Traffic Balancing

Philipp Gühring <p.guehring@futureware.at>

27. September 2000

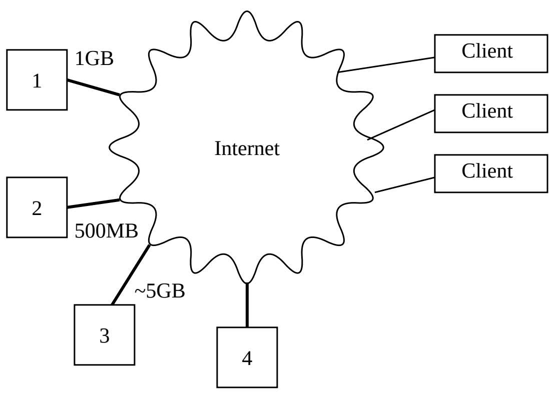
Zusammenfassung

Der erste Teil dieses Artikels beschäftigt sich mit dem Hintergrund des Traffic Balancing, zeigt die vorhandenen Probleme und deren Lösungen auf. Der zweite Teil des Artikels erklärt Details der praktischen Umsetzung von Traffic Balancing bei Futureware. Der dritte Teil soll als Anleitung für Administratoren dienen, wie weitere Server in das System eingebunden werden müssen, und wie diese Server konfiguriert werden müssen.

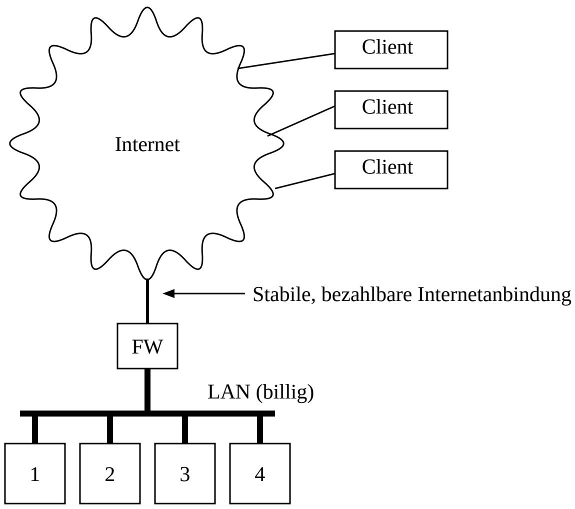
1 Hintergrund des Traffic Balancing

Wo bekommt man 10-20 GB Traffic pro Monat für eine Website gratis? (September 2000)

Nun, bei den meisten Servern, die man gratis auftreiben kann, bekommt man 500 MB - 2 GB Traffic pro Monat gratis, Stichwort "Fair-use". Wenn man 10-20 von diesen Rechnern bekommt, müßte das ganze gehen. Doch wie balanciert man den Traffic richtig aus?



Da man Traffic Balancing leicht mit Load Balancing verwechseln kann, hier das Diagramm für Load Balancing:



Beim Load Balancing geht es darum, einen Server-Cluster, der über ein billiges (Traffic kostet nichts) LAN verbunden ist, die Anfragen gleichmäßig auf die Server zu verteilen. Die Bezahlung der Internet Anbindung ist hier meist nicht das Problem. Die Server können hier im Sekundentakt Statusinformationen untereinander verbreiten.

Beim Traffic Balancing hingegen spielt der Systemload eine untergeordnete Rolle, Statusinformationen können bestenfalls alle 5 Minuten ausgetauscht werden. Nun, die einfachste Lösung, die Server gleichmäßig zu belasten wäre DNS Round-Robin.

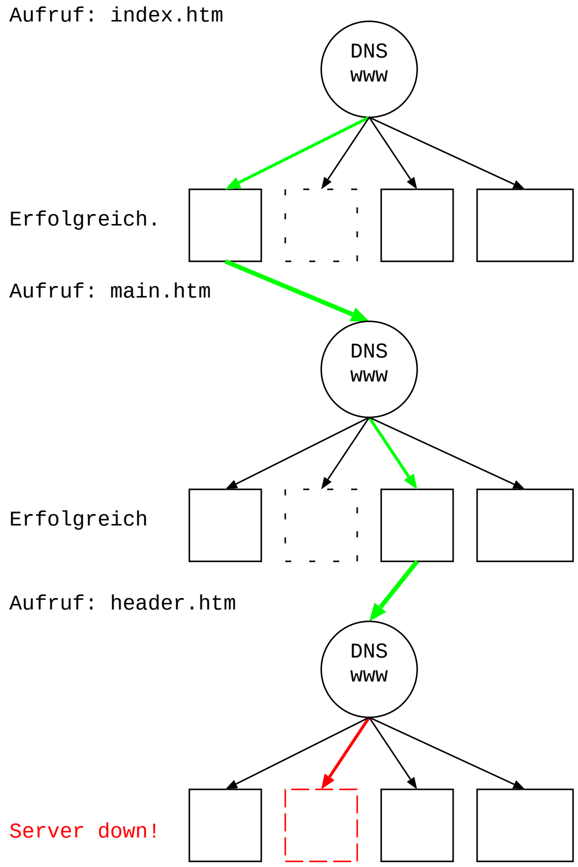
Doch mit der Zeit erkennt man die Probleme, die im laufenden Betrieb entstehen:

1.1 Serverausfälle

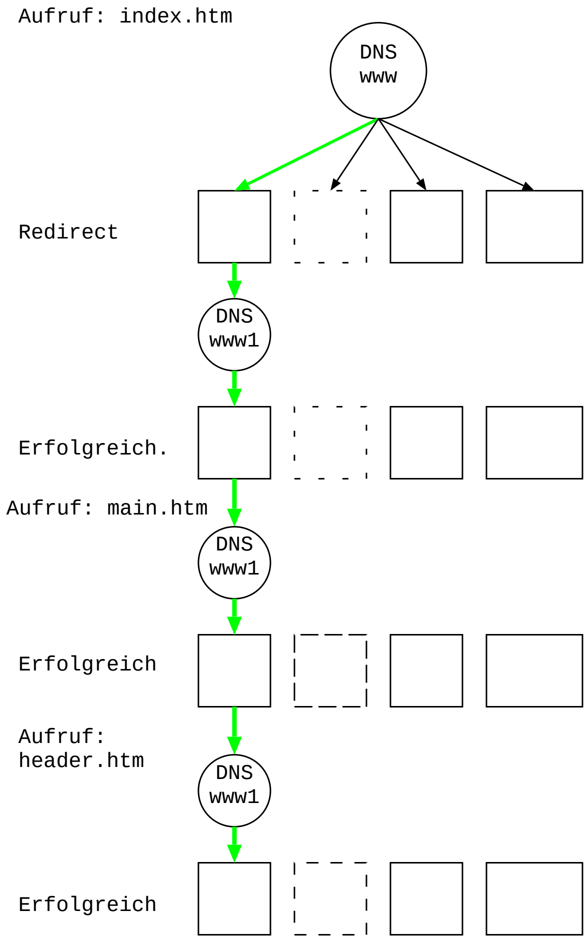
Einzelne Server könnten ausfallen. Da man ja keine administrative Kontrolle über die Server hat (und es vielleicht auch geographisch unmöglich ist), kann man die Probleme oft auch nicht lösen, und nur darauf warten, daß sie gelöst werden. Als Problemlösung bietet sich hier an, die Server automatisch im DNS auszublenden, sobald man erkennt, daß die Server nicht richtig laufen.

Die erste Idee wäre, jede Sekunde die DNS Server alle anderen Webserver abfragen zu lassen. Allerdings würde dies eine enorme Menge an Traffic produzieren, und den wollen wir vermeiden. Wir haben uns daher entschieden, nur jede Stunde, vielleicht jede halbe Stunde diese Aktualisierungen vorzunehmen.

Bei Websites, die mehr als eine (generierte) HTML Seite beinhalten, kommt dann auch noch folgendes Problem dazu: Jede Abfrage, ob nach HTML Seiten oder nach Grafiken, verwendet eine eigene TCP/IP Verbindung, und bekommt daher unterschiedliche IP Adressen. Sehen wir uns mal ein Beispiel an, mit einer Website mit 3 verlinkten Seiten, die der Benutzer der Reihe nach anschauen will:



Wenn man zum 4 Server hat, und eine HTML Seite, mit einer Vielzahl von Bildern darauf, und einer der Server ausfällt, wird man nur 3/4 der Bilder sehen. Die Lösung für dieses Problem sind Redirects.



Die Gesamtlösung des Problems besteht kurzfristig aus Redirects, um “halbe” Seiten zu vermeiden, und ausblenden im DNS langfristig, um Serverausfälle, die sich länger hinziehen ohne größerer Probleme zu überstehen.

1.2 Traffic Limits

Das zweite Problem ist, daß die Server Betreiber aus verschiedenen Gründen (Traffic Limit überschritten, braucht die Bandbreite kurzfristig für anderes, ...) den Traffic einschränken wollen. Wenn man ihnen keine saubere Art gibt, dies zu tun, werden sie einfach den Webserver abdrehen, und die Website fällt teilweise aus. Zwei verschiedene Lösungen hätte ich für dieses Problem gesehen: Eine Weboberfläche mit einem Formular, einer Tabelle aller Server, mit der der Admin durch ein paar Mausklicks den DNS-Server steuern kann. Allerdings hätten wir da erst die Oberfläche entwickeln müssen, und vor allem bräuchten wir einen Webserver auf jedem DNS Server, was sicherheitstechnisch bedenklich ist. Die andere Möglichkeit ist, die Existenz und den

Inhalt einer Datei oder eines Verzeichnisses per HTTP von den Webservern abzufragen. In der Datei (traffic.dat) werden alle generischen Hostnamen eingetragen, auf die der Server eingetragen werden will. Wenn die Webserver komplett ausfallen, die Datei gelöscht, umbenannt, oder mit *chmod 000 <Datei>* unlesbar gemacht wird. Interessant wäre noch ein Mechanismus, mit dem der Provider den Traffic für den Rest des Monats sperren kann. Dies ließe sich über einen cron Job erledigen (Jeden Monatsersten *chmod 644 traffic.dat)*

2 Umsetzung

Die Grundidee hinter Traffic Balancing ist die, daß der DNS-Server regelmäßig (wahrscheinlich jede Stunde) alle Server abfragt, welche Dienste sie anbieten wollen. Wenn diese Abfrage fehl schlägt, weil der Server außer Betrieb ist, oder der Administrator die Dienste deaktiviert hat, wird der Server automatisch aus dem DNS ausgeblendet.

Um das ganze System so einfach wie möglich zu gestalten, nahmen wir möglichst viele vorhandene Software Komponenten. Bind als Domainname-Server, Perl und Bash als Programmier- und Scriptsprachen.

Eine der offenen Fragen war, wie man einem Domainnameserver im laufenden Betrieb neue Daten einspielen kann, ohne den Betrieb zu stören. Bei dem Domainname Server Bind sind dynamische DNS Updates mittels *nsupdate* die Lösung für dieses Problem.

Bei Diensten, die nur einmal abgefragt werden, oder nur eine TCP/IP Verbindung lange offenbleibt, kann der Redirect auch weggelassen werden.

Das System wurde auf zwei Teile aufgeteilt. Die Basis-Einträge, und die dynamischen Einträge. Die Basis Einträge erledigt das *base* Script, das nur aufgerufen werden muß, wenn neue Server/Services hinzukommen, oder sich IP Adressen von Servern ändern. Das *dyn* Script sollte als cronjob aufgerufen werden. Die Refreshrate sollte zwischen 10 Minunten und 12 Stunden liegen, wir empfehlen eine Stunde. Dies ist abhängig von den Traffickosten und der gewünschten Zeit, bis ein Server ausgeblendet wird, der down ist.

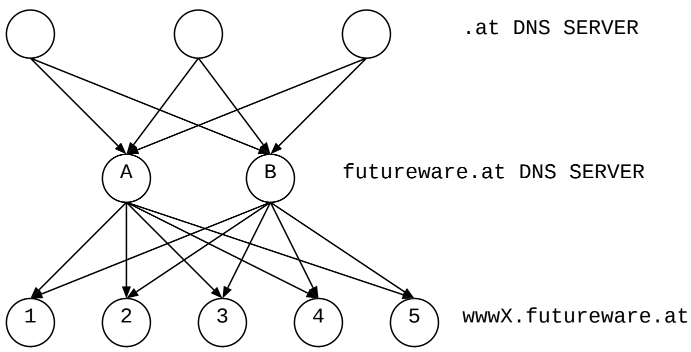
Das *base* Script erstellt die folgende Tabelle, und trägt die Zuordnungen von Hostnamen zu IP Adressen in den Domainname Server ein:

Nr.	IP	WWW	Eigentümer	www	quick	ftp	Hostname
1	62.116.5.149	www1.futureware.at	Globesoft	X	X	X	www.globesoft.at
2	195.26.213.73	www2.futureware.at	Papa.at	X	?	-	tux.bhfnet.at
3	62.116.11.3	www3.futureware.at	Enemy.org	X	?	-	www.enemy.org
4	-	www4.futureware.at	Sourceforge	?	-	-	
5	-	www5.futureware.at	Xoom	?	-	-	members.xoom.com
6	212.69.165.14	www6.futureware.at	Inca	?	X	-	bootnix.chello.or.at
7		www7.futureware.at	Inca	?	X	-	ftp.inca.at
8		www8.futureware.at	Inode	?	X	-	motorradseiten.at
9		www9.futureware.at		?	X	-	
10		www10.futureware.at	Inca	?	X	-	

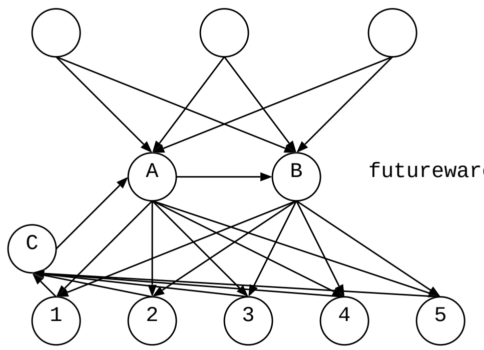
Das *dyn* Script fragt die Rechner über die bereits eingetragenen www Hostnamen ab (traffic.dat), welche Dienste zur Verfügung gestellt werden sollen, und trägt dann die Dienste mit allen Rechnern, die diese anbieten im DNS ein.

2.1 Single Point of Failure

Um den gefürchteten Single-Point of Failure zu vermeiden, kann man folgende Struktur aufziehen:



Wir haben bei Futureware zur Zeit DNS Server B als Secondary DNS für Server A laufen, und die Aktualisierung der DNS Daten passiert über dynamisches DNS von einem unabhängigen Rechner C:



Diese Methode hat den Vorteil, daß man keinen besonderen Zugriff, keine neuen Services auf den DNS Servern benötigt, nur die Konfiguration für Dynamische DNS Updates ist notwendig. Der Nachteil ist, daß bei einem Ausfall von C oder A die DNS Daten veraltern, und dann Ausfälle der Server nicht mehr zeitgerecht oder gar nicht mehr ausgeblendet werden.

Die Alternative wäre, A und B beide Master Nameserver für die Zone spielen lassen, und auf beiden Rechnern das *dyn* Skript laufen lassen.

3 Anleitung zum Anlegen eines Mirrors

1. Der Serverbetreiber gibt die IP Adresse des Servers bekannt.

2. Der Admin vergibt eine neue, eindeutige Servernummer (<*NR*>), und gibt diese dem Serverbetreiber.

3. Der Serverbetreiber richtet nun einen Webserver ein, der www<*NR*>.domain.name hostet. (zum Beispiel www6.futureware.at)

4. Nun werden die Server für alle weiteren benötigten Dienste eingerichtet, und denen die entsprechenden Namen zugeordnet (quick6.futureware.at)

5. Zusätzlich zu den spezifischen Hostnamen (quick6.futureware.at) müssen die Dienste auch auf die generischen Hostnamen (quick.futureware.at) hören. Dies sollte im Falle von Webservern mittels Redirect geschehen, um die Clients an den Server zu binden.

6. Bei Webservern gilt dir Grundregel, nicht nur http://www.domain.name/ , sondern auch http://domain.name/ gemeinsam zu bedienen.

7. Wenn alle Services richtig laufen und getestet wurden, ist die Datei "traffic.dat" zu erstellen, die dann unter http://www<*NR*>.domain.name/traffic.dat abrufbar ist. (Bsp.: http://www1.futureware.at/traffic.dat)

8. In der traffic.dat werden nun zeilenweise alle generischen Hostnamen eingetragen, auf die der Server hören will:

http://www1.futureware.at/traffic.dat:

```
www.futureware.at
futureware.at
ftp.futureware.at
```

4 Beispielkonfiguration

4.1 DNS: Bind

Die dynamischen DNS Updates können bei Bind8 mit folgenden Einstellungen in der /etc/named.conf aktiviert werden. Die IP Adresse 127.0.0.1 ist dann durch die jeweilige IP Adresse des Rechners zu ersetzen, auf dem die Traffic Balancing Scripte laufen, wenn dies nicht der DNS Server selber ist.

```
zone "futureware.at" IN {
    type master;
    file "futureware.at";
    check-names fail;
    allow-update { 127.0.0.1; };
};
```

4.2 Dienste bei Futureware

4.2.1 www

Dies ist die Website von Futureware 2001. Sie umfaßt ca. 10-20 MB an HTML Seiten und Bildern. Der generische Hostname dafür ist www.futureware.at und futureware.at. Für diesen Dienst muß ein Redirect von http://www.futureware.at/ und http://futureware.at/ auf http://www<NR>.futureware.at/ eingerichtet werden, und http://www<NR>.futureware.at/ muß auf ein per FTP verfügbares Verzeichnis zeigen. Nach Möglichkeit bitte auch SSH Zugang ermöglichen.

4.2.2 ftp

Dies ist unser interner FTP Server, um intern Daten auszutauschen.

4.2.3 download

Dies sind die Programm und Datenpakete, zur Zeit ca. 100 MB

Früher hatten wir www und packages getrennt, da wir einerseits Fileserver wie Xoom hatten, die 500 MB Daten unterbrachten, aber auf alle HTML Seiten und Grafiken Banner gegeben haben, andererseits Webserver mit 5 - 20 MB Kapazität. Heute haben wir fast ausschließlich Server mit 200 MB Datenkapazität und Unterstützung für eigene CGI Programme. Sollte es wieder in die andere Richtung gehen, werden wir vielleicht beides wieder trennen

4.2.4 quick

Dieses ist die Online Version von *Quick, ein CGI Programm. Es ist zur Zeit unser einzigster Bedarfd an CGI Programmen. Platzbedarf: Zwischen 20 und 150 MB, je nachdem, wieviele Datenbanken installiert werden. Bei einem Linux Server sind zur Zeit die QT und KDE Bibliotheken erforderlich. Der generische Hostname ist quick.futureware.at. Der Webserver muß http://quick.futureware.at/cgi-bin/quick?... und http://quick<NR>.futureware.at/cgi-bin/quick? bedienen können, ein Redirect von quick auf quick<NR> ist nicht nötig, da dies keine Ausfall-Sicherheit bringt, und nur die Abfragen verzögert. Die traffic.dat bitte nicht zu vergessen.

4.2.5 traffic

Die traffic.dat wird über den www Dienst übertragen, und muß auf allen Servern egal, welche Dienste sie anbeiten abrufbar sein. Der Zugriff erfolgt über http://www<NR>.futureware.at/traffic.dat

4.3 Webserver: Apache

```
<VirtualHost 62.116.5.149>
ServerName www.futureware.at
Redirect / http://www1.futureware.at/
ErrorLog /home/futureware/error_log
TransferLog /home/futureware/access_log
</VirtualHost>

<VirtualHost 62.116.5.149>
```

```
ServerName futureware.at
Redirect / http://www1.futureware.at/
ErrorLog /home/futureware/error_log
TransferLog /home/futureware/access_log
</VirtualHost>

<VirtualHost 62.116.5.149>
DocumentRoot /home/futureware/html/
ServerName futureware.globesoft.at
ErrorLog /home/futureware/error_log
TransferLog /home/futureware/access_log
ScriptAlias /cgi-bin/ /home/futureware/cgi-bin/
ServerAdmin webmaster@futureware.at
</VirtualHost>

<VirtualHost 62.116.5.149>
DocumentRoot /home/futureware/html/
ServerName www1.futureware.at
ErrorLog /home/futureware/error_log
TransferLog /home/futureware/access_log
ScriptAlias /cgi-bin/ /home/futureware/cgi-bin/
ServerAdmin webmaster@futureware.at
</VirtualHost>

<VirtualHost 62.116.5.149>
DocumentRoot /home/futureware/html/
ServerName quick1.futureware.at
ErrorLog /home/futureware/error_log
TransferLog /home/futureware/access_log
ScriptAlias /cgi-bin/ /home/futureware/cgi-bin/
ServerAdmin webmaster@futureware.at
```

```
</VirtualHost>

<VirtualHost 62.116.5.149>
DocumentRoot /home/futureware/html/
ServerName quick.futureware.at
ErrorLog /home/futureware/error_log
TransferLog /home/futureware/access_log
ScriptAlias /cgi-bin/ /home/futureware/cgi-bin/
ServerAdmin webmaster@futureware.at
</VirtualHost>
```