Proxy-Server: Squid

Im folgenden Kapitel wird erläutert, wie das Caching von Webseiten mit Hilfe eines Proxy-Servers funktioniert und welchen Nutzen Squid für Ihr System bietet.

Squid ist der am weitesten verbreitete Proxy-Cache tür Linux/UNIX-Plattformen. Wir werden beschreiben, wie er zu konfigurieren ist, welche Systemanforderungen bestehen, wie das eigene System konfiguriert sein muss, um transparentes Proxying durchzuführen, wie man Statistiken über den Nutzen des Cache mithilfe von Programmen wie Calamaris und cachemgr erhält oder wie man Web-Inhalte mit squidgrd filtriert.

7.1 Was ist ein Proxy_Cache?

Squid fungiert als Proxy-Cache. Es verhält sich wie ein Makler, der Anfrragen von Clients erhält (in diesem Fall Web-Browser) und an den zuständigen Server-Provider weiterleitet. Wenn die angeforderten Objekte beim Vermittler ankommen, behält er eine Kopie davon in einem Festplatten-Cache.
Der Vorteil zeigt sich, wenn mehrere Clients dasselbe Objekt anfordern. Sie können nun direkt aus dem Festplatten-Cache bedient werden, also wesentlich schneller als aus dem Internet. Dies spart gleichzeitig eine Menge Systembandbreite.

Tipp
Squid bietet ein großes Spektrum an Features, z. B. die Festlegung von Hierarchien für die Proxy-Server zum Verteilen der Systemlast, Aufstellen fester Zugriffsregeln an alle Clients, die auf den Proxy zugreifen wollen, Erteilen oder Verweigern von Zugriffsrechten auf bestimmte Webseiten mit Hilfe anderer Applikationen oder die Ausgabe von Statistiken der meistbesuchten Webseiten, wie z. B. das Surfverhalten der Benutzer u. v. m.

Squid ist kein generischer Proxy. Normalerweise vermittelt er nur zwischen HTTP-Verbindungen. Außerdem unterstützt er die Protokolle FTP, Gopher, SSL und WAIS, jedoch keine anderen Internet-Protokolle wie Real Audio, News oder Videokonferenzen. Squid greift auf das UDP-Protokoll nur zur Unterstützung der Kommunikation zwischen verschiedenen Caches zurück. Aus diesem Grund werden auch andere Multimedia-Programme nicht unterstützt.

7.2 Informationen zu Proxy_Cache

7.2.1 Squid und Sicherheit

Man kann Squid zusammen mit einer Firewall verwenden, um interne Netzwerke durch den Einsatz von Proxy-Cache nach außen zu schützen. Die Firewall verweigert mit Ausnahme von Squid alle externen Dienste, alle WWW-Verbindungen müssen durch den Proxy aufgebaut werden.

Im Falle einer Firewall-Konfiguration mit einem DMZ würden wir dort unseren Proxy setzen. In diesem Fall ist es wichtig, dass alle Rechner im DMZ ihre Protokolldateien an Rechner innerhalb des gesicherten Netzwerks senden.

Ein Möglichkeit der Implementierung dieser Features mit Hilfe eines so genannten "transparenten" Proxy wird in Abschnitt 7.6 behandelt.

7.2.2 Mehrere Caches

Man kann mehrere Caches so konfigurieren, dass Objekte zwischen ihnen ausgetauscht werden können, um die Systemlast zu reduzieren und die Möglichkeit zu steigern, ein bereits im lokalen Netzwerk vorhandenes Objekt zu finden. Möglich sind auch Cache-Hierarchien, so dass ein Cache in der Lage ist, Objektanfragen an Caches der gleichen Hierarchie weiterzuleiten oder einen übergeordneten Cache zu veranlassen, die Objekte von einem anderen Cache im lokalen Netzwerk oder direkt aus der Quelle herunterzuladen.

Die Wahl der richtigen Topologie fur die Cache-Hierarchie ist sehr wichtig, da Netzwerkverkehr insgesamt nicht erhöht werden soll. In einem großen Netzwerk z.B. ist es möglich, für jedes Subnetz einen Proxy-Server zu konfigurieren und diesen dann mit einem übergeordneten Proxy zu verbinden, der wiederum an den Proxy-Cache vom ISP angeschlossen wird.

Die gesamte Kommunikation wird vom ICP (*engl. Internet Cache Protocol*) gesteuert, das aufdem UDP-Protokoll aufgesetzt ist. Der Datenaustausch zwischen Caches geschieht mittels HTTP (*engl. Hyper Text Transmission Protocol*) basierend auf TCP. Allerdings sollten für solche Verbindungen schnellere und einfachere Protokolle verwendet werden, die innerhalb von maximal einer oder zwei Sekunden auf eingehende Anfragen reagieren können.

Um den besten Server für die gewünschten Objekte zu finden, schickt ein Cache an alle Proxies der gleichen Hierarchie eine ICP-Anfrage. Die Proxies werden mittels ICP-Antworten mit dem Code "HIT" auf die Anfragen reagieren, falls das Objekt gefunden wurde oder, falls nicht, mit dem Code "MISS". Im Falle mehrerer HIT-Antworten wird der Proxy-Server einen Server fur das Herunterladen bestimmen. Diese Entscheidung wird unter anderem dadurch bestimmt, welcher Cache die schnellste Antwort sendet oder welcher näher ist. Bei einer nicht zufrieden stellenden Antwort gesendet wurde, wird die Anfrage an den übergeordneten Cache geschickt.

Tipp

Zur Vermeidung von mehrfacher Speicherung von Objekten in verschiedenen Caches unseres Netzwerks werden andere ICP-Protokolle verwendet, wie z.B. CARP (*engl. Cache Array Routing Protocol*) oder HTCP (*engl.Hyper-Text Cache Protocol*).
Je mehr Objekte sich im Netzwerk befinden, desto leichter wird es, das Gesuchte zu finden.

7.2.3 Zwischenspeichern von Internetobjekten

Nicht alle im Netzwerk verfugbaren Objekte sind statisch. Es existieren viele dynamisch generierte CGI-Seiten, Zugriffszähler oder verschlüsselte SSL-Dokumente für eine höhere Sicherheit. Aus diesem Grund werden solche Objekte nicht im Cache gehalten. Bei jedem neuen Zugriff hat sich das Objekt bereits wieder verändert.

Für alle anderen im Cache befindlichen Objekte stellt sich jedoch die Frage, wie lange sie dort bleiben sollen. Für diese Entscheidung werden alle Objekte im Cache drei verschiedenen Stadien zugeordnet:

1. **FRESH:** Wenn dieses Objekt angefordert wird, wird es gesendet, ohne dass ein Abgleich mit dem Originalobjekt im Web stattfindet.

2. **NORMAL:** Der Server, von dem das Objekt ursprünglich stammt, wird daraufhin überpüft, ob sich das Objekt geändert hat. Falls dies der Fall ist, wird die Kopie im Cache aktualisiert.

3. **STALE:** Das Objekt wird als veraltet angesehen und wird neu vom Server heruntergeladen.

Durch Header wie "Last modifed" ("zuletzt geändert") oder "Expires" ("läuft ab") und dem entsprechenden Datum informieren sich Web- und Proxy-Server über den Status eines Objekts. Es werden auch andere Header verwendet, die z.B. anzeigen, dass ein Objekt nicht zwischengespeichert werden muss.

Objekte im Cache werden normalerweise aufgrund fehlenden Speichers ersetzt, und zwar durch Algorithmen wie LRU (*engl. Last Recently Used*), die zum Ersetzen von Cache-Objekten entwickelt wurden. Das Prinzip besteht im Wesentlichen darin, zuerst die am seltensten gewünschten Objekte zu ersetzen.

7.3 Systemanforderungen

Zuerst sollte die maximale Systemlast bestimmt werden. Es ist wichtig, den Systemspitzen besondere Aufmerksamkeit zu schenken, da diese mehr als viermal so hoch wie der Tagesdurchschnitt sein können. Im Zweifelsfall ist es besser, die Systemanforderungen zu überschätzen, vorausgesetzt, dass ein am Limit arbeitender Squid zu einem ernsthaften Qualitätsverlust des Dienstes führen kann.
Geordnet nach Wichtigkeit werden in den folgenden Abschnitten die verschiedenen Systemfaktoren aufgezeigt.

7.3.1 Festplatte

Für das Zwischenspeichern spielt Geschwindigkeit eine hohe Rolle. Man sollte sich also um diesen Faktor besonders kümmern. Bei Festplatten ist dieser Parameter als "zutällige Positionierzeit" in Millisekunden beschrieben. Als Faustregel gilt: Je niedriger dieser Wert, desto besser. Für eine hohe Geschwindigkeit empfiehlt es sich, schnelle Festplatten zu wählen.

Nach dem Squid-Benutzer-Guide (`http://www.squid-cache.org`) ist bei Systemen mit nur einer Festplatte die Formel für die Berechnung der Anzahl von Anfragen pro Sekunde von der Positionierzeit der Festplatten ganz einfach:

Anfragen pro Sekunde = 1OOO / Positionierzeit

Squid erlaubt die gleichzeitige Verwendung von mehreren Festplatten und damit eine höhere Anzahl von Anfragen pro Sekunde. Hat man z.B. drei Festplatten mit der gleichen Positionierzeit von l2 Millisekunden, ergibt sich unter Verwendung der vorherigen Formel folgendes:

Anfragen pro Sekunde = 1OOO / (Positionierzeit / Anzahl der Festplatten) =
1OOO / (12/3) = 250 Anfragen pro Sekunde

Im Vergleich zum Einsatz von IDE-Festplatten sind SCSI-Festplatten zu bevorzugen. Allerdings haben neuere IDE-Festplatten ähnliche Positionierzeiten wie SCSI und zusammen mit DMA-kompatiblen IDE-Controllern erreichen sie eine ähnliche Geschwindigkeit für den Datentransfer ohne dabei die Systemlast beträchtlich zu steigern.

Größe des Festplatten-Cache

In einem kleinen Cache ist die Wahrscheinlichkeit eines HIT (das gewünschte Objekt befindet sich bereits dort) sehr gering, da der Cache schnell gefüllt sein wird. In diesem Fall werden die selten gewünschten Objekte durch neue ersetzt.
Steht jedoch 1 GB für den Cache zur Verfugung und die Benutzer benötigen nur 10 MB pro Tag zum Surfen, dann dauert es mehr als hundert Tage, bis der Cache voll ist.

Am leichtesten lässt sich die Größe des Cache durch die maximale Übertragungsrate der Verbindung bestimmen. Mit einer Verbindung von 1 MB/Sek wird die maximale übertragungsrate bei 125 KB/Sek liegen. Landet der gesamte Datenverkehr im Cache, kommen innerhalb einer Stunde 450 MB zusammen. Wenn man nun annimmt, dass der gesamte Datenverkehr lediglich während acht Arbeitsstunden erzeugt wird, erreicht man innerhalb eines Tages 3,6 GB.
Da die Verbindung nicht bis zur Kapazitätsgrenze ausgeschöpft wurde, konnten wir davon ausgehen, dass die gesamte Datenmenge, die durch den Cache geht, bei ungefähr 2 GB liegt. In unserem Beispiel werden 2 GB Speicher für Squid benötigt, um die Daten aller aufgerufenen Seiten *eines* Tages im Cache zu halten.

Zusammenfassend lässt sich sagen, dass Squid dazu tendiert, kleinere Datenblöcke von der Festplatte zu lesen oder darauf zu schreiben, so dass es wichtiger ist, wie schnell er diese Objekte auf der Festplatte findet, als eine Festplatte mit hohem Durchsatz zu haben.

7.3.2 RAM

Der von Squid benötigte Speicher ist abhängig von der Anzahl der im Cache zugewiesenen Objekte. Squid speichert Cache-Objektverweise und häufig angeforderte Objekte zusätzlich im Speicher, damit diese Daten schneller abgefragt werden können. Der Speicher ist eine Million mal schneller als eine Festplatte!

Jedes Objekt im RAM-Speicher hat eine Größe von 72 Bytes (für "kleine" Pointer-Architektu-ren wie **Intel**, **Sparc**, **MIPS**; für Alpha sind es 104 Bytes), wenn die Durchschnittsgröße eines Objekts im Internet ungefähr 8 KB beträgt und wir 1 GB Festplattenspeicher für den Cache haben, werden wir ungefähr 130.000 Objekte speichern, was alleine für die Meta-Daten fast 10 MB RAM ergibt.

Squid hält auch andere Daten im Speicher, z.B. eine Tabelle mit allen vergebenen IP-Adressen, einen genau festgelegten Domainnamen-Cache, die am häufigsten gewünschten Objekte, Puffer, Zugriffskontrolllisten, etc.
Es ist sehr wichtig, dass ausreichend Speicher für den Squid-Prozess zur Verfügung steht. Sollte er ausgelagert werden müssen, wird sich die Systemleistung nämlich drastisch reduzieren. Für die Cache-Speicherverwaltung wird das Tool cachemgr.cgi verwendet. Es wird im Abschnitt 7.7.1 erläutert.

7.3.3 CPU

Das Programm Squid benötigt nicht viel CPU. Nur beim Start und während der Überprüfung des Cache-Inhalts ist die Prozessorlast höher. Der Einsatz eines Multiprozessorrechners steigert nicht die Systemleistung. Zur Effektivitätssteigerung ist es besser, schnellere Festplatten zu verwenden oder mehr Speicher hinzuzufugen.

Einige Beispiele von konfigurierten Systemen, auf denen Squid läuft, finden sich unter `http://wwwcache.ja.net/servers/squids.html`.

7.4 Squid starten

Der Squid auf SuSE Linux ist bereits soweit vorkonfiguriert, dass man ihn problemlos sofort nach der Installation starten kann. Als Voraussetzung für einen reibungslosen Start sollte das Netzwerk soweit konfiguriert sein, dass mindestens ein Nameserver und sinnvollerweise auch das Internet erreichbar sind. Probleme kann es bereiten, wenn man eine Wählverbindung mit dynamischer DNS-Konfiguration verwendet. In so einem Fall sollte mindestens der Nameserver fest eingetragen sein, da Squid erst gar nicht startet, wenn er in der `/etc/resolv.conf` keinen DNS findet.
Um Squid zu starten, gibt man auf der Kommandozeile (als 'root')

```
rcsquid start
```

ein. Beim ersten Mal wird zunächst die Verzeichnisstruktur in `/var/squid/cache` angelegt. Dies wird vom Startskript `/etc/init.d/squid` automatisch durchgeführt und kann ein paar Sekunden bis Minulen dauern. Erscheint rechts in grün `done`, wurde Squid erfolgreich gestartet. Auf dem lokalen System kann man die Funktionsfahigkeit von Squid sofort testen, indem man im Browser als Proxy `localhost` und `Port 3128` einträgt. Um den Zugriff auf Squid und
somit das Internet für alle zu ermöglichen, braucht man in der Konfigurationsdatei `/etc/squid.conf` lediglich den Eintrag `http_acces deny all` auf `http_acces allow all` zu ändern. Allerdings sollte man dabei bedenken, dass man den Squid damit komplett für jedermann öffnet. Von daher sollte man unbedingt so genannte "ACL's" definieren, die den Zugriff auf den Proxy regeln.
Dazu mehr im nächsten Kapitel.
Hat man Änderungen an der Konfigurationsdatei `/etc/squid.conf` vorgenommen, muss man Squid dazu bringen, diese neu einzulesen. Das gelingt mit:

```
rcsquid reload
```

Alternativ kann man Squid auch komplett neu starten.

```
rcsquid restart
```

Wichtig ist noch folgendes Kommando.

```
rcsquid status
```

Damit kann man feststellen, ob der Proxy läuft und mit

```
rcsquid stop
```

wird Squid beendet. Letzteres kann eine Weile dauern, da Squid bis zu einer halben Minute (`shutdown_lifetime`) wartet, bevor die Verbindungen zu den Clients unterbrochen werden und er dann noch seine Daten auf Platte schreiben muss.
Beendet man Squid mit einem kill oder killall, kann das einen zerstörten Cache zur Folge

haben, den man dann löschen muss, um Squid wieder starten zu können.

Beendet sich Squid nach kurzer Zeit, obwohl er scheinbar erfolgreich gestartet wurde, kann das an einem fehlerhaften Nameserver-Eintrag oder einer fehlenden `/etc/resolv.conf` liegen. Den Grund für einen gescheiterten Start protokolliert Squid dabei in der Datei `/var/squid/logs/cache.log`.

Soll Squid bereits beim Booten automatisch gestartet werden, braucht man in `/etc/rc.config` lediglich den Eintrag `START_SQUID=no` auf `START_SQUID=yes` abzuändern.

Bei einer Deinstallation von Squid werden weder Cache noch Log-Dateien entfernt. Man muss das Verzeichnis `/var/squid` manuell löschen.

Lokaler DNS-Server

Einen lokalen DNS-Server wie **BIND-8** oder **BIND-9** aufzusetzen, ist durchaus sinnvoll, auch wenn er keine eigene Domain verwaltet. Er fungiert dann lediglich als "Caching-only DNS" und kann ohne spezielle Konfiguration DNS-Anfragen über die Root-Nameserver auflösen. Trägt man diesen in der `/etc/resolv.conf` mit der IP-Adresse `127.0.0.1` für localhost ein, findet Squid beim Starten immer einen gülligen Nameserver. Es reicht aus, das Paket zu installieren
und BIND zu starten. Den Nameserver des Providers sollte man in der Konfigurationsdatei `/etc/named.conf` unter forwarders mit seiner IP-Adresse eintragen. Falls man eine Firewall laufen hat, und sei es nur die Personal-Firewall, muss man aber darauf achten, dass die DNS-Anfragen auch durchgelassen werden.

7.5 Die Konfigurationsdatei `/etc/squid.conf`

Alle Einstellungen zum HTTP-Proxyserver Squid sind in der Datei `/etc/squid.conf` vorzunehmen. Um Squid erstmalig starten zu können, sind darin keine Änderungen erforderlich, der Zugriff von externen Clients ist jedoch erst einmal gesperrt. Für localhost ist der Proxy freigegeben und als Port wird standardmäßig `3128` verwendet. Die Optionen sind ausführlich und mit vielen Beispielen in der vorinstallierten `/etc/squid.conf` dokumentiert. Annähernd
alle Einträge sind am Zeilenanfang durch ein #-Zeichen auskommentiert und immer am Ende der zugehörigen Beschreibung zu finden. Die angegebenen Werte entsprechen fast immer den voreingestellten Werten, so dass das Entfernen des Kommentarzeichens, ohne den Parameter der Option zu andern, bis auf wenige Ausnahmen keine Wirkung hat. Besser ist es, das Beispiel stehen zu lassen und die Option mit dem geänderten Parameter in der Zeile darunter neu einzufügen. So kann man die voreingestellten Werte und Änderungen problemlos nachvollziehen.

Hat man ein Update von einer älteren Squid-Version durchgefuhrt, ist es unbedingt zu empfehlen, die neue `/etc/squid.conf` zu verwenden und nur die Änderungen von der ursprünglichen Datei zu übernehmen. Versucht man die alte `squid.conf` weiter zu verwenden, läuft man Gefahr, dass die Konfiguration nicht mehr funktioniert, da Optionen immer wieder geändert werden und neue hinzukommen.

Allgemeine Konfigurations-Optionen

http_port 3128
Das ist der Port, auf dem Squid auf Anfragen der Clients lauscht. Voreingestellt ist 3128, gebräuchlich ist auch 8080. Es ist möglich, hier mehrere Portnummern, durch Leerzeichen getrennt, anzugeben.

cache_peer <hostname> <type> <proxy-port> <icp-port>
Hier kann man einen übergeordneten Proxy als "Parent" eintragen, z.B. wenn man den des Providers nutzen will oder muss. Als `<hostname>` trägt man den Namen bzw. die IP-Adresse des zu verwendenden Proxies und als `<type>` parent ein. Für `<proxy-port>` trägt man die Portnummer ein, die der Betreiber des Parent auch zur Verwendung im Browser angibt, meist `8080`. Den `<icp-port>` kann man auf `7` oder `0` setzen, wenn man den ICP-Port des Parent nicht kennt und die Benutzung dieses mit dem Provider nicht vereinbart wurde. Zusätzlich sollte man dann noch default und `no-query` nach den Portnummern angeben, um die Vervendung des ICP-Protokolls ganz zu unterbinden. Squid verhält sich dann gegenüber dem Proxy des Providers wie ein normaler Browser.

cache-mem 8 MB
Dieser Eintrag gibt an, wie viel Arbeitsspeicher von Squid für das Cachen maximal verwendet wird. Voreingestellt sind 8 MB.

cache-dir ufs /var/squid/cache 1OO 16 256
Der Eintrag cache-dir gibt das Verzeichnis an, in dem alle Objekte auf Platte abgelegt werden. Die Zahlen dahinter geben den maximal zu verwendenden Plattenplatz in MB und die Anzahl der Verzeichnisse in erster und zweiter Ebene an. Den Parameter `ufs` sollte man unverändert lassen. Voreingestellt sind `100 MB` Plattenplatz im Verzeichnis `/var/squid/cache` zu belegen und darin 16 Unterverzeichnisse anzulegen, diejeweils wiederum 256 Verzeichnisse enthalten. Bei Angabe des zu verwendenden Plattenplatzes sollte man genügend Reserven lassen, sinnvoll sind Werte zwischen 50 und maximal 80 Prozent des verfügbaren Platzes. Die beiden letzten Zahlen für die Anzahl der Verzeichnisse sollte man nur mit Vorsicht vergrößern, da zu viele Verzeichnisse auch wieder auf Kosten der Performance gehen können. Hat man mehrere Platten, auf die der Cache verteilt werden soll, kann man entsprechend viele `cache-dir`-Zeilen eintfagen.

cache_access_log /var/squid/logs/access.log
Pfadangabe für Log-Datei.

cache_log /var/squid/logs/cache.log
Pfadangabe für Log-Datei.

cache_store_log /var/squid/logs/store.log
Pfadangabe fur Log-Datei.
Diese drei Einträge geben den Pfad zur Protokolldatei von Squid an. Normalerweise wird man daran nichts ändern. Wird der Squid stark beansprucht, kann es sinnvoll sein, den Cache und die Log-Dateien auf verschiedene Platten zu legen.

emulate_httpd_log off
Ändert man diesen Eintrag auf on, erhält man lesbare Log-Dateien. Allerdings kommen manche Auswerteprogramme damit nicht zurecht.

client_netmask 255.255.255.255
Mit diesem Eintrag kann man die protokollierten IP-Adressen in den Log-Dateien maskieren, um die Identität der Clients zu verbergen. Trägt man hier `255.255.255.0` ein, wird die letzte Stelle der IP-Adresse auf Null gesetzt.

ftp-user Squid@
Hiermit kann man das Passwort setzen, welches Squid für den anonymen FTP-Login verwenden soll. Beim Zugriff auf öffentliche FTP-Server wird im allgemeinen als Login '`anonymous`' und als Passwort die eigene Mail-Adresse verwendet, was das Eingeben von Benutzername und Passwort für jeden FTP-Download erspart. Voreingestellt ist `Squid@` ohne Domain, da die Clients aus beliebigen Domains kommen können. Es kann aber sinnvoll sein, hier eine gültige E-Mail-Adresse in der eigenen Domain anzugeben, da einige FTP-Server diese auf Gültigkeit überprüfen.

cache-mgr webmaster
Eine E-Mail-Adresse, an die Squid eine Nachricht schickt, wenn er unerwartet abstürzt. Voreingestellt ist `webmaster`.

log_file_rotate 0
Squid ist in der Lage, die gesicherten Log-Dateien zu rotieren, wenn man `squid -k rotate` aufruft. Die Dateien werden dabei, entsprechend der angegebenen Anzahl, durchnummeriert, und nach Erreichen des angegebenen Wertes wird diejeweils älteste Datei wieder überschrieben. Dieser Wert steht standardmäßig auf `0`, weil das Archivieren und Löschen der Log-Dateien bei SuSE Linux von einem eigenen Cronjob durchgeführt wird, dessen Konfiguration man in der Datei `/etc/logfiles` findet. Der Zeitraum, nach dem die Dateien gelöscht werden, wird in der `/etc/rc.config` mit dem Eintrag `MAX_DAYS_FOR_LOG_FILES` festgelegt.

append_domain <domain>
Mit append_domain kann man angeben, welche Domain automatisch angehängt wird, wenn keine angegeben wurde. Meist wird man hier die eigene Domain eintragen, dann genügt es, im Browser www einzugeben, um auf den eigenen Webserver zu gelangen.

forwarded_for on
Setzt man diesen Eintrag auf off, entfernt Squid die IP-Adresse bzw. den Systemnamen des Clients aus den HTTP-Anfragen.

negative_ttl 5 minutes; negative_dns_ttl 5 minutes
Normalerweise braucht man diese Werte nicht zu verandern. Hat man aber eine Wahlleitung, kann es vorkommen, dass das Internet zeitweilig nicht erreichbar ist. Squid merkt sich dann die erfolglosen Anfragen und weigert sich, diese neu anzufragen, obwohl die Verbindung in das Internet wieder steht. Für diesen Fall sollte man die minutes in seconds ändern, dann führt auch ein Reload im Browser, wenige Sekunden nach der Einwahl, wieder zum Erfolg.

never_direct allow <acl-name>

Will man verhindern, dass Squid Anfragen direkt aus dem Internet fordert, kann man hiermit die Verwendung eines anderen Proxies erzwingen. Diesen muss man zuvor unter `cache-peer` eingetragen haben. Gibt man als `<acl_name>` all an, erzwingt man, dass sämtliche Anfragen direkt an den parent weitergegeben werden. Das kann zum Beispiel nötig sein, wenn man einen Provider verwendet, der die Verwendung seines Proxies zwingend vorschreibt oder die Firewall keinen direkten Zugriff auf das Internet durchlässt.

Optionen zur Zugriffskontrolle

Squid bietet ein ausgeklügeltes System, um den Zugriff auf den Proxy zu steuern. Durch die Verwendung so genannter "ACLs" ist es einfach und vielseitig konfigurierbar. Dabei handelt es sich um Listen mit Regeln, die der Reihe nach abgearbeitet werden. ACLs müssen zuerst definiert werden, bevor sie verwendet werden können. Einige Standard-ACLs wie `all` und `localhost` sind bereits vorhanden. Das Festlegen einer ACL an sich bewirkt aber noch gar nichts. Erst wenn sie zur Anwendung kommt, z. B. in Verbindung mit http_access, werden die definierten Regeln genutzt.

acl <acl_name> <type> <data>

Eine ACL benötigt zur Definition mindestens drei Angaben. Der Name `<acl_name>` kann frei gewählt werden. Für `<type>` kann man aus einer Vielzahl unterschiedlicher Möglichkeiten auswählen, die man im Abschnitt `ACCESS CONTROLS` in der `/etc/squid.conf` nachlesen kann. Was für `<data>` anzugeben ist, hängt vom jeweiligen Typ der ACL ab und kann auch aus einer Datei, z. B. mit Rechnernamen, IP-Adressen oder URLs eingelesen werden. Im folgenden einige einfache Beispiele:

```
acl meinesurfer srcdomain .meine-domain.com
acl lehrer src 192.168.1.0/255.255.255.0
acl studenten src 192.168.7.0-192.168.9.0/255.255.255.0
acl mittags time MTWHF 12:00-15:00
```

http_access allow <acl_name>

Mit `http_access` wird festgelegt, wer den Proxy verwenden darf und auf was er im Internet zugreiffen darf. Dabei sind ACLs anzugeben, localhost und all sind weiter oben bereits definiert, die mit `deny` oder `allow` den Zugriff sperren oder freigeben. Man kann hier eine Liste mit vielen `http-access`-Einträgen erstellen, die von oben nach unten abgearbeitet werden, je nachdem, was zuerst zutrifft, wird der Zugriff auf die angeforderte URL freigegeben oder gesperrt. Als letzter Eintrag sollte immer `http-access deny all` stehen. Im folgenden Beispiel hat `localhost`, also der lokale Rechner, freien Zugriff auf alles, während er für alle anderen komplett gespert ist:

```
http_access allow localhost
http_access deny all
```

Noch ein Beispiel, in dem die zuvor definierten ACLs verwendet werden. Die Gruppe `'lehrer'` hat jederzeit Zugriff auf das Internet, während die Gruppe `'studenten'` nur Montags bis Freitags, und da nur mittags, surfen darf.

```
http_access deny  localhost
http_access allow lehrer
```

```
http_access allow studenten mittags
http_access aeny  all
```

Die Liste mit den eigenen `http_access`-Einträgen sollte man der übersichtlichkeit halber nur an der dafür vorgesehenen Stelle in der `/etc/squid.conf` eintragen. Das bedeutet zwischen dem Text

```
# INSERT YOUR OWN RULE(S) HERE TO ATLLOW ACCESS FROM YOUR CLIENTS
```

und dem abschließenden

```
http_access deny all
```

`redirect_proqram /usr/bin/squidGuard`

Mit dieser Option kann man einen "redirector", wie z.B. SquidGuard angeben, der in der Lage ist, unerwünschte URLs zu sperren. In Verbindung mit Proxy-Authentifizierung und den passenden ACLs kann man so den Zugriff auf das Internet für verschiedene Benutzergruppen sehr differenziert steuern. SquidGuard ist ein eigenes Paket, das separat zu installieren und kofigurieren ist.

`authenticate_program /usr/sbin/pam_auth`

Sollen sich die Benutzer am Proxy authentifizieren müssen, kann man hier ein entsprechendes Programmwie z.B. pam_auth angeben. Bei der Verwendung von pam_auth öffnet sich für den Anwender beim ersten Zugriff ein Loginfenster, in dem er Benutzername und Passwort eingeben muss. Zusätzlich ist noch eine ACL erforderlich, damit nur Clients mit gültigem Login surfen können.

```
acl password proxy_auth REQUIRED

http_access allow password
http_access deny all
```

Das `REQUIRED` nach `proxy_auth` kann man auch durch eine Liste von erlaubten Benutzernamen ersetzen.

`ident_lookup_access allow <acl_name>`

Hiermit erreicht man, dass auf alle durch die ACL definierten Clients eine Ident-Anfrage ausgefuhrt wird, um die Identität des jeweiligen Benutzers zu ermitteln. Setzt man für `<acl_name>` `all` ein, erfolgt dies generell für alle Clients. Auf den Clients muss dazu ein Ident-Daemon laufen, bei Linux kann man dafür das Paket pidentd installieren, für Windows gibt es freie Software, die man sich aus dem Internet besorgen kann. Damit nur Clients mit erfolgreichem Ident-Lookup zugelassen werden, ist auch hier wieder eine entsprechende ACL zu definiefen:

```
acl identhosts ident REQUIRED

http_access allow identhosts
http_access deny all
```

Auch hier kann man das `REQUIRED` wieder durch eine Liste erlaubter Benutzernamen ersetzen. Die Verwendung von `Ident` kann den Zugriff merklich verlangsamen, da die Ident-Lookups durchaus für jede Anfrage wiederholt werden.

7.6 Transparente Proxy-Konfiguration

Normalerweise schickt der Web-Browser an einen bestimmten Port des Proxy-Servers Anfragen und der Proxy stellt die angeforderten Objekte zur Verfugung, ob sie nun im Cache sind oder nicht. Innerhalb eines echten Netzwerks können verschiedene Situationen auftreten:

Aus Sicherheitsgrunden ist es besser, wenn alle Clients zum Surfen im Internet einen Proxy verwenden.

Es ist notwendig, dass alle Clients einen Proxy verwenden, egal ob sie sich dessen bewusst sind oder nicht.

In großen Netzwerken, die bereits einen Proxy verwenden, ist es möglich, veränderte Konfigurationen der einzelnen Rechner zu speichern, falls sich Änderungen am System ergeben.

In jedem dieser Fälle kann ein transparenter Proxy eingesetzt werden. Das Prinzip ist denkbar einfach: Der Proxy nimmt die Anfragen des Web-Browsers entgegen und bearbeitet sie, sodass der Web-Browser die angeforderten Seiten erhält ohne zu wissen, woher sie kommen. Der gesamte Prozess wird transparent ausgeführt, daher der Name für den Vorgang.

7.7 Squid und andere Programme

7.7.2 SquidGuard

Dieses Kapitel soll lediglich eine Einführung zur Konfiguration von SquidGuard sowie ein paar Ratschläge zu dessen Einsatz geben. Auf eine umfangreiche Erklarung wird an dieser Stelle verzichtet. Tiefergehende Informationen finden sich auf den Webseiten zu SquidGuard: `http://www.squidguard.org`

SquidGuard ist ein freier (GPL), flexibler und ultraschneller Filter, ein Umleiter und "Zugriffs-Controller-PlugIn" für Squid. Er ermöglicht das Festlegen einer Vielzahl von Zugriffsregeln mit unterschiedlichen Beschränkungen für verschiedene Benutzergruppen für einen Squid-Cache. SquidGuard verwendet die Standardschnittstelle von Squid zum Umleiten.

squidGuard kann u.a. für Folgendes verwendet werden:

Beschränkung des Internetzugriffs für einige Benutzer auf bestimmte akzeptiertelbekannte Web-Server und/oder URLs.

Zugrlffsverweigerung für einige Benutzer auf bestimmte Web-Server und/oder URLs. Zugriffsverweigerung fur einige Benutzer auf URLs, die bestimmte reguläre Ausdrucke oder Wörter enthalten.

Umleiten gesperrter URLs an eine "intelligente" CGI-basierte Infoseite.

Umleiten nicht registrierter Benutzer an ein Registrationsformular.

Umleiten von Bannern an ein leeres GIF.

Unterschiedliche Zugriffsregeln abhängig von der Uhrzeit, dem Wochentag, dem Datum etc.

Unterschiedliche Regeln für die einzelnen Benutzergruppen

Weder mit squidGuard noch mit Squid ist folgendes möglich:

Text innerhalb von Dokumenten filtern/zensieren/editieren

In HTML eingebettete Skriptsprachen wie JavaScript oder VBscript filtern/zensieren/editieren

Verwendung von SquidGuard

Installieren Sie das Paket `squidGuard` aus der Serie n. Editieren Sie die Konfigurationsdatei `/etc/squidguard.conf`. Es gibt zahlreiche andere Konfgurationsbeispiele unter `http:/www.squidguard.org/config/`. Sie können später mit komplizierteren Konfigurationsenstellungen experimentieren.

Der nächste Schritt besteht darin, eine Dummy-Seite "Zugriff verweigert" oder eine mehr oder weniger intelligente CGI-Seite zu erzeugen, um Squid umzuleiten, falls der Client eine verbotene Webseite anfordert. Der Einsatz von Apache wird auch hier wieder empfohlen.

Nun müssen wir Squid sagen, dass er SquidGuard benutzen soll. Dafür vewenden wir folgende Einträge in der Datei `/etc/squid.conf`:

```
redirect_program /usr/sbin/squidGuard
```

Eine andere Option namens `redirect_children` konfguriert die Anzahl der verschiedenen auf dem Rechner laufenden "`redirect`", also Umleitungsprozesse (in diesem Fall SquidGuard). SquidGuard ist schnell genug, um eine Vielzahl von Anfragen (SquidGuard ist wirklich schnell. 1OO.OOO Anfragen innerhalb von 1O Sekunden auf einem 500MHz Pentium mit 5900 Domains, 7880
URLs, gesamt 13780) zu bearbeiten. Es wird daher nicht empfohlen, mehr als 4 Prozesse festzusetzen, da die Zuweisung dieser Prozesse unnötig viel Speicher braucht.

```
redirect_children 4
```

Als Letztes senden Sie ein HUP-Signal zum Squid, damit die neue Konfiguration eingelesen wird:

```
squid -k reconfigure
```

Nun können Sie Ihre Einstellungen in einem Browser testen.

7.7.3 Erzeugen von Cache-Berichten mit Calamaris

Calamaris ist ein Perl-Skript, das zur Erzeugung von Aktivitätsberichten des Cache im ASCII- oder HTML-Format verwendet wird. Es arbeitet mit Squid-eigenen Zugriffsprotokolldateien. Die Homepage zu Calamaris befindet sich unter: `http://Calamaris.Cord.de/`

Das Programm ist einfach zu verwenden. Melden Sie sich als 'root' an und geben Sie folgendes ein:

```
cat access.log.files | calamaris [optionen] > reportfile
```

Beim Verketten mehrerer Prolokolldateien ist die Beachtung der chronologischen Reihenfolge wichtig, d.h. ältere Dateien kommen zuerst.

Die verschiedenen Optionen:

`-a` wird normalerweise zur Ausgabe aller verfugbaren Berichte verwendet, mit

`-w` erhält man einen HTML-Bericht und mit

`-l` eine Nachricht oder ein Logo im Header des Berichts.

Weitere Informationen über die verschiedenen Optionen finden Sie in der Manual Page zu calamaris: **`man calamaris`**

Ein übliches Beispiel:

```
cat access.log.2 access.log.1 access.log | calamaris -a -w > \
     /usr/local/httpd/Squid/squidreport.html
```

Der Bericht wird im Verzeichnis des Web-Servers abgelegt. Wieder wird Apache benötigt, um die Berichte anzeigen zu können.

Ein weiteres, leistungsstarkes Tool zum Erzeugen von Cache-Berichten ist SARG (Squid Analysis Report Generator). Weitere Informationen dazu finden sich auf der entsprechenden Internetseite unter: `http://web.onda.com.br/orso/`

7.8 Weitere Informationen zu Squid

Besuchen Sie die Homepage von Squid: `http:/www.squid-cache.org/`. Hier fnden Sie den Squid User Guide und eine sehr umfangreiche Sammlung von FAQs zu Squid. Das Mini-Howto zu einem transparenten Proxy im Paket `howtoen`, unter:

`/usr/share/doc/howto/en/mini/TransparentProxy.gz`

Des Weiteren gibt es Mailinglisten für Squid unter:

`squid-users@squid-cache.org.`

Das Archiv dazu befindet sich unter:

`http://www.squid-cache.org/mail-archive/squid-users/`