Ubuntu 18.04 und 20.04: Apache Webserver

Ich verwende in dieser Anleitung das MPM ITK - http://mpm-itk.sesse.net/ .

Installation der Pakete

```
# aptitude install apache2 libapache2-mpm-itk
```

Konfigurationsanpassungen

Ist Apache installiert, ist er sofort einsatzbereit. Ich mache noch folgende Anpassungen:

/etc/apache2/conf-available/security.conf

```
[...]
#
# ServerTokens
# This directive configures what you return as the Server HTTP response
# Header. The default is 'Full' which sends information about the OS-
Type
# and compiled in modules.
# Set to one of:  Full | OS | Minimal | Minor | Major | Prod
# where Full conveys the most information, and Prod the least.
#ServerTokens Minimal
ServerTokens Prod
[...]
```

Sicherheistskritische Module ausschalten:

```
# a2dismod info
# a2dismod status
```

Webserver-User für virtuellen Host

Ich lege mittels eines Bash-Skripts die Benutzer und Verzeichnisse für die benötigten virtuellen Hosts an:

```
#!/bin/bash

# Sicherheitsstopp, vorher auskommentieren!
exit 0
```

```
arr=(user1 user2 user3)

for item in ${arr[*]}
do
        mkdir -p /srv/www/$item
        mkdir -p /srv/www/$item/htdocs
        mkdir -p /srv/www/$item/log
        useradd -d "/srv/www/$item" -s /bin/false $item
        chown -R $item.$item /srv/www/$item
done

exit 0
```

Virtuellen Host konfigurieren

Folgend eine Beispielkonfiguration, wie sie im Pfad `/etc/apache2/sites-available` abgelegt werden muss und mit einem Dateinamen, der mit `.conf` endet, benannt werden muss.

```
<VirtualHost *:80>
        ServerAdmin webmaster@meine-domain.de
        ServerName meine-domain.de
        ServerAlias www.meine-domain.de
        DocumentRoot /srv/www/meine-domain/htdocs

        # Mod ITK configuration
        <IfModule mpm_itk_module>
                AssignUserId meine-domain meine-domain
        </IfModule>


        <Directory "/srv/www/meine-domain/htdocs/">
                Options +Includes +MultiViews -Indexes +FollowSymLinks
                AllowOverride AuthConfig FileInfo Options
                Require all granted
        </Directory>

        ErrorLog /srv/www/meine-domain/log/error.log

        # Possible values include: debug, info, notice, warn, error, crit,
        # alert, emerg.
        LogLevel warn

        CustomLog /srv/www/meine-domain/log/access.log combined

</VirtualHost>
```

Der virtuelle Host muss noch aktiviert werden. Hinter dem Befehl `a2ensite` gehört der Dateiname des virtuellen Hosts ohne .conf.

```
# a2ensite meine-domain
# apache2ctl configtest
# service apache2 reload
```

HTTPS mit Let's Encrypt

Certbot installieren

Für Ubuntu 18.04 musste noch ein PPA eingebunden werden. Bitte nachfolgende Befhele als root ausführen:

```
apt-get update
apt-get install software-properties-common
add-apt-repository universe
add-apt-repository ppa:certbot/certbot
apt-get update
apt-get install python-certbot-apache
```

Bei Ubuntu 20.04 ist dies nicht mehr notwendig:

```
aptitude install certbot
aptitude install python3-certbot-apache
```

Cerbot: Zertifikate anfordern und Webserver anpassen

Ein Beispiel:

```
# certbot --apache --agree-tos --redirect --hsts --staple-ocsp --email
certadmin@meine-domain.de --domain www.meine-domain.de --domain meine-
domain.de
```

Cerbot: Zertifikatserneuerung testen

```
# certbot renew --dry-run
```

Cerbot Zertifikatserneuerung automatisieren

Als User root:

```
# crontab -e
```

Beispiel für die Crontab:

```
[...]
```

```
# m h  dom mon dow   command
35 7 * * * /usr/bin/certbot renew  >> /var/log/le-renew.log
```

Verschlüsselung: Protokolle und Ciphers

/etc/letsencrypt/options-ssl-apache.conf

```
# This file contains important security parameters. If you modify this
file
# manually, Certbot will be unable to automatically provide future
security
# updates. Instead, Certbot will print and log an error message with a
path to
# the up-to-date file that you will need to refer to when manually
updating
# this file.

SSLEngine on

# Intermediate configuration, tweak to your needs
SSLProtocol             all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
SSLCipherSuite          ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-
CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:DHE-
RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES128-
SHA256:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-
SHA384:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES256-SHA384:ECDHE-ECDSA-
AES256-SHA:ECDHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-
SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES256-SHA:ECDHE-ECDSA-DES-CBC3-
SHA:ECDHE-RSA-DES-CBC3-SHA:EDH-RSA-DES-CBC3-SHA:AES128-GCM-
SHA256:AES256-GCM-SHA384:AES128-SHA256:AES256-SHA256:AES128-SHA:AES256-
SHA:DES-CBC3-SHA:!DSS
SSLHonorCipherOrder     on
SSLCompression          off

SSLOptions +StrictRequire

# Add vhost name to log entries:
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-
agent}i\"" vhost_combined
LogFormat "%v %h %l %u %t \"%r\" %>s %b" vhost_common

#CustomLog /var/log/apache2/access.log vhost_combined
#LogLevel warn
#ErrorLog /var/log/apache2/error.log

# Always ensure Cookies have "Secure" set (JAH 2012/1)
#Header edit Set-Cookie (?i)^(.*)(;\s*secure)??((\s*;)?(.*)) "$1;
Secure$3$4"
```

In der Konfiguration des virtuellen Hosts:

```
[...]
<IfModule mod_headers.c>
                Header always set Strict-Transport-Security "max-
age=15552000; includeSubDomains"
</IfModule>
[...]
```

Logfiles rotieren lassen

/etc/logrotate.d/apache2

```
/var/log/apache2/*.log /srv/www/*/log/*.log {
        daily
        missingok
        rotate 14
        compress
        delaycompress
        notifempty
        create 640 root adm
        sharedscripts
        postrotate
                if invoke-rc.d apache2 status > /dev/null 2>&1; then \
                    invoke-rc.d apache2 reload > /dev/null 2>&1; \
                fi;
        endscript
        prerotate
                if [ -d /etc/logrotate.d/httpd-prerotate ]; then \
                        run-parts /etc/logrotate.d/httpd-prerotate; \
                fi; \
        endscript
}
```

PHP

```
# aptitude install php php-cli php-json php-pdo php-mysql php-zip php-gd
php-mbstring php-curl php-xml php-pear php-bcmath libapache2-mod-php php-
imagick php-common
```

ModSecurity

Dieses Apache-Modul erkennt Angriffmuster (z. B. Session Hijacking, Cross-Site-Scripting) und verweigert bei Angriff die Anfrage.

Installation

```
aptitude install libapache2-mod-security2
```

Konfiguration

```
cd /etc/modsecurity/
cp modsecurity.conf-recommended modsecurity.conf
```

ModSecurity sollte nun im DetectOnly-Modus laufen und erkannte Angriffsmuster unter `/var/log/apache2/modsec_audit.log` aufzeichnen.

Dies kann folgendermaßen geändert werden:

/etc/modsecurity/modsecurity.conf

```
[...]
# Enable ModSecurity, attaching it to every transaction. Use detection
# only to start with, because that minimises the chances of post-
installation
# disruption.
#
#SecRuleEngine DetectionOnly
SecRuleEngine On
[...]
```

/etc/modsecurity/modsecurity.conf

```
[...]
# Log everything we know about a transaction.
#SecAuditLogParts ABDEFHIJZ
SecAuditLogParts ABCEFHJKZ
[...]
```

Aktuelles OWASP Core Rule Set

```
cd /etc/modsecurity
git clone https://github.com/SpiderLabs/owasp-modsecurity-crs.git
cd owasp-modsecurity-crs
cp crs-setup.conf.example crs-setup.conf
```

Das Regelwerk hinterlegen:

/etc/apache2/mods-enabled/security2.conf

```
<IfModule security2_module>
        # Default Debian dir for modsecurity's persistent data
        SecDataDir /var/cache/modsecurity

        # Include all the *.conf files in /etc/modsecurity.
        # Keeping your local configuration in that directory
        # will allow for an easy upgrade of THIS file and
        # make your life easier
        # IncludeOptional /etc/modsecurity/*.conf
        IncludeOptional /etc/modsecurity/modsecurity.conf
        IncludeOptional /etc/modsecurity/owasp-modsecurity-crs/crs-
setup.conf
        IncludeOptional /etc/modsecurity/owasp-modsecurity-
crs/rules/*.conf

        # Include OWASP ModSecurity CRS rules if installed
        # IncludeOptional /usr/share/modsecurity-crs/*.load
</IfModule>
```

Aktualisieren kann man das Ruleset, indem man einfach eine aktuelle Kopie von github zieht und diese in sein Conf-Verzeichnis kopiert. Bitte vorher die Konfig-Dateien an Seite legen.

Logfiles

Da sehr viele Daten geloggt werden, sollte man das Logfile täglich rotieren lassen:

/etc/logrotate.d/modsecurity

```
/var/log/apache2/modsec_audit.log
{
        rotate 14
        daily
        missingok
        compress
        delaycompress
        notifempty
}
```

Regelausnahmen: Nextcloud

Virtuellen Host anpassen für WebDAV

```
[...]
<IfModule mod_headers.c>
                Header always set Strict-Transport-Security "max-
```

```
age=15552000; includeSubDomains"
                Redirect 301 /.well-known/carddav /remote.php/dav
                Redirect 301 /.well-known/caldav /remote.php/dav
</IfModule>
[...]
```

Ausnahmeregelungen für Nextcloud

REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf

```
[...]
# Own ip address not monitored
SecRule REMOTE_ADDR "^20\.21\.22\.23"
"id:1002,phase:1,nolog,allow,ctl:ruleEngine=Off"

SecRule REQUEST_URI "@beginsWith /remote.php/dav"
"id:1003,phase:1,nolog,allow,ctl:ruleRemoveById=932110"
```

Globale Ausnahme-Presets für Nextcloud

Achtung: Gilt für alle virtuellen Hosts!

```
[...]
SecAction \
 "id:900130,\
  phase:1,\
  nolog,\
  pass,\
  t:none,\
  setvar:tx.crs_exclusions_nextcloud=1"
[...]
```

> https://www.linuxbabe.com/security/modsecurity-apache-debian-ubuntu