30 秒，你可以在 shutdown_lifetime 指令里更改它。

假如因为某些理由，squid.pid 文件丢失或不可读，squid -k 命令会失败。在此情形下，你可以用 ps 找到 squid 的进程 ID，然后手工杀死 squid。例如：

```
%ps ax |grep squid
```

假如你看到不止一个 squid 进程，请杀死以(squid)显示的那个。例如：

```
% ps ax | grep squid
  294  ??  Is      0:00.01 squid -sD
  296  ??  S       0:00.27 (squid) -sD (squid)
% kill -TERM 296
```

在发送 TERM 信号后，你也许想查看日志，以确认 squid 已关闭：

```
% tail -f logs/cache.log
2003/09/29 21:49:30| Preparing for shutdown after 9316 requests
2003/09/29 21:49:30| Waiting 10 seconds for active connections to finish
2003/09/29 21:49:30| FD 11 Closing HTTP connection
2003/09/29 21:49:31| Shutting down...
2003/09/29 21:49:31| FD 12 Closing ICP connection
2003/09/29 21:49:31| Closing unlinkd pipe on FD 9
2003/09/29 21:49:31| storeDirWriteCleanLogs: Starting...
2003/09/29 21:49:32| Finished.  Wrote 253 entries.
2003/09/29 21:49:32| Took 0.1 seconds (1957.6 entries/sec).
2003/09/29 21:49:32| Squid Cache (Version 2.5.STABLE4): Exiting normally.
```

假如你使用 squid -k interrupt 命令，squid 立即关闭，不用等待完成活动请求。这与在 kill 里发送 INT 信号相同。

5.9 重配置运行中的 squid 进程

在你了解了更多关于 squid 的知识后，你会发现对 squid.conf 文件做了许多改动。为了让新设置生效，你可以关闭和重启 squid,或者在 squid 运行时，重配置它。

重配置运行中的 squid 最好的方法是使用 squid -k reconfigure 命令：

%squid -k reconfigure

当你运行该命令时，HUP 信号被发送到运行中的 squid 进程。然后 squid 读取和解析 squid.conf 文件。假如操作成功，你可以在 cache.log 里看到这些：

2003/09/29 22:02:25| Restarting Squid Cache (version 2.5.STABLE4)...

2003/09/29 22:02:25| FD 12 Closing HTTP connection

2003/09/29 22:02:25| FD 13 Closing ICP connection

2003/09/29 22:02:25| Cache dir '/usr/local/squid/var/cache' size remains unchanged at 102400 KB

2003/09/29 22:02:25| DNS Socket created on FD 5

2003/09/29 22:02:25| Adding nameserver 10.0.0.1 from /etc/resolv.conf

2003/09/29 22:02:25| Accepting HTTP connections at 0.0.0.0, port 3128, FD 9.

2003/09/29 22:02:25| Accepting ICP messages at 0.0.0.0, port 3130, FD 11.

2003/09/29 22:02:25| WCCP Disabled.

2003/09/29 22:02:25| Loaded Icons.

2003/09/29 22:02:25| Ready to serve requests.

在使用 reconfigure 选项时你须谨慎，因为所做的改变可能会导致致命错误。例如，请注意 squid 关闭和重新打开进来的 HTTP 和 ICP 套接字；假如你将 http_port 改变为 squid 不能打开的端口，它会发生致命错误并退出。

在 squid 运行时，某些指令和和选项不能改变，包括：

删除 cache 目录（cache_dir 指令）

改变 store_log 指令

改变 coss cache_dir 的块大小数值。事实上，无论何时你改变了该值，你必须重新初始化 coss cache_dir.

coredump_dir 指令在重配置过程中不被检查。所以，在 squid 已经启动了后，你不能让 squid 改变它的当前目录。

solaris 用户在重配置 squid 过程中可能遇到其他问题。solaris 的 stdio 执行组件里的 fopen() 调用要求使用小于 256 的未用文件描述符。FILE 结构以 8 位值存储该文件描述符。正常情况下这不构成问题，因为 squid 使用底层 I/O（例如 open()）来打开 cache 文件。然而，在重配置过程中的某些任务使用 fopen()，这就有可能失败，因为前面的 256 个文件描述符已被分配出去。

5.10 滚动日志文件

除非你在 squid.conf 里禁止，squid 会写大量的日志文件。你必须周期性的滚动日志文件，以阻止它们变得太大。squid 将大量的重要信息写入日志，假如写不进去了，squid 会发

生错误并退出。为了合理控制磁盘空间消耗，在 cron 里使用如下命令：

%squid -k rotate

例如，如下任务接口在每天的早上 4 点滚动日志：

0 4 * * * /usr/local/squid/sbin/squid -k rotate

该命令做两件事。首先，它关闭当前打开的日志文件。然后，通过在文件名后加数字扩展名，它重命名 cache.log,store.log,和 access.log。例如，cache.log 变成 cache.log.0,cache.log.0 变成 cache.log.1,如此继续，滚动到 logfile_rotate 选项指定的值。

squid 仅仅保存每个日志文件的最后 logfile_rotate 版本。更老的版本在重命名过程中被删除。假如你想保存更多的拷贝，你需要增加 logfile_rotate 限制，或者编写脚本用于将日志文件移动到其他位置。

请见 13.7 章关于滚动日志的其他信息。