Der APACHE Web-Server

Handreichung des
Kompetenzzentrums für Multimedia und Telematik
am
Deutschen Institut für Fernstudienforschung

zum Seminar
Web-Server Software
Abschnitt I

Autor: M. Knobloch
KMMT am DIFF
Konrad Adenauer Str. 40
72072 Tübingen

1 Aufgabe und Funktion eines Webservers?

Die Aufgabe eines Web - Servers ist es eine angeforderte URL

- in einen Dateinamen zu übersetzen und die Datei über das Internet zurückzuschicken oder
- in einen Programmnamen zu übersetzen, das Programm auszuführen und die Programmausgabe über das Internet zurückzuschicken

Diese schlichte Anforderung wird angereichert durch eine Menge von Aufgaben, die sich um diese ursprünglichen Funktionenen herum entwickelt haben:

- Beantwortung von mehreren Anfragen (scheinbar) gleichzeitig
- Verwalten von Berechtigungen und Prüfen von Berechtigungen
- Reaktion auf Fehler
- Anpassbarkeit an unterschiedliche Sprachen (Lokalisierung)
- Unterstützung von Media Dateiformaten
- Puffern von häufig angefragten Informationen für schnellern Zugriff (Proxy Funktion)
- Unterstützung von Sicherheitsprotokollen
- to be continued

Die verschiedenen Web - Server realisieren diese Anforderungen auf unterschiedliche Weise und decken unterschiedliche Unter- und Obermengen dieser Funktionen ab. Hinzu kommen weitere Kriterien wie Offenheit für Erweiterungen, Bedienerfreundlichkeit, Systemintegration, Geschwindigkeit..., die die Entwicklungsteams der unterschiedlichen Produkte auf ihre Weise gewichtet haben. Entsprechend unterschiedlich sind die nachfolgen dargestellten Produkte in ihrem Verhalten. Ein Produktvergleich im Sinne von "Besser" oder "Schlechter" ist nicht Ziel dieser Informationen. Vielmehr soll der Frage "was ist wofür in welcher Weise geeignet?" nachgegangen werden.

2 Der Apache Web-Server

Der Apache Web-Server ist der Server mit der weltweit größten Installationsbasis. Die Gründe hierfür werden aus der nachfolgenden Beschreibung ersichtlich.

Entstehung und Geschichte

- Der Apache Web Server entstand zu Beginn des Jahres 1995 aus dem Code des NCSA[1] Web-Servers und einigem Flickwerk (patches) zur Verbesserung des NCSA Produkts. Daher trägt der Server seinen Namen "a patche server". Apache war also "auf dem Markt" als der WWW Boom begann. Die Entwickler des Apache Servers, eine Gruppe von Freiwilligen, lehnen sich heute jedoch lautlich an die indianische Tradition an.

Verfügbarkeit und Plattformen

- Der Apache Server ist für eine große Zahl von UNIX - Derivaten verfügbar. Da der Programm - Quellcode ausgeliefert wird, sind Anpassungen jeglicher Art leicht möglich, Erweiterungen durch kompetente Endnutzer geplant. Die aktuelle Version 1.3 bedient auch Windows NT
Die Quellen des Apache Servers sind frei verfügbar. Sie stellen die wichtigste Distributionsbasis dar. Die Nutzungslizenz verfügt lediglich, daß bei Nutzung oder Vertrieb der Copyright-Hinweis auf die Apache Group dokumentiert sein muß und daß ein Produkt, das aus verändertem Apache Quellcode besteht nicht mehr den Namen Apache tragen darf.

wesentliche Merkmale

- Was den Apache von anderen Webservern abhebt, ist sein Modulkonzept. Wird Apache aus dem Quellcode installiert, muß vor dem Kompilieren des Produkts ausgewählt werden, welche Module zu Apache hinzugelinkt werden sollen und welche nicht. Eine große Zahl von Modulen wird mitgeliefert, weitere Module sind im Internet verfügbar[2], darunter auch Schnittstellenmodule für kommerzielle Produkte.
Es ist auch möglich eigene Module zu schreiben, die mit dem Apache API kommunizieren. Da es jedoch bereits einige Module gibt, die eine Erweiterung des Apache ohne Rücksicht auf das Apache API erlauben, wird in Zukunft eher diese Möglichkeit genutzt werden (siehe CGI, PHP, mod_jserv, mod_perl...)

2.1 Schritte zur Inbetriebname

Konfiguration und Installation der Software gelten als nicht ganz einfach. Die Komplexität der Fragen, die bei der Konfiguration geklärt werden müssen, resultiert aus der Möglichkeit, den Server den individuellen Anforderungen anzupassen. Diese Anpassungsmöglichkeit wird in den zwei Schritten Konfiguration und Installation realisiert.
Hierbei steuert die Konfiguration initial die Erstellung der make-Datei, die make-Datei steuert die Erstellung (compile) und Installation des Apache.

2.1.1 Konfiguration[3], make und erster Start des Apache Servers

Alte Form bis V. 1.2 Configure[4]

- Die `Configuration` Datei steuert die Erzeugung des makefiles bis Apache Version 1.2. Diese Datei liegt in der 1.3 Distribution im /src Verzeichnis. Hier können Einträge zu Compiler flags, zusätzlich notwendigen Bibliotheken und den modulen, die gelinkt werden sollen gemacht werden.
- Im gleichen Verzeichnis liegt auch ein script mit Namen `Configure` (Großschreibung beachten!), dessen Aufruf die Informationen aus der `Configure` Datei liest und das eigentliche `makefile` generiert.

neue Form ab V. 1.3 (bevorzugt) configure

- `./configure --prefix=/pfad/fuer/apache`
  erzeugt ein makefile mit Standardeinträgen. Sollen weitere Module gelinkt werden können optionen beim configure script angegeben werden, z.B. das PHP Modul soll benutzt werden:
- `./configure --prefix=/usr --activate-module=src/modules/php3/libphp3.a`[5]
  ist dann der entsprechende Aufruf. Bei mehreren Modulen wird für jedes Modul eine --activate-module Anweisung eingegeben.
- zur Kontrolle vor einem configure Lauf sollte unbedingt
  `./configure --prefix=/usr --activate-module=src/modules/php3/libphp3.a --layout`
  angegeben werden. Die --layout Option zeigt die Installationsverzeichnisse die bei nachfolgender Installation angelegt und benutzt werden an, ohne die Konfiguration durchzuführen.
- die Option `--enable-module=status` muß benutzt werden, wenn es sich um mitgelieferte module im `/modules/standard` Verzeichnis handelt.

make

- nach erfolgtem configure Lauf kann
  `make`
  aufgerufen werden. Das Programm make liest die Datei mit Namen `makefile`, die von `configure` erzeugt wurde und übersetzt den C-Quellcode unter Einbindung der angegebenen Module. Dieser Vorgang kann einige Minuten in Anspruch nehmen. Das Ergebnis ist ein Programm mit Namen `httpd`. Um zu überprüfen, ob die gewünschte Modulkombination erstellt wurde, kann das Programm mit dem Parameter -l aufgerufen werden:
  ```
  httpd -l
  ```
  gibt die Namen aller Module aus, die gelinkt wurden

make install

- der Aufruf von `make install` kopiert Programm und Standarddateien in die entsprechenden Installationsverzeichnisse.

apachectl[6]

- mit dem mitgelieferten script apachectl kann der Server gestartet und gestoppt werden:
- `/PREFIX/sbin/apachectl start`
- `/PREFIX/sbin/apachectl stop`

2.1.2 Starten und stoppen des Servers

Der Apache Server verfügt nicht über die Möglichkeit der graphischen Konfiguration. Alle Angaben, die das Laufzeitverhalten des Servers beeinflussen, müssen in Dateien eingetragen werden. Jeder Eintrag in einer .conf Datei wirkt hierbei auf den Kern des Produkts oder auf ein einzelnes Modul, das zu Apache hinzugelinkt wurde. Die Sinnhaftigkeit der einzelnen Optionen in den Laufzeit-Konfigurationsdateien hängt also auch davon ab, ob die ausführenden Module aktiviert wurden.

Beim Start des Programms liest Apache folgende Konfigurationsdateien[7]:

1. Die Server Konfiguration *httpd.conf*
2. Die Zugriffskonfiguration *srm.conf*
3. Server Resource Management access.*conf*

Änderungen an den Konfigurationsdateien wirken erst bei einem Neustart des Servers. Das Script `apachectl` akzeptiert die Parameter `start, stop, restart` und `graceful`. Der Parameter `graceful` führt den Restart nur auf Prozesse durch, die augenblicklich keinen Request bearbeiten.

2.1.3 Konfigurationsdateien für den Start

Nachfolgend sollen einige Einträge in .conf Dateien vorgestellt werden[8]. Viele der Einträge können mit ihrer Standardeinstellung benutzt werden.

ServerType
- `Standalone` legt fest, daß der Server permanent läuft und nicht bei Anfrage jedesmal neu gestartet wird. (Option `inetd`)

Port
- `Port 80` ist Voreinstellung
- Der Server kann jedoch auf anderen Ports betrieben werden, z.B. parallel als experimenteller Server mit dem Eintrag `Port 8080`. Bei Portnummern unter 1024 muß der Start mit root-Berechtigung erfolgen.

User
- Legt fest unter welcher Userid httpd nach dem Start läuft
- User `nobody` ist Standardeinstellung

Group
- `Group #-1` analog zu User

ServerRoot
- `ServerRoot /PREFIX` gibt an wo config-, error- und log - Dateien liegen

ServerAdmin
- Die email Adresse, die hier angegeben wird, wird im Fehlerfall angezeigt. Beispiel:
- ServerAdmin manfred_knobloch@diff.uni-tuebingen.de

ServerName
- DNS Name des Webservers. Der Standardbetrieb funktioniert meist auch ohne diesen Eintrag. Manche Erweiterungsmodule benötigen jedoch hier einen Eintrag!
- `ServerName tsatsiki.diff.uni-tuebingen.de`

Mit der Anpassung dieser Einstellungen an die eigenen Gegebenheiten ist es möglich einen Standardserver zu betreiben.

2.2 Betrieb des Servers

2.2.1 Den Überblick wahren

Bei der Menge der Optionen, switches und Einstellungen geschieht es leicht, daß das eine oder andere in Vergessenheit gerät. Deshalb hier ein Versuch zusammenzufassen, was für Informationsquellen sich nutzen lassen, um die aktuellen Einstellungen nachschlagen zu können.

- Der Einstieg in die Online Dokumentation zu Apache befindet sich in Form der Datei `index.html` im Standard DocumentRoot, die Texte selbst im Unterverzeichnis `/manual`.
- Ein Lauf des `configure` - Befehls mit der option `--layout` zeigt an welche Verzeichnisstrukturen angelegt werden, die Ergebnisse werden nicht in das makefile geschrieben

- Nach erfolgtem `make` steht das Programm `httpd` im `/src` Verzeichnis zur Verfügung. Mit
  `./httpd -l` kann überprüft werden, welche Module zu Apache hinzugelinkt wurden,
  `./httpd -S` zeigt an welche `VirtualHost` Anweisungen eingelesen wurden,
  `./httpd -V` zeigt Informationen zum build Prozess.
- beim configure Lauf sollten die Module `mod_info` und `mod_status` aktiviert werden. Sie liegen im Standard-Verzeichnis, können deshalb mit Kurzschreibweise angegeben werden.
  `--enable-module=info`
  `--enable-module=status`

  In der `access.conf` Datei sollten folgende Einträge gemacht werden:

```
# Abfrage von Server-Info nur für bestimmte ip-Adressen
# mod_info muss aktiviert sein
<Location /intern/server-info>
SetHandler server-info
order deny,allow
deny from all
allow from irgendwo.domain.com
</Location>

# Abfrage von Server-Stauts nur für bestimmte ip-Adressen
# mod_status muss aktiviert sein
<Location /intern/server-status>
SetHandler server-status
order deny,allow
deny from all
allow from irgendwo.domain.com
</Location>
```

  durch Eingabe der URL `/intern/server-info` oder `/intern/server-status` können die Informationen über Auslastung und Konfiguration des Servers abgerufen werden.

2.2.2 Log Dateien

Die Möglichkeit Benutzeraktionen zu protokollieren, wird durch das Modul mod_log_config[9] realisiert. Welche Informationen mitgeschrieben werden und wie diese Informationen dargestellt werden sollen, läßt sich über Einstellungen in httpd.conf steuern.

2.2.2.1 Fehlerprotokoll

Den Ort für Fehler- und sonstige Infos, die Apache produziert läßt sich mit der Anweisung

ErrorLog

- `ErrorLog /PREFIX/var/apache/log/error_log` festlegen
- `ErrorLog /dev/null` schaltet die Fehlerprotokollierung aus
- ErrorLog syslog kann verwendet werden, wenn das Trägersystem einen syslogd besitzt.

2.2.2.2 Zugriffsprotokoll

Auslastung und Benutzerverhalten werden in der Datei access_log festgehalten. Dies erfolgt über die Direktiven LogFormat, CustomLog und HostnameLookups.

LogFormat

- Mit dieser Anweisung wird bestimmt, welche Informationen protokolliert werden sollen.
- `LogFormat "%h %l %u %t \"%r\" %>s %b" common`
  Die LogFormat Anweisung besteht aus einem Format-String und einem Kurznamen, unter dem die Formatdefiniton angesprochen werden kann. Die abgebildeten Formatkürzel bedueten:
- `%h` - hostname / IP Adresse des zugreifenden Rechners[10]
- `%l` - remote Logname (meist nicht gesetzt)
- `%u` - über Authentifizierung erhaltene userid
- `%t` - Zeit des Zugriffs
- `%r` - erste (Befehlszeile) des Requests
- `%s` - HTTP Statuscode
- `%b` - content length des ausgelieferten Dokuments

CustomLog

- Mit CustomLog wird festgelegt in welche Datei und mit welchem Aufbau die Protokolldaten geschrieben werden.
- `CustomLog /usr/var/apache/log/access_log common`

HostnameLookups

- Dieser Eintrag in httpd.conf bewirkt, daß Client-Requests nicht mit der IP-Adresse des Clients sondern mit dem Rechnernamen in die log-datei geschrieben wird. Hierzu führt Apache bei jedem Request einen DNS lookup durch.
- Bei Zugriffen aus online-Diensten wie AOL oder T-Online ist das nicht möglich, da hier lediglich DHCP Adressen vergeben werden.
- Standardeinstellung ist deshalb:
  `HostnameLookups off`

2.2.2.3 Auswertung und Auswertungstools

Die Apache Logfiles können mit z.T. mitgelieferten tools ausgewertet werden. Die Aussagekraft der entstehenden Statistiken ist jedoch nur bedingt verläßlich. Da in vielen Netzen Proxy Server verwendet werden, enden viele Client-Zugriffe auf oft genutzte Seiten bereits im Cache und erreichen den Webserver und damit die Log-Programme nicht.
Das Caching zu unterbinden ist keine echte Lösung, weil sich damit die Antwortzeiten für die Clients verlängern. Dies wird in der Regel zu einer Abnahme der Nutzungsfrequenz führen. In Test- oder Versuchsumgebungen kann dies jedoch ein Weg sein.

Eine Zusammenfassung der access_log Einträge zu einem Gesamtüberblick bietet das Perlscript `wwwstat` oder das frei verfügbare Programm `webinator`.

2.2.3 virtual host

Virtual Host beschreibt die Fähigkeit eines Web-Servers, unter verschiedenen Namen angesprochen und unter jedem Namen ein eigenes Dokumentenverzeichnis, eigene Logfiles etc. verwalten zu können. Zwei Arten von virtuellen Servern werden unterschieden:

1. IP-basierte virtual hosts, d.h. jeder Name entspricht einer IP-Adresse. Da sicher nicht für jeden virtuellen Rechner eine Netzwerkkarte eingebaut werden soll, ist Voraussetzung dafür, daß das Betriebsystem es unterstützt, mehrere IP-Nummern auf eine Netzwerkkarte abzubilden[11].
2. Namensbasierte[12] virtuelle Server, d.h. in der Nameservertabelle wird lediglich ein CNAME alias eingetragen.

Beispiel: Webserver auf tsatsiki.diff.uni-tuebingen.de soll unter folgenden Namen angesprochen werden:

Eintrag in named.hosts

```
tsatsiki                    IN      CNAME   node195
rfcliste                    IN      CNAME   node195
kmmt.servletdokus           IN      CNAME   node195
kmmtlogin                   IN      CNAME   node195
kmmtsecure                  IN      CNAME   node195
www.webuser1                IN      CNAME   node195
www.webuser2                IN      CNAME   node195
www.webuser3                IN      CNAME   node195
www.webuser4                IN      CNAME   node195
```

Eintrag in httpd.conf

```
NameVirtualHost 134.2.196.195

<VirtualHost 134.2.196.195>
ServerName tsatsiki.diff.uni-tuebingen.de
DocumentRoot /usr/share/apache/htdocs
DirectoryIndex index.html index.htm
</VirtualHost>

<VirtualHost 134.2.196.195>
ServerName rfcliste.diff.uni-tuebingen.de
DocumentRoot /usr/share/apache/htdocs/rfc
</VirtualHost>

<VirtualHost 134.2.196.195>
ServerName kmmtlogin.diff.uni-tuebingen.de
DocumentRoot /home
DirectoryIndex index.html index.htm tree.html
</VirtualHost>
```

Der erste virtuelle Server, der nach einer `NameVirtualHost` Anweisung angegeben ist fungiert hierbei als Default-Server. Alle weiteren Einträge definieren die Priorität der Server in absteigender Reihenfolge.

2.2.4 Dateien und Verzeichnisse

Mit den Anweisungen `<Directory>....</Directory>`, `<Files>....</Files>` und `<Location>.....</Location>` kann die Konfiguration des Zugriffs auf Dateien gesteuert werden.

2.2.4.1 Zugriffskontrolle auf Verzeichnisse und Dateien

Informationen über Zugriffsberechtigungen auf Dateien und Verzeichnisse werden in der Regel über die Datei `access.conf` gesteuert. Die hierzu notwendigen Anweisungen sind `Directory`, `Files` und `Location`.

2.2.4.2 <Directory>

Die Directory Anweisung beschreibt mit einer absoluten Pfadangabe Zugriffsrichtlinien für die genannten Verzeichnisse, inclusive zugehöriger Unterverzeichnisse. Neben einem festen Verzeichnisnamen können auch Platzhalter (wildcards und regular expressions) im Namen verwendet werden, um ganze Gruppen von Verzeichnissen zu kennzeichnen.

Zwischen `<Directory>....</Directory>` eingetragene Anweisungen gelten für das bezeichnete Verzeichnis inclusive aller Unterverzeichnisse. Als Verzeichnisangabe muß ein absoluter Pfad verwendet werden. Im nachfolgenden Beispiel werden die Verzeichnisse `/home/webuser1...4` und das Verzeichnis `/home/manne` mit einem Authentifizierungsmechanismus geschützt.

```
<VirtualHost 134.2.196.195>
ServerName kmmtlogin.diff.uni-tuebingen.de
DocumentRoot /home

<Directory /home/w*>
Options Indexes FollowSymLinks
AllowOverride None
order allow,deny
allow from all
AuthType Basic
AuthName "user Verzeichnis"
AuthUserFile /usr/etc/apache/passwd
AuthGroupFile /usr/etc/apache/group
require valid-user
</Directory>

<Directory /home/manne>
Options Indexes FollowSymLinks
AllowOverride None
order allow,deny
allow from all
AuthType Basic
AuthName "user Verzeichnis"
AuthUserFile /usr/etc/apache/passwd
AuthGroupFile /usr/etc/apache/group
require user manne
</Directory>
</VirtualHost>
```

2.2.4.3 <Files>

Die `<Files>....</Files>` Anweisung ist analog zur Directory Anweisung aufgebaut und wirkt entsprechend auf Dateien, die innerhalb der Files Deklaration angegeben werden.

2.2.4.4 <Location>

Wie die vorhergehenden Anweisungen beschreibt Location das Verhalten von Verzeichnissen und Dateien, allerdings mit einem URL als Deskriptor.

`<Directory /home/manne>`
entspricht dem Eintrag

`<Location /manne>`

2.2.4.5 Authentifizierung

Die Berechtigung Verzeichnisse oder Dateien öffnen zu dürfen an IP Adressen zu binden ist oft nicht ausreichend, vor allem, wenn mehrere Benutzer eine Arbeitsstation teilen. Die gängige Form der Prüfung von Zugriffsberechtigungen ist die Authentifizierung anhand von Benutzername und Password. Für Apache existiert eine Vielzahl von Modulen, die Benutzername und Password gegen Dateien, Datenbanken oder externe Zugriffgsysteme zu prüfen. Die einfachste Form ist die Prüfung gegen eine Password oder Gruppendatei, die mit dem Modul mod_auth realisiert werden kann. Diese Dateien sollten nicht im Dokument Verzeichnisbaum liegen. Sie werden mit einem Hilfsprogramm, das mitgeliefert wird verwaltet.

Die Folgenden Beispieleinträge fordern eine Berechtigung, wenn versucht wird ein Unterverzeichnis von /home, das mit dem Buchstaben "w" beginnt zu öffnen, also alle webuser Verzeichnisse. Berechtigt sind alle Benutzer, die in der Authentifizierungsdatei gefunden werden.

Im Verzeichnis /home/manne ist nur der Benutzer manne berechtigt, die Verzeichnisse /home/soft, /home/*admin sind vollständig gesperrt.

```
<VirtualHost 193.12...>
<Directory /home/w*>
Options Indexes FollowSymLinks
AllowOverride None
order allow,deny
allow from all
AuthType Basic
AuthName "user Verzeichnis"
AuthUserFile /usr/etc/apache/passwd
AuthGroupFile /usr/etc/apache/group
require valid-user
</Directory>

<Directory /home/manne>
Options -Indexes FollowSymLinks
DirectoryIndex index.html
AllowOverride None
order allow,deny
allow from all
AuthType Basic
AuthName "user Verzeichnis"
AuthUserFile /usr/etc/apache/passwd
AuthGroupFile /usr/etc/apache/group
require user gueltiger_benutzer
</Directory>

<Directory ~ "^/home/(soft|(.*)admin)">
```

```
order allow,deny
deny from all
</Directory>
</VirtualHost >
```

Sicherheitslücken können hier leicht entstehen, wenn über einen anderen `VirtualHost` Eintrag Zugriff auf das selbe Verzeichnis unter einem anderen URL gewährt wird. Authentifikation innerhalb eines VirtualHost Eintrags bleibt auf diesen beschränkt.

2.2.4.6 Content Negotiation

Die Auslieferung eines Dokuments kann von den Anforderungen des Clients abhängig gemacht werden. Der Client handelt beim Zugriff die Variante des gewünschten Dokuments aus. Dies wird über die Anweisungen `MultiViews` und `type-map` gesteuert.

Für selbstdefinierte Error Dokumente, die im Unterverzeichnis `/error` des Dokumentenpfades des Standard Default-Servers liegen definieren folgende Einträge, daß nach Varianten der Dokumente gesucht wird. Im Verzeichnis liegen die Variantentdateien mit dem jeweiligen Sprachkennzeichen, die angeboten wird.

Beispiel: die deutsche und englische Variante für die Meldung, daß Zugriff nicht erlaubt ist

```
forbidden.de.html
forbidden.en.html
```

Die Anweisung, daß nach Varianten gesucht werden soll, sieht in der configuration so aus:

```
<Location /error>
Options +MultiViews +Includes -Indexes
AddHandler server-parsed .html
</Location>
```

2.2.5 Types, Handler und SSI

2.2.5.1 Types

Bei Generierung eines `Content-Type` Header Eintrags einer HTTP-Response werden Informationen aus den Einträgen der Datei `mime.types` verwendet, um die Zuordnung zwischen Dateiendung und Inhaltstyp herzustellen. Die Lage der `mime.types` wird mit der `TypesConfig` Anweisung bekannt gegeben, mit `DefaultType` der Standard Inhaltstyp (`text/plain`), der benutzt wird, wenn die Zuordnung nicht bekannt ist. Ein `DefaultType` oder `ForceType` Eintrag kann auch innerhalb eines Verzeichnisses benutzt werden. Neue Einträge sollten mit `AddType` in der Apache Laufzeitkonfiguration gemacht werden.

2.2.5.2 Handler

Für jede Datei mit einer bestimmten Endung oder für eine bestimmte Lokation kann eine definierte Aktion festgelegt werden. Mit einem `Handler` Konfigurationseintrag kann Apache mitgeteilt werden, welche Aktionen welchen Dateiendungen oder Lokationen zugeordnet sind.

`AddHandler` weist den Dateien mit entsprechenden Endungen einen Handler zu. Sollen generell alle Dateien mit der Endung `.cgi` oder `.pl` als scripte interpretiert werden, wird der Handler wie folgt zugeordnet:

```
AddHandler cgi-script .cgi
```

```
AddHandler cgi-script .pl
```

Mit der `Action` Anweisung kann ein Handler Name erzeugt und an ein script gebunden werden, d.h. das script als Handler definiert werden.

Die Anweisung `SetHandler` wirkt innerhalb von `<Directory>-, <Location>-` oder `<Files>-` Blöcken und legt fest, daß dieser Handler auf alle Dateien innerhalb des Bereichs angewendet wird.

Apache verfügt über vorgefertigte `Handler` - sofern die entsprechenden Module eingebunden sind. Beispiel dafür ist der `Handler: cgi-script`. Zu seiner Aktivierung sind folgende Einträge notwendig:

```
ScriptAlias /cgi-bin/ /usr/share/apache/cgi-bin/
...
<Location /cgi-bin>
AllowOverride None
Options ExecCGI
SetHandler cgi-script
</Location>
```

Die Einträge legen fest, daß alle Dateien im `/cgi-bin` Verzeichnis als CGI-Scripte interpretiert werden sollen. Weitere wichtige interne Handler sind `server-info, server-parsed, server-status`.

Mit Hilfe der `<Files>` Anweisung kann auch eine einzelne Datei als CGI-Script bekannt gemacht werden.

```
<Files /usr/share/apache/htdocs/test-cgi >
SetHandler cgi-script
</Files>
```

2.2.5.3 Server Side Includes - SSI

Ein Dokument, das Server Side Includes enthält wird vor seiner Auslieferung vom Server durchgelesen und nach speziellen Anweisungen, die in Form von HTML Kommentaren abgelegt sind, untersucht.

```
<!--#include file= ''Dateiname'' -->
```

oder z.B. innerhalb einer Referenz zur Anzeige einer Umgebungsvariablen.

```
<a href="mailto:<!--#echo var="SERVER_ADMIN"-->">Webmaster</a>
```

Gefundene Einträge werden ausgewertet und die darin festgelegten Anweisungen werden ausgeführt. Der einfachste und häufigste Fall ist die Anzeige von Umgebungsvariablen. Es ist jedoch auch möglich, bedingte Anweisungen auszuführen oder CGI-Scripts aufzurufen. Bei komplexeren Anwendungen wird CGI oder anderen Erweiterungen sicher der Vorzug gegeben.

Um Server Side Includes zu verwenden, muß das Modul `mod_include` eingebunden sein. Die Aktivierung geschieht über den `Handler server-parsed`.

```
<Location /error >
Options +Includes
AddHandler server-parsed .html
</Location>
```

2.2.6 Server Erweiterungen

2.2.6.1 Apache API

Im Modules Verzeichnis der Apache Distribution findet sich das mod_example. Dies ist eine Beispielimplementierung eines lauffähigen Moduls, das eine große Zahl der verfügbaren API-calls enthält. Es sollte lediglich zu Lernzwecken verwendet werden, wenn beabsichtigt ist ein eigenes Modul zu schreiben. Module sind auf die Programmierung in C festgelegt, haben sicherlich den Vorteil der guten Ausführungsgeschwindigkeit, bringen aber einiges an Komplexität mit sich.

2.2.6.2 CGI und darüber hinaus

Das Common Gateway Protokoll[13] beschreibt eine Programmschnittstelle zwischen einem Webserver und einem Anwendungsprogramm. Das Gateway Protokoll legt die Form der Übergabe von Werten (Parametern) aus dem Webserver fest. Die Schnittstelle ist auf keine bestimmte Programmiersprache festgelegt. Der Sprachgebrauch "CGI-Script" resultiert aus dem Umstand, daß bislang vor allem Perl scripts oder shell-scripts als CGI Programme zum Einsatz kamen. Es können jedoch auch kompilierte Sprachen wie C eingesetzt werden. Relevant ist lediglich, daß die Parameter, die an das Programm übergeben werden entsprechend den Konventionen von CGI verarbeitet werden. CGI Programme können über eine Referenz oder Grafikreferenz

```
<a href="/cgi-bin/counter.pl">Zaehler inkrementieren</a>.
```

```
<img src="/cgi-bin/counter.pl">.
```

oder über ein SSI

```
<!-- #exec cgi="/cgi-bin/counter.pl" -->.
```

aktiviert werden.

Wichtigstes Einsatzgebiet von CGI ist die Verarbeitung von interaktiven HTML Seiten in Form von Formularen. Mit der action Anweisung im HTML Dokument wird festgelegt, welches Programm bei Klicken auf den submit Button ausgeführt werden soll.

```
<form action="/cgi-bin/guestbook.pl" method="get">).
```

Ist ein HTML Formular wie folgt definiert[14]:

```
<html><head><title>Kommentarseite</title>
</head><body>
<h1>Ihr Kommentar</h1>
<form action="/cgi-bin/note.pl" method=post>
  Name:   <input name="AnwenderName" size=40>
..Text:   <textarea rows=5 cols=34></textarea>
  <input type=submit value="absenden">
</form></body></html>
```

erhält der Server nach Klicken des Absenden-Buttons folgenden Request:

`POST /cgi-bin/note.pl` mit folgenden Parametern in STDIN

```
Name=Olaf+M%FCller&Text=Das+ist+ein+kleiner+Text
```

und bearbeitet die Daten mit nachfolgendem Script

```
#!/usr/bin/perl
# Formulardaten werden über STDIN eingegeben und haben die Länge
# CONTENT_LENGTH
read(STDIN, $Daten, $ENV{'CONTENT_LENGTH'});

print "Content-type: text/html\n\n";
print "<html><head><title>CGI-Feedback</title></head>\n";
print "<body><h1>CGI-Feedback vom Programm <i>note.pl</i></h1>\n";

# Parameterblock in eine Liste zerlegen. Elementtrennzeichen ist $
@Formularfelder = split(/&/, $Daten);

# durch Liste der Parameter iterieren, aktueller Listenwert ist in
# $Feld pro iterationsschritt zwischengespeichert
foreach $Feld (@Formularfelder)
 {
  # key und value aufsplitten, Trennzeichen Gleichheitszeichen
  ($name, $value) = split(/=/, $Feld);

  # Pluszeichen in Leerzeichen rückkonvertieren
  $value =~ tr/+/ /;

..# urlencoding von Sonderzeichen rückverwandeln, z.B. Müller
  $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;

..# Kommentare entfernen
  $value =~ s/<!--(.|\n)*-->//g;

print "$name = $value", "<br>\n";
 }
print "</body></html>\n";
```

Die bekanntesten Nachteile von CGI sind schlechte Performance, umständliches Session handling, Probleme mit persistenenten Informationen, etc. Sehr verbreitet im praktischen Einsatz sind deshalb Apache Module, die versuchen diese Probleme zu mildern.

mod_php

PHP vereint Fähigkeiten von SSI und CGI. Die Sprache selbst ist eine Mischung aus C und PERL. Da Zugriffsroutinen auf sehr viele verschiedene Datenbanksysteme angeboten werden und PHP sehr performant ausgeführt wird, hat sich PHP in der Programmierung von Webserver Erweiterungen etabliert. Parameterzerlegung wie bei Perlscripts ist weiterhin möglich. Die Werte können jedoch auch über Variablennamen, die den Feldnamen im Formular entsprechen, direkt angesprochen werden. PHP entwickelt sich derzeit trendentsprechend in Richtung Objektorientierung.

mod_jserv

Dieses Modul realisiert eine Servlet Engine, d.h. eine JAVA VM, die als Teil des Webservers ausgeführt wird. Der Java Interpreter wird also nicht erst geladen, wenn ein Servlet ausgeführt werden soll, sondern kann mit dem WebServer gestartet werden. Servlets werden beim ersten Aufruf in den Speicher geladen und verbleiben dort. Da sie mit JAVA Objekten jeder Art umgehen und via JDBC Verbindung zu relationale Datenbanken aufbauen können, wächst ihnen derzeit starkes Interesse zu.

2.3 Informationsquellen

2.3.1 Kommentierte Literatur

[Eile98]

Eilebrecht, Lars: Apache Web-Server; 2.Aufl. 1998; International Thomson publishing, Bonn.

Das Buch deckt ein breites Spektrum an Themen rund um den APACHE ab und gibt gute Anwendungsbeispiele für die besprochenen Themen. Die vorliegende zweite Auflage bietet einige Ergänzungen zur Version 1.3, orientiert sich jedoch an der UNIX Umgebung. Das Thema Apache unter Windows NT wird nur sehr randständig behandelt, serverseitiges JAVA nicht erwähnt. Das Buch ist das einzige gedruckte Werk, das die neue Autoconf Möglichkeiten im Anhang zu Apache 1.3 gut dokumentiert. Ein separates Kapitel geht auf frei verfügbare Indexsysteme, ein weiteres auf den Umgang mit Robots ein.

[Fiel97]

Fielding, R.; u.a.: Hypertext Transfer Protocol -HTTP/1.1. Januar 1997; RFC 2068

http://rfc.fh-koeln.de/rfc/html/rfc2068.html

[Laury97]

Laurie, Ben; Laurie, Peter: Apache - The Definitive Guide; 1.Aufl. 1997, O'Reilly & Associates, Inc, California.

Das Buch behandelt die Versionen 1.1 und 1.2 des Apache Webservers, damit lediglich die UNIX - Versionen. Themen wie Authentifizierung, Indexing, Proxy Funktion und Redirection werden ausführlich besprochen. Ein natürlicher Schwerpunkt liegt dabei auf Besprechung der Konfigurationsdateien. Zwei Kapitel widmen sich der Möglichkeit, eigene Apache module zu schreiben

[Roßbach98]

Roßbach, Stephan: Der Apache Webserver, Installation, Konfiguration, Verwaltung; 1.Aufl. 1998, Addison Wesley Longman, Bonn.

Das Buch geht auf die Unix und die NT Version von APACHE ein. Eigene Kapitel zum Thema Logdateien, serverseitiges JAVA und Sicherheitskonzepte bieten aktuelles KnowHow. Die Besprechung der Konfigurationsdateien gibt zu wenig Erklärung über den Sinn der einzelnen Parameter.

2.3.2 Wichtige Links zu Apache

Die Apache Homepage: http://www.apache.org

Der mirror in Deutschland: http://www.apache.de

Zentrale Registratur von Modulen: http://module.apache.org

Anmerkungen

[1] NCSA - National Center for Supercomputing Applications. Aus dem Personalstamm von NCSA rekrutierten viele Gründungsmitarbeiter von Netscape.

[2] Eine zentrale Registratur weiterer Module findet sich unter: http://modules.apache.org. Hier finden sich z.B. die Schnittstellen zu kommerziellen Zugriffsberechtigungssystemen wie Radius, Zerberus, Session Tracking Module, SecureSocketLayer Module etc.

[3] Konfiguration, Installation und Start des Servers sollte als root ausgeführt werden. Die Version für Microsoft Windows NT wird als selbstextrahierende .exe Datei geliefert und installiert sich eigenständig.

[4] Die gedruckte Literatur geht lediglich auf diese alte Version der Konfiguration ein. Sie wird deshalb hier nicht ausführlich dargestellt, zumal die aktuelle Version bequemer und übersichtlicher ist. Zu beachten ist, daß die scripten sich lediglich durch die Gross-Klein-Schreibung unterscheiden! Bei mischen beider Formen entsteht Verzeichnischaos!

[5] Das PHP Modul muß natürlich zuvor erzeugt worden sein. Siehe dazu README.configure im apache Distributionsverzeichnis und http://www.php.net

[6] Die bis Version 1.2 übliche Methode via "httpd -f /usr/local/httpd/conf/httpd.conf", bzw. "kill -TERM pid" kann noch verwendet werden, apachectl ist jedoch vorzuziehen

[7] In der aktuellen Version von Apache können alle Anweisungen auch in einer Datei (httpd.conf) untergebracht werden. Im Sinne der Übersichtlichkeit wird jedoch allgemein empfohlen weiterhin die drei Dateien zu verwenden. Im Sinne einer Kompakteren Darstellungsweise wurde jedoch die Möglichkeit der Zusammenfassung genutzt.

[8] Hier kann nur ein kleiner Bruchteil der Einstellungsmöglichkeiten beschrieben werden. Weitere Einträge werden im Zusammenhang mit den Funktionen für die sie dienen besprochen. Viele Beispiele und Erklärungen in Kommentarform finden sich nach der Installation in /PREFIX/etc/apache/httpd.conf.default. Diese Datei sollte als Anleitung erhalten bleiben. Anpassung an eigene Bedürfnisse sollte an Kopien vorgenommen werden.

[9] Die Module mod_log_agent, mod_log_referer und mod_cookies sollten nicht mehr verwendet werden, da ihre Funktionen von mod_log_config übernommen wurden.

[10] Mit dem Program `logresolve` können auch nachträglich die IP-Adressen in Namen umgewandelt werden. Das Programm ist in der Distribution enthalten.

[11] IP basierte virtual hosts waren aufgrund der HTTP 1.0 Spezifikation notwendig, da es Clients nur möglich war mit IP-Adresse auf einen Webserver zuzugreifen. Der Webserver konnte dann über DNS lookup den Namen aus einem A-Record erheben und hiermit feststellen, welches Dokument geliefert werden muß.

[12] Namensbasierte virtual hosts sind nur möglich, wenn Client (Browser) und Server HTTP 1.1 unterstützen. Wird der entsprechende HTTP Header nicht mitgesendet, landet der Request beim default host.

[13] Die Spezifikation von CGI findet sich unter anderem hier: http://hoohoo.ncsa.uiuc.edu/cgi/interface.html

[14] Nachfolgendes Beispiel ist leicht verändert aus SELFHTML von Stefan Münz übernommen (http://www.teamone.de/selfaktuell)