Datenbanken unter Redhat / CentOS

Nachdem Oracle die Firma Sun Microsystems im Jahr 2009 aufkaufte, entschloss sich der Hauptentwickler Ulf Michael Widenius der OpenSource Datenbank mySQL als eigenständigen Fork weiter zu entwickeln. Für das neue relationales Open-Source-Datenbankverwaltungssystem, welcher grundsätzlich zu MySQL kompatibel ist, wählte Widenius den Namen MariaDB. Im folgenden Artikel findet man tiefergehende Informationen zu den Unterschieden beider Datenbank-Daemon.

MariaDB löste mit RHEL7 / CentOS7 die bis dahin verwendete MySQL-Datenbank ab. Nachfolgend wird aufgezeit, wie man die Mariadb korrekt auf seinem System implementiert / härtet.

Installation MariaDB

Die Installation des Datenbankservers gestaltet sich recht einfach, da das notwendige Paket als RPM aus dem Base-Repository unserer Redhat-Installation zur Verfügung gestellt wird. Die Installation selbst erfolgt mit einem einfachen **yum** Kommando aus RHEL7.

```
# yum install mariadb-server mariadb -y
```

Neben dem Server-Part mysql-server wird auch der Client-Part mysql sowie weitere Perl-Datenbankmodule installiert. Was uns die einzelnen Programmpakete mitbringen, erkunden wir bei Bedarf mit folgendem Befehl:

```
# rpm -qil mariadb-server
```

```
Name        : mariadb-server
Epoch       : 1
Version     : 5.5.56
Release     : 2.el7
Architecture: x86_64
Install Date: Wed 13 Sep 2017 04:17:44 PM CEST
Group       : Applications/Databases
Size        : 61123351
License     : GPLv2 with exceptions and LGPLv2 and BSD
Signature   : RSA/SHA256, Fri 09 Jun 2017 08:13:00 AM CEST, Key ID
199e2f91fd431d51
Source RPM  : mariadb-5.5.56-2.el7.src.rpm
Build Date  : Thu 08 Jun 2017 07:16:25 PM CEST
Build Host  : x86-017.build.eng.bos.redhat.com
Relocations : (not relocatable)
Packager    : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
Vendor      : Red Hat, Inc.
URL         : http://mariadb.org
Summary     : The MariaDB server and related files
Description :
```

```
MariaDB is a multi-user, multi-threaded SQL database server. It is a
client/server implementation consisting of a server daemon (mysqld)
and many different client programs and libraries. This package contains
the MariaDB server and some accompanying files and directories.
MariaDB is a community developed branch of MySQL.
/etc/logrotate.d/mariadb
/etc/my.cnf.d/server.cnf
/usr/bin/innochecksum
/usr/bin/myisam_ftdump
/usr/bin/myisamchk
/usr/bin/myisamlog
/usr/bin/myisampack
/usr/bin/mysql_convert_table_format
/usr/bin/mysql_fix_extensions
/usr/bin/mysql_install_db
/usr/bin/mysql_plugin
...
```

Konfiguration MariaDB

Daemon Konfiguration - my.cnf

Die Konfiguration des Datenbankservers erfolgt über die Konfigurationsdatei **/etc/my.cnf**, die bei der Installation bereits mitgeliefert wird.

```
# vim /etc/my.cnf
```

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
# Disabling symbolic-links is recommended to prevent assorted security risks
symbolic-links=0
# Settings user and group are ignored when systemd is used.
# If you need to run mysqld under a different user or group,
# customize your systemd unit file for mariadb according to the
# instructions in http://fedoraproject.org/wiki/Systemd

[mysqld_safe]
log-error=/var/log/mariadb/mariadb.log
pid-file=/var/run/mariadb/mariadb.pid

#
# include all files from the config directory
#
!includedir /etc/my.cnf.d
```

Eine genauere Beschreibung aller Konfigurations-Variablen findet sich **hier**.

Zusätzliche Konfigurations-Beispiele finden man übrigens auch im Verzeichnis **/usr/share/mysql/**.

```
# ls -lisa /usr/share/mysql/*.cnf
```

```
-rw-r--r--. 1 root root  4920 Feb  3 15:44 /usr/share/mysql/my-huge.cnf
-rw-r--r--. 1 root root 20438 Feb  3 15:44 /usr/share/mysql/my-innodb-
heavy-4G.cnf
-rw-r--r--. 1 root root  4907 Feb  3 15:44 /usr/share/mysql/my-large.cnf
-rw-r--r--. 1 root root  4920 Feb  3 15:44 /usr/share/mysql/my-medium.cnf
-rw-r--r--. 1 root root  2846 Feb  3 15:44 /usr/share/mysql/my-small.cnf
```

Der erste Start

Nun ist es an der Zeit unseren Datenbank-Server das erste mal zu starten.

```
# systemctl start mariadb
```

Der Start wird im Logfile des Datenbankservers **/var/log/mariadb/mariadb.log** entsprechend dokumentiert.

```
# less /var/log/mariadb/mariadb.log
```

```
150307 22:10:14 mysqld_safe Starting mysqld daemon with databases from
/var/lib/mysql/data
150307 22:10:14 InnoDB: The InnoDB memory heap is disabled
150307 22:10:14 InnoDB: Mutexes and rw_locks use GCC atomic builtins
150307 22:10:14 InnoDB: Compressed tables use zlib 1.2.7
150307 22:10:14 InnoDB: Using Linux native AIO
150307 22:10:14 InnoDB: Initializing buffer pool, size = 128.0M
150307 22:10:14 InnoDB: Completed initialization of buffer pool
InnoDB: The first specified data file ./ibdata1 did not exist:
InnoDB: a new database to be created!
150307 22:10:14  InnoDB: Setting file ./ibdata1 size to 10 MB
InnoDB: Database physically writes the file full: wait...
150307 22:10:14  InnoDB: Log file ./ib_logfile0 did not exist: new to be
created
InnoDB: Setting log file ./ib_logfile0 size to 5 MB
InnoDB: Database physically writes the file full: wait...
150307 22:10:14  InnoDB: Log file ./ib_logfile1 did not exist: new to be
created
InnoDB: Setting log file ./ib_logfile1 size to 5 MB
```

```
InnoDB: Database physically writes the file full: wait...
InnoDB: Doublewrite buffer not found: creating new
InnoDB: Doublewrite buffer created
InnoDB: 127 rollback segment(s) active.
InnoDB: Creating foreign key constraint system tables
InnoDB: Foreign key constraint system tables created
150307 22:10:15  InnoDB: Waiting for the background threads to start
150307 22:10:16 Percona XtraDB (http://www.percona.com) 5.5.40-MariaDB-36.1
started; log sequence number 0
150307 22:10:16 [Note] Plugin 'FEEDBACK' is disabled.
150307 22:10:16 [Note] Server socket created on IP: '0.0.0.0'.
150307 22:10:16 [Note] Event Scheduler: Loaded 0 events
150307 22:10:16 [Note] /usr/libexec/mysqld: ready for connections.
Version: '5.5.41-MariaDB'  socket: '/var/lib/mysql/mysql.sock'  port: 3306
MariaDB Server
```

In unserem Datenbankverzeichnis **/var/lib/mysql/data** wurden auch die ersten Datenbankdateien angelegt.

```
# ls -lisa /var/lib/mysql/data
```

```
total 28700
-rw-rw----. 1 mysql mysql    16384 Mar  3 22:10 aria_log.00000001
-rw-rw----. 1 mysql mysql       52 Mar  3 22:10 aria_log_control
-rw-rw----. 1 mysql mysql 18874368 Mar  3 22:10 ibdata1
-rw-rw----. 1 mysql mysql  5242880 Mar  3 22:10 ib_logfile0
-rw-rw----. 1 mysql mysql  5242880 Mar  3 22:10 ib_logfile1
drwx------. 2 mysql mysql     4096 Mar  3 22:10 mysql
srwxrwxrwx. 1 mysql mysql        0 Mar  3 22:10 mysql.sock
drwx------. 2 mysql mysql     4096 Mar  3 22:10 performance_schema
drwx------. 2 mysql mysql        6 Mar  3 22:10 test
```

Möchten wir überprüfen, ob der MariaDB-Server läuft, haben wir mehrere Möglichkeiten.

1. Daemon Überprüfung mit **systemctl**:

   ```
   # systemctl status mariadb -l
   ```

   ```
   mariadb.service - MariaDB database server
      Loaded: loaded (/usr/lib/systemd/system/mariadb.service; disabled)
      Active: active (running) since Sat 2015-03-07 22:10:17 CET; 27min
   ago
     Process: 27040 ExecStartPost=/usr/libexec/mariadb-wait-ready $MAINPID
   (code=exited, status=0/SUCCESS)
     Process: 26961 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir %n
   (code=exited, status=0/SUCCESS)
   ```

```
 Main PID: 27039 (mysqld_safe)
   CGroup: /system.slice/mariadb.service
           ├─27039 /bin/sh /usr/bin/mysqld_safe --basedir=/usr
           └─27198 /usr/libexec/mysqld --basedir=/usr --
datadir=/var/lib/mysql/data --plugin-dir=/usr/lib64/mysql/plugin --log-
error=/var/log/mariadb/mariadb.log --pid-
file=/var/run/mariadb/mariadb.pid --socket=/var/lib/mysql/mysql.sock
```

2. Prozess Überprüfung mit **ps**:

```
# ps aux | grep mariadb
```

```
mysql    27039  0.0  0.1 115344  1620 ?        Ss   22:10   0:00
/bin/sh /usr/bin/mysqld_safe --basedir=/usr
mysql    27198  0.0  8.3 905348 84784 ?        Sl   22:10   0:01
/usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysql/data --
plugin-dir=/usr/lib64/mysql/plugin --log-
error=/var/log/mariadb/mariadb.log --pid-
file=/var/run/mariadb/mariadb.pid --socket=/var/lib/mysql/mysql.sock
root     27302  0.0  0.0 112640   924 pts/0    R+   22:41   0:00 grep -
-color=auto mysql
```

3. Netzwerk Überprüfung mit **netstat**:

```
# netstat -tulpn | grep 3306
```

```
Proto Recv-Q Send-Q Local Address           Foreign Address
State       PID/Program name
tcp        0      0 127.0.0.1:25            0.0.0.0:*
LISTEN      1728/master
tcp        0      0 0.0.0.0:3306            0.0.0.0:*
LISTEN      27198/mysqld
tcp        0      0 0.0.0.0:22              0.0.0.0:*
LISTEN      1357/sshd
tcp6       0      0 ::1:25                  :::*
LISTEN      1728/master
tcp6       0      0 :::80                   :::*
LISTEN      26249/httpd
tcp6       0      0 :::22                   :::*
LISTEN      1357/sshd
udp        0      0 0.0.0.0:40525           0.0.0.0:*
571/avahi-daemon: r
udp        0      0 0.0.0.0:52944           0.0.0.0:*
588/chronyd
udp        0      0 0.0.0.0:5353            0.0.0.0:*
571/avahi-daemon: r
udp        0      0 127.0.0.1:323           0.0.0.0:*
588/chronyd
```

Autostart beim booten des Systems

Damit nun unser MariaDBL-Server beim Booten automatisch gestartet wird, nehmen wir noch folgenden Konfigurationsschritt vor.

```
# systemctl enable mariadb
```

```
ln -s '/usr/lib/systemd/system/mariadb.service' '/etc/systemd/system/multi-user.target.wants/mariadb.service'
```

Wollen wir überprüfen, ob der Datenbank-Daemon beim Serverstart automatisch gestartet wird, fragen wir dies mit folgendem Befehl ab.

```
# systemctl is-enabled mariadb
```

```
enabled
```

Startet der Datenbank-Daemon nicht automatisch, wird ein **disabled** zurück gemeldet.

Firewall Konfiguration für distributed Systems

Falls sich die frisch installierte Mariadb nicht auf dem gleichen Server wie die spätere Applikation befindet, so müssen wir nun zum erlauben der externen Verbindung noch diverse Änderungen mittels **firewalld** vornehmen.

ACHTUNG: Nur durchführen, wenn sich die mariadb und der Apache (Die Applikation) nicht auf dem ein und selben Server befinden!

Unter **RHEL 7** und **CentOS 7** wird als Standard-Firewall die dynamische **firewalld** verwendet. Ein grosser Vorteil der dynamischen Paketfilterregeln ist unter anderem, dass zur Aktivierung der neuen Firewall-Regel(n) nicht der Daemon durchgestartet werden muss und somit alle aktiven Verbindungen kurz getrennt werden. Sondern unsere Änderungen können **on-the-fly** aktiviert oder auch wieder deaktiviert werden.

In meinem kleinen Konfigurationsbeispiel hat der **MariaDB-Server** die IP-Adresse **10.0.0.37** und der **Applikations-Server** die **10.0.0.27**. **Nun braucht man also eine neue Firewall-Regen, die ausschliesslich Verbindungen der Source-IP: 10.0.0.27 auf die Destination-IP: 10.0.0.37 auf Port 3306 gestattet.** Mit nachfolgendem Befehl wird diese restriktive Regel angelegt:

```
# firewall-cmd --permanent --zone=public --add-rich-rule="rule family="ipv4"
source address="10.0.0.27/32" port protocol="tcp" port="3306" destination
address="10.0.0.37/32" accept"
```

Zum Aktivieren der neuen Regel, brauchtes nun nur noch einen reload des Firewall-Daemons.

```
# firewall-cmd --reload
```

Fragt man nun den Regelsatz unserer **iptables**-basieten Firewall ab, finden wir in der Chain **IN_public_allow** unsere aktive Regel.

```
# iptables -nvL IN_public_allow
```

```
Chain IN_public_allow (1 references)
pkts bytes target     prot opt in     out     source               
destination
10K   25K ACCEPT     tcp  --  *      *       10.0.0.27            10.0.0.37
tcp dpt:3306 ctstate NEW
2656  159K ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
tcp dpt:22 ctstate NEW
```

Natürlich kann auch mit dem Befehl **firewall-cmd** abgefragt werden, welche Dienste in der Zone **public** geöffnet sind.

```
# firewall-cmd --zone=public --list-services
```

```
mysql ssh
```

Härten der Mariadb Datenban (Security relevant!)

Wie bei der doch grossen Ausgabe beim erstmaligen Start des Datenbank-Daemons angeraten, werden wir nun die sicherheitsrelevanten Konfigurationsänderungen vornehmen.

Zuvor, generieren wir jedoch noch einen sicheren DB-Schlüssel für den MariaDB-Root User und speichern ihn unter /root/ mit folgenden Befehl ab:

```
# openssl rand -base64 30 > /root/.mariadb-root-pw && cat /root/.mariadb-
root-pw
```

Anschliessend, benutzen wir einfach das mitgelieferte Script **/usr/bin/mysql_secure_installation**, welches folgende Änderungen vornimmt:

1. Datenbankpasswort des MySQL-Datenbankuser **root** setzen

2. Anonyme Benutzerkonten löschen
3. Deaktivieren der Remote-Zugriffsmöglichkeit für den MySQL-Datenbankuser **root**
4. Löschen der nicht benötigten Testdatenbank **test**

Wichtig: Das anschliessende Hardenig sollte auf produktiv-Systemen keinesfalls übersprungen werden!!

```
# /usr/bin/mysql_secure_installation
```

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none): ENTER
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] y
New password:           *EINGABE DES ZUVOR GENERIERTEN MARIADB-ROOT-PWs*
Re-enter new password:  *EINGABE DES ZUVOR GENERIERTEN MARIADB-ROOT-PWs*
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
 ... Success!

By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.
```

```
Remove test database and access to it? [Y/n] y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

Erstellen einer Initialen Datenbank

Um sich in Mariadb eine neue Datenbank zu erstellen, wird folgendermassen vorgegangen:

1. Einloggen in die Mariadb als root:

   ```
   # mysql -u root --password=$(cat /root/.mariadb-root-pw)
   ```

2. Erstellen der ersten Datenbank:

   ```
   CREATE DATABASE mydatabase;
   ```

   Note: Every MySQL statement or command must end in a semi-colon (;), so check to make sure that this is present if you are running into any issues.

3. Next, we are going to create a new MySQL user account that we will use exclusively to operate on our new database. I am going to call the **new account** “dbuser” and will assign it a **password** of “dbuserpasswort”.

   ```
   CREATE USER dbuser@localhost IDENTIFIED BY 'dbuserpasswort';
   ```

4. At this point, you have a database and user account that are each specifically made for our new environment. However, the user has no access to the database. We need to link the two components together by granting our user access to the database.

   ```
   GRANT ALL PRIVILEGES ON mydatabase.* TO dbuser@localhost IDENTIFIED BY
   'dbuserpasswort';
   ```

5. Now that the user has access to the database, we need to flush the privileges so that MySQL

knows about the recent privilege changes that we've made:

```
FLUSH PRIVILEGES;
```

6. Once these commands have all been executed, we can exit out of the MySQL command prompt by typing:

```
exit
```

ACHTUNG: UM Zugriff mit php auf die mariadb zu erhalten, muss hierfür noch das php-mysql Packet installiert werden!

Vorhandenen Datenbank-Dump in neu erstellte Datenbank schreiben:

```
# mysql -u root --password=$(cat /root/.mariadb-root-pw) -h localhost
mydatabase < mysqldump.sql
```

Logrotate

Bei einem unter Last stehendem MariaDB-Sserver kann unter Umständen das zugehörige Logfile **/var/log/mariadb/mariadb.log** recht schnell anwachsen. In der Datei **mariadb** aus dem Verzeichnis **/etc** finden wir dazu alle nötigen Informationen.

```
 # less /etc/logrotate.d/mariadb
```

```
# This logname can be set in /etc/my.cnf
# by setting the variable "log-error"
# in the [mysqld_safe] section as follows:
#
# [mysqld_safe]
# log-error=/var/log/mariadb/mariadb.log
#
# If the root user has a password you have to create a
# /root/.my.cnf configuration file with the following
# content:
#
# [mysqladmin]
# password = <secret>
# user= root
#
# where "<secret>" is the password.
#
# ATTENTION: This /root/.my.cnf should be readable ONLY
# for root !
```

```
# Then, un-comment the following lines to enable rotation of mysql's log
file:

#/var/log/mariadb/mariadb.log {
#        create 640 mysql mysql
#        notifempty
#       daily
#        rotate 3
#        missingok
#        compress
#    postrotate
#       # just if mysqld is really running
#       if test -x /usr/bin/mysqladmin && \
#          /usr/bin/mysqladmin ping &>/dev/null
#       then
#          /usr/bin/mysqladmin flush-logs
#       fi
#    endscript
#}
```

In der MariaDB-Konfigurationsdatei **/etc/my.cnf** aus dem RPM ist in der Sektion **[mysqld_safe]** das Error-Log gesetzt:

```
log-error=/var/log/mariadb/mariadb.log
```

Da wir dem im Kapitel installation_absichern für den User **root** ein Passwort vergeben haben, werden wir nun die Authentifizierungsdaten im Verzeichnis des Users **root** ablegen. Hierzu legen wir erst einmal diese Datei an.

```
 # touch /root/.my.cnf
```

Anschliessend stellen wir sicher dass auch wirklich **NUR root** diese Datei lesen kann.

```
 # chmod 600 /root/.my.cnf
```

Bevor wir die Daten in der gerade angelegten Datei hinterlegen, überprüfen wir noch, ob auch wirklich die Berechtigungen passen.

```
-rw-------. 1 root root 0 Mar  8 00:15 /root/.my.cnf
```

Da alles passt, befüllen wir nun die Datei **/root/.my.cnf**.

```
 # vim /root/.my.cnf
```

```
 [mysqladmin]
password = dxiFHdig10JXyRAec74j7bcPdyVGX9I1BxcYcoFs
```

```
user= root
```

Nun aktivieren wir noch in der Datei **/etc/logrotate.d/mariadb** die Zeilen in der unteren Hälfte, damit der logrotate-Mechanismus auch scharf geschalten ist.

```
 # vim /etc/logrotate.d/mariadb
```

```
# This logname can be set in /etc/my.cnf
# by setting the variable "log-error"
# in the [mysqld_safe] section as follows:
#
# [mysqld_safe]
# log-error=/var/log/mariadb/mariadb.log
#
# If the root user has a password you have to create a
# /root/.my.cnf configuration file with the following
# content:
#
# [mysqladmin]
# password = <secret>
# user= root
#
# where "<secret>" is the password.
#
# ATTENTION: This /root/.my.cnf should be readable ONLY
# for root !

# Then, un-comment the following lines to enable rotation of mysql's log
file:

# rebermi : 2017-02-06
# logrotate aktiviert
/var/log/mariadb/mariadb.log {
        create 640 mysql mysql
        notifempty
        daily
        rotate 3
        missingok
        compress
    postrotate
        # just if mysqld is really running
        if test -x /usr/bin/mysqladmin && \
           /usr/bin/mysqladmin ping &>/dev/null
        then
           /usr/bin/mysqladmin flush-logs
        fi
    endscript
}
```

Datenbankhandling

mysqladmin

Mit Hilfe des Hilfsprogrammes **mysqladmin** aus dem Clientpaket **mysql** können umfangreiche Abfrage gegen unsere Datenbank durchgeführt werden. Startet man das Programm ohne weitere Angaben von Optionen, werden die möglichen Optionen am Bildschirm ausgegeben.

```
 # mysqladmin
```

```
mysqladmin  Ver 9.0 Distrib 5.5.41-MariaDB, for Linux on x86_64
Copyright (c) 2000, 2014, Oracle, MariaDB Corporation Ab and others.

Administration program for the mysqld daemon.
Usage: mysqladmin [OPTIONS] command command....

Default options are read from the following files in the given order:
/etc/mysql/my.cnf /etc/my.cnf ~/.my.cnf
The following groups are read: mysqladmin client client-server client-
mariadb
The following options may be given as the first argument:
--print-defaults        Print the program argument list and exit.
--no-defaults           Don't read default options from any option file.
--defaults-file=#       Only read default options from the given file #.
--defaults-extra-file=# Read this file after the global files are read.

  -c, --count=#       Number of iterations to make. This works with -i
                      (--sleep) only.
  --debug-check       Check memory and open file usage at exit.
  --debug-info        Print some debug info at exit.
  -f, --force         Don't ask for confirmation on drop database; with
                      multiple commands, continue even if an error occurs.
  -C, --compress      Use compression in server/client protocol.
  --character-sets-dir=name
                      Directory for character set files.
  --default-character-set=name
                      Set the default character set.
  -?, --help          Display this help and exit.
  -h, --host=name     Connect to host.
  -b, --no-beep       Turn off beep on error.
  -p, --password[=name]
                      Password to use when connecting to server. If password
is
                      not given it's asked from the tty.
  -P, --port=#        Port number to use for connection or 0 for default to,
```

```
in
                      order of preference, my.cnf, $MYSQL_TCP_PORT,
                      /etc/services, built-in default (3306).
  --protocol=name     The protocol to use for connection (tcp, socket, pipe,
                      memory).
  -r, --relative      Show difference between current and previous values
when
                      used with -i. Currently only works with extended-
status.
  -s, --silent        Silently exit if one can't connect to server.
  -S, --socket=name   The socket file to use for connection.
  -i, --sleep=#       Execute commands repeatedly with a sleep between.
  --ssl               Enable SSL for connection (automatically enabled with
                      other flags).
  --ssl-ca=name       CA file in PEM format (check OpenSSL docs, implies
                      --ssl).
  --ssl-capath=name   CA directory (check OpenSSL docs, implies --ssl).
  --ssl-cert=name     X509 cert in PEM format (implies --ssl).
  --ssl-cipher=name   SSL cipher to use (implies --ssl).
  --ssl-key=name      X509 key in PEM format (implies --ssl).
  --ssl-verify-server-cert
                      Verify server's "Common Name" in its cert against
                      hostname used when connecting. This option is disabled
by
                      default.
  -u, --user=name     User for login if not current user.
  -v, --verbose       Write more information.
  -V, --version       Output version information and exit.
  -E, --vertical      Print output vertically. Is similar to --relative, but
                      prints output vertically.
  -w, --wait[=#]      Wait and retry if connection is down.
  --connect-timeout=#
  --shutdown-timeout=#
  --plugin-dir=name   Directory for client-side plugins.
  --default-auth=name Default authentication client-side plugin to use.

Variables (--variable-name=value)
and boolean options {FALSE|TRUE}  Value (after reading options)
--------------------------------- ----------------------------------------
count                             0
debug-check                       FALSE
debug-info                        FALSE
force                             FALSE
compress                          FALSE
character-sets-dir                (No default value)
default-character-set             auto
host                              (No default value)
no-beep                           FALSE
port                              0
relative                          FALSE
socket                            (No default value)
```

```
sleep                           0
ssl                             FALSE
ssl-ca                          (No default value)
ssl-capath                      (No default value)
ssl-cert                        (No default value)
ssl-cipher                      (No default value)
ssl-key                         (No default value)
ssl-verify-server-cert          FALSE
user                            root
verbose                         FALSE
vertical                        FALSE
connect-timeout                 43200
shutdown-timeout                3600
plugin-dir                      (No default value)
default-auth                    (No default value)

Where command is a one or more of: (Commands may be shortened)
  create databasename     Create a new database
  debug                   Instruct server to write debug information to log
  drop databasename       Delete a database and all its tables
  extended-status         Gives an extended status message from the server
  flush-all-statistics    Flush all statistics tables
  flush-all-status        Flush status and statistics
  flush-client-statistics Flush client statistics
  flush-hosts             Flush all cached hosts
  flush-index-statistics  Flush index statistics
  flush-logs              Flush all logs
  flush-privileges        Reload grant tables (same as reload)
  flush-slow-log          Flush slow query log
  flush-status            Clear status variables
  flush-table-statistics  Clear table statistics
  flush-tables            Flush all tables
  flush-threads           Flush the thread cache
  flush-user-statistics   Flush user statistics
  kill id,id,...        Kill mysql threads
  password [new-password] Change old password to new-password in current
format
  old-password [new-password] Change old password to new-password in old
format
  ping                  Check if mysqld is alive
  processlist           Show list of active threads in server
  reload                Reload grant tables
  refresh               Flush all tables and close and open logfiles
  shutdown              Take server down
  status                Gives a short status message from the server
  start-slave           Start slave
  stop-slave            Stop slave
  variables             Prints variables available
  version               Get version info from server
```

So können wir z.B. auch die verwendete Version von **MariaDB** abfragen.

```
# mysqladmin version
```

```
mysqladmin  Ver 9.0 Distrib 5.5.41-MariaDB, for Linux on x86_64
Copyright (c) 2000, 2014, Oracle, MariaDB Corporation Ab and others.

Server version          5.5.41-MariaDB
Protocol version        10
Connection              Localhost via UNIX socket
UNIX socket             /var/lib/mysql/mysql.sock
Uptime:                 2 hours 33 min 19 sec

Threads: 1  Questions: 27  Slow queries: 0  Opens: 1  Flush tables: 2  Open
tables: 27  Queries per second avg: 0.002
```

---

mysql

Der Zugriff auf unseren MariaDB-Server wird in der Regel mit dem Werkzeug **mysql** vorgenommen. So kann man auch z.B. sehr leicht und einfach den Status unseres Datenbankservers abfragen.

```
 # mysql -h localhost -u root -p
```

```
 Enter password: dxiFHdig10JXyRAec74j7bcPdyVGX9I1BxcYcoFs
```

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 12
Server version: 5.5.41-MariaDB MariaDB Server

Copyright (c) 2000, 2014, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

MariaDB [(none)]>
```

Auch hier kann man nun den Status des Daemon anzeigen lassen. Hierzu verwenden wir den SQL-Befehl **status**, den wir mit einem Strichpunkt ; abschliessen.

```
MariaDB [(none)]> status;
--------------
mysql  Ver 15.1 Distrib 5.5.41-MariaDB, for Linux (x86_64) using readline
5.1
```

```
Connection id:          12
Current database:
Current user:           root@localhost
SSL:                    Not in use
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server:                 MariaDB
Server version:         5.5.41-MariaDB MariaDB Server
Protocol version:       10
Connection:             Localhost via UNIX socket
Server characterset:    latin1
Db     characterset:    latin1
Client characterset:    utf8
Conn.  characterset:    utf8
UNIX socket:            /var/lib/mysql/mysql.sock
Uptime:                 2 hours 38 min 10 sec

Threads: 1  Questions: 31  Slow queries: 0  Opens: 1  Flush tables: 2  Open
tables: 27  Queries per second avg: 0.003
--------------

MariaDB [(none)]>
```

Die Verbindung zum Datenbank-Daemon beenden wir mit dem Befehl **quit**.

```
MariaDB [(none)]> quit;
```

```
 Bye
```

Datenbank-Dump

Zur Sicherung unserer MariaDB-Tabellen legen wir uns ein kleines Script an, mit dessen Hilfe wir täglich eine Sicherung der kompletten Datenbank vornehmen können.

```
 # touch /root/bin/mariadb_fulldump
```

Damit das Script später auch nur vom User **root** gelesen und ausgeführt werden kann, setzen wir nun noch die entsprechenden Dateirechte.

```
 # chmod 700 /root/bin/mariadb_fulldump
```

Nun füllen wir noch unser Script.

```
 # vim /root/bin/mariadb_fulldump
```

```
#!/bin/bash

#########################################################################
######
# Script-Name : mysqldump.sh
#
# Description : Datenbank-Dump der kompletten (alle Tabellen) unserer
#
#               MariaDB nach /root/mysql/dumps
#
#               Drei Datensicherungen werden aufgehoben, ältere werden
gelöscht. #
#
#
#
#
#
#
#
# Last update : 03.02.2017
#
# Version     : 0.01
#
#########################################################################
######

#########################################################################
######
#                                   H I S T O R Y
#
#########################################################################
######
# Version     : 0.01
#
# Description : initial release
#
# ------------------------------------------------------------------------
---- #
# Version     : x.xx
#
# Description : <Description>
#
#########################################################################
######

# Source function library.
. /etc/init.d/functions

# Definition der systemindividuellen Variablen

# Script-Name.
SCRIPT_NAME='mariadb_fulldump'
```

```
# Backup-Verzeichnis.
DIR_TARGET='/root/mysql/dump'
DUMP_FILES="$DIR_TARGET/*.sql"

# Mail-Empfänger
MAIL_RECIPIENT='michael.r467@gmail.com'

# Status-Mail versenden? [J|N].
MAIL_STATUS='J'

# Datenbankdefinitionen
DB_HOST="127.0.0.1"
DB_USER="root"
DB_SECRET="immNI+32$cHU551n5Kn13gn1uS4W6HYu0SAJwH8W"

# Variablen
MYSQLDUMP_COMMAND=`command -v mysqldump`
TOUCH_COMMAND=`command -v touch`
RM_COMMAND=`command -v rm`
PROG_SENDMAIL=`command -v sendmail`
CAT_COMMAND=`command -v cat`
DATE_COMMAND=`command -v date`
MKDIR_COMMAND=`command -v mkdir`
FILE_NAME='/'$SCRIPT_NAME'.'`$DATE_COMMAND '+%Y-%m-%d-%H%M%S'`'.sql'
FILE_LOCK='/tmp/'$SCRIPT_NAME'.lock'
FILE_LOG='/var/log/'$SCRIPT_NAME'.log'
FILE_LAST_LOG='/tmp/'$SCRIPT_NAME'.log'
FILE_MAIL='/tmp/'$SCRIPT_NAME'.mail'
VAR_HOSTNAME=`uname -n`
VAR_SENDER='root@'$VAR_HOSTNAME
VAR_EMAILDATE=`$DATE_COMMAND '+%a, %d %b %Y %H:%M:%S (%Z)'`

# Functionen
function log() {
        echo $1
        echo `$DATE_COMMAND '+%Y/%m/%d %H:%M:%S'` " INFO:" $1
>>${FILE_LAST_LOG}
}

function movelog() {
        $CAT_COMMAND $FILE_LAST_LOG >> $FILE_LOG
        $RM_COMMAND -f $FILE_LAST_LOG
        $RM_COMMAND -f $FILE_LOCK
}

function sendmail() {
        case "$1" in
        'STATUS')
                MAIL_SUBJECT='Status execution '$SCRIPT_NAME' script.'
        ;;
        *)
```

```
                MAIL_SUBJECT='ERROR while execution '$SCRIPT_NAME' script
!!!'
        ;;
        esac

$CAT_COMMAND <<MAIL >$FILE_MAIL
Subject: $MAIL_SUBJECT
Date: $VAR_EMAILDATE
From: $VAR_SENDER
To: $MAIL_RECIPIENT

MAIL

$CAT_COMMAND $FILE_LAST_LOG >> $FILE_MAIL

$PROG_SENDMAIL -f $VAR_SENDER -t $MAIL_RECIPIENT < $FILE_MAIL

$RM_COMMAND -f $FILE_MAIL

}

# Main.
log ""
log "+-------------------------------------------------------------------
---------+"
log "| ....................... Start des MariaDB-Dumps
........................... |"
log "+-------------------------------------------------------------------
---------+"
log ""
log "Das Datenbank-Backupscript wurde mit folgenden Parametern aufgerufen:"
log ""
log "SCRIPT_NAME     : $SCRIPT_NAME"
log "ZIEL-VERZEICHNIS: $DIR_TARGET"
log "MAIL_EMPFÄNGER  : $MAIL_RECIPIENT"
log "MAIL_STATUS     : $MAIL_STATUS"
log ""

# Prüfung ob alle benötigten Programme und Befehle vorhanden sind.
if [ ! -s "$MYSQLDUMP_COMMAND" ]; then
        log "Prüfen, ob das Programm '$MYSQLDUMP_COMMAND' vorhanden
ist.................[FEHLER]"
        sendmail ERROR
        movelog
        exit 10
else
        log "Prüfen, ob das Programm '$MYSQLDUMP_COMMAND' vorhanden
ist.................[  OK  ]"
fi

if [ ! -s "$TOUCH_COMMAND" ]; then
```

```
        log "Prüfen, ob das Programm '$TOUCH_COMMAND' vorhanden
ist.......................[FEHLER]"
        sendmail ERROR
        movelog
        exit 11
else
        log "Prüfen, ob das Programm '$TOUCH_COMMAND' vorhanden
ist.......................[  OK  ]"
fi

if [ ! -s "$RM_COMMAND" ]; then
        log "Prüfen, ob das Programm '$RM_COMMAND' vorhanden
ist.........................[FEHLER]"
        sendmail ERROR
        movelog
        exit 12
else
        log "Prüfen, ob das Programm '$RM_COMMAND' vorhanden
ist.........................[  OK  ]"
fi

if [ ! -s "$CAT_COMMAND" ]; then
        log "Prüfen, ob das Programm '$CAT_COMMAND' vorhanden
ist.........................[FEHLER]"
        sendmail ERROR
        movelog
        exit 13
else
        log "Prüfen, ob das Programm '$CAT_COMMAND' vorhanden
ist.........................[  OK  ]"
fi

if [ ! -s "$DATE_COMMAND" ]; then
        log "Prüfen, ob das Programm '$DATE_COMMAND' vorhanden
ist.........................[FEHLER]"
        sendmail ERROR
        movelog
        exit 14
else
        log "Prüfen, ob das Programm '$DATE_COMMAND' vorhanden
ist.........................[  OK  ]"
fi

if [ ! -s "$MKDIR_COMMAND" ]; then
        log "Prüfen, ob das Programm '$MKDIR_COMMAND' vorhanden
ist.........................[FEHLER]"
        sendmail ERROR
        movelog
        exit 15
else
        log "Prüfen, ob das Programm '$MKDIR_COMMAND' vorhanden
```

```
ist.......................[  OK  ]"
fi

if [ ! -s "$PROG_SENDMAIL" ]; then
        log "Prüfen, ob das Programm '$PROG_SENDMAIL' vorhanden
ist.................[FEHLER]"
        sendmail ERROR
        movelog
        exit 16
else
        log "Prüfen, ob das Programm '$PROG_SENDMAIL' vorhanden
ist.................[  OK  ]"
fi

if [ ! -e "$FILE_LOCK" ]; then
        log "Prüfen, ob das Programm nicht bereits oder noch
läuft.....................[  OK  ]"

        $TOUCH_COMMAND $FILE_LOCK
else
        log "Prüfen, ob das Programm nicht bereits oder noch
läuft.....................[FEHLER]"
        log ""
        log "FEHLER: Das Script läuft bereits bzw. immer noch, oder die
LOCK-Datei"
        log "existiert noch von einem früheren Programmaufruf!"
        log ""
        sendmail ERROR
        movelog
        exit 20
fi

if [ ! -d "$DIR_TARGET" ]; then
        log "Prüfen, ob Zielverzeichnis
existiert.....................................[FEHLER]"
        log ""
        log " INFO: Erstelle Zielverzeichnis!"
        log " INFO: --> "$DIR_TARGET
        log ""

        $MKDIR_COMMAND -p $DIR_TARGET
else
        log "Prüfen, ob Zielverzeichnis
existiert.....................................[  OK  ]"
fi

if [ "$UID" -ne 0 ]; then
        log "Prüfen, ob das Script mit root-Rechten gestartet
wurde.....................[FEHLER]"
        log ""
        sendmail ERROR
```

```
        movelog
        exit 21
else
        log "Prüfen, ob das Script mit root-Rechten gestartet
wurde.....................[  OK  ]"
fi

# Start dumping.
log ""
log "+-------------------------------------------------------------------
---------+"
log "| ..................... Start des Datenbank-Dumps
.............................. |"
log "+-------------------------------------------------------------------
---------+"
log ""

log "$MYSQLDUMP_COMMAND -h "$DB_HOST" -u "$DB_USER" --all-databases --events
> $DIR_TARGET$FILE_NAME"

$MYSQLDUMP_COMMAND -h $DB_HOST -u $DB_USER --password=$DB_SECRET --all-
databases --events > $DIR_TARGET$FILE_NAME

if [ "$?" != 0 ]; then
        log ""
        $RM_COMMAND -f $FILE_LOCK
        sendmail ERROR
        movelog
        exit 99
else
        log ""
        log "+-------------------------------------------------------------
-----------------+"
        log "| ......................... Datenbank-Dump beendet
............................. |"
        log "+-------------------------------------------------------------
-----------------+"
        log ""
fi

# Bis auf die letzten drei Datenbankbackups alle anderen Dateien löschen.
cd $DIR_TARGET/
(ls $DUMP_FILES -t|head -n 3;ls $DUMP_FILES )|sort|uniq -u|xargs rm
if [ "$?" != "0" ]; then
        log "alte Datenbanksicherungen aus Zielverzeichnis $DIR_TARGET
gelöscht....[FEHLER]"
        log ""
        sendmail ERROR
        movelog
        exit 69
else
```

```
        log "alte Datenbanksicherungen aus Zielverzeichnis $DIR_TARGET
gelöscht....[  OK  ]"
        log ""
fi

# Finish syncing.
log "+-------------------------------------------------------------------
---------+"
log "| ......................... Ende des MariaDB-Dumps
........................... |"
log "+-------------------------------------------------------------------
---------+"
log ""

# Status eMail versenden
if [ $MAIL_STATUS = 'J' ]; then
        sendmail STATUS
fi

# Temporäres Logfile permanent sichern
movelog

exit 0
```

Anschliessend machen wir noch einen Eintrag in der **/etc/crontab** , damit das Script auch täglich selbständig um 03:20 Uhr lauft.

```
# vim /etc/crontab
```

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin

# For details see man 4 crontabs

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR
sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed

# rebermi: 2017-02-03 täglicher MariaDB-Datenbankdump
20  3  *  *  root /root/bin/mariadb_fulldump 1>/dev/null 2>&1
```

phpMyAdmin

Zur komfortablen Administration unserer **MariaDB unter RHEL 7 und CentOS 7** kann man auf das PHP-Projekt phpMyAdmin zurück greifen.

Im Kapitel **phpMyAdmin für Redhat** ist die Installation und Konfiguration des PHP Projektes für RHEL7 beschrieben.

Last update: **2019/03/08 13:22**