w: www.mariadb.com

e: info@mariadb.com

Getting Started With MariaDB MaxScale

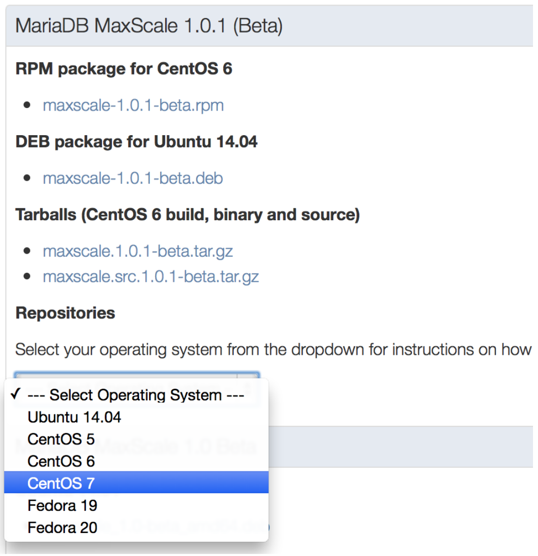
First Steps With MaxScale

In this introduction to MaxScale the aim is to take the reader from the point of installation to making the decision as to which of the various setups that are possible with MaxScale should be the initial configuration to use. One of the problems that new users to MaxScale suffer is deciding exactly what they should consider as a base configuration to start exploring what MaxScale is capable of. MaxScale is highly configurable, with new plugins expanding the capabilities of MaxScale, whilst this makes it a very adaptable tool it does lead to an initial hurdle in configuring MaxScale.

Installation

The simplest way to install MaxScale is to use one of the binary packages that are available for download from the MariaDB website.

- Simply go to www.mariadb.com
- Select the Downloads option from the Resources menu
- Find and click on the button "Download MariaDB MaxScale Binaries"
- Find the section on that page entitled MariaDB MaxScale
- Select your operating system from the drop down box



MariaDB MaxScale 1.0.1 (Beta)

RPM package for CentOS 6

- maxscale-1.0.1-beta.rpm

DEB package for Ubuntu 14.04

- maxscale-1.0.1-beta.deb

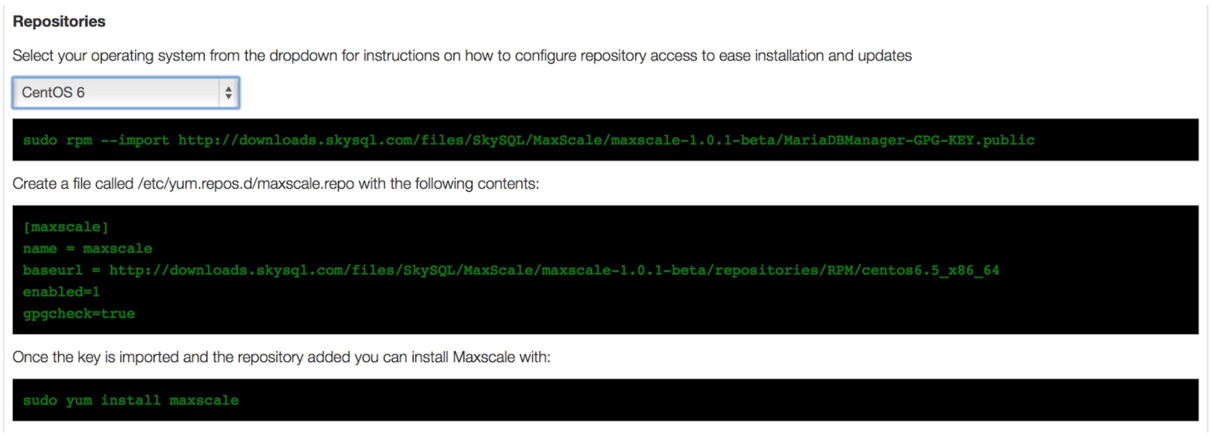
Tarballs (CentOS 6 build, binary and source)

- maxscale.1.0.1-beta.tar.gz
- maxscale.src.1.0.1-beta.tar.gz

Repositories

Select your operating system from the dropdown for instructions on how

- Instructions that are specific for your operating system will then appear



Repositories

Select your operating system from the dropdown for instructions on how to configure repository access to ease installation and updates

CentOS 6

```
sudo rpm --import http://downloads.skysql.com/files/SkySQL/MaxScale/maxscale-1.0.1-beta/MariaDBManager-GPG-KEY.public
```

Create a file called /etc/yum.repos.d/maxscale.repo with the following contents:

```
[maxscale]
name = maxscale
baseurl = http://downloads.skysql.com/files/SkySQL/MaxScale/maxscale-1.0.1-beta/repositories/RPM/centos6.5_x86_64
enabled=1
gpgcheck=true
```

Once the key is imported and the repository added you can install Maxscale with:

```
sudo yum install maxscale
```

- Follow these instructions to install MaxScale on your machine

Upon successful completion of the installation process you have a version of MaxScale that is missing only a configuration file before it can be started.

Building MaxScale From Source Code

Alternatively you may download the MaxScale source and build your own binaries. You will need a number of tools and libraries in order to achieve this.

- cmake version 2.8.12 or later
- gcc recommended version 4.4.7 or later
- libaio
- MariaDB Develop libraries version 5.5.38 or later
- libedit 2.11 or later (used by MaxAdmin tool)

First clone the GitHub project to your machine either via the web interface, your favorite graphical interface or the git command line

```
$ git clone https://github.com/mariadb-corporation/MaxScale
Cloning into 'MaxScale'...
remote: Counting objects: 16228, done.
...
```

Change directory to the MaxScale directory, create a build directory and change directory to that build directory

```
$ cd MaxScale
$ mkdir build
$ cd build
```

The next step is to run the cmake command to build the Makefile you need to compile Maxscale. There are a number of options you may give to configure cmake and point it to the various packages it requires. These are documented in the MaxScale README file, in this example we will assume the MariaDB developer packages have been installed in a non-standard location and set all the options required to locate these, along with options to build the unit tests and configure the installation target directory.

```
$ cmake -DMYSQL_DIR=~/usr/include/mysql \
-DEMBEDDED_LIB=~/usr/lib64/libmysqld.a \
-DMYSQLCLIENT_LIBRARIES=~/usr/lib64/libmysqlclient.so \
-DERRMSG=~/usr/share/mysql/english/errmsg.sys \
-DINSTALL_DIR=/usr/local/maxscale -DBUILD_TESTS=Y \
-DINSTALL_SYSTEM_FILES=N ../MaxScale
-- CMake version: 2.8.12.2
-- The C compiler identification is GNU 4.4.7
-- The CXX compiler identification is GNU 4.4.7
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Library was found at: /lib64/libaio.so
-- Library was found at: /usr/lib64/libssl.so
-- Library was found at: /usr/lib64/libcrypt.so
-- Library was found at: /usr/lib64/libcrypto.so
-- Library was found at: /usr/lib64/libz.so
-- Library was found at: /usr/lib64/libm.so
-- Library was found at: /usr/lib64/libdl.so
-- Library was found at: /usr/lib64/librt.so
-- Library was found at: /usr/lib64/libpthread.so
-- Using errmsg.sys found at: /home/maxscale/usr/share/mysql/english/
errmsg.sys
-- Using embedded library: /home/mpinto/usr/lib64/libmysqld.a
-- Valgrind found: /usr/bin/valgrind
-- Found dynamic MySQL client library: /home/maxscale/usr/lib64/
libmysqlclient.so
```

```
-- Found static MySQL client library: /usr/lib/libmysqlclient.a
-- C Compiler supports: -Werror=format-security
-- Linking against: /home/mpinto/usr/lib64/libmysqlclient.so
-- Installing MaxScale to: /usr/local/maxscale/
-- Generating RPM packages
-- Found Doxygen: /usr/bin/doxygen (found version "1.6.1")
-- Configuring done
-- Generating done
-- Build files have been written to: /home/maxscale/develop/build
-bash-4.1$ make depend
-bash-4.1$ make
```

Once the cmake command is complete simply run make to build the MaxScale binaries.

```
$ make
Scanning dependencies of target utils
[  1%] Building CXX object utils/CMakeFiles/utils.dir/skygw_utils.cc.o
Linking CXX static library libutils.a
[  1%] Built target utils
Scanning dependencies of target log_manager
[  2%] Building CXX object log_manager/CMakeFiles/log_manager.dir/
log_manager.cc.o
...
```

After the completion of the make process the installation can be achieved by running the make install target.

```
$ make install
...
```

This will result in an installation being created which is identical to that which would be achieved by installing the binary package.

Configuring MaxScale

The first step in configuring your MaxScale is to determine what it is you want to achieve with your MaxScale and what environment it will run in. The later is probably the easiest starting point for choosing which configuration route you wish to take. There are two distinct database environments which the first GA release of MaxScale supports; MySQL Master/Slave Replication clusters and Galera Cluster.

Master/Slave Replication Clusters

There are two major configuration options available to use MaxScale with a MySQL Replication cluster; connection routing with separate read and write connections, or read/write splitting with a single connection. A separate tutorial is available for each of these configurations that describes how to build the configuration file for MaxScale that will work with your environment.

Using a MySQL Master/Slave Replication cluster will provide one node server within the cluster that is the master server and the remainder of the servers will be slaves. The slaves are read replicas of the master. In a replication cluster like this all write operations must be performed on the master. This can provide not just added security of your data, but also read scalability. In an application environment with a substantial proportions of read operations, directing those read operations to the slave servers can increase the total load which the system can handle by offloading the master server from the burden of these read operations.

Making the choice between these two setups is relatively simple, if you have an application that understands that there are some database servers that it can only read from and one it must send all of the writes to, then the connection routing option can be used. Applications that are not written to separate read and write statements must use a service within MaxScale that will split the incoming stream of SQL statements into operations that can be executed on the master and those that can be set to the slave. These applications should use the statement based routing provided by the Read/Write Splitter router.

Galera Cluster

A Galera Cluster provides a true multi-master cluster option for MariaDB and MySQL database environments. In such a setup any node that is part of the cluster can be used to both execute read and write operations. MaxScale again offers two different configurations that can be used with Galera; a connection balancing configuration or a statement splitting mechanism that can be used to isolate write operations to a single node within the cluster. Again there is a tutorial guide available for both of these major configurations.

The connection based load balancing configuration is used in an environment in which you have a cluster that you want to be available to an application without the application needing to be aware of the cluster configuration or state of the database nodes. MaxScale will monitor the nodes within the database cluster and will route connections from the application to database nodes that are active members of the cluster. MaxScale will also keep track of the number of connections to each database node keep equal numbers of connections to each node, at the time the connection is established.

It is also possible to use the Read/Write Splitter with Galera. Although it is not necessary to segregate the write operations to a single node, there are advantages in doing this if you have an application where the write load is not too great to be handled by a single node in the cluster. Galera Cluster uses an optimistic locking strategy that will allow transactions to progress independently on each node within the cluster. It is only when the transaction commits that the transaction is checked for conflicts with other transactions that are committing on the other nodes. At this stage the commit can fail with a deadlock detection error. This can be inconvenient for applications and, some older applications, that are not aware that the transaction can fail at this stage may not check for this failure. Using the Read/Write Splitter will allow this to be avoided since it will isolate the write to one node and no deadlock detection will occur. MaxScale provides a monitoring module that will maintain pseudo states of master and slave for the Galera cluster that allows for this type of configuration.

Other MaxScale Configuration

As well as the four major configuration choices outlined above there are also other configurations sub-options that may be mixed with those to provide a variety of different configuration and functionality. The MaxScale filter concept allows the basic configurations to be built upon in a large variety of ways. A separate filter tutorial is available that discusses the concept and gives some examples of ways to use filters.

Administration Of MaxScale

There are various administration tasks that may be done with MaxScale, a client command, maxadmin, is available that will interact with a running MaxScale and allow the status of MaxScale to be monitored and give some control of the MaxScale functionality. There is a separate reference guide for the maxadmin utility and also a short administration tutorial that covers the common administration tasks that need to be done with MaxScale.