phpMyAdmin unter Redhat / CentOS

Zu einer komfortablen Administration eines **MariaDB-Datenbankservers** unter Redhat7, greift man am einfachsten auf das PHP-Projekt phpMyAdmin zurück.

Die Software ist in PHP implementiert; daher kommt der Name phpMyAdmin. Die meisten Funktionen können ausgeführt werden, ohne selbst SQL-Anweisungen zu schreiben, wie z. B. Datensätze auflisten, Tabellen anlegen/löschen, Spalten hinzufügen, Datenbanken anlegen/löschen und Benutzer verwalten.

Voraussetzungen

Für unseren komfortablen Weg der Administration des **MariaDB-Servers** durch phpMyAdmin, muss natürlich ein erstmals ein funktionstüchtiger MariaDB-Server und zweitens ein passender Apache-Webserver oder der NGiNX Webserver zur Verfügung stehen. Weiter werden auch folgende zusätzliche PHP-Module benötigt:

- **php**
- **php-cli**
- **php-common**
- **php-gd**
- **php-mbstring**
- **php-mcrypt**
- **php-mysql**
- **php-pdo**

Diese Pakete, müssen noch vor der Installation installiert werden; sofern dies nicht schon bei der Grundinstallation des Apache-Webservers erfolgt ist. Anschliessend kann mit der Installation begonnen werden!

Installation von phpMyAdmin

1. Herunterladen der neusten Version von phpMyAdmin.

```
# wget $(curl -s https://www.phpmyadmin.net/downloads/ | grep -oP
'http[^"]*(?=">phpMyAdmin-)' | head -1 | rev | cut -c5- | rev)
```

2. Erstellen der benötigten Ordner für die phpMyAdmin Instanz.

```
# mkdir -p /var/www/html/php_my_admin/tmp/
```

3. Entpacken des heruntergeladenen Skriptes..

```
# unzip phpMyAdmin-*-all-languages.zip
```

4. Kopieren in das neu angelegte phpMyAdmin Webverzeichnis:

```
# cp -av phpMyAdmin-*-all-languages/* /var/www/html/php_my_admin/
```

5. Setzen der korrekten Berechtigungen:

```
# chown -R apache:apache /var/www/html/php_my_admin/
# chmod -R 775 /var/www/html/php_my_admin/
```

6. Umbenennen der Konfigurationsdatei, für den Produktiven Betrieb:

```
# mv /var/www/html/php_my_admin/config.sample.inc.php
/var/www/html/php_my_admin/config.inc.php
```

7. Anschliessend, muss noch in der Konfiguration unter: `/var/www/html/php_my_admin/config.inc.php` beim Wert `$cfg['blowfish_secret']` ein mindestens **zwanzig stelliger MD5-Hash** zur internen Verschlüsselung von phpMyAdmin hinterlegt werden! Dies wird durch folgenden Befehl automatisiert:

```
# sed -e "s/\['blowfish_secret'\] = ''/\['blowfish_secret'\] = '`date
+%s | sha256sum | base64 | head -c 32; echo`'/" -i
/var/www/html/php_my_admin/config.inc.php
```

8. Zum Abschluss werden die SELinux Berechtigungen gesetzt und diverse Aufräumungsarbeiten durchgeführt:

```
# semanage fcontext -a -t httpd_sys_rw_content_t
'/var/www/html/php_my_admin/tmp(/.*)?'
# restorecon -Rv '/var/www/html/'
# rm -Rf phpMyAdmin-*
```

Installation complete!

Konfiguration des Zugriffs auf phpMyAdmin

ACHTUNG bei Variante 1 als sowohl Variante 2 müssen natürlich noch die beiden ersten `Require ip XXX.XXX.X.X` Werte auf die eigenen IP-Adressen angepasst werden. Es wird nur diesen IP's den Zugriff auf phpmyadmin gewährt.

Variante 1 - Verwenden des Standard Ports und einschränken des Zugriffs

Um den Zugriff auf unser phpMyAdmin Verzeichnis → `php_my_admin` noch zusätzlich abzusichern und den Zugriff von aussen zu verbieten, kann wie folgt vorgegangen werden:

```
# vim /etc/httpd/conf.d/phpMyAdmin.conf
```

ODER: Für die SCL Repository Version

```
# vim /opt/rh/httpd24/root/etc/httpd/conf.d/phpMyAdmin.conf
```

```
# Web application to manage MySQL
#
<Directory "/var/www/html/php_my_admin">
    Options -Indexes +FollowSymLinks
    AllowOverride None
    Require ip 192.168.1.5
    Require ip 192.168.1.65
    Require ip 127.0.0.1
    Require ip ::1
</Directory>

<Directory "/var/www/html/php_my_admin/libraries">
    Require all denied
</Directory>

<Directory "/var/www/html/php_my_admin/setup/lib">
    Require all denied
</Directory>

<Directory "/var/www/html/php_my_admin/setup/frames">
    Require all denied
</Directory>
```

Nach gemachten Änderungen, muss zum übernehmen der Konfiguration zwingend ein reload des Apache Webservers ausgeführt werden!

```
# systemctl reload httpd
```

Hardening complete!

Variante 2 - Verwenden des Standard Ports aber einschränken des Zugriffs

Erstellen eines eigenen VirtualHosts (mit eigenen Port) auf welchem dann ausschliesslich auf phpmyadmin zugegriffen wird:

```
# vim /etc/httpd/conf.d/phpMyAdmin.conf
```

ODER: Für die SCL Repository Version

```
# vim /opt/rh/httpd24/root/etc/httpd/conf.d/phpMyAdmin.conf
```

```
Listen 6060

<VirtualHost *:6060>
    ServerAdmin michael.r467@gmail.com

    ErrorLog "/var/log/httpd/phpmyadmin-error_log"
    CustomLog "/var/log/httpd/phpmyadmin-access_log" combined

    # Web application to manage MySQL
    #
    DocumentRoot "/var/www/html/php_my_admin/"

    <Directory "/var/www/html/php_my_admin">
        Options -Indexes +FollowSymLinks
        AllowOverride None
        Require ip 192.168.1.5
        Require ip 192.168.1.65
        Require ip 127.0.0.1
        Require ip ::1
    </Directory>

    <Directory "/var/www/html/php_my_admin/libraries">
        Require all denied
    </Directory>

    <Directory "/var/www/html/php_my_admin/setup/lib">
        Require all denied
    </Directory>

    <Directory "/var/www/html/php_my_admin/setup/frames">
        Require all denied
    </Directory>

</VirtualHost>
```

Damit der Webserver auch andere Ports als 80 / 8080 und 443 binden kann muss folgende SELinux boolean aktiviert werden:

```
# setsebool -P nis_enabled 1
```

Um die gemachten Änderungen zu übernehmen wird nun ein reload des Apache Webservers ausgeführt und anschliessend noch der separate Port 6060 geöffnet!

```
# systemctl reload httpd

# firewall-cmd --zone=public --add-port=6060/tcp --permanent
# firewall-cmd --reload
```

Hardening complete!

Last update: **2020/04/17 13:45**