




Hilfe, mein Webserver ist lahm!

18. Juni 2009

Lahmende Webserver können durch Einsatz eines 'Reverse Proxy' erheblich beschleunigt werden. Als kostenloser Nebeneffekt ergibt sich ein wirksamer Schutz gegen Angriffe von aussen. Auch für Lastverteilung und Hochverfügbarkeit bieten sich einfache und effiziente Konfigurationsmöglichkeiten an. Ein 'Reverse Proxy' kann mit geringen Aufwand oft mehr bewirken als der Einsatz aufwändiger Hardware. Entgegen der weitverbreiteten Meinung kann ein Proxy auch bei dynamischen Webseiten erhebliche Leistungssteigerungen erzielen.

Im CCCS-Vortrag vom 18.Juni.2009 wird an einem real existierenden Beispiel gezeigt, wie mit Hilfe des  Openourceservers *Squid* ein 'Reverse Proxy' konfiguriert wird und welche Leistungswerte erreichbar sind.

Referenten: E.R. Sexauer, P. Tomko

Email: Cccs-Liste, Subject: Reverse Proxy

## 1 Was ist ein Reverse Proxy?

Terminologie: Http-Accelerator, surrogate Proxy

Proxy: Browser  $\rightarrow$  *PROXY*  $\Rightarrow$  Internet  $\rightarrow$  Webserver

Reverse Proxy: Browser  $\Rightarrow$  Internet  $\rightarrow$  *PROXY*  $\rightarrow$  Webserver

## 2 Warum Reverse Proxy?

### 2.1 Entlastung des Webservers

Ein Proxy ist ein einfacheres Programm als ein Webserver. Er kann einmal erzeugte Webseiten schneller ausliefern als der Webserver. Da sich auch bei dynamischen Webseiten viele Elemente ständig wiederholen, kann die Auslieferung

dieser Elemente aus dem Cache des Proyservers eine erhebliche Beschleunigung bewirken. Wikipedia z.B. gibt an, dass rund 75% aller Zugriffe direkt aus den Proxyservern bedient werden.

### 2.1.1 Sicherheit

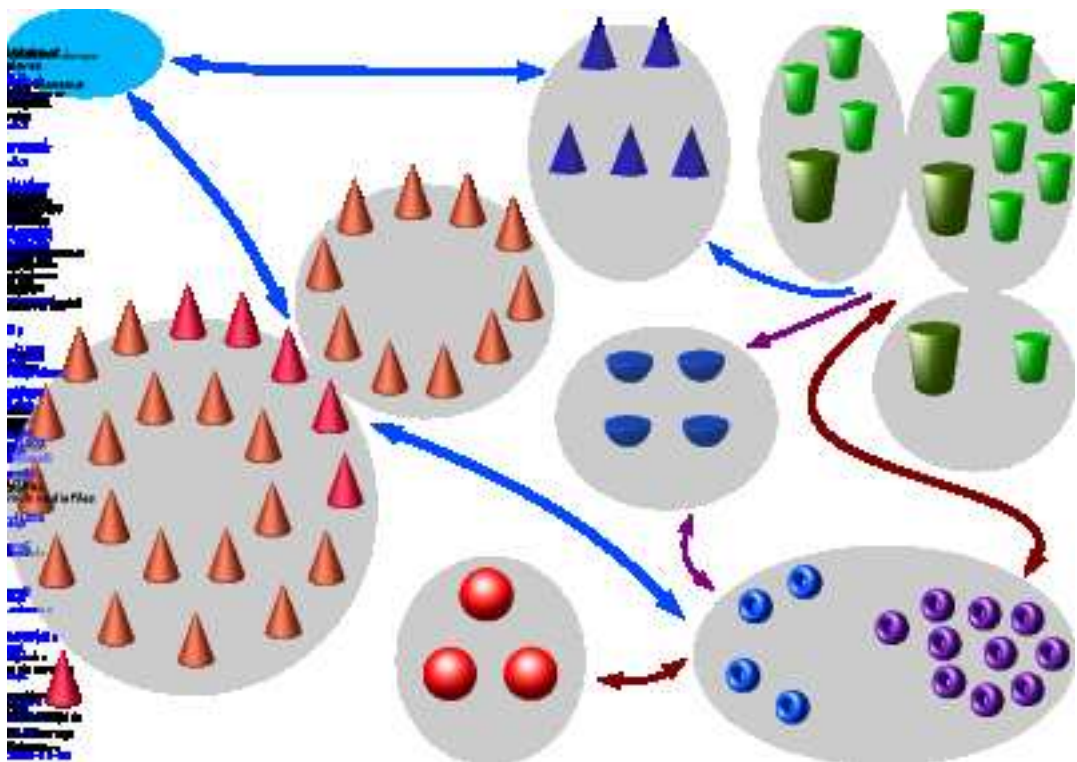
Nach aussen ist der Proxy die Schnittstelle zum Internet; der eigentliche Webserver ist unsichtbar. Durch Zugriffsregeln können unerwünschte Zugriffe - z.B. vergiftete URL's - im Vorfeld abgeblockt werden.

Wenn der Webserver eine öffentliche IP hat, sollte diese per Firewall gegen Fremdzugriff gesperrt werden:

```
- iptables -j INPUT -a ACCEPT -p tcp -s proxyserver -dport 80  
- iptables -j INPUT -a DROP -p tcp -dport 80
```

### 2.1.2 Lastverteilung

Proxyserver ermöglichen auf einfache Weise die Verteilung der Arbeit auf mehrere Rechner. Beispiele finden sich bei Wikipedia.



## 3 Drei Beispiele

- Erdbebenwarte Pasadena, <http://pasadena.wr.usgs.gov/office/stans/slashdot.html>

- Wikipedia, [http://meta.wikimedia.org/wiki/Wikimedia\\_servers](http://meta.wikimedia.org/wiki/Wikimedia_servers)
- NA (Neue Arbeit), <http://neuearbeit.de>

## 4 Konfiguration

Die Konfigurationsdatei ist `/etc/squid.conf` (u.U. `squid3.conf`)

### 4.1 Webhost

```
- cache_peer master.neuearbeit.org parent 80 0 no-query no-digest originserver
login=PASS name=nanormal
```

Der Webhost liegt normalerweise auf einer anderen Maschine. Es ist aber auch möglich, Squid und den Webserver auf der gleichen Maschine laufen zu lassen. In diesem Falle sollten die Datenverzeichnisse von Squid und dem Webserver möglichst auf unterschiedlichen Platten liegen.

### 4.2 Netze und Ports

```
- http_port 80 vhost (Port, auf das Squid von aussen hört)
- acl mynet 172.16.1.0/24, für einen öffentlichen Server 0.0.0.0/0
```

### 4.3 Verzeichnisse

Die Verzeichnisse von Squid können und sollten auf das Attribut 'access-time' verzichten; dies erhöht die Leistung beträchtlich. ('noatime' beim Mounten der Partition oder 'chattr -RV +A' bei ext2/3)

#### 4.3.1 Logfiles

```
access_log /usr/local/squid/var/logs/access.log common
cache_log /usr/local/squid/var/logs/cache.log (Systemereignisse)
cache_store_log none (nützlich für Testzwecke)
# logfile_rotate 10
```

#### 4.3.2 Cachefiles

```
cache_dir ufs /usr/local/squid/var/cache 2000 16 256
```

*Hinweis:* Der Cache sollte nicht grösser sein als notwendig - Suchzeit!

## 4.4 ACL's (Access Control List)

ACL's steuern die Zugriffsrechte auf Squid. Die allgemeine Form ist:

```
- acl aclname acltype argument
```

Mögliche Typen sind u.a.: Source, Destination, Methode, IP, Port und Zeit. Reguläre Ausdrücke für Domain's und Url's werden unterstützt.

Beispiel:

```
- acl myacl src 172.16.1.0/24
- http_access myacl allow
```

## 4.5 Redirector (URL-Rewriter)

Der Redirector kann verwendet werden, um Requests zu überprüfen und ggf. umzuleiten. Z.B. kann der Zugriff auf ein internes Wiki nur erlaubt werden, wenn die anfordernde IP aus dem eigenen Netz kommt.

```
- url_rewrite_program /usr/local/squid/etc/squid-redir.pl
- url_rewrite_children 60
- acl REDIRAKDIAK dstdomain ak-diak.neuearbeit.de
- redirector_access allow REDIRAKDIAK
```

Siehe: <http://www.comfsm.fm/computing/squid/FAQ-15.html>

## 4.6 Open Proxy verhindern

Per Default bearbeitet Squid alle einkommenden Anforderungen. Um Mißbrauch als offener Proxy zu verhindern, sollte der Zugriff auf die eigenen Zieldomains eingeschränkt werden.

```
- acl hetznergw dst 78.46.108.164/32 (eigene IP = IP der eigenen Domains)
- http_access allow hetznergw
- http_access deny all
```

Beispiel:

```
222.145.191.21 -- [02/Apr/2009:13:16:44 +0200] "GET http://plage.cool.ne.jp/ranking/ranklink.cgi?
HTTP/1.0" 403 1410 TCP_DENIED:NONE
```

# 5 Proxy und SSL

## 5.1 Warum SSL-Proxy?

Der Proxy kann die Ver- und Entschlüsselung für den Webserver erledigen. Proxy und Webserver verkehren miteinander im Klartext. Vorteile sind:

- Entlastung des Webservers
- Elemente im Klartext können gecached werden, verschlüsselte Elemente nicht.

## 5.2 Konfiguration

Squid muss für diese Option kompiliert werden.

Es kann das gleiche Zertifikat wie für den Webserver verwendet werden. Selbsterzeugte Zertifikate sind möglich, werden aber vom Browser der Clients als unbekannt ausgewiesen. Um den unverschlüsselten Zugriff auf Https-Seiten zu unterbinden, ist eine Zugriffsregel erforderlich.

```
- https_port 443 key=/usr/local/squid/etc/unencrypted_key_neuearbeit.pem
cert=/usr/local/squid/etc/neuearbeit.crt vhost version=1
- acl qbank url_regex ^http://qbank.* (Acl für unverschlüsselten Zugriff)
- http_access deny qbank (Zugriff verbieten)
```

## 5.3 Alternativen:

Pound, <http://www.apsis.ch/pound/>. Ein schmaler, effizienter und lizenzfreier Proxy für SSL und Lastverteilung. Kein Caching.

Crossroads, <http://directory.fsf.org/project/crossroadsloadbalancer/>. Wie Pound.

### Anmerkung:

Im Gegensatz zum kernelbasierten Loadbalancing LVS, <http://www.linuxvirtualserver.org/software/index.htm> berücksichtigen die genannten Proxyserver die Protokolle Http/Https. In der Doku von Crossroads findet sich ein Leistungsvergleich; LVS ist dort nur 10% schneller.

# 6 Caching

## 6.1 Statische und Dynamische Seiten

Obwohl die meisten Webseiten heute technisch gesehen dynamisch sind, enthalten sie doch viele Elemente, die sich selten oder nie ändern - z.B. Bilder oder Textdokumente. Es gibt mehrere Methoden, diese Elemente cachebar zu machen.

## 6.2 Http-Header

Der Http-Header kann spezifizieren, ob und wie lange ein Element cachebar ist. Verfügbare Attribute sind:

- last modified, expires, pragma no-cache [Http-1.0] und cache-control [no-cache, public, private]

Leider unterstützen gängige Html-Generatoren nicht das Setzen dieser Attribute.

### 6.2.1 Php.ini

Bei Php können globale Parameter definiert werden, die von Php selbst ausgewertet werden:

- no-cache: Erzeuge keine cachebaren Inhalte.
- public: Erzeuge Inhalte, die für jeden Client und Proxy cachebar sind
- private: Erzeuge Inhalte, die für jeden Client cachebar sind.

Literatur (PHP): George Schlossnagel, Professionelle PHP5-Programmierung, Addison-Wesley

## 6.3 Dateiendungen (override expire)

Squid erlaubt es, Attribute des Http-Headers zu überschreiben oder zu setzen. Beispiel:

- refresh\_pattern -i .jpg 1440 20% 990080 override-expire ignore-no-cache  
setzt die Lebensdauer für 'jpg' auf 990080 Sekunden. "-i" ignoriert GroSS/Kleinschreibung.

## 7 Einige Ergebnisse

Gesamtzugriffe: 2654550, Hits: 2429898 = 92%

Logins: 16633

Zugriffe Qbank (reines PHP/MySQL, nur Https): 1903712, Hits Qbank: 1857625 = 94%

Zugriffe Joomla-CMS: 265894, Hits Joomla: 205965 = 75%

Zugriffe Egroupware (PHP/MySQL, nur Https): 344179, Hits Egroupware: 292557 = 85%

## 8 Logging oder 'Wer misst, misst Mist

Wenn ein Proxy vor dem Webserver steht, werden die Zugriffe, die der Proxy direkt bearbeitet, im Logfile des Webserver nicht aufgezeichnet. Eine vollständige

Statistik muSS daher die Logfiles des Proxyservers auswerten. Um Belastung der Server zu untersuchen, werden beide Logfiles benötigt.

Weitere Tools:

- Webalizer liefert ein grobes Profil der zeitlichen Auslastung
- Mrtg, <http://www.cyberciti.biz/nixcraft/linux/docs/uniqlinuxfeatures/mrtg/>, ermöglicht eine genaue zeitliche Aufzeichnung der Systembelastung
- Mysql, `log_slow_queries`, ermöglicht die Aufzeichnung von Queries, die eine einstellbare Zeitdauer überschreiten.

## 8.1 Logfiles

Squid kann Logfiles im Apacheformat (Http) oder im eigenen Format erzeugen. Es ist möglich, individuelle Formate zu definieren. Beispiel:

```
- logformat common %>a %ui %un [%t] "%rm %ru HTTP/%rv" %Hs %<st  
%Ss:%Sh. "rm" z.B. bedeutet Request-Methode.
```

## 8.2 Squidlog versus Weblog

```
emulate_htdocs_log on (on=Http/ Apache, off=Squid)
```

## 8.3 Tools:

Webalizer (Http), Calamaris (Squid) und eigene Skripts.

# 9 Ausblick, Lastverteilung

Da die Leistung eines Servers, insbesondere der Platten, auch mit viel Geld nicht beliebig gesteigert werden kann, ist es bei hoher Last zwingend, die Arbeit auf mehrere Rechner zu verteilen.

## 9.1 Nach aussen via DNS

In dieser Konstellation hat eine Domain mehrere IP-Adressen, die von DNS abwechselnd geliefert werden. Beispiel:

```
- neuarbeit.de. 3591 IN A 85.214.133.148
```

```
- neuarbeit.de. 3591 IN A 78.46.108.164
```

Auf jeder dieser IP's läuft ein Proxy mit dem gleichen Parent, dem Webserver. Die Zugriffe von auSSen werden statistisch gleichmäSSig auf die 2 Proxyserver verteilt.

Die TTL (hier 3600 Sekunden) steuert, wie lange die IP eines ausgefallenen Servers schlimmstenfalls noch verwendet werden darf.

Es empfiehlt sich, die Logfiles mit Syslog-NG auf einen remote Logserver zu spielen, um zeitfolgerichtige Zugriffsprotokolle zu erhalten.

Zur Erkennung von Ausfällen gibt es zwei Möglichkeiten:

- Ein Skript auf den DNS-Servern überprüft die Verfügbarkeit der Proxyserver und modifiziert die Zonendateien entsprechend
- DNS selbst läuft auf den Proxyservern und liefert jeweils nur die eigene IP.

## 9.2 Nach innen mit mehreren Parents (Session, Cluster)

Squid erlaubt es, mehrere Parents zu definieren. Der optionale Parameter "weight" erlaubt eine Gewichtung. Squid erkennt automatisch, wenn ein Parent ausfällt oder wieder verfügbar wird. Diese Ereignisse werden in "cache.log" aufgezeichnet.

Wenn mehrere Webserver eingesetzt werden, müssen diese synchronisiert werden. Bei Anwendungen, die Transaktionen ausführen, muSS zusätzlich sichergestellt werden, dass eine Session auf dem ursprünglich gewählten Server verbleibt.

### 9.2.1 Tools zu Synchronisation

- Copy
- Rsync
- Transactionlog (MySQL)
- Mysql-Cluster ([http://de.wikipedia.org/wiki/MySQL\\_Cluster](http://de.wikipedia.org/wiki/MySQL_Cluster))

### 9.2.2 Tools zur Verteilung von Zugriffen

- ACL's innerhalb von Squid
  - `cache_peer qbankmaster.neuearbeit.org parent 90 0 no-query no-digest originserver login=PASS`
  - `cache_peer_domain qbanksrv qbank.neuearbeit.de`
- Mysql-Proxy, [http://forge.mysql.com/wiki/MySQL\\_Proxy](http://forge.mysql.com/wiki/MySQL_Proxy). Dieser Proxy wird vor den (oder die) Mysql-Server geschaltet und erlaubt u.a:
  - Trennen der Zugriffe nach Datenbanken
  - Trennen der Lese- und Schreibzugriffe.

### **9.2.3 Sessionmanagement**

Squid3.0 hat keine Möglichkeit, eine Session zu erkennen und auf einem Parent zu halten. Durch Zwischenschalten beispielsweise eines Pound-Proxy können Sessions anhand von Cookies oder Html-Parametern erkannt und festgehalten werden. Bei Mysql-Cluster erfolgt das Sessionmanagement innerhalb des Clusters.

# Inhaltsverzeichnis

<b>1</b>	<b>Was ist ein Reverse Proxy?</b>	<b>1</b>
<b>2</b>	<b>Warum Reverse Proxy?</b>	<b>1</b>
2.1	Entlastung des Webservers . . . . .	1
2.1.1	Sicherheit . . . . .	2
2.1.2	Lastverteilung . . . . .	2
<b>3</b>	<b>Drei Beispiele</b>	<b>2</b>
<b>4</b>	<b>Konfiguration</b>	<b>3</b>
4.1	Webhost . . . . .	3
4.2	Netze und Ports . . . . .	3
4.3	Verzeichnisse . . . . .	3
4.3.1	Logfiles . . . . .	3
4.3.2	Cachefiles . . . . .	3
4.4	ACL's (Access Control List) . . . . .	4
4.5	Redirector (URL-Rewriter) . . . . .	4
4.6	Open Proxy verhindern . . . . .	4
<b>5</b>	<b>Proxy und SSL</b>	<b>4</b>
5.1	Warum SSL-Proxy? . . . . .	4
5.2	Konfiguration . . . . .	5
5.3	Alternativen: . . . . .	5
<b>6</b>	<b>Caching</b>	<b>5</b>
6.1	Statische und Dynamische Seiten . . . . .	5
6.2	Http-Header . . . . .	6
6.2.1	Php.ini . . . . .	6
6.3	Dateiendungen (override expire) . . . . .	6
<b>7</b>	<b>Einige Ergebnisse</b>	<b>6</b>
<b>8</b>	<b>Logging oder 'Wer misst, misst Mist</b>	<b>6</b>
8.1	Logfiles . . . . .	7
8.2	Squidlog versus Weblog . . . . .	7
8.3	Tools: . . . . .	7

<b>9</b>	<b>Ausblick, Lastverteilung</b>	<b>7</b>
9.1	Nach aussen via DNS . . . . .	7
9.2	Nach innen mit mehreren Parents (Session, Cluster) . . . . .	8
9.2.1	Tools zu Synchronisation . . . . .	8
9.2.2	Tools zur Verteilung von Zugriffen . . . . .	8
9.2.3	Sessionmanagement . . . . .	9