Debian/Apache HOWTO

Johannes Franken
<jfranken@jfranken.de>

Auf dieser Seite erkläre ich die Grundlagen von Webservern und zeige die Installation und Konfiguration des **Apache-Webservers Version 2 unter Debian GNU/Linux 3.1.**

Inhalt

Grundlagen

Webserver

Als "Webserver" bezeichnet man Programme, die über das HTTP-Protokoll Anfragen entgegennehmen (z.B. von einem Webbrowser über ein Internet) und Daten zurücksenden, die für die in der Anfrage genannte URL vorgesehen sind.

Die Daten sind entweder

1. statisch (Dateiinhalte) oder
2. dynamisch (Ausgabe von Programmen).

Der erste Webserver ("W3 demon") wurde 1989 von Tim Berners-Lee programmiert. Inzwischen gibt es hunderte verschiedene Webserver-Implementierungen, die sich in Ihrer

- Arbeitsgeschwindigkeit,
- Zuverlässigkeit,
- Erweiterbarkeit und
- den von ihnen unterstützten Betriebssystemen

unterscheiden.

Mehr zum Thema:
siehe

- http://www.ietf.org/rfc/rfc2616.txt (HTTP/1.1-protocol)
- http://de.wikipedia.org/wiki/Webserver

Die Debian-3.1-Distribution enthält folgende Webserver:

Debian-Paket	Beschreibung
aolserver	AOL Web Server 3 (Program)
aolserver4	AOL Web Server 4 (Program)
apache	versatile, high-performance HTTP server
apache2	next generation, scalable, extendable web server
boa	Lightweight and high performance webserver
caudium	An extensible webserver written in Pike
dhttpd	minimal secure webserrver without cgi-bin support
micro-httpd	A really small http server
roxen4	The Roxen Challenger Webserver
thttpd	tiny/turbo/throttling HTTP server
webfs	a lightweight web server for static content
zope	open source web application server

Tabelle: Webserver in der Debian-Distribution

Dieser Vortrag beschreibt ausschließlich **apache2**, den "Apache"-Webserver in der Version 2.0 0

Apache

Apache ist Open-Source-Software, arbeitet effizient und zuverlässig und bietet in seinem modularen Aufbau einen Leistungsumfang, der alle anderen Webserver übertrifft. Etwa 60% aller Websites laufen zurzeit unter Apache.

Mehr zum Thema "Apache":
Eine kurze Übersicht zum Apache-Webserver finden Sie auf den Webseiten von Wikipedia und der Apache Foundation.

Die Apache Foundation bietet den HTTP-Server zurzeit (Dezember 2005) in drei Versionen an:

- **Version 1.3.34** ist die letzte Version der "alten Baureihe". Obwohl Sie nicht mehr weiterentwickelt wird, ist sie immer noch auf vielen Servern im Einsatz. Sie sollten 1.3er-Versionen nur noch einsetzen, wenn Sie ein Modul benötigen, das für die Version 2 noch nicht zur Verfügung steht.
- **Version 2.0.55** ist die "aktuelle" Version der 2.0er-Baureihe und Gegenstand dieses Vortrags. Im Vergleich zu den 1.3er-Versionen arbeitet sie etwas schneller und ist flexibler in der Konfiguration.
- **Version 2.2.0** ist die allerneueste Version (erschienen: 30.11.2005). Sie bietet zwar einige neue Möglichkeiten, ist aber für einen Praxiseinsatz noch nicht hinreichend getestet.

Ab Version 2.0 greift der Apache-Kern nicht mehr direkt (über System-Calls), sondern über eines der "Multi-Processing-Module" (MPM) auf das Betriebssystem zu.

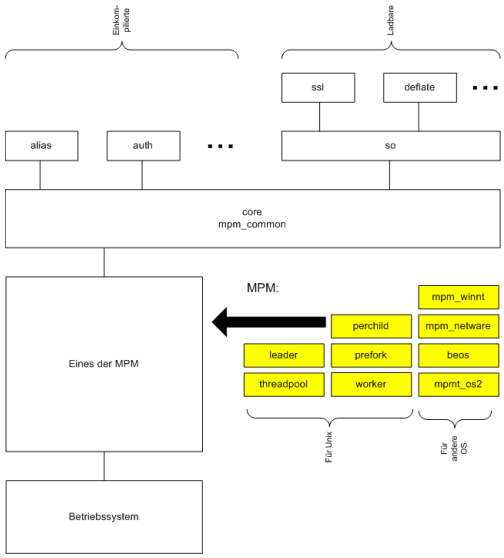

Abbildung: Apache-Architektur (ab Version 2)

Es gibt ein MPM für jedes unterstützte Betriebssystem und für Unix-Systeme sogar fünf verschiedene.

Unter Debian 3.1 stehen Ihnen drei MPM zur Auswahl. Wählen Sie das MPM aus, das die Anforderungen an Ihren Webserver am besten trifft.

Debian-Paket	Beschreibung
apache2-mpm-prefork	ist das "normale" MPM (wie bei Apache 1.3). Jede Anfrage wird von einem separaten Prozess bearbeitet und alle Prozesse laufen unter demselben Unix-User. **Mehr zum Thema:** siehe http://httpd.apache.org/docs-2.0/mod/prefork.html
apache2-mpm-worker	arbeitet schneller als das **prefork**-MPM und benötigt weniger Speicher als dieses. Anfragen werden auf die Threads mehrerer Prozesse verteilt, die alle unter demselben Unix-User laufen. Dieses MPM wird automatisch installiert, wenn Sie bei der Installation von Apache kein anderes MPM angeben. Leider funktionieren einige Module (z.B. PHP) nicht mit diesem MPM. **Mehr zum Thema:** siehe http://httpd.apache.org/docs-2.0/mod/worker.html
apache2-mpm-perchild	ist eine Variante des **worker**-MPM, bei der die Prozesse unter unterschiedlichen Unix-Usern laufen können. Dieses MPM soll ungewollte Übergriffe auf gemeinsam genutzten Webservern verhindern, doch leider arbeitet es noch nicht zuverlässig. **Mehr zum Thema:** siehe http://httpd.apache.org/docs-2.0/mod/perchild.html

Tabelle: Beschreibung der erhältlichen MPM

Mehr zum Thema:
siehe http://httpd.apache.org/docs-2.0/mpm.html (kurze MPM-Einführung)
und http://httpd.apache.org/docs-2.0/de/mod/ (Links zu den Beschreibungen der MPMs).

Installation

Installieren Sie die neueste Version des Apache-Webservers:

```
$ aptitude update
[...]
$ aptitude install apache2
[...]
Die folgenden Pakete werden zusätzlich installiert:
  apache2-common apache2-mpm-worker apache2-utils libapr0 libexpat1
  openssl ssl-cert
0 Pakete aktualisiert, 7 zusätzlich installiert,
0 werden entfernt und 0 nicht aktualisiert.
Muss 2193kB/2193kB an Archiven herunterladen.
Nach dem Entpacken werden 5368kB zusätzlich belegt sein.
Wollen Sie fortsetzen? [Y/n/?] Y
[...]
Starting web server: Apache2.
```

Listing: Installation von Apache2

Wie Sie sehen, wählt **`aptitude`** das **`worker`**-MPM aus. Wenn Sie später zusätzlich PHP installieren, können Sie beobachten, dass das **`worker`**-MPM automatisch gegen das **`prefork`**-MPM ausgetauscht wird.

Begeben Sie sich an einen anderen Rechner und testen Sie mit einem Browser, ob Ihr frisch installierter Webserver antwortet:

Abbildung: Apache antwortet

Konfiguration

Die Debian-Pakete legen bei der Installation eine gebrauchsfertige Grundkonfiguration an. Diese besteht aus mehreren Konfigurationsdateien, die sich in folgenden Verzeichnissen befinden:

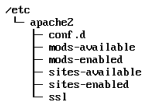

Abbildung: Apache2-Konfigurations-Verzeichnisse

Die von anderen Distributionen bekannte, zentrale Konfigurationsdatei **`/etc/httpd/conf/httpd.conf`** heißt unter Debian **`/etc/apache2/apache2.conf`**. Sie enthält im Wesentlichen **`Include`**-Anweisungen auf die anderen Konfigurationsdateien sowie einige grundlegende Konfigurationsanweisungen, die Sie vermutlich niemals ändern müssen.

Beschreibung der includeten Konfigurationsdateien:

`/etc/apache2/`	**Beschreibung**
`conf.d/*`	Hier können Sie (oder andere Debian-Pakete) Dateien anlegen, die weitere Konfigurationsanweisungen enthalten. Alle Dateien/Links in diesem Verzeichnis werden automatisch included. **Achtung:** In der Debian-Konfiguration ignoriert Apache Dateien, deren Dateiname mit einem Punkt oder #-Zeichen beginnt. Daher können Sie Konfigurationsdateien durch entsprechendes Umbenennen "auskommentieren".
`httpd.conf`	Eine leere Datei (zur Kompatibilität mit Apache 1.3).
`mods-enabled/*.load` **`mods-enabled/*.conf`**	Symlinks auf gleichnamige Dateien im Verzeichnis **`mods-available`**. Alle Dateien/Links in diesem Verzeichnis werden automatisch included. Sie können die Symlinks für alle Module, die Apache laden soll, mit dem Programm **`a2enmod`** anlegen und mit **`a2dismod`** entfernen (Beispiel: siehe unten).
`ports.conf`	Anweisungen zur Konfiguration der IP-Adressen und TCP-Ports, auf denen Apache lauschen soll.
`sites-enabled/*`	Symlinks auf gleichnamige Dateien im Verzeichnis **`sites-available`**. Alle Dateien/Links in diesem Verzeichnis werden automatisch included. Sie können die Symlinks für alle VirtualHosts, auf denen Apache antworten soll, mit dem Programm **`a2ensite`** anlegen und mit **`a2dissite`** entfernen (Beispiel: siehe unten). **Achtung:** Auch hier werden Dateien, deren Dateiname mit einem Punkt oder #-Zeichen beginnt, ignoriert.

Tabelle: Includete Apache2-Konfigurationsdateien

Die folgenden Konfigurationsdateien werden *nicht* automatisch included:

`/etc/apache2/`	**Beschreibung**
`mods-available/*.load` **`mods-available/*.conf`**	Dateien, die Lade- und Konfigurations-Anweisungen für Apache-Module enthalten.
`sites-available/*`	Dateien, die Konfigurations-Anweisungen für VirtualHosts enthalten.

Tabelle: weitere Konfigurationsdateien

Das folgende Listing zeigt die VirtualHost-Konfigurationsdatei, die standardmäßig verwendet wird und die Sie auf jeden Fall anpassen sollten:

```
01  NameVirtualHost *
02  <VirtualHost *>
03     ServerAdmin webmaster@localhost
04
05     DocumentRoot /var/www/
06     <Directory />
07        Options FollowSymLinks
08        AllowOverride None
09     </Directory>
10     <Directory /var/www/>
11        Options Indexes FollowSymLinks MultiViews
12        AllowOverride None
13        Order allow,deny
14        allow from all
15        # This directive allows us to have apache2's default start page
16        # in /apache2-default/, but still have / go to the right place
17        RedirectMatch ^/$ /apache2-default/
18     </Directory>
19
20     ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
21     <Directory "/usr/lib/cgi-bin">
22        AllowOverride None
23        Options ExecCGI -MultiViews +SymLinksIfOwnerMatch
24        Order allow,deny
25        Allow from all
26     </Directory>
27
28     ErrorLog /var/log/apache2/error.log
29
30     # Possible values include: debug, info, notice, warn, error, crit,
31     # alert, emerg.
32     LogLevel warn
33
34     CustomLog /var/log/apache2/access.log combined
35     ServerSignature On
36
37      Alias /doc/ "/usr/share/doc/"
38      <Directory "/usr/share/doc/">
39          Options Indexes MultiViews FollowSymLinks
40          AllowOverride None
41          Order deny,allow
42          Deny from all
43          Allow from 127.0.0.0/255.0.0.0 ::1/128
44      </Directory>
45
46  </VirtualHost>
```

*Listing: **`/etc/apache2/sites-available/default`***

Mehr zum Thema:
Genaue Beschreibungen aller Konfigurationsanweisungen finden Sie auf http://httpd.apache.org/docs-2.0/de/mod/quickreference.html.

Bevor Sie Ihre Webseiten auf den Server kopieren, sollten Sie die Apache-Begrüßungsseite wie folgt deaktivieren: Kommentieren Sie einfach die **`RedirectMatch`**-Anweisung in Zeile 17 der VirtualHost-Konfiguraitonsdatei mit einem #-Zeichen aus. Anschließend aktivieren Sie die geänderte Konfiguration, wie im nächsten Abschnitt beschrieben.

Starten und stoppen

Sie können den Apache-Webserver mit dem Skript **`/etc/init.d/apache2`** starten, stoppen und dazu bringen, die Konfigurationsdateien neu einzulesen. Unter Debian sollte man dieses Script aber über **`invoke-rc.d`** aufrufen.

```
$ invoke-rc.d apache2 start
Starting web server: Apache2.

$ invoke-rc.d apache2 stop
Stopping apache 2.0 web server....

$ invoke-rc.d apache2 restart   # entspr. -reload und stop+start
Starting apache 2.0 web server....

$ invoke-rc.d apache2 reload
Reloading apache 2.0 configuration....
```

Listing: Apache starten und stoppen

Beim Aufruf mit den Parametern **`restart`** oder **`force-reload`** beendet das **`apache2`**-Skript alle Apache-Prozesse und startet sie anschließend neu.

Achtung:
Wenn die Konfigurationsdatei schwere Fehler enthält, bleibt der Apache beim **`restart`** oder **`force-reload`** beendet.

Beim Aufruf mit dem Parameter **`reload`** weist das **`apache2`**-Skript den (unter **`root`** laufenden) Haupt-Prozess des Apache-Webservers an, seine Konfigurationsdateien neu einzulesen und die Kindprozesse nach Abschluss ihrer aktuellen Client-Verbindungen neu zu starten.

Webseiten einspielen

In der Konfigurationsdatei (siehe Listing) erkennen Sie an den Anweisungen **`DocumentRoot`** (Zeile 5) und **`ScriptAlias`** (Zeile 20), dass Apache Ihre Webseiten in **`/var/www/`** und CGI-Scripts in **`/usr/lib/cgi-bin/`** sucht. Testen Sie dies, bevor Sie Ihre Webseiten einspielen, indem Sie hier eine einfache Startseite und ein CGI-Script anlegen und diese im Browser aufrufen.

```
$ cd /var/www
$ cat <<EOF >index.html
<HTML><BODY>
click <a href= "/cgi-bin/ps">here</a> for ps
</BODY></HTML>
EOF
$ chmod 664 index.html


$ cd /usr/lib/cgi-bin
$ cat <<EOF >ps
#!/bin/sh
echo Content-Type: text/html
echo
echo '<html><body><pre>'
ps -ef    # <-- Beispiel fuer ein Unix-Kommmando
echo '</pre></body></html>'
EOF
$ chmod 775 ps
```

Listing: Eine Webseite und ein CGI-Script anlegen

Achtung:
Die Webseiten-Dateien, CGI-Scripts und Verzeichnisse müssen für den User **`www-data`** oder die Gruppe **`www-data`** oder für *alle* lesbar und ggf. auch ausführbar sein.

VirtualHosts

Wenn Sie auf einem physischen Server mehrere Websites gleichzeitig betreiben möchten, muss Apache bei jedem http-Request feststellen, an welche Website der Request gerichtet ist. Je nach Konfiguration achtet Apache dabei entweder auf

- die angesprochene IP-Adresse ("IP-based, virtual host") oder
- den im Request enthaltenen Hostnamen ("Name-based, virtual host").

"Name-based, virtual hosts"

Mit "Name based, virtual hosts" können Sie nach dem Einrichten entsprechender DNS-Einträge und Site-Konfigurationsdateien beliebig viele Websites auf derselben IP-Adresse betreiben. "Name based, virtual hosts" sind jedoch nicht für https geeignet und man sieht ihnen von außen (z.B. am PTR-Record) leicht an, dass mehrere Websites auf derselben Hardware laufen.

Um einen "Name-based, virtual host" einzurichten, gehen Sie bitte folgendermaßen vor:

1. Legen Sie im DNS einen A-Record an, der dem virtuellen Hostnamen eine IP-Adresse zuordnet.
2. Erstellen Sie eine Konfigurationsdatei unter **`/etc/apache2/sites-available`**.
   Das folgende Beispiel zeigt die Konfigurationsdatei für die Websites **`http://debian2.jfranken.de`**, **`http://debian2`** und **`http://debian2.jfranken`**:

```
01  NameVirtualHost *
02  <VirtualHost *>
03    ServerName debian2
04    ServerAlias debian2.jfranken.de debian2.jfranken
05    DocumentRoot /var/www2
06    ServerAdmin webmaster@jfranken.de
07
08    # Logfiles:
09    CustomLog /var/log/apache2/access2.log combined
10    ErrorLog /var/log/apache2/error2.log
11    LogLevel warn
12
13    # Umleitungs-Beispiele:
14    ScriptAlias /cgi-bin/         /usr/lib/cgi-bin/
15    Alias       /Dokumentationen  /usr/share/doc
16    Redirect    /doc   http://debian2/Dokumentationen
17
18    <Location />
19      Options Indexes FollowSymLinks MultiViews
20      AllowOverride None
21      Order allow,deny
22      allow from all
23    </Location>
24
25  </VirtualHost>
```

Listing: /etc/apache2/sites-available/debian2

   Das **`*`**-Zeichen bei den Anweisungen **`NameVirtualHost`** und **`<Virtualhost>`** bewirkt, dass Apache diese Website auf allen IP-Adressen anbietet, die mit Listen-Anweisungen (z.B. in **`/etc/apache2/ports.conf`**) benannt wurden. Wenn Ihnen das zu weit geht, können Sie statt des *-Zeichens in beiden Anweisungen eine oder mehrere IP-Adressen (ggf. mit TCP-Port) (z.B. **`192.168.143.1:80 192.168.143.2:80`**) angeben.
3. Aktivieren Sie die neue Konfiguration:

```
$ a2ensite debian2
Site debian2 installed; run /etc/init.d/apache2 reload to enable.
$ tail -0f /var/log/apache2/error.log &
[1] 2186
$ invoke-rc.d apache2 reload
Reloading apache 2.0 configuration....
[Wed Jul 06 16:56:52 2005] [notice] Graceful restart requested, doing restart
[Wed Jul 06 16:56:52 2005] [notice] Apache/2.0.54 (Debian GNU/Linux) PHP/4.3.10-15 configured --
resuming normal operations
```

Listing: Aktivieren des neuen VirtualHosts

"IP-based, virtual hosts"

Es gibt komplexere Konfigurationen (z.B. in Verbindung mit mehreren Netzschnitstellen oder https-VirtualHosts), die sich nicht mehr mit den bequem einzurichtenden "Name-based, virtual hosts", sondern nur noch mit "IP-based, virtual hosts" abbilden lassen.

Für jeden "IP-based, virtual host" benötigen Sie eine IP-Adresse, die sich von den IP-Adressen aller anderen VirtualHosts auf diesem Server unterscheidet. Die IP-Adresse kann auf einer separaten Schnittstelle (z.B. **`eth1`**) liegen oder als weitere ("virtuelle" oder "Alias"-) Adresse (z.B. **`eth0:1`**) auf einer bereits verwendeten Netzschnittstelle mitlaufen.

Die Konfiguration eines "IP-based, virtual hosts" unterscheidet sich in einigen Punkten von der eines "Name based, virtual host":

Beim "IP-based, virtual hosts"...

- ... tragen Sie Im DNS zusätzlich zu dem A-Record des Hostnamens den PTR-Record der IP-Adresse ein.
- ... entfällt die **`NameVirtualHost-Anweisung`** entfällt.
- ... sollten Sie in der **`<VirtualHost>`**-Anweisung an Stelle des *-Zeichens die IP-Adressen (ggf. mit TCP-Port) der Website angeben.

Ein Beispiel einer derartigen Konfigurationsdatei finden Sie im Listing unten.

Mehr zum Thema:
Weitere Informationen zur Konfiguration von VirtualHosts finden Sie auf http://httpd.apache.org/docs-2.0/vhosts/.

https

"https" bezeichnet die Protokoll-Kombination "http-over-SSL". Dabei baut der Browser einen SSL-Kanal zu einem TCP-Port (meist Port 443) des Webservers auf und überträgt darüber http.

Der SSL-Kanal sichert die

- Vertraulichkeit (Daten sind vor Mitschneiden geschützt)
- Integrität (Daten können unterwegs nicht manipuliert werden)
- Authentizität (Schutz vor gefälschten Webservern)

der http-Kommunikation, die er überträgt.

Zusätzlich werden die Daten komprimiert, was Netzverkehr spart.

Mehr zum Thema:

- **SSL** ("Secure Sockets Layer") wurde eine zeitlang **TLS** ("Transport Layer Security") genannt.
- http://www.bsi.de/fachthem/verwpki/dokumente/BSI-SSL-Studie_34.pdf (Einführung in SSL, Unterschiede zwischen SSL und TLS)
- http://www.ietf.org/rfc/rfc2818.txt (HTTP over TLS)
- http://de.wikipedia.org/wiki/Transport_Layer_Security (Übersicht SSL)

Server-Zertifikate erstellen

Bevor Sie https auf Ihrem Webserver aktivieren können, benötigen Sie neben einer entsprechenden Apache-Konfiguration ein signiertes Server-Zertifikat (auch "SSL"- oder "X509"-Zertifikat genannt).

Das Server-Zertifikat erstellen und signieren Sie zunächst selbst:

```
$ cd /etc/apache2/ssl
$ openssl req -new -x509 -nodes -out debian1.jfranken.de.crt -keyout debian1.jfranken.de.key
Generating a 1024 bit RSA private key
.....++++++
.......++++++
writing new private key to 'debian1.jfranken.de.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [DE]:
State or Province Name (full name) [Hessen]:
Locality Name (eg, city) [Frankfurt]:
Organization Name (eg, company) [Franken EDV-Konzepte]:
Organizational Unit Name (eg, section) []:Web-Team
Common Name (eg, YOUR name) []:debian1.jfranken.de
Email Address []:
```

Listing: Ein selbstsigniertes Serverzertifikat erstellen

Achtung:
Bei der Frage nach dem "Common Name" müssen Sie den Hostnamen Ihres Webservers *exakt* so angeben, wie er später in die Adresszeile des Webbrowsers eingegeben wird.

Wenn Sie mehrere Zertifikate erstellen müssen, lohnt es sich, Default-Werte für das Land, das Bundesland, den Ort und den Firmennamen in der Datei **`/etc/ssl/openssl.cnf`** einzutragen.

Server-Zertifikate beglaubigen lassen

Damit Browser beim Aufruf Ihrer Website keine Warnung anzeigen, sollten Sie Ihr Zertifikat von einer Zertifizierungsstelle (CA) beglaubigen (d.h. signieren) lassen, deren Zertifikat bereits im Browser hinterlegt ist. Hierzu müssen Sie eine Signierungsanfrage (CSR) an die entsprechende CA senden. Einige CA (z.B. cacert.org) stellen signierte Zertifikate kostenlos zur Verfügung, während andere (z.B. VeriSign) jährlich eine Gebühr für die Verlängerung ihrer Signatur verlangen.

Achtung:
Einige Browserhersteller (z.B. Microsoft) vertrauen nur "kommerziellen" CAs. Daher müssen Nutzer des Internet-Explorers das Root-Zertifikat und die CRL von https://www.cacert.org/index.php?id=3 importieren, bevor sie die Serverzertifikate der "kostenlosen" cacert.org prüfen können.

Wenn Sie Ihr Zertifikat von einer Zertifizierungsstelle signieren lassen möchten, erstellen Sie mit **`openssl`** eine CSR zu dem Zertifikat:

```
$ cd /etc/apache2/ssl
$ openssl x509 -x509toreq -signkey debian1.jfranken.key -in debian1.jfranken.crt
Getting request Private Key
Generating certificate request
[...]
-----BEGIN CERTIFICATE REQUEST-----
MIIBXTCBxwIBADAeMRwwGgYDVQQDExNkZWJpYW4xLmpmcmFua2VuLmRlMIGfMA0G
CSqGSIb3DQEBAQUAA4GNADCBiQKBgQCvLg7abZr3ZkTMIFFrCF+g7j7InDNb9bdQ
wZrgoHUwujpH53DUku+aR0HTNoaOouOkja/uQu5OF/fBNYRjR7a7bzq41yhG0h2x
WqqdgJ2HLDgdotdGWPC5tZt4rc+OvEC01gCnzdkgMjEQ6FcBVR1JdTSmQhXQA95O
QWx519UlyQIDAQABoAAwDQYJKoZIhvcNAQEEBQADgYEAmXgXQUeZJAjBZl8NBVvp
YCUjx5/shrpMmUVcffKQu0dazS7SIbn6S9VN6Mh2DcOdbnOByIiNHNz+oUqZN33R
HT/9x6Uw3KJUOMwVYYh5kPexz7NP1bnmD21awVVpXMysIh4L51FZg5pQztvAVE5P
NYHa2SQpT+V6mMoX1ubmoBE=
-----END CERTIFICATE REQUEST-----
```

Listing: Eine Signierungsanfrage (CSR) erstellen

Die CSR (den Abschnitt ab **`-----BEGIN`**) kopieren Sie in die Zwischenablage und senden sie per E-Mail oder Webformular an die CA:

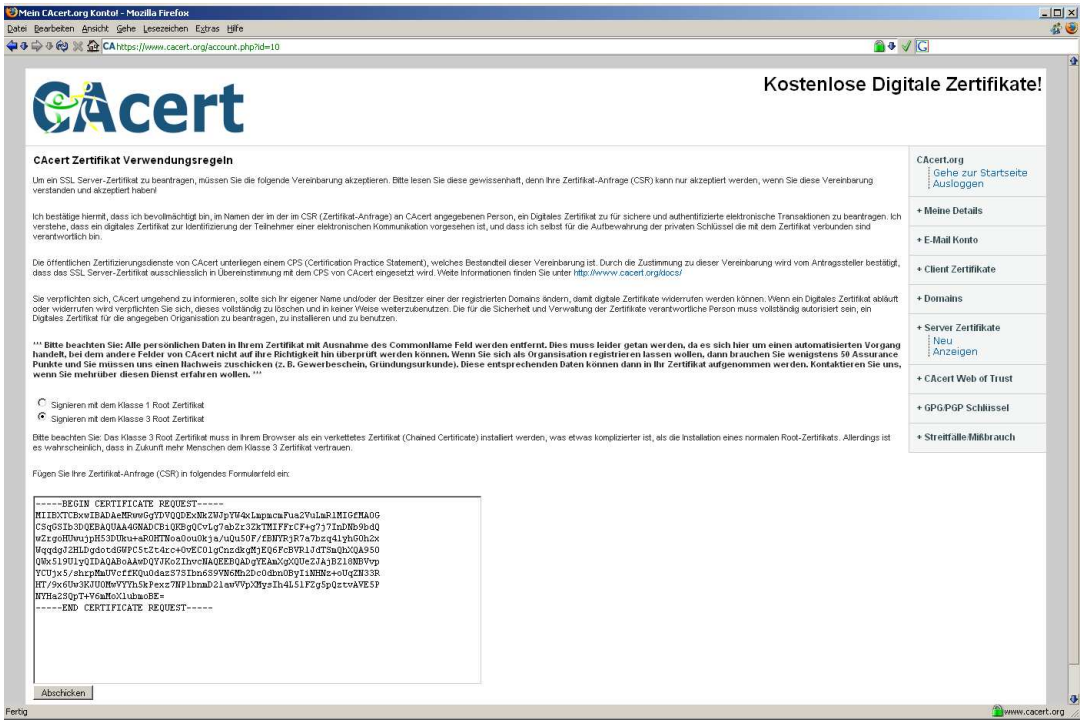

Abbildung: Eingabeformular einer Certification Authority (CA)

Im Gegenzug sendet Ihnen die CA Ihr signiertes Zertifikat zurück (siehe Abbildung), mit dem Sie das selbstsignierte Zertifikat in **`/etc/apache2/ssl/debian1.jfranken.de.crt`** ersetzen:

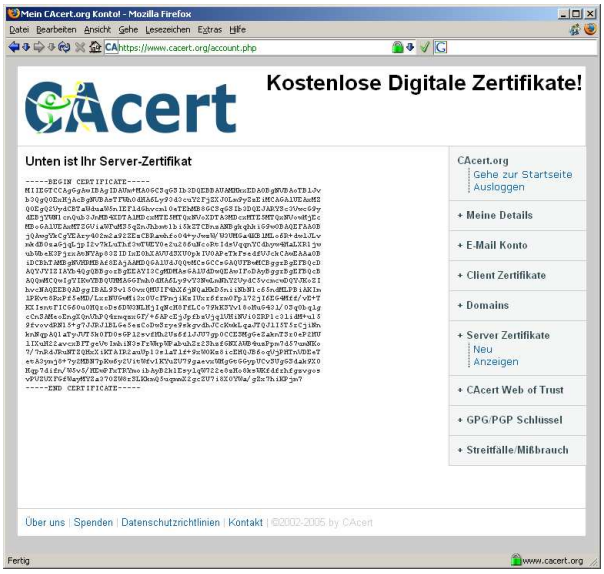

Abbildung: Das von der CA signierte Zertifikat

VirtualHost-Konfiguration

Legen Sie im Verzeichnis **`/etc/apache2/sites-available`** eine Konfigurationsdatei für den VirtualHost an, den Sie per https erreichen möchten:

```
01  Listen 192.168.134.2:443
02  <VirtualHost 192.168.134.2:443>
03    ServerName debian1.jfranken.de
04    DocumentRoot /var/www2
05    ServerAdmin webmaster@jfranken.de
06
07    # SSL
08    SSLEngine On
09    SSLCipherSuite HIGH:MEDIUM
10    SSLCertificateFile    /etc/apache2/ssl/debian1.jfranken.de.crt
11    SSLCertificateKeyFile /etc/apache2/ssl/debian1.jfranken.de.key
12
13    # Logfiles:
14    CustomLog /var/log/apache2/access-debian1.jfranken.de combined
15    ErrorLog /var/log/apache2/error-debian1.jfranken.de
16    LogLevel warn
17
18    <Location />
19      Options Indexes FollowSymLinks MultiViews
20      AllowOverride None
21      Order allow,deny
22      allow from all
23    </Location>
24
25  </VirtualHost>
```

Listing: /etc/apache2/sites-available/debian1

Aktivieren Sie die neue Konfiguration:

```
$ a2ensite debian1
Site debian1 installed; run /etc/init.d/apache2 reload to enable.
$ a2enmod ssl
Module ssl installed; run /etc/init.d/apache2 force-reload to enable.
$ tail -0f /var/log/apache2/error* &
[1] 3402
$ invoke-rc.d apache2 force-reload
Forcing reload of apache 2.0 web server: Apache2
==> /var/log/apache2/error.log <==
[Wed Jul 06 22:15:22 2005] [notice] caught SIGTERM, shutting down
[Wed Jul 06 22:15:23 2005] [notice] Apache/2.0.54 (Debian GNU/Linux) PHP/4.3.10-15 mod_ssl/2.0.54 OpenSSL/0.9.7e configured -- resuming normal operations

$ kill %%
[1]+  Beendet    tail -0f /var/log/apache2/error.log /var/log/apache2/error-debian1.jfranken.de
```

Listing: Apache für https konfigurieren

Zugriffsschutz

Sie können Ihre Website vor unberechtigtem Zugriff schützen, indem Sie Apache so konfigurieren, dass er unautorisierte Zugriffe auf bestimmte Dateien oder Verzeichnisse ablehnt. Die Autorisierung bezieht sich auf eine Authentisierung (Identifikation) des Clients, die der Webserver mittels eines Authentifizierungs-Mechanismus (z.B. Passwortvergleich) authentifizieren (überprüfen) kann. Man unterscheidet zwischen

- starker Authentisierung, die sich auf fest verdrahtete Eigenschaften des Clients bezieht (z.B. IP-Adresse oder Hardware-Dongles), und
- schwacher Authentisierung, auf die der Anwender Einfluss nehmen kann (z.B. durch Eingeben von Passwörtern oder Installieren von Client-Zertifikaten).

Authentisierung per IP-Adresse

Wenn Sie Webseiten vor Zugriffen unberechtigter Clients schützen möchten, ersetzen Sie in der Konfigurationsdatei des entspr. VirtualHosts bei der Anweisung **`Allow from all`** das **`all`** durch eine Liste berechtigter IP-Adressen, -Netze oder Domains. Apache lehnt dann alle unberechtigten Clients mit der Meldung **`403 Forbidden`** ab.

Abbildung: 403 Forbidden

Beispiel: Ersetzen Sie die Zeilen 10-18 der Datei **`/etc/apache2/sites-available/default`** (siehe oben) mit den folgenden:

```
<Directory /var/www/>
  Options Indexes FollowSymLinks MultiViews
  AllowOverride None
  Order allow,deny

  # Einzelne IP-Adressen freischalten
  Allow from 192.168.134.2 192.168.134.3

  # Ganze IP-Netze freischalten
  Allow from 192.168.1 192.168.2
  Allow from 10.1.0.0/16
  Allow from 10.2.0.0/255.255.0.0

  # Freischalten fast aller Rechner einer DNS-Domain:
  Allow from apache.org
  Deny from foo.apache.org
</Directory>
```

Listing: Anweisungen zur starken Authentifizierung

Mehr zum Thema:
Die Dokumentation der **`Order`**-, **`Allow`**- und **`Deny`**-Anweisungen finden Sie auf http://httpd.apache.org/docs/2.0/mod/mod_access.html.

Authentisierung per Passwort

Wenn Sie Webseiten nur Anwendern zugänglich machen möchten, die eine vorher festgelegte Kombination aus Username und Passwort kennen, ergänzen Sie in der Konfigurationsdatei des entspr. VirtualHosts (z.B. **`/etc/apache2/sites-available/default`**, siehe Listing) die bestehenden **`Allow from`**-Anweisungen mit den folgenden:

```
AuthType Basic
AuthName "Geschuetzter Bereich"
AuthUserFile /etc/apache2/htpasswd
Require valid-user
```

Listing: Anweisungen zur Passwortabfrage (Autorisierung aller bekannten User)

Dies bewirkt, dass der Webserver auf alle Anfragen (außer denen, die eine in **`/etc/apache2/htpasswd`** vorkommende Username/Passwort-Kombination enthalten) mit einer 401-Seite antwortet.

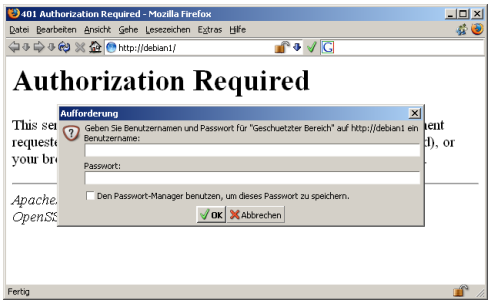

Abbildung: 401 Authorization Required

Den ersten Eintrag der Passwortdatei legen Sie mit dem Kommando **`htpasswd -c /etc/apache2/htpasswd *Username*`** an.
Bei weiteren Einträgen lassen Sie das **`-c`** weg (sonst verlieren Sie alle bisherigen Einträge).
Mit **`htpasswd -D /etc/apache2/htpasswd *Username*`** löschen Sie einen Eintrag aus der Passwortdatei.

Sie können statt *aller* User (**`Require valid-user`**) einzelne User explizit autorisieren:

```
Require user1 user2 user3
```

Listing: Explizite Autorisierung mehrerer User

oder die Benutzer in Gruppen zusammenfassen die Gruppen autorisieren:

```
# Anweisungen in der Apache-Konfigurationsdatei:
AuthGroupFile /etc/apache2/htgroups
Require group Vertrieb

# Einträge in /etc/apache2/htgroups:
Vertrieb: User1 User2
IT: User2 User3 User4
```

Listing: Autorisierung von Benutzergruppen

Mehr zum Thema:
Die Dokumentation der vorgestellten Konfigurationsanweisungen finden Sie auf:
http://httpd.apache.org/docs/2.0/mod/core.html#authtype
http://httpd.apache.org/docs/2.0/mod/core.html#authname
http://httpd.apache.org/docs/2.0/mod/mod_auth.html
http://httpd.apache.org/docs/2.0/mod/core.html#require

Authentisierung per Client-Zertifikat

Apache bietet die Möglichkeit, Zugriffe nur den Clients zu gewähren, die über ein berechtigtes Client-Zertifikat (auch "SSL-", "X509-" oder "User-"Zertifikat genannt) verfügen. Dazu muss ein entsprechendes Client-Zertifikat vorher

1. hergestellt,
2. mit dem Schlüssel einer CA, deren Zertifikat auf dem Webserver hinterlegt ist, signiert,
3. auf dem Webserver autorisiert und
4. in den Browser importiert

worden sein.

Es gibt verschiedene Tools, die diese Aufgaben automatisieren. Der folgende Abschnitt beschreibt das *manuelle* Ausführen der erforderlichen Schritte.

Eine Root-CA einrichten

Zum Signieren von Zertifikaten benötigen Sie die Dienste einer CA. Im einfachsten Fall ist dies eine lokale Root-CA, die Sie mit folgenden Befehlen auf dem Webserver anlegen:

```
$ cd /etc/ssl
$ echo 1001 > serial
$ touch index.txt
$ mkdir newcerts csr

$ vi openssl.cnf
    # openssl.cnf wie folgt bearbeiten:
    # dir auf /etc/ssl setzen
    # countryName_default             = DE
    # stateOrProvinceName_default     = Hessen
    # 0.organizationName_default      = Franken EDV-Konzepte
    # localityName_default            = Frankfurt

$ openssl req -new -x509 -keyout private/cakey.pem -out cacert.pem
Generating a 1024 bit RSA private key
..........++++++
.......................++++++
writing new private key to 'privkey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [DE]:
State or Province Name (full name) [Hessen]:
Locality Name (eg, city) [Frankfurt]:
Organization Name (eg, company) [Franken EDV-Konzepte]:
Organizational Unit Name (eg, section) []:CA
Common Name (eg, YOUR name) []:Franken CA
Email Address []:ca@jfranken.de
```

Listing: Anlegen einer Root-CA

Dabei entsteht das Root-Zertifikat Ihrer CA (**`cacert.pem`**) sowie ein per Passphrase geschützer Schlüssel (**`cakey.pem`**).

Achtung:
Bewahren Sie die Passphrase an einem sicheren Ort auf. Sie benötigen die Passphrase später zum Signieren und Zurückziehen von Client-Zertifikaten.

Legen Sie eine "Certificate-Revocation-List" (**`cacert.crl`**) an, in der Sie später die Zertifikate vermerken können, die Apache als "ungültig" betrachten soll. Auf diese Weise können Sie ausgestellte Zertifkate vor dem Ablauf ihrer Gültigkeit annulieren.

```
$ openssl ca -gencrl -out cacert.crl
```

Listing: Anlegen der CRL

Sie können das Zertifikat und die CRL im Web veröffentlichen, damit Clients sie importieren können:

```
$ cp /etc/ssl/cacert.pem /var/www/FrankenCA-Root-Zertifikat.crt
$ ln -sf /etc/ssl/cacert.crl /var/www/FrankenCA-Revocationlist.crl
```

Listing: Veröffentlichen der Root-CA/CRL

Achtung:
Veröffentlichen Sie niemals den Schlüssel (**`cakey.pem`**).

Client-Zertifikate erstellen

Bevor Sie ein Client-Zertifikat herstellen können, müssen Sie zuerst die Informationen über den zukünftigen Nutzer in Form einer Zertifikatsanforderung (CSR-Datei) notieren:

```
$ cd /etc/ssl
$ openssl req -new -nodes -out csr/jfranken.csr -keyout private/jfranken.key
Generating a 1024 bit RSA private key
..............++++++
....++++++
writing new private key to 'private/jfranken.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [DE]:
State or Province Name (full name) [Hessen]:
Locality Name (eg, city) [Frankfurt]:
Organization Name (eg, company) [Franken EDV-Konzepte]:
Organizational Unit Name (eg, section) []:IT-Abteilung
Common Name (eg, YOUR name) []:Johannes Franken
Email Address []:jfranken@jfranken.de

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Listing: Erstellen einer Client-Zertifikatsanforderung (CSR-Datei)

Erstellen Sie das Client-Zertifikat (PEM-Datei), indem Sie die Zertifikatsanforderung mit dem Schlüssel Ihrer CA signieren. Hierzu benötigen Sie die Passphrase des Schlüssels.

```
$ openssl ca -in csr/jfranken.csr
Using configuration from /usr/lib/ssl/openssl.cnf
Enter pass phrase for /etc/ssl/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
    Serial Number: 4097 (0x1001)
    Validity
        Not Before: Jul 24 22:01:36 2005 GMT
        Not After : Jul 24 22:01:36 2006 GMT
    Subject:
        countryName               = DE
        stateOrProvinceName       = Hessen
        organizationName          = Franken EDV-Konzepte
        organizationalUnitName    = IT-Abteilung
        commonName                = Johannes Franken
        emailAddress              = jfranken@jfranken.de
    X509v3 extensions:
        X509v3 Basic Constraints:
            CA:FALSE
        Netscape Comment:
            OpenSSL Generated Certificate
        X509v3 Subject Key Identifier:
            53:23:97:FC:B5:E4:8D:7A:3E:B2:05:4C:C5:33:74:27:F4:F5:48:2D
        X509v3 Authority Key Identifier:
            keyid:E2:93:41:5C:89:C9:3C:81:42:6D:3C:76:CF:49:1F:8A:91:5F:4E:FC
            DirName:/C=DE/ST=Hessen/L=Frankfurt/O=Franken EDV-Konzepte/OU=CA
                    /CN=Franken CA/emailAddress=jfranken@jfranken.de
            serial:B0:1F:97:8F:4A:C3:84:07

Certificate is to be certified until Jul 24 22:01:36 2006 GMT (365 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
[...]
Data Base Updated
```

Listing: Erstellen eines Client-Zertifikats (PEM-Datei)

Das Zertifikat liegt nun in **`/etc/apache2/ssl/newcerts/1001.pem`**. Der Dateiname ergibt sich aus der hexadezimalen Seriennummer des Zertifikats. Wandeln Sie die PEM-Datei in das PFX-Format (auch PKCS12- oder P12-Format genannt) um, das für Browser bekömmlicher ist:

```
$ openssl pkcs12 -export \
  -in        newcerts/1001.pem  \
  -inkey     private/jfranken.key \
  -certfile  cacert.pem \
  -name      "Johannes Franken => FrankenEDV" \
  -out       newcerts/1001.p12
Enter Export Password:
Verifying - Enter Export Password:
```

Listing: Umwandeln PEM- zu PFX-Datei

Die PFX-Datei wurde mit einem Passwort verschlüsselt und liegt in **`/etc/apache2/ssl/newcerts/1001.p12`**. Geben Sie die Datei und das Passwort an den berechtigen User weiter.

Client-Zertifikate autorisieren

Aktivieren Sie zunächst SSL (wie oben beschrieben) und die Abfrage von Client-Zertifikaten (mit folgenden Zeilen) in der Konfigurationsdatei des VirtualHosts:

```
# Client-Zertifikatsabfrage aktivieren:
SSLCACertificateFile /etc/ssl/cacert.pem
SSLCARevocationFile /etc/ssl/cacert.crl
SSLVerifyClient require
SSLUserName SSL_CLIENT_S_DN_CN
```

Listing: Client-Zertifikatsabfrage aktivieren

In der Konfigurationsdatei des VirtualHosts können Sie z.B. mit **<Directory>**-Blocks für jedes Verzeichnis festlegen, welche Zertifikate zum Zugriff erforderlich sind. Dabei können Sie die Zertifikate entweder

1. über ihre *Eigenschaften* (z.B. "OU=’IT-Abteilung’") oder
2. *direkt*, d.h. über den gesamten DN (siehe Beispiel unten)

referenzieren.

Zertifikate über ihre Eigenschaften referenzieren

Fügen Sie der Konfigurationsdatei des VirtualHosts folgende Zeilen hinzu:

```
<Location />
  SSLOptions +ExportCertData
  SSLRequire (%{SSL_CLIENT_S_DN_OU} in {"IT-Abteilung", "Vorstand"})
# [...]
</Location>
```

Listing: SSLRequire-Beispiel

Mehr zum Thema:
Die Dokumentation und weitere Beispiele zur **SSLRequire**-Anweisung finden Sie auf http://httpd.apache.org/docs/2.0/de/mod/mod_ssl.html#sslrequire.

Zertifikate über eine htpasswd-Datei direkt referenzieren

Fügen Sie der Konfigurationsdatei des VirtualHosts folgende Zeilen hinzu:

```
<Location />
  SSLOptions    +FakeBasicAuth
  AuthName      "Snake Oil Authentication"
  AuthType      Basic
  AuthUserFile  /etc/apache2/htpasswd.debian1
  require       valid-user
# [...]
</Location>
```

Listing: FakeBasicAuth-Beispiel

Schreiben Sie die DNs aller erlaubten Zertifikate in die Datei **/etc/apache2/htpasswd.debian1** und hängen Sie jeweils **:xxj31ZMTZzkVA** an.

```
/C=DE/ST=Hessen/O=Franken EDV-Konzepte/OU=IT-Abteilung/CN=Johannes \
Franken/emailAddress=jfranken@jfranken.de:xxj31ZMTZzkVA
```

*Listing: **/etc/apache2/htpasswd.debian***

Der Passwordhash **xxj31ZMTZzkVA** ist eine Verschlüsselung des Wortes **password** (probieren Sie mal **openssl passwd -crypt -salt xx password**) und im **FakeBasicAuth**-Modus das Erkennungszeichen dafür, dass dieser Eintrag ein Client-Zertifikat referenziert.

Kombinationen

IP-Adresse und Passwort/Zertifikat erforderlich

Wenn Sie Ihre Website besonders gut schützen möchten, können Sie den Zugriff nur den Clients erteilen, die

sowohl
: über ein passendes Passwort oder Client-Zertifikat verfügen (schwache Authentisierung)

als auch
: von einer freigeschalteten IP-Adresse aus anrufen (starke Authentisierung).

Hierzu erweitern Sie die **<Directory>**- oder **<Location>**-Blocks der Konfigurationsdatei Ihres VirtualHosts (z.B. **/etc/apache2/sites-available/default**, siehe Listing) wie folgt:

```
<Location />    # z.B. alles unterhalb "/" schützen
  # [...]
  Satisfy All   # Sowohl starke als auch schwache Auth. erforderlich

  # Starke Authentisierung:
  Order Allow, Deny  # das bedeutet: "Deny" ist Default-Policy.
  Allow from 192.168.134.2 192.168.134.3
  Allow from 10.1.0.0/16

  # Schwache Authentisierung:
  AuthType Basic
  AuthName "Geschuetzter Bereich"
  AuthUserFile /etc/apache2/htpasswd
  Require valid-user

  # [...]
</Location>
```

Listing: IP-Adresse und Passwort erforderlich

Passwort/Zertifikat nur "von draussen" erforderlich

Wenn Sie Ihre Website nur bei Zugriffen von "draussen" mit einem Passwort/Zertifikat schützen möchten, erweitern Sie die **<Directory>**- oder **<Location>**-Blocks der Konfigurationsdatei Ihres VirtualHosts (z.B. **/etc/apache2/sites-available/default**, siehe Listing) wie folgt:

```
<Location />    # z.B. alles unterhalb "/" schützen
  # [...]
  Satisfy Any   # Entweder starke oder schwache Auth. erforderlich

  # Starke Authentisierung:
  Order Allow, Deny  # das bedeutet: "Deny" ist Default-Policy.
  Allow from 192.168.134.2 192.168.134.3
  Allow from 10.1.0.0/16

  # Schwache Authentisierung:
  AuthType Basic
  AuthName "Geschuetzter Bereich"
  AuthUserFile /etc/apache2/htpasswd
  Require valid-user

  # [...]
</Location>
```

Listing: IP-Adresse oder Passwort erforderlich

Mehr zum Thema:
Die Dokumentation der **`satisfy`**-Anweisung finden Sie auf:
http://httpd.apache.org/docs/2.0/mod/core.html#satisfy

Apache-Module

Sie können die Funktionalität des Apache Webservers durch das Laden sog. "Module" erweitern. Wenn Sie **`apache2 -l`** aufrufen, erhalten Sie eine Liste der *statischen* (d.h. bereits in den Apache einkompilierten) Module:

```
$ apache2 -l
Compiled in modules:
  core.c
  mod_access.c
  mod_auth.c
  mod_log_config.c
  mod_logio.c
  mod_env.c
  mod_setenvif.c
  prefork.c
  http_core.c
  mod_mime.c
  mod_status.c
  mod_autoindex.c
  mod_negotiation.c
  mod_dir.c
  mod_alias.c
  mod_so.c
```

Listing: Statische Module

Einige *dynamische* Module (DSOs) sind in Form von **`mod_*.so`**-Dateien im Debian-Paket **`apache2-common`** enthalten und somit bereits auf Ihrer Festplatte installiert. Sie können diese Module mit den Programmen **`a2enmod`** und **`a2dismod`** auflisten, auswählen und abschalten:

```
$ a2enmod
Which module would you like to enable?
Your choices are: actions asis auth_anon auth_dbm auth_digest auth_ldap cache cern_meta cgid cgi
dav_fs dav deflate disk_cache expires ext_filter file_cache headers imap include info ldap
mem_cache mime_magic php4 proxy_connect proxy_ftp proxy_http proxy rewrite speling ssl suexec
unique_id userdir usertrack vhost_alias
Module name?
```

Listing: a2enmod

Mehr zum Thema:
`a2enmod` und **`a2dismod`** ersetzen das von Debian/Apache-1.3 bekannte **`apache-modconf`**-Kommando.

Eine Liste zusätzlicher Apache-Module, die als Debian-Pakete auf den Debian-FTP-Servern bereitstehen, können Sie mit **`aptitude`** ausgeben:

```
$ aptitude update
[...]
$ aptitude search apache2-mod-
p   libapache2-mod-auth-kerb   - Apache2 module for Kerberos
p   libapache2-mod-auth-mysql  - Apache 2 module for MySQL
[...]
```

Listing: Liste verfügbarer Apache-Module

Die folgenden Abschnitte befassen sich mit der Konfiguration einiger häufig eingesetzter Module.

deflate

Das Modul **mod_deflate** ermöglicht dem Apache-Webserver, Antworten vor der Übertragung zu komprimieren. Bei langsamen Internet-Verbindungen kann dies die Darstellung von Webseiten beschleunigen.

Sie können die Kompression global aktivieren oder auf einzelne VirtualHosts, Verzeichnisse, Dateitypen, Browser usw. einschränken.

Legen die Datei **/etc/apache2/mods-available/deflate.conf** mit folgendem Inhalt an, um die Kompression global zu aktivieren (mit Ausnahme von Bilddateien und bei veralteten Browsern, welche die Komprimierung nicht beherrschen):

```
01  # Das Filtermodul aktivieren.
02  SetOutputFilter DEFLATE
03
04  # Einige Browser verstehen kein gzip:
05  BrowserMatch ^Mozilla/4 gzip-only-text/html
06  BrowserMatch ^Mozilla/4\.0[678] no-gzip
07
08  # Andere schon...
09  BrowserMatch \bMSIE !no-gzip !gzip-only-text/html
10
11  # Bilddateien nicht komprimieren
12  SetEnvIfNoCase Request_URI \.(?:gif|jpe?g|png)$ no-gzip dont-vary
13
14  # "Vary: Accept-Encoding"-Header einfuegen (wichtig bei Proxys)
15  # (benötigt das headers-Modul)
16  Header append Vary User-Agent env=!dont-vary
```

Listing: /etc/apache2/mods-available/deflate.conf

Aktivieren Sie die Module im Apache:

```
$ a2enmod headers
Module headers installed; run /etc/init.d/apache2 force-reload to enable.
$ a2enmod deflate
Module deflate installed; run /etc/init.d/apache2 force-reload to enable.
$ invoke-rc.d apache2 force-reload
Forcing reload of apache 2.0 web server...
```

Listing: Die deflate-Konfiguration aktivieren

Mehr zum Thema:
siehe http://httpd.apache.org/docs-2.0/mod/mod_deflate.html

server-status

Das **server-status**-Modul zeigt die aktuelle Auslastung des Webservers sowie die Werte einiger Zähler an:

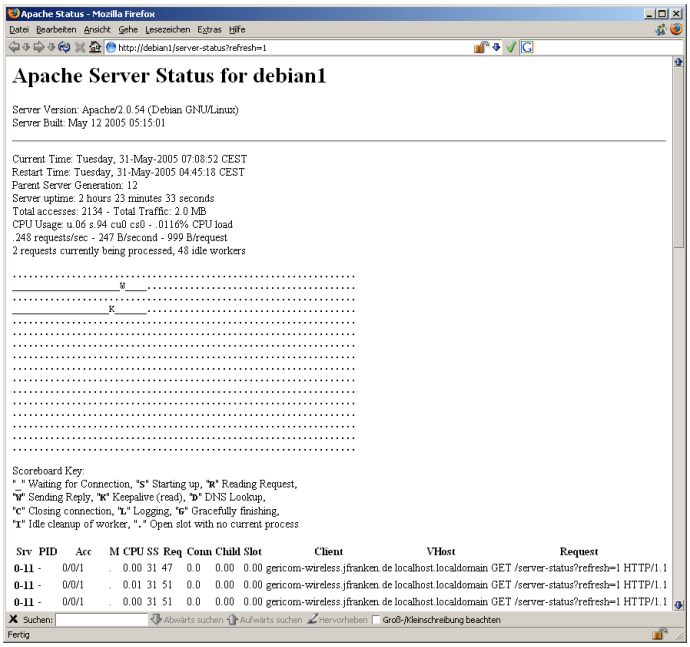

Abbildung: /server-status?refresh=1

Bevor Sie das **server-status**-Modul aufrufen können, müssen Sie es konfigurieren:

```
$ touch /etc/apache2/mods-available/status.load
$ cat > /etc/apache2/mods-available/status.conf <<EOF
<Location /server-status>
    SetHandler server-status
    # Wenn Sie die Seite vor fremden Augen schützen möchten:
    Order deny,allow
    Deny from all
    Allow from .IHRE.DOMAIN.COM
</Location>
EOF
$ a2enmod status
Module status installed; run /etc/init.d/apache2 force-reload to enable.
$ invoke-rc.d apache2 force-reload
Forcing reload of apache 2.0 web server...
```

Listing: server-status konfigurieren

Weil das **server-status**-Modul bereits in den Apache einkompiliert ist, benötigen Sie keine **LoadModule**-Anweisung. Die leere **status.load**-Datei ist für **a2enmod** erforderlich.

Mehr zum Thema:
siehe http://httpd.apache.org/docs-2.0/mod/mod_status.html

server-info

Mit dem **server-info**-Modul können Sie die Konfiguration des laufenden Apache und aller Module anzeigen.

Abbildung: /server-info

Bevor Sie das **server-info**-Modul aufrufen können, müssen Sie es konfigurieren:

```
$ cat > /etc/apache2/mods-available/info.conf <<EOF
<Location /server-info>
    SetHandler server-info
    # Wenn Sie die Seite vor fremden Augen schützen möchten:
    Order deny,allow
    Deny from all
    Allow from .IHRE.DOMAIN.COM
</Location>
EOF
$ a2enmod info
Module info installed; run /etc/init.d/apache2 force-reload to enable.
$ invoke-rc.d apache2 force-reload
Forcing reload of apache 2.0 web server...
```

Listing: server-info konfigurieren

Die Datei **mods-available/info.load** (aus dem **apache2-common**-Paket) wird beim **a2enmod** automatisch nach **mods-enabled** gelinkt.

Mehr zum Thema:
siehe http://httpd.apache.org/docs-2.0/mod/mod_info.html

libphp4 (PHP-Interpretierer)

PHP ist eine Skriptsprache, die aufgrund ihrer vielen Funktionsbibliotheken sehr beliebt und weit verbreitet ist. Das Apache-Modul **mod_php4** interpretiert PHP-Code, der in Webseiten untergebracht ist.

Installation des Apache-Moduls und Anlegen einer Testdatei:

```
$ aptitude install libapache2-mod-php4
[...]
Die folgenden Pakete werden zusätzlich automatisch installiert:
  libzzip-0-12 php4-common
Die folgenden Pakete werden zusätzlich installiert:
  libapache2-mod-php4 libzzip-0-12 php4-common
0 Pakete aktualisiert, 3 zusätzlich installiert,
0 werden entfernt und 0 nicht aktualisiert.
Muss 1813kB an Archiven herunterladen.Nach dem Entpacken werden 3521kB zusätzlich belegt sein.
Wollen Sie fortsetzen? [Y/n/?] Y
[...]
Forcing reload of apache 2.0 web server...

$ echo '<?php phpinfo() ?>'   >> /var/www/test.php
```

Listing: Installation von PHP4

Leider ist PHP4 nicht kompatibel zum **worker**-MPM. Daher wird das Debian-Paketsystem Ihren Webserver bei der Installation des PHP-Moduls automatisch auf das (langsamere) **prefork**-MPM umrüsten, wenn auf bisher das **worker**-MPM installiert war.

Rufen Sie die Testdatei im Browser auf. Die **phpinfo()**-Funktion gibt eine Liste der einkompilierten und verfügbaren PHP-Module aus:

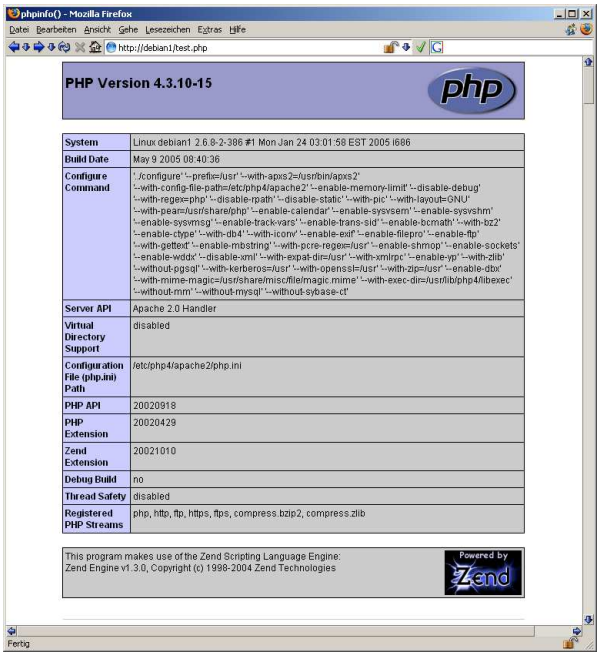

Abbildung: Aufruf der PHP-Testdatei

Sie können den PHP-Interpreter und die PHP-Module in der Datei **/etc/php4/apache2/php.ini** konfigurieren.

Achtung:
Änderungen an der **php.ini** werden erst nach einem Neustart des Webservers (z.B. mit **apache2ctl graceful**) aktiv.

PHP-Module und PEAR

Eine Liste der als Debian-Pakete verfügbaren PHP-Module erhalten Sie mit **`aptitude search ^php{4,}-`** :

```
$ aptitude search ^php{4,}-
p   php-auth    - PHP PEAR modules for creating an authentication system
[...]
p   php4-xslt   - XSLT module for php4
v   php4-yaz    -
```

Listing: Verfügbare Debian-paketierte PHP-Module

Besonders hervorzuheben ist das PEAR-Paket, das den Zugriff auf eine grosse Sammlung weiterer PHP-Module ermöglicht.

Installation des PEAR-Pakets:

```
$ aptitude install php4-pear
[...]
Die folgenden Pakete werden zusätzlich automatisch installiert:
  php4-cli
Die folgenden Pakete werden zusätzlich installiert:
  php4-cli php4-pear
0 Pakete aktualisiert, 2 zusätzlich installiert,
0 werden entfernt und 0 nicht aktualisiert.
Muss 1859kB an Archiven herunterladen.
Nach dem Entpacken werden 4944kB zusätzlich belegt sein.
Wollen Sie fortsetzen? [Y/n/?] y
[...]
Richte php4-pear ein (4.3.10-15) ...
$ apache2ctl graceful
```

Listing: Installation von PEAR

Die unter PEAR veröffentlichten PHP-Module können Sie mit dem **`pear`**-Kommando verwalten. Mit dem **`pear`**-Kommando können Sie z.B.

- eine Liste bereits installierter PEAR-Module ausgeben (**`pear list`**)
- Eine Liste von Modulen ausgeben, die Sie aus dem Internet laden können (**`pear list-all`**)
- Die Beschreibung eines Moduls aus dem Internet ausgeben (**`pear remote-info Image_Barcode`**)
- Ein Modul aus dem Internet installieren (**`pear install Modulname`**)
- Alle PEAR-Module automatisch auf den aktuellen Stand bringen (**`pear upgrade-all`**)

Wenn Sie **`pear`** ohne Parameter aufrufen, erhalten Sie eine Liste aller Möglichkeiten.

mod_perl (Perl-Interpretierer)

Mit dem Apache-"Perl"-Modul können die Ladezeit von CGI-Scripts, die in der Programmiersprache Perl geschrieben sind, wesentlich reduzieren. Das Modul bewirkt, dass der Perl-Interpretierer beim Start des Webservers in den Webserver integriert und nicht mehr bei jedem Aufruf eines CGI-Scripts erneut geladen wird.

Installation:

```
$ aptitude install libapache2-mod-perl2
[...]
Die folgenden Pakete werden zusätzlich installiert:
  libapache2-mod-perl2 libcompress-zlib-perl libdevel-symdump-perl
  libfont-afm-perl libhtml-format-perl libhtml-parser-perl
  libhtml-tagset-perl libhtml-tree-perl libmailtools-perl libperl5.8
  libtimedate-perl liburi-perl libwww-perl
Die folgenden Pakete werden aktualisiert:
  perl perl-base perl-modules
Die folgenden Pakete werden EMPFOHLEN, aber NICHT installiert:
  perl-doc
3 Pakete aktualisiert, 13 zusätzlich installiert,
0 werden entfernt und 0 nicht aktualisiert.
Muss 8603kB an Archiven herunterladen.
Nach dem Entpacken werden 8434kB zusätzlich belegt sein.
Wollen Sie fortsetzen? [Y/n/?] y
[...]
Hole:7 http://ftp.freenet.de stable/main libapache2-mod-perl2 1.999.21-1 [642kB]
[...]
Richte libapache2-mod-perl2 ein (1.999.21-1) ...
Module perl installed; run /etc/init.d/apache2 force-reload to enable.
[...]
```

*Listing: Installation von **mod_perl***

Achtung:
Wenn Sie Debian in der Version "Sarge" einsetzen, werden Sie feststellen, dass die Version 1.999 von **libapache2_mod_perl2** defekt ist ("**Can't locate Apache.pm in @INC**"). Ich empfehle Ihnen das Upgrade auf einen Backport der Version 2.0 aus Debian "Etch".

```
$ wget http://packages.aquabolt.com/dists/sarge/main/binary-i386/libapache2-mod-perl2_2.0.1-4.0.aquabolt.2_i386.deb
$ wget http://packages.aquabolt.com/dists/sarge/main/binary-i386/libcgi-pm-perl_3.15-0.aquabolt.1_i386.deb
$ dpkg -r libapache2-mod-perl2
$ dpkg -i libcgi-pm-perl_3.15-0.aquabolt.1_i386.deb
$ dpkg -i libapache2-mod-perl2_2.0.1-4.0.aquabolt.2_i386.deb
$ a2enmod perl
```

*Listing: Upgrade auf **mod_perl** 2.0 (nur bei Debian Sarge erforderlich)*

Konfiguration:
Sie müssen Apache mitteilen, welche Dateien er an **mod_perl** übergeben soll. Das geht z.B. mit **<Directory>**-, **<Location>**- und **<Files>**-Anweisungen.

Die folgende Konfiguration bewirkt, dass Apache alle Dateien mit der Endung **.pl** an mod_perl übergibt.

```
$ cat <<EOF > /etc/apache2/mods-available/perl.conf
PerlModule ModPerl::Registry
<Files *.pl>
        AddHandler perl-script .pl
        PerlSendHeader On
        PerlResponseHandler ModPerl::Registry
        Options +ExecCGI
</Files>
EOF
```

*Listing: Konfiguration von **mod_perl***

Aktivieren Sie die neue Konfiguration:

```
$ tail -0f /var/log/apache2/error.log &
[1] 2402
$ invoke-rc.d apache2 force-reload
Forcing reload of apache 2.0 web server...
[Sun Jan 22 19:40:33 2006] [notice] caught SIGTERM, shutting down.
[Sun Jan 22 19:40:35 2006] [notice] Apache/2.0.55 (Debian)
PHP/4.4.0-4 mod_ssl/2.0.55 OpenSSL/0.9.8a mod_perl/2.0.1 Perl/v5.8.7
configured -- resuming normal operations
$ kill %%
[1]+  Beendet                 tail -0f /var/log/apache2/error.log
```

Listing: Restart des Webservers

Sie können folgendes Perl-Script zum Testen verwenden:

```
$ cat <<EOF >/usr/lib/cgi-bin/test.pl
#!/usr/local/bin/perl -w
use strict;
use CGI;
sleep 10;
my $q = CGI->new;
print $q->header,
      $q->start_html,
      $q->h1('Hello World!'),
      'This is perl',
      $q->end_html;
EOF
```

Listing: Anlegen eines Perl-CGI-Scripts

Wenn Sie das Script im Browser aufrufen, erscheint nach zehn Sekunden die Meldung "Hello World!". Rufen Sie während der Wartezeit auf dem Webserver **`ps -ef|grep perl`** auf. Wenn die Liste *keinen* **`test.pl`**-Prozess enthält, ist **`mod_perl`** korrekt installiert.

Mehr zum Thema:
siehe http://perl.apache.org/

mod_jk ("Tomcat-Modul")

Die Installation von **`mod_jk`** wird weiter unten beschrieben.

Java Applicationserver

Immer mehr Anwendungen werden in der Programmiersprache "Java" erstellt. Dieser Abschnitt beschreibt die Installation der Komponenten, die zur Integration von *Java-Anwendungen* in den Apache-Webserver benötigt werden.

Die Java-Anwendung kommuniziert mit den Browsern über ein **`servlet`**-Objekt, das über einen Servlet-Container (hier: "Tomcat") im Netz bereitgestellt wird.

Eine Java Virtual Machine installieren [Sun J2SE 5.0]

Java-Anwendungen werden nicht direkt vom Betriebssystem. sondern von einer Software ("Java Virtual Machine", JVM) ausgeführt, welche die speziellen Eigenschaften der zugrundeliegenden Hardware und des Betriebssystems abstrahiert. Es gibt verschiedene JVM-Implementierungen, von denen einige als Open-Source-Software erhältlich sind, während andere zu einem der drei Java-Standards kompatibel sind.

Die Firma Sun Microsystems hat folgende Java-Standards definiert:

- "Java2 Standard Edition" (J2SE) für Workstations
- "Java2 Micro Edition" (J2ME) für PDAs und Mobiltelefone
- "Java2 Enterprise Edition" (J2EE) für Applicationserver

Das Debian-Projekt bietet aus lizenzrechtlichen Gründen auf seinen FTP- und Webservern nur OpenSource-JVM an (die bisher zu keinem der Java-Standards kompatibel sind).

Server	Debian-Paket	Beschreibung
`ftp.debian.org` (Sarge)	**`gcj-4.0`**	The GNU compiler for Java(TM)
	`kaffe`	A JVM to run Java bytecode
	`sablevm`	Free implementation of Java Virtual Machine (JVM) second edition

Tabelle: OpenSource-JVM, die als Debian-Paket erhältlich sind

Wenn Sie Wert auf eine Standard-konforme Umgebung legen, sollten Sie eine JVM von Sun oder IBM als Debian-Paket vom Server **`ftp.debian-unofficial.org`** installieren.

Server	Debian-Paket	Beschreibung
`ftp.debian-unofficial.org`	**`ibm-j2se5.0-jdk-binary`**	IBM Java 2 Standard Edition J2SE Development Kit (JDK)
	`ibm-j2se5.0-jre-binary`	IBM Java 2 Standard Edition J2SE Runtime Environment (JRE)
	`sun-j2se5.0-jdk-binary`	Sun Java 2 Platform Standard Edition 5.0 Development Kit (JDK)
	`sun-j2se5.0-jre-binary`	Sun Java 2 Platform Standard Edition 5.0 Runtime Environment (JRE)

Tabelle: Kommerzielle JVM, die als Debian-Paket erhältlich sind

Sun und IBM bieten ihre JVM in zwei Varianten an:

- als "Java Runtime Engine" (JRE, enthält nur den Interpretierer) und
- als "Java Development Kit" (JDK, enthält neben der JRE eine Entwicklungsumgebung mit Compilern und Dokumentation).

Installation des Sun JDK:

```
$ echo 'deb http://ftp.debian-unofficial.org/debian sarge main contrib non-free restricted
' >> /etc/apt/sources.list
$ aptitude update
[...]
$ aptitude install sun-j2se5.0-jdk-binary
[...]
Die folgenden Pakete werden zusätzlich installiert:
  libasound2 libglib1.2 libgtk1.2 libgtk1.2-common libxi6
    sun-j2se5.0-jdk-binary
0 Pakete aktualisiert, 6 zusätzlich installiert,
0 werden entfernt und 0 nicht aktualisiert.
Muss 65,8MB an Archiven herunterladen.
Nach dem Entpacken werden 148MB zusätzlich belegt sein.
Wollen Sie fortsetzen? [Y/n/?] y
[...]
Richte sun-j2se5.0-jdk-binary ein (1.5.0.06+debian-1.unofficial.sarge.1) ...
  * configuring alternatives: done.
$ java -version
java version "1.5.0_06"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_06-b05)
Java HotSpot(TM) Client VM (build 1.5.0_06-b05, mixed mode, sharing)
```

Listing: Installation des Sun JDK

Auf dem **`unofficial`**-Server liegen die J2SE-Versionen 1.4, 5.0 und 6.0 (wahlweise inkl. der **`jce`** Kryptographie-Tools). Im Zweifel wählen Sie die Version 1.4.

Einen Java-EE-kompatiblen Applicationserver installieren [Geronimo 1.0 und Tomcat 5.5]

Die "Java Platform Enterprise Edition" (Java EE) ist eine von SUN Microsystems herausgegebene Spezifikation eines Java-Applicationservers.

Die Java EE Version 5 ist der direkte Nachfolger der "Java 2 Platform Enterprise Edition" (J2EE) Version 1.4.

Mehr zum Thema:
siehe http://www.wikipedia.de/wiki/J2EE

Es gibt verschiedene Implementierungen J2EE-kompatibler Applicationserver, z.B.

- **OpenSource:** Apache Geronimo, JBoss, JOnAS, Sun GlassFish
- **Kommerziell:** IBM WebSphere, BEA Weblogic, Oracle Application Server, SAP Web Application Server

Interessant: IBM vertreibt den Geronimo Applicationsrever auch unter dem Produktnamen "IBM WebSphere CE" (für "Community Edition").

Installation von Apache Geronimo:

```
$ cd /usr/local
$ wget http://mirror.serversupportforum.de/apache/geronimo/1.0/geronimo-tomcat-j2ee-1.0.tar.gz
[...]
$ tar xzf geronimo-tomcat-j2ee-1.0.tar.gz
$ cat >>geronimo-1.0/bin/setenv.sh <<EOF
#!/bin/sh
export JAVA_HOME=/usr/lib/sun-j2se5.0-jdk
EOF
```

Listing: Geronimo/Tomcat installieren

Sie könnten Geronimo nun über das Shellscript **`/usr/local/geronimo-1.0/bin/geronimo.sh`** starten. Besser ist es jedoch, ein Initscript anzulegen:

```
$ cd /etc/init.d
$ wget http://www.jfranken.de/homepages/johannes/vortraege/apache/geronimo-1.0
[...]
$ chmod 755 geronimo-1.0
$ update-rc.d geronimo-1.0 defaults 98 15
 Adding system startup for /etc/init.d/geronimo-1.0 ...
   /etc/rc0.d/K15geronimo-1.0 -> ../init.d/geronimo-1.0
   /etc/rc1.d/K15geronimo-1.0 -> ../init.d/geronimo-1.0
   /etc/rc6.d/K15geronimo-1.0 -> ../init.d/geronimo-1.0
   /etc/rc2.d/S98geronimo-1.0 -> ../init.d/geronimo-1.0
   /etc/rc3.d/S98geronimo-1.0 -> ../init.d/geronimo-1.0
   /etc/rc4.d/S98geronimo-1.0 -> ../init.d/geronimo-1.0
   /etc/rc5.d/S98geronimo-1.0 -> ../init.d/geronimo-1.0
```

Listing: Geronimo-Initscript anlegen

Das Initscript standardisiert den Aufruf und bewirkt, dass der Applicationserver beim Booten automatisch gestartet und beim Herunterfahren sauber beendet wird:

```
$ invoke-rc.d geronimo-1.0 start
Starting Java application server: Geronimo-1.0 .

$ invoke-rc.d geronimo-1.0 restart
Stopping Java application server: Geronimo-1.0 .
Starting Java application server: Geronimo-1.0 .

$ invoke-rc.d geronimo-1.0 stop
Stopping Java application server: Geronimo-1.0 .
```

Listing: Geronimo starten und stoppen

Auf Port 8080 bietet Geronimo Ihnen eine Administrationskonsole:

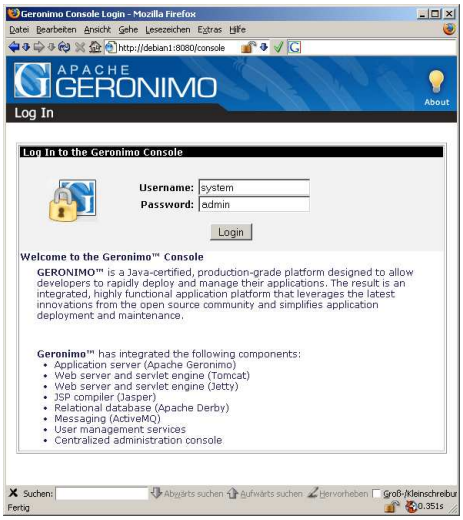

Abbildung: Geronimo-Login (http://server:8080/console)

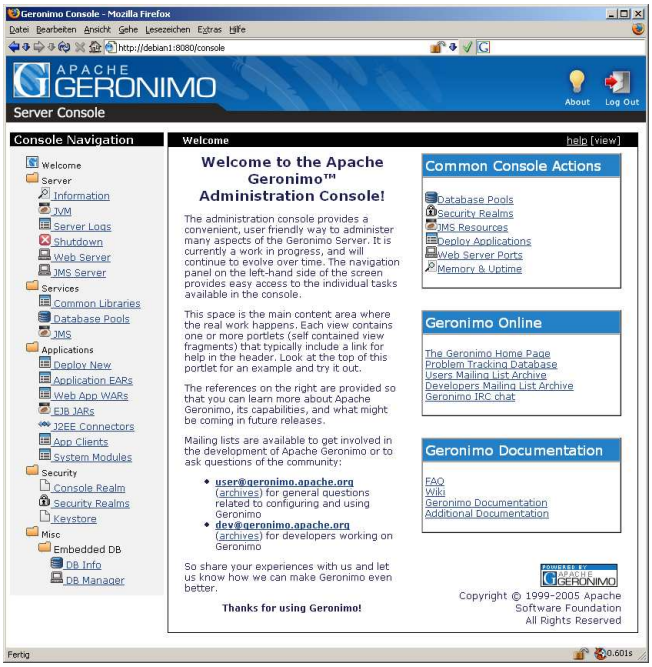

Abbildung: Geronimo-Konsole

Die Portnummern sind in der Datei **`/usr/local/geronimo-1.0/var/config/config.xml`** konfiguriert.

Alternativ: Nur einen Servlet-Container installieren [Tomcat 4.1]

Wenn Ihre Web-Anwendung keine "Enterprise Java Beans" (EJBs) benötigt, reicht die Installation eines Servlet-/JSP-Containers wie z.B. Tomcat, Jetty (beides OpenSource) oder Resin (kommerziell).

Installation von Tomcat (als Debian-Paket verfügbar):

```
$ aptitude install tomcat4-webapps
[...]
Die folgenden Pakete werden zusätzlich automatisch installiert:
  libant1.6-java libbcel-java libcommons-beanutils-java
  libcommons-collections-java libcommons-dbcp-java libcommons-digester-java
  libcommons-fileupload-java libcommons-logging-java
  libcommons-modeler-java libcommons-pool-java libjaxp1.2-java
  liblog4j1.2-java libmx4j-java libregexp-java libservlet2.3-java
  libtomcat4-java libxerces2-java tomcat4
[...]
0 Pakete aktualisiert, 19 zusätzlich installiert,
0 werden entfernt und 0 nicht aktualisiert.
Muss 8856kB an Archiven herunterladen.
Nach dem Entpacken werden 34,1MB zusätzlich belegt sein.
Wollen Sie fortsetzen? [Y/n/?] y
[...]
Richte tomcat4 ein (4.1.31-3) ...
Lege Systembenutzer tomcat4 an...
Adding new user ‘tomcat4’ (103) with group ‘nogroup’.
Erstelle kein Homeverzeichnis.
Installing /var/lib/tomcat4/conf/tomcat-users.xml.
Installing /var/lib/tomcat4/conf/jk2.properties
Could not start Tomcat 4.1 servlet engine because no Java Development Kit
(JDK) was found. Please download and install JDK 1.3 or higher and set
JAVA_HOME in /etc/default/tomcat4 to the JDK’s installation directory.
Richte tomcat4-webapps ein (4.1.31-3) ...
$ echo JAVA_HOME=/usr/lib/sun-j2se5.0-jdk/ >> /etc/default/tomcat4
$ invoke-rc.d tomcat4 start
Starting Tomcat 4.1 servlet engine using Java from /usr/lib/sun-j2se5.0-jdk/: tomcat4.
```

Listing: Tomcat installieren

Informationen zu den Pfaden und Konfigurationsvorgaben finden Sie in der Logdatei:

```
$ cat /var/log/tomcat4/catalina_*.log
Using CATALINA_BASE:   /var/lib/tomcat4
Using CATALINA_HOME:   /usr/share/tomcat4
Using CATALINA_TMPDIR: /var/lib/tomcat4/temp
Using JAVA_HOME:       /usr/lib/sun-j2se5.0-jdk/
Using Security Manager
26.01.2006 20:01:49 org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8180
Starting service Tomcat-Standalone
Apache Tomcat/4.1
26.01.2006 20:01:53 org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8180
26.01.2006 20:01:53 org.apache.jk.common.ChannelSocket init
INFO: JK2: ajp13 listening on /0.0.0.0:8009
26.01.2006 20:01:53 org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=3/215  config=/var/lib/tomcat4/conf/jk2.properties
Using CATALINA_BASE:   /var/lib/tomcat4
Using CATALINA_HOME:   /usr/share/tomcat4
Using CATALINA_TMPDIR: /var/lib/tomcat4/temp
Using JAVA_HOME:       /usr/lib/sun-j2se5.0-jdk/
Using Security Manager
26.01.2006 20:07:38 org.apache.coyote.http11.Http11Protocol init
INFO: Initializing Coyote HTTP/1.1 on http-8180
Starting service Tomcat-Standalone
Apache Tomcat/4.1
26.01.2006 20:07:40 org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-8180
26.01.2006 20:07:40 org.apache.jk.common.ChannelSocket init
INFO: JK2: ajp13 listening on /0.0.0.0:8009
26.01.2006 20:07:40 org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=2/90  config=/var/lib/tomcat4/conf/jk2.properties
```

Listing: Tomcat-Logfile ("Catalina.out")

Tomcat enthält einen eigenen Webserver ("Coyote"), der auf Port 8180 lauscht:

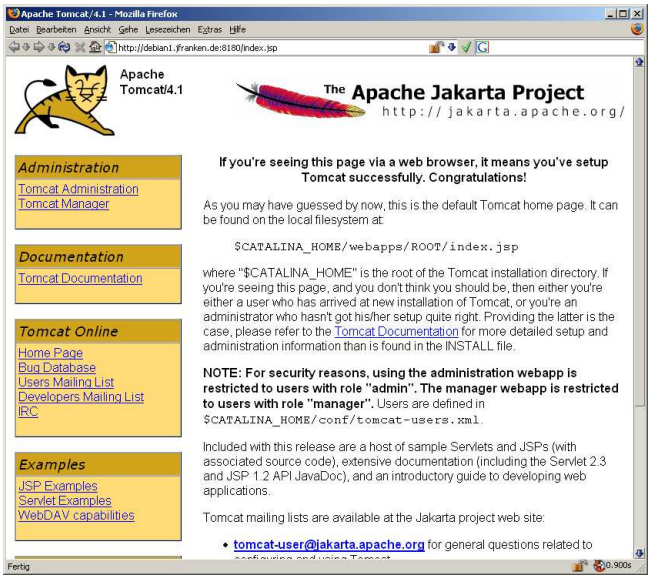

Abbildung: Tomcat antwortet

Tomcat in den Apache Webserver einbinden [mod-jk2]

Sie können den Apache-Webserver als *Reverse-Proxy* einsetzen mit dem Vorteil, dass Apache

- statischen Content (z.B. Icons) selbst ausliefert,
- die Authentifizierung/Autorisierung und die Verschlüsselung übernimmt,
- die Anfragen auf mehrere Tomcat-Instanzen verteilt, die auf verschiedenen Applicationservers liegen (Loadbalancing)
- eine Netz-Trennung zwischen dem Webserver (DMZ) und dem Applicationserver (Intranet) ermöglicht.

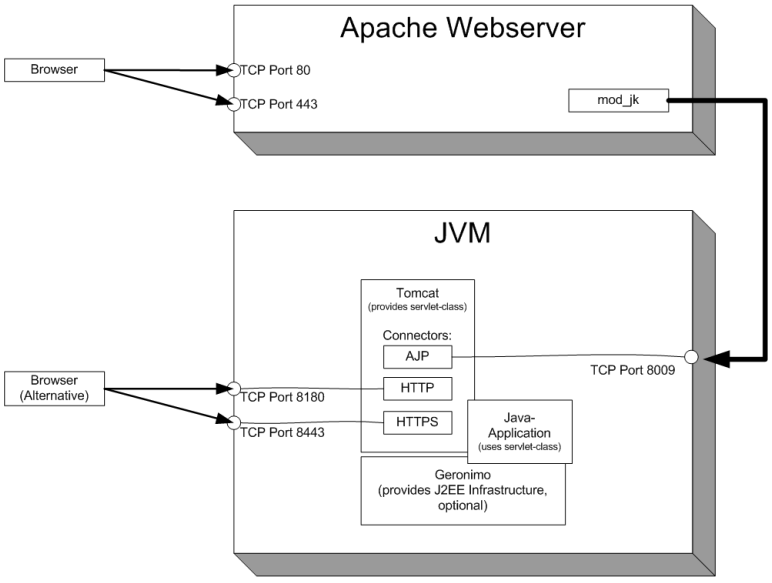

Abbildung: Kommunikations-Schema (Tomcat)

Die folgende Abbildung zeigt ein Beispiel für die Architektur einer hochverfügbaren Internet-Anwendung, bei der die Webserver hinter Loadbalancern in einer DMZ stehen und die Zugriffe auf Applicationservers im Intranet verteilen.

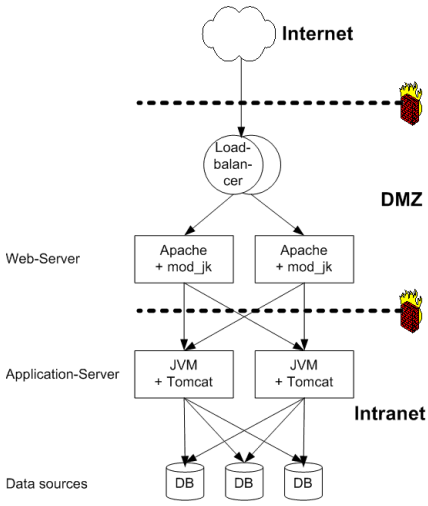

Abbildung: Beispiel-Architektur (mod_jk)

Aktivieren Sie das Jakarta-Modul (**`mod_jk`**), damit der Apache Webserver Anfragen an Tomcat weiterleiten kann:

```
$ aptitude install libapache2-mod-jk2
[...]
Die folgenden Pakete werden zusätzlich installiert:
  libapache2-mod-jk2
0 Pakete aktualisiert, 1 zusätzlich installiert,
0 werden entfernt und 0 nicht aktualisiert.
Muss 157kB an Archiven herunterladen.
Nach dem Entpacken werden 705kB zusätzlich belegt sein.
[...]
Richte libapache2-mod-jk2 ein (2.0.4-3) ...
Module jk2 installed; run /etc/init.d/apache2 force-reload to enable.
$ cp /usr/share/doc/libapache2-mod-jk2/examples/workers2.properties /etc/apache2/
$ vi /etc/apache2/workers2.properties
$ invoke-rc.d apache2 force-reload
Forcing reload of web server: Apache2
```

*Listing: Installation von **`mod_jk`***

Die Standardkonfiuration blendet den Tomcat im Documentroot unter **`/examples`** und **`/jkstatus`** ein:

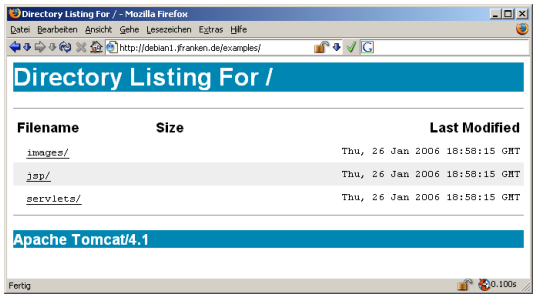

Abbildung: Jakarta antwortet

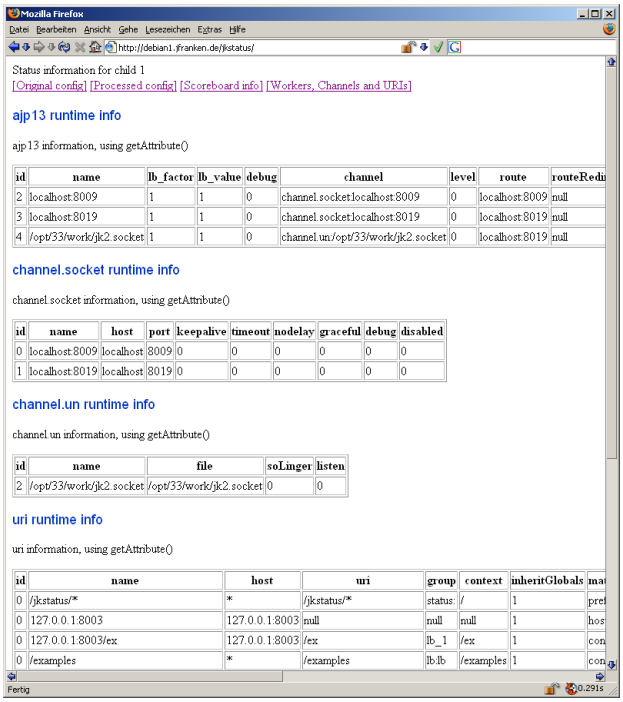

Abbildung: jkstatus