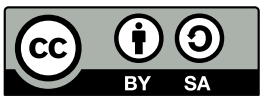
Apache HTTP-Server Teil 1

Zinching Dang

13. Juni 2014



Apache HTTP-Server

Apache HTTP-Server allgemein

- offizielle Namensherkunft: Apachen-Stamm in Nordamerika
- wurde 1994 auf Basis des NCSA HTTPd-Webservers entwickelt
- weit verbreiteter HTTP-Server für verschiedene Betriebssysteme
- Module für verschiedene Protokolle und Skriptsprachen
- modularer Aufbau ermöglicht flexible Einsatzzwecke

Der HTTP-Server wird häufig nur „Apache" genannt, da er das bekannteste Projekt der Apache Software Foundation ist. Die inoffizelle Namensherkunft „a patchy web server" kommt daher, dass er zu Anfangszeiten aus Patches für den NCSA HTTPd-Webserver bestand.

Neben üblichen Protokollen wie HTTP, HTTPS, WebDAV, etc. werden auch viele Skriptsprachen wie Perl, PHP, Python oder Ruby unterstützt. Diese können über Module für den jeweiligen Anwendungsfall aktiviert werden.

Apache HTTP-Server

Apache HTTP-Server Features

- „virtual hosts" (oder auch „VHosts")
  - mehrere Webseiten auf einem Server
  - „name-based": eigene Webseite unter jeder Domain, jedoch nur eine IP-Adresse
  - wichtig, da IPv4-Adressen knapp sind
  - passender VHosts wird anhand des ServerNames gewählt
  - „IP-based": eigene IP-Adresse für jede Webseite
- Verschlüsselung mit SSL/TLS
- Benutzer-Authentifizierung

VHosts ermöglichen es Webseiten verschiedener Domains auf einem Server mit einer einzelnen IP-Adresse zu hosten (name-based VHost). Die Konfiguration von einer Webseite pro IPv4 Adresse ist auch möglich (IP-based VHost) ist aber wegen der knappen Anzahl an IPv4-Adressen nicht das Mittel der Wahl.

Außerdem können pro VHost, d. h. pro Webseite, Verschlüsselung oder Benutzer-Authentifizierung individuell konfiguriert werden.

Server-Konfiguration

Server-Konfiguration

- mehrere Konfigurationsdateien, u. a.:
  - `/etc/apache2/apache2.conf`
  - `/etc/apache2/conf.d/`
  - `/etc/apache2/mods-available/`
  - `/etc/apache2/sites-available/`

`/etc/apache2/apache2.conf` ist die Haupt-Konfigurationsdatei. Darin werden viele Parameter wie z. B. der Pfad für Log-Files oder die Zeit für das Timeout gesetzt.

`/etc/apache2/conf.d` enthält darüberhinaus weitere Konfigurationsdateien, die beispielsweise für bestimmte Web-Anwendungen benötigt werden.

Server-Konfiguration

Server-Konfiguration

- Modul-Konfigurationen befinden sich in `/etc/apache2/mods-available`
- aktivierte Module sind Symlinks in `/etc/apache2/mods-enabled` auf das jeweilige Modul
- Module werden mit `a2enmod <modul>` und `a2dismod <modul>` aktiviert bzw. deaktiviert

Server-Konfiguration

Server-Konfiguration

- verschiedene Webseiten als VHosts
- VHosts-Konfigurationen befinden sich im Verzeichnis `/etc/apache2/sites-available`
- aktivierte Webseiten sind Symlinks in `/etc/apache2/sites-enabled` auf die jeweilige Webseite
- Webseiten werden mit `a2ensite <seite>` und `a2dissite <seite>` eingeschaltet bzw. abgeschaltet

Server-Konfiguration

Konfigurationsbeispiel

- Beispiel-Konfiguration für eine Webseite:

```
<VirtualHost 192.168.2.123:80>
  ServerAdmin webmaster@example.com
  ServerName www.example.com
  DocumentRoot /var/www
  <Directory />
    Options FollowSymLinks
    AllowOverride None
  </Directory>
  <Directory /var/www/>
    Options Indexes FollowSymLinks MultiViews
    AllowOverride None
    Order allow,deny
    allow from all
  </Directory>
</VirtualHost>
```


Für jeden VHost werden IP-Adresse und Port benötigt. Das `DocumentRoot` Verzeichnis gibt an, wo sich die Dateien befinden, die der Webserver ausliefert und darf **nicht** mit einem „/” enden. Die `Directory`-Einträge setzen Optionen für das jeweilige Verzeichnis, welches mit absolutem Pfad angegeben wird.

Server-Konfiguration

Konfigurationsbeispiel

- Beispiel-Konfiguration für name-based VHosts:

```
1 <VirtualHost 192.168.2.123:80>
2   ServerName www.example.com
3   DocumentRoot /var/www/site1
4 </VirtualHost>
5 <VirtualHost 192.168.2.123:80>
6   ServerName www.beispiel.net
7   DocumentRoot /var/www/site2
8 </VirtualHost>
```


Für die name-based VHost-Konfiguration wird ein `ServerName` benötigt, über den der entsprechende VHost ausgewählt wird. Jedem VHost (und damit auch jedem `ServerName`) kann eine eigene Webseite zugeordnet werden. Da die einzelnen VHosts durch den `ServerName` unterschieden werden, können sie alle über die gleiche IP-Adresse bereitgestellt werden.

Server-Konfiguration

Konfigurationsbeispiel

- Beispiel-Konfiguration für IP-based VHosts:

```
<VirtualHost 192.168.2.123:80>
  ServerName www.example.com
  DocumentRoot /var/www/site1
</VirtualHost>
<VirtualHost 192.168.2.234:80>
  ServerName www.beispiel.net
  DocumentRoot /var/www/site2
</VirtualHost>
```


Der Unterschied zu der name-based Host-Konfiguration ist, dass pro VHosts zusätzlich eine andere IP-Adresse vergeben wird.

Lab 4.1: Apache installieren und VHosts konfigurieren

Lab: Apache installieren und VHosts konfigurieren

- Apache installieren
- einen IP-based VHost konfigurieren
- Startseite des VHosts ändern

Verschlüsselung mit SSL/TLS

Verschlüsselung mit SSL/TLS

- wichtig, um das Mitlesen sensibler Daten zu verhindern
- basiert auf asymmetrischer Kryptographie
- statt HTTP wird HTTPS verwendet
- Server hat Private und Public Key
- Public Key ist üblicherweise durch eine Certificate Authority (CA) signiert
- allgemein anerkannte CAs sind in Browsern eingetragen
- Public Keys können auch „self-signed" sein, führen jedoch zu Warnmeldungen

TLS steht für „Transport Layer Security" und ist ein Verschlüsselungs-Protokoll, welches unter anderem für HTTPS oder IMAPS verwendet wird. „Secure Sockets Layer" (SSL) ist der Vorgänger. Zu den bekanntesten Implementierungen gehören OpenSSL und GnuTLS.

CAs bieten eine Möglichkeit die Echtheit eines Zertifikats zu überprüfen und damit den Benutzern zu versichern, dass die verschüsselte Verbindung auch zu dem richtigen Server aufgebaut wurde. Dies basiert jedoch darauf, dass der CA vertraut wird, die das Zertifikat signiert hat.

Konfiguration von SSL/TLS

Konfiguration von SSL/TLS

- Zertifikate (Private und Public Key, Zertifikat der Authority) müssen auf dem Server verfügbar sein
- das SSL-Modul muss aktiviert sein
- muss in der Konfigurationsdatei aktiviert werden
- wahlweise SSL/TLS für IP-based oder name-based VHosts
- bei name-based muss der Browser SNI[1] unterstützen
- wird von allen modernen Browser unterstützt

[1] http://de.wikipedia.org/wiki/Server_Name_Indication

Mit der Installation des Apache HTTP-Servers wurden auch Zertifikate erstellt. Diese sind jedoch „self-signed", d. h. von keiner allgemein anerkannten CA signiert, sind aber für den Eigengebrauch verwendbar.

Bei der name-based Konfiguration muss der Browser SNI unterstützen, um dem HTTP-Server mitzuteilen, welcher VHost aufgerufen werden soll.

Konfiguration von SSL/TLS

Konfigurationsbeispiel

- Beispiel-Konfiguration für einen VHost mit SSL:

```
1 <IfModule mod_ssl.c>
2 <VirtualHost 192.168.2.123:443>
3   ServerName www.example.com
4   DocumentRoot /var/www
5   SSLEngine on
6   SSLCertificateFile /etc/ssl/certs/www.example.
      com-cert.pem
7   SSLCertificateKeyFile /etc/ssl/private/www.
      example.com-key.pem
8 </VirtualHost>
9 </IfModule>
```


Zur Aktivierung von SSL/TLS für den VHost muss SSL mit `SSLEngine on` eingeschaltet werden und die Pfade zum SSL-Zertifikat und dem Private Key müssen angegeben werden. Außerdem wird bei HTTPS üblicherweise Port 443 statt 80 verwendet. Dies muss in der `VirtualHost`-Zeile angegeben werden.

Mehrere VHosts mit SSL/TLS

Mehrere VHosts mit SSL

- Konfiguration für mehrere name-based VHost mit SSL wie mit mehreren name-based VHosts
- zusätzliche SSL-Konfigurationseinträge für jeden VHosts einzeln
- Konfiguration für mehrere IP-based VHosts mit SSL wie mit mehreren IP-based VHosts
- zusätzliche SSL-Konfigurationseinträge für jeden VHosts einzeln

Lab 4.2: SSL/TLS für VHosts Konfigurieren

Lab: SSL/TLS für VHosts konfigurieren

- SSL aktivieren
- HTTPS-Zugriff für die Webseite aus Lab 4.1 aktivieren