Rekursiver Download mit Wget

ALEXANDER MAYER, TECHN. UNIVERSITÄT MÜNCHEN

Der Download ganzer Websites oder großer Dateien kann selbst mit modernen Webbrowsern nervenaufreibend sein, da nur wenige die Möglichkeit vorsehen, abgebrochene Downloads weiterzuführen oder beim Speichern Links rekursiv zu verfolgen. Es gibt aber das GNU-Tool Wget, das diese und viele andere Aufgaben problemlos meistert.

Wget gehört bereits seit vielen Jahren zum Handwerkszeug von Kommandozeilen-Puristen, auch wenn das GNU-Tool in jüngster Zeit aufgrund seines komplexen und unübersichtlichen Quelltextes in die Kritik geriet. Einige Sicherheitslücken in älteren Versionen von Wget erlaubten bei einem entsprechend präparierten Server das Löschen oder Überschreiben lokaler Dateien, denn der Quelltext ist fehlerhaft und schlecht kommentiert. Nach Meinung des Maintainers wäre eigentlich ein völliger Neuanfang die beste Lösung. Trotz der Probleme ist Wget aber weiterhin in den meisten Fällen in der Praxis ein verläßlicher Partner.

Bereits beim einfachen Download einer Datei leistet Wget gute Dienste, möchte man beispielsweise ein CD-Image ohne Einsatz eines Download-Managers aus dem Netz laden: Wie Bild 1 zeigt, versteht Wget die Protokolle FTP und HTTP gleichermaßen. Befindet man sich hinter einer Firewall und erhält die Fehlermeldung »Invalid PORT«, muß man den passiven Modus von FTP mit der Option *--passive-ftp* einschalten:

```
$ wget --passive-ftp ...
```

Sollte der Download, aus welchen Gründen auch immer, abgebrochen werden, versucht Wget standardmäßig bis zu zwanzig Mal, die Verbindung wieder aufzunehmen. Erst wenn diese Versuche gescheitert sind, bricht das Tool endgültig ab. Allerdings sind auch dann die bereits geladenen Daten nicht verloren. Ist die Verbindung wieder funktionsfähig, wird mit der Option *-c* der Download an der abgebrochenen Stelle wieder aufgenommen:

```
$ wget -c ftp://iso3.de.netbsd.org/\
pub/NetBSD/iso/2.0.2/i386cd.iso
```

Vergißt man beim Wiederaufnehmen eines Downloads die Option *-c*, überschreibt Wget die vorhandene Datei nicht, sondern hängt einen neuen Zähler an die neue Version an. Befindet sich beispielsweise bereits eine Datei *i386-cd.iso* im Zielverzeichnis, heißt die neue Datei *i386cd.iso.1*. Auch wenn es trivial klingt: Man sollte sich dieses Verhaltens stets bewußt sein. Nur zu leicht kommt es vor, daß man im Eifer des Gefechts die unvollständige Datei *i386cd.iso* und nicht die neue und vollständige Datei *i386cd.iso.1* auf einen Rohling gebrannt wird.

Beim Download von FTP-Servern ist es manchmal erforderlich, gleich mehrere Dateien auf einmal herunterzuladen. Dafür versteht Wget die Jokerzeichen, die man auch aus der Shell kennt. Es ist nur mit Hilfe eines Quotings darauf zu achten, daß die Shell die Jokerzeichen nicht selbst auflöst:

```
$ wget 'ftp://ftp.kernel.org/pub/*'
```

Rekursiver Download

Standardmäßig erkennt Wget die Jokerzeichen automatisch. Falls sie jedoch in einer Konfigurationsdatei ausgeschaltet sind, lassen sie sich mit der Option *--glob=on* explizit aktivieren. Man muß darauf achten, daß die Suchmuster nur auf Unix-FTP-Servern zuverlässig funktionieren, da die Ausgabe der Dateiliste

```
$ wget ftp://iso3.de.netbsd.org/pub/NetBSD/iso/2.0.2/i386cd.iso
--17:47:12--
ftp://iso3.de.netbsd.org/pub/NetBSD/iso/2.0.2/i386cd.iso
           => `i386cd.iso'
Resolving iso3.de.netbsd.org... 195.22.142.121
Connecting to iso3.de.netbsd.org[195.22.142.121]:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /pub/NetBSD/iso/2.0.2 ... done.
==> PASV ... done.    ==> RETR i386cd.iso ... done.
Length: 179,470,336 (unauthoritative)

 0% [>                    ] 1,560,960    111.69K/s    ETA 25:37
```

Bild 1: Eine typische Download-Session mit Wget

von FTP-Servern nicht standardisiert ist und Wget aus dieser Ausgabe die Dateinamen extrahieren muß.
Wofür Wget sehr prädestiniert ist, ist der rekursive Download ganzer Server, der mit der Option *-r* aktiviert wird:

```
$ wget -r http://www.myserver.de
```

Bevor man einen solchen rekursiven Download startet, muß man allerdings die betreffende Startseite genau darauf analysieren, was tatsächlich benötigt wird. Standardmäßig lädt Wget nämlich alle Dateien, die auf der angegebenen HTML-Seite verlinkt sind und die sich auf dem Webserver befinden. Neben HTML-Seiten und Grafiken betrifft das auch alle anderen Dateitypen wie ZIP-Archive oder ISO-Images. Nur externe Links bleiben unberührt.
Wget beläßt es auch nicht bei der ersten Ebene der HTML-Seiten. Es prüft alle heruntergeladenen HTML-Seiten und holt sich dann die darin angegebenen verlinkten Dateien. Mit denen geht es dann genauso weiter. Falls man es nicht anders angibt, beendet Wget die Arbeit erst in der fünften Ebene. Es liegt auf der Hand, daß solch ein rekursiver Download großer Server eine große Datenmenge auf der lokalen Platte anhäuft. Deshalb ist eine genauere Präzisierung der gewünschten Daten sehr zu empfehlen.

ZÜ

Eine deutliche Steigerung der Effektivität erreicht man, wenn man den rekursiven Download von vornherein mit der Option *-l* auf eine bestimmte Tiefe beschränkt. Möchte man beispielsweise nur eine HTML-Seite und die darin direkt verlinkte Dateien herunterladen, reicht die Angabe nur dieser einen Ebene:

```
$ wget -l 1 -r http://www.myserver.de
```

Wget legt bei allen Aufrufen mit der Option *-r* einen Unterordner mit dem Namen des Webservers, im Beispiel *www.myserver.de*, an und speichert alle von dort geladenen Dateien in diesem Verzeichnis ab.
Will man nur eine einzelne HTML-Seite aus dem Internet laden, ist also eigentlich nur am HTML-Quelltext und den zur Darstellung außerdem benötigten Dateien interessiert, erreicht man mit der Option *-p*, daß im Gegensatz zur Angabe von *-r* nicht alle verlinkten Dateien geladen werden:

```
$ wget -p http://www.myserver.de
```

Eine weitere Möglichkeit, den Download zu beschränken, besteht in der Angabe der Option *-L*. Sie veranlaßt Wget, nur relative Links zu verfolgen. Relative Links sind solche, die nicht auf einen anderen Server verweisen und nicht mit »/« beginnen. Man muß hierbei aber darauf achten, daß mittlerweile nur noch die wenigsten Webseiten konsequent relative Links verwenden. In der Regel wird diese Option also wenig hilfreich sein.
Eine wesentlich bessere Methode, die gewünschten Dateien einzugrenzen, ist die Angabe von *-np*. Damit geht Wget nur in die Tiefe und verfolgt keine Links, die in der Hierarchie der Verzeichnisse auf dem Webserver zurückgehen. Mit dem folgenden Aufruf werden nur die Dateien und Verzeichnisse im Unterverzeichnis */man* gespeichert:

```
$ wget -r -np http://www.myserver.de/man
```

Möchte man Seiten mit Wget aus dem Internet laden, um sie lokal lesen zu können, wird das Resultat des rekursiven Downloads, wie bisher vorgestellt, in den meisten Fällen nicht zufriedenstellend sein. Da die HTML-Seiten unangetastet bleiben, werden viele Links absolute Pfade sein. Lautet beispielsweise ein Link */man/index.html*, wird er in der lokalen Kopie nicht funktionieren, da sie der Browser im Verzeichnis */man/index.html* im Dateisystem sucht. Es wäre also wünschenswert, diese absoluten Links in Links umzuwandeln, die relativ zum aktuellen Verzeichnis sind.
Angenommen, die Startseite des Servers *http://www.myserver.de* enthielte einen Link auf die HTML-Seite */man/index.html* und die lokale Kopie wäre im Unterverzeichnis *www.myserver.de*. Der äquivalente relative Pfad dieses Links lautet damit

Authentifizierung und Proxies

Manche Webserver erwarten eine Authentifizierung durch Benutzername und Paßwort. Solche Daten übergibt man Wget mit folgenden Optionen:

```
$ wget --http-user=benutzer \
       --http-passwd=passwort \
       http://www.myserver.de
```

Man muß aber beachten, daß die angegebenen Daten während des Downloads in der Prozeßliste des Systems für jeden eingeloggten Benutzer lesbar sind!

Besonders in Firmennetzwerken ist es üblich, daß HTTP nur über einen Proxy erreichbar ist. Um den entsprechenden Proxy-Server auch mit Wget zu verwenden, muß man die Umgebungsvariable *http_proxy* beziehungsweise *ftp_proxy* für FTP setzen:

```
$ export http_proxy=proxy:8080
```

Im obigen Beispiel heißt der Proxy-Server *proxy* und wartet an Port 8080 auf Anfragen. Falls der Proxy-Server auch noch Benutzername und Paßwort erwartet, kann man diese Daten Wget direkt übergeben:

```
$ wget --proxy-user=benutzer \
       --proxy-passwd=passwort \
      http://www.myserver.de
```

Download-Aktivitäten begrenzen

Rekursive Downloads sind bei Webserver-Administratoren im allgemeinen nicht besonders beliebt, da durch sie über längere Zeit hohe Last auf dem Rechner erzeugt wird. Findige Administratoren versuchen deshalb, anhand einiger Kriterien solche Download-Aktivitäten zu unterbinden. Um eine solche Ausgrenzung zu verhindern und die Last auf dem anderen Rechner möglichst gering zu halten, kann man deshalb mit Wget beispielsweise die Downloadrate auf einen festen Wert begrenzen:

```
$ wget -m --limit-rate=5k \
       http://www.myserver.de
```

Dieser Aufruf von Wget sorgt dafür, daß nur mit fünf KByte/Sekunde geladen wird. Eine weitere Möglichkeit der Entlastung des Webservers besteht darin, diverse Timeouts festzulegen. Im allgemeinen sollte es ausreichen, zwischen jedem Download einige Sekunden zu warten:

```
$ wget -w 5 -m http://www.myserver.de
```

Hiermit wartet Wget nach jedem erfolgtem Download fünf Sekunden. In besonders schwerwiegenden Fällen kann Wget mit der Option *--random-wait* auch eine zufällige Anzahl von Sekunden zwischen den Downloads warten. Es ist aber zu berücksichtigen, daß das Erstellen einer lokalen Kopie nur mit ausdrücklicher Genehmigung des Webserver-Betreibers erlaubt ist. Ein Administrator, der Wget derart geschickt abwimmeln möchte, hat sicher gute Gründe dafür.

man/index.html. Außerdem befindet sich in dieser Datei ein Link zur Startseite. Er lautet im Original */index.html*, der relative Link ist dementsprechend *../index.html*. Wget beherrscht dafür das Konvertieren absoluter Links in relative Links. Diese schaltet man mit *-k* ein:

```
$ wget -r -k http://www.myserver.de
```

Es werden nur die Links konvertiert, die auf Dateien zeigen, die sich auch tatsächlich auf der lokalen Platte befinden. Alle anderen Links werden in vollständige URLs umgewandelt, also beispielsweise *http://www.myserver/home/index.html*. Falls man zusätzlich die Option *-K* angibt, speichert Wget die Originalversion einer jeden konvertierten Datei unter dem gleichen Namen, aber mit der Endung *.orig*.

Wget wird sehr häufig dazu eingesetzt, vollständige lokale Kopien von Webservern aktuell zu halten. Wird eine solche Kopie dann erneut online gestellt, spricht man auch vom Erstellen eines »Mirrors«. In der Regel wird bei solchen Anwendungsfällen Wget in regelmäßigen Abständen über einen Cron-Job aufgerufen. Um Bandbreite und Zeit zu sparen, empfiehlt es sich allerdings, dabei die Option *-N* einzuschalten. Sie veranlaßt Wget dazu, nur solche Dateien zu laden, die neuer sind als die auf der Festplatte. Um Änderung erkennen zu können, wertet Wget bei HTTP-Servern die Header-Angabe *Last-Modified* aus und setzt für die lokalen Dateien den vom Server gelieferten Zeitstempel. Nur wenn der Zeitstempel der lokalen Datei älter als der vom Webserver gelieferte ist, wird die Datei geladen:

```
$ wget -N -r -l inf \
  http://www.myserver.de
```

Mirrors erstellen

Mit dem beschriebenen Aufruf erstellt man eine vollständige Kopie eines Webservers auf der lokalen Platte. Der Begriff *inf* steht dabei für »infinitly« und sorgt dafür, daß der Webserver bis in die letzten Winkel ohne Beschränkung durchsucht wird. Anstatt der hier angegebenen einzelnen Optionen wäre auch *-m* (für Mirror) denkbar:

```
$ wget -m http://www.myserver.de
```

Privat wird sich kaum jemand einen klassischen Mirror ziehen. Viel häufiger möchte man aus Gründen der Bequemlichkeit eine lokal benutzbare Kopie eines wichtigen Servers haben. Setzt man jedoch die Option *-k* in Verbindung mit *-N* ein, wird Wget die meisten HTML-Seiten neu herunterladen, auch wenn sie auf dem Originalserver gar nicht verändert wurden. Der Grund liegt in der Konvertierung der Links. Durch das Verändern der Originaldateien verändert sich naturgemäß auch deren Zeitstempel. Abhilfe schafft die Option *-K* für das Sichern der Originaldateien. Sind die drei Optionen *-kKN* angegeben, prüft Wget die Zeitstempel der .orig-Dateien und nicht die der konvertierten Dateien:

```
$ wget -m -kK http://www.myserver.de
```

Da der rekursive Download mit Wget unter Umständen sehr lange dauern kann und oft auch automatisiert vorgenommen wurde, möchte man ihn im Hintergrund starten. Im einfachsten Fall geschieht das mit dem Aufruf:

```
wget -o myserver.log -m \
     http://www.myserver.de &
```

Der Parameter *-o* sorgt dafür, daß die Statusmeldungen von Wget, die normalerweise auf der Standardfehlerausgabe landen, in die angegebene Datei geschrieben werden. Je nachdem, von wo aus man einen solchen Befehl eingibt, wird die Shell vor dem Ausloggen blockieren, da noch ein Hintergrundprozeß läuft. Um diesen Problemen aus dem Weg zu gehen, sollte man deshalb Wget mit der Option *-b* aufrufen. Damit begibt sich Wget von selbst in den Hintergrund und schreibt die Ausgaben in die Datei *wgetlog*:

```
wget -b -m http://www.myserver.de
```

Standardmäßig lädt Wget alle Dateien aus dem Netz, die verlinkt werden. Die Optionen *-A* und *-R* beschränken den Download auf bestimmte Dateitypen. Nach *-A* gibt man kommasepariert alle Dateiendungen an, die gespeichert werden sollen, und nach *-R* alle Dateiendungen, die nicht gespeichert werden sollen. Der folgende Aufruf lädt beispielsweise nur die Grafiken einer HTML-Seite:

```
$ wget -r -l1 \
   -A .gif,.GIF,.jpg,.JPG,.png,.PNG \
   http://www.myserver.de/bilder.html
```

Man muß aber wissen, daß Wget die HTML-Seiten *immer* lädt, um die Suche nach verlinkten Dateien gewährleisten zu können, auch wenn letztendlich die HTML-Dateien nicht auf der Festplatte verbleiben.

Download mehrerer URLs

Immer mehr Webpräsenzen werden dynamisch mit Hilfe von ASP, PHP oder CGI generiert. Spiegelt man URLs wie *http://www.myserver.de/index.php*, wird in den meisten Fällen die gespeicherte Datei auch *index.php* heißen. Da ein Webbrowser mit der Dateiendung *.php* beim lokalen Betrachten der Datei nichts anfangen kann, sollte man mit der Option *-E* dafür sorgen, daß solche Dateien automatisch die Endung *.html* erhalten:

```
$ wget -E \
  http://www.myserver.de/index.php
```

In manchen Fällen möchte man gleich mehrere Webserver auf einmal spiegeln. Wget lassen sich mehrere URLs übergeben, wenn diese in einer eigenen Datei, am besten zeilenweise, zu finden sind:

```
$ cat urlliste
 http://www.myserver1.de
 http://www.myserver2.de
 $ wget -i urlliste -mkK
```

In der Praxis ist diese Methode zu empfehlen, wenn diese Server in regelmäßigen Abständen automatisiert gespiegelt werden sollen. Möchte man allgemein einen größeren Downloadauftrag an Wget übergeben, ist es günstiger, diesen in einem Shellskript zu starten:

```
#!/bin/sh
# wgetlist
# Beispielaufruf: ./wgetlist urllist.lst

IFS=
cat "$@" | while read url; do
  until wget -m "$url"; do
    echo '*** Wget endete mit Fehler, \
Neustart'

    # hier kann man irgendetwas machen
    # als Beispiel warten wir 10 Sekunden
    sleep 10
  done
done
```

Das Shellskript sorgt dafür, daß im Fehlerfall, wenn also beispielsweise die Internetverbindung abbricht oder der Webserver kurzzeitig nicht erreichbar ist, nur der Download der Seite neu gestartet wird, die noch nicht vollständig geladen wurden. Bricht beispielsweise Wget im obigen Beispiel beim Download von *www.myserver2.de* ab und ist der andere Server bereits vollständig auf der Festplatte gespiegelt, beginnt Wget mit der Option *-i* bei einem weiteren Aufruf nochmals unnötigerweise bei *www.myserver1.de*.
Eine andere Lösung, das Problem der Option *-i* zu umgehen, besteht darin, zu verhindern, daß Wget abbricht. Mit der Option *-t* bestimmt man die Anzahl der Downloadversuche, wobei »inf« für »unendlich« steht:

```
wget -i urlliste -t inf -mkK
```

Wget kann auf eine lange Entwicklungsgeschichte zurückblicken. Die erste Version erschien bereits im Jahr 1995. Da ist es nicht verwunderlich, daß es noch wesentlich mehr Optionen besitzt, wie in diesem Beitrag angesprochen. Ein Blick in die Manpage lohnt also.
Es ist aber immer zu berücksichtigen, daß Wget der Komplexität und Inkompatibilität vieler HTML-Seiten Rechnung tragen muß. Falls beispielsweise Links über Javaskript realisiert werden, erkennt Wget sie im Regelfall nicht. Auch schlecht geschriebener HTML-Code kann das Programm verwirren – von Homepages, die auf Plugins wie Flash basieren, ganz abgesehen. Wget kann eben vieles, aber auch nicht alles. Allerdings ist es auch fraglich, ob solche Seiten wirklich die Spiegelung wert sind.