

Такой разный Squid

Сергей Яремчук

Одним из самых популярных решений для организации совместного доступа в Интернет, кэширования трафика и борьбы с баннерами является прокси-сервер Squid. В различных HOWTO можно найти готовые настройки и инструкции, но работают они не всегда.

Зачем это необходимо?

Прокси-сервер Squid [1] развивается в течение уже многих лет, за это время было выпущено несколько версий, настройки в которых, хотя и незначительно, но отличаются. В последние дни декабря 2007-го в разряд STABLE перешла третья версия Squid, которая сегодня уже рекомендуема к использованию. В репозиториях дистрибутивов встречается сразу несколько версий Squid, и только это подчас вызывает путаницу у новичков. Например, возьмем Ubuntu. Для сервера рекомендуемым является Ubuntu 6.06, который пока единственный из всех Ubuntu обладающий титулом LTS (Long Term Support) и его поддержка будет продолжаться до середины 2011 года. Но при установке Squid при помощи команды:

```
$ sudo apt-get install squid
```

мы обнаружим, что имеем дело с версией 2.5.STABLE12. В репозитории Gutsy Gibbon уже присутствует версия 2.6.14. Но это еще не все. Начиная с Festy Fawn (7.04) в репозитории доступны пакеты и с третьей версией Squid, для установки которой следует вводить:

```
$ sudo apt-get install squid3
```

Причем в 7.04, 7.10 и 8.04 версиях тройки разные, но на настройках это никак не отражается, поэтому трогать их не будем.

В FreeBSD ситуация аналогична, хотя и не так запутанна. В портах FreeBSD присутствуют обе версии. Для установки 2.6.x (на момент написания 2.6.18) вводим:

```
# cd /usr/ports/www/squid
```

Или для 3.0:

```
# cd /usr/ports/www/squid30
```

И далее стандартные:

```
# make install
```

Таким образом, сегодня некоторым администраторам, вполне вероятно, приходится иметь дело с разными версиями Squid. А в случае обновлений столкнуться с тем, что с новым Squid нельзя использовать старый файл настроек. Теперь перейдем непосредственно к отличиям.

Подключение внешних файлов

Одним из главных нововведений, появившихся в 3.0, является поддержка директивы include, при помощи кото-

рой можно подключать внешние файлы с настройками. Вроде этого:

```
include /path/to/included/file/squid.acl.config
```

При попытке использовать подобную конструкцию в Squid второй ветки получаем предупреждение:

```
$ sudo squid -k parse
```

```
2008/03/31 12:57:52| parseConfigFile: line 1860 unrecognized:
'include /etc/squid/acl'
```

Но эта ошибка не является критической, поэтому прокси-сервер ее пропускает. При запуске с консоли администратор будет уведомлен, но при загрузке вместе с системой можно и не догадаться, почему параметры в файлах, подключенных при помощи include, не работают. Внутри таких файлов также допускается применение include, но увлекаться здесь не стоит. Во избежание проблем установлено ограничение в 16 уровней вложенности, чего вполне достаточно.

Напомню, что в некоторых правилах и раньше было разрешено использовать внешние файлы. Например, вместо того чтобы писать:

```
acl blockfiles url_regex -i ftp \.exe \.mp3 \.zip \.rar \.
\avi \.mpeg \.mpg \.iso \.raw \.wav
```

Удобнее такие файлы указывать во внешнем файле:

```
acl blockfiles urlpath_regex -i "/etc/squid/blocks.files.acl"
```

И в файл blocks.files.acl прописываем регулярные выражения:

```
$ sudo nano /etc/squid/blocks.files.acl
\.exe$
\.avi$
\.mpg$
\.mpeg$
\.mp3$
```

И так далее.

Начиная с версии 2.6 Squid стал дополнительно поддерживать несколько типов ACL:

- **атрибуты SSL-сертификата** – user_cert и ca_cert;
- **имя пользователя, полученное с внешней ACL** – ext_user, extuser_regex.

Появление их в конфигурационном файле версии 2.5 вызовет остановку прокси-сервера.

Сетевые параметры

Для включения прозрачного проксирования в версиях 2.6 и 3.0 достаточно использовать параметр transparent:

```
http_port 3128 transparent
```

Но версия 2.5 его не поддерживает, при попытке его включения получаем критическую ошибку:

```
$ sudo squid -k parse
```

```
FATAL: Bungled squid.conf line 53: http_port 192.168.0.1:3128 transparent
Squid Cache (Version 2.5.STABLE12): Terminated abnormally.
```

Причем ошибка является фатальной, и дальнейшая работа сервера прекращается.

Директива urlgroup, используемая в параметре https_ports, при помощи которого указываются адрес и порт для SSL-подключений через Squid, будет работать только в 2.6. Ни в 2.5, ни в 3.0 ее нет. Хотя за это время она, правда, так и не успела набрать популярности. Но теперь при помощи параметра name можно задать имя порта для более удобного использования в правилах. Напомним, что https_ports, как и поддержка других параметров для работы с SSL, появилась только начиная с версии 2.5, да и то не все. Так, в 2.6 добавились ssl_engine и целый набор параметров sslproxy_*.

Списки доступа

Для обработки ACL Squid может задействовать внешнюю программу, описание которой дается в параметре external_acl_type. В версии Squid-2.5.STABLE3 и ранее вместо директивы children, при помощи которой указывается количество процессов, выполняющихся для external_acl_type, использовался concurrency. В более поздних версиях 2.5 в целях совместимости поддерживаются оба параметра, но начиная с 2.6 старый параметр уже не поддерживается. Поэтому вместо правила вроде:

```
external_acl_type nt_group %LOGIN concurrency=10
/usr/lib/squid/wbinfo_group.pl
```

пишем такое:

```
external_acl_type nt_group %LOGIN children=10
/usr/lib/squid/wbinfo_group.pl
```

В версии 3.0 убран параметр http_access2, который выполнял те же функции, что и http_access, но после редиректа. Впрочем, особой популярностью он также не пользовался.

Аутентификация

С версии 2.5 и выше Squid может предлагать клиенту несколько схем аутентификации (Proxy-Authenticate), а браузер выбирает самую безопасную из поддерживаемых. Для задания параметров аутентификации используется auth_param. В версии 3.0 в схеме basic убрана директива casesensitive, отвечающая за регистрозависимость имени пользователя, подключающегося к Squid. В большинстве примеров, которые мне приходилось видеть, он был отключен. Поэтому при переносе squid.conf на новую версию прокси-сервера строку:

```
auth_param basic casesensitive off
```

можно убирать, хотя ее наличие не приводит к фатальной ошибке.

Работа с кэшем

Расположение и размер кэша Squid определяется параметром cache_dir. Причем в squid.conf может быть несколько

ко таких описаний, что очень удобно, так как можно расположить кэш на разных дисках. Версии 2.5 и 2.6 поддерживают дополнительную директиву `read-only`, указывающую на режим «только чтение» для этого `cache_dir`. В Squid 3.0 она убрана.

Начиная с версии 2.6 появился еще один параметр `read_ahead_gap`, отвечающий за размер упреждающего буфера при передаче данных клиенту от другого сервера. При использовании его в 2.5 администратор получает сообщение об ошибочном параметре, а Squid без проблем запускается.

Еще об одном параметре изменения. Это `refresh_pattern`, используемый для определения устаревания объекта в кэше. Так, в версии Squid 3.0 по умолчанию используются инструкции:

```
refresh_pattern ^ftp:      1440      20%      10080
refresh_pattern ^gopher:  1440      0%       1440
refresh_pattern (cgi-bin|?) 0        0%        0
refresh_pattern .         0        20%      4320
```

В версиях 2.5 и 2.6 Squid третье правило отсутствует. Но не это главное изменение. В поле `options` через пробел указываются дополнительные параметры. В версии 2.x параметров семь, в версии 3.x добавилось еще два. Большинство из них идут в разрез со стандартами HTTP, и их использование может вызвать проблемы при работе с некоторыми серверами. Но обычно серверы корректно с этим справляются, и до каких-либо конфликтов не доходит.

Они полезны для оптимизации кэша, и чтобы показать общую картину, расскажу обо всех:

- **override-expire** – в нарушение стандарта заставляет игнорировать параметр `expire`, то есть время актуальности объекта;
- **override-lastmod** – игнорирование времени последней модификации объекта, переданное сервером;
- **reload-into-ims, ignore-reload** – изменяет или игнорирует клиентские запросы `posache` или `reload` и принудительно выдает объект, хранящийся в кэше;
- **ignore-no-cache, ignore-private, ignore-auth** – игнорирует заголовки «Pragma: no-cache», «Cache-control: no-cache», «Cache-control: private» и «Cache-control: public», принудительно кэшируя такой объект.

И два новых параметра, появившиеся в третьей версии Squid:

- **ignore-no-store** – игнорирование заголовка «Cache-control: no-store»;
- **refresh-ims** – заставляет проверять наличие новой версии файла при получении от клиента `If-Modified-Since`.

Теперь легко вместо правил по умолчанию можно написать всего одно правило, заставляющее принудительно кэшировать объекты на целый год и игнорирующее всякие попытки его обновления:

```
refresh_pattern .          518400 80% 518400
override-expire override-lastmod reload-into-ims
ignore-no-cache ignore-private ignore-auth
ignore-no-store
```

Это, конечно, крайний случай, но администратор в Squid 3.0 получает фактически полный контроль над кэшем.

Изменение в журналах

Также стоит обратить внимание на изменение форматов в `logformat`, при помощи которой описывается, как должен выглядеть файл журнала запросов `access.log`. Вроде этого:

```
logformat squid %ts.%03tu %6tr %>a %Ss/%03Hs
%<st %rm %ru %un %Sh/%<A %mt
access_log /var/log/squid/access.log squid
```

Но, например, в версии 3.0 появился новый формат «<S», показывающий размер потока объекта, а убран «st», показывающий общий (Request+Reply) размер, включая HTTP-заголовки. Кстати, параметр `logformat` впервые появился в версии 2.6. Поэтому при запуске в более раннем Squid получаем ошибку.

```
$ sudo squid -k parse
```

```
2008/03/31 19:57:55| parseConfigFile: line 3512 unrecognized:
'logformat Squid %ts.%03tu %6tr %>a %Ss/%03Hs %<st %rm %ru %un
%Sh/%<A %mt'
```

Да и вместо `access_log` в версии 2.5 следует использовать `cache_access_log`.

Работа с HTTP-заголовками

В 2.4 вместо `anonymize_headers`, который отвечал за анонимизацию заголовков (в нарушение HTTP), был введен параметр `header_access`, он и используется в версиях от 2.4 до 2.6. В третьей версии вместо `header_access` следует применять `request_header_access`. Ничего нового в дополнительных параметрах придумывать не нужно, достаточно просто добавить префикс `request_` к имеющимся правилам:

```
request_header_access From deny all
request_header_access Referer deny all
request_header_access Server deny all
request_header_access User-Agent deny all
request_header_access WWW-Authenticate deny all
request_header_access Link deny all
```

Так же, как и раньше, по умолчанию используется правило «request_header_access Allow allow all», то есть все заголовки разрешены.

Заключение

Это далеко не все отличия, с которыми можно столкнуться, но я думаю, этого достаточно, чтобы понять, насколько отличаются разные версии Squid. Совет может быть только один – обязательно проверяйте перед запуском корректность файла `squid.conf` с помощью команды:

```
squid -k parse
```

Успехов! 

1. Сайт проекта Squid – <http://www.squid-cache.org>.
2. Список некоторых нововведений в Squid-3.0.STABLE1 – <http://solaris.opennet.ru/opennews/art.shtml?num=13298>.