MaxScale

Configuration & Usage Scenarios

Mark Riddoch

Last Updated: 8th May 2014

Contents

Document History

Date	Change	Who
21st July 2013	Initial version	Mark Riddoch
23rd July 2013	Addition of default user and password for a monitor and discussion of monitor user requirements New monitor documented for Galera clusters Addition of example Galera cluster configuration	Mark Riddoch
13th November 2013	state for Galera Monitor is “synced”	Massimiliano Pinto
2nd December 2013	Updated the description of the command line arguments to match the code updates. Improved descriptions and general documentation. Enhanced example configurations	Mark Riddoch
6th February 2014	Added “enable_root_user” as a service parameter	Massimiliano Pinto
7th February 2014	Addition of bind address information Clarification of user configuration required for monitoring users and the user needed to fetch the user data	Mark Riddoch
3rd March 2014	MySQL authentication with hostnames	Massimiliano Pinto
3rd March 2014	Addition of section that describes authentication requirements and the rules for creating user credentials	Mark Riddoch
28th March 2014	Unix socket support	Massimiliano Pinto
8th May 2014	Added “version_string” parameter in service	Massimiliano Pinto

Introduction

The purpose of this document is to describe how to configure MaxScale and to discuss some possible usage scenarios for MaxScale. MaxScale is designed with flexibility in mind, and consists of an event processing core with various support functions and plugin modules that tailor the behaviour of the MaxScale itself.

Terms

Term	Description
service	A service represents a set of databases with a specific access mechanism that is offered to clients of MaxScale. The access mechanism defines the algorithm that MaxScale will use to direct particular requests to the individual databases.
server	A server represents an individual database server to which a client can be connected via MaxScale.
router	A router is a module within MaxScale that will route client requests to the various database servers which MaxScale provides a service interface to.
connection routing	Connection routing is a method of handling requests in which MaxScale will accept connections from a client and route data on that connection to a single database using a single connection. Connection based routing will not examine individual quests on a connection and it will not move that connection once it is established.
statement routing	Statement routing is a method of handling requests in which each request within a connection will be handled individually. Requests may be sent to one or more servers and connections may be dynamically added or removed from the session.
protocol	A protocol is a module of software that is used to communicate with another software entity within the system. MaxScale supports the dynamic loading of protocol modules to allow for increased flexibility.
module	A module is a separate code entity that may be loaded dynamically into MaxScale to increase the available functionality. Modules are implemented as run-time loadable shared objects.
monitor	A monitor is a module that can be executed within MaxScale to

	monitor the state of a set of database. The use of an internal monitor is optional, monitoring may be performed externally to MaxScale.
listener	A listener is the network endpoint that is used to listen for connections to MaxScale from the client applications. A listener is associated to a single service, however a service may have many listeners.
connection failover	When a connection currently being used between MaxScale and the database server fails a replacement will be automatically created to another server by MaxScale without client intervention
backend database	A term used to refer to a database that sits behind MaxScale and is accessed by applications via MaxScale.

Configuration

The MaxScale configuration is read from a file which can be located in a number of placing, MaxScale will search for the configuration file in a number of locations.

1. If the environment variable MAXSCALE_HOME is set then MaxScale will look for a configuration file called MaxScale.cnf in the directory $MAXSCALE_HOME/etc
2. If MAXSCALE_HOME is not set or the configuration file is not in the location above MaxScale will look for a file in /etc/MaxScale.cnf

Alternatively MaxScale can be started with the -c flag and the path of the MaxScale home directory tree.

An explicit path to a configuration file can be passed by using the -f option to MaxScale.

The configuration file itself is based on the “ini” file format and consists of various sections that are used to build the configuration, these sections define services, servers, listeners, monitors and global settings.

Global Settings

The global settings, in a section named [MaxScale], allow various parameters that affect MaxScale as a whole to be tuned. Currently the only setting that is supported is the number of threads to use to handle the network traffic. MaxScale will also accept the section name of [gateway] for global settings. This is for backward compatibility with versions prior to the naming of MaxScale.

Threads

To control the number of threads that poll for network traffic set the parameter threads to a number. It is recommended that you start with a single thread and add more as you find the performance is not satisfactory. MaxScale is implemented to be very thread efficient, so a small number of threads is usually adequate to support reasonably heavy workloads. Adding more threads may not improve performance and can consume resources needlessly.

```
# Valid options are:
#       threads=<number of epoll threads>
[MaxScale]
threads=1
```

It should be noted that additional threads will be created to execute other internal services within MaxScale, this setting is merely used to configure the number of threads that will be used to manage the user connections.

Service

A service represents the database service that MaxScale offers to the clients. In general a service consists of a set of backend database servers and a routing algorithm that determines how MaxScale decides to send statements or route connections to those backend servers.

A service may be considered as a virtual database server that MaxScale makes available to its clients.

Several different services may be defined using the same set of backend servers. For example a connection based routing service might be used by clients that already performed internal read/write splitting, whilst a different statement based router may be used by clients that are not written with this functionality in place. Both sets of applications could access the same data in the same databases.

A service is identified by a service name, which is the name of the configuration file section and a type parameter of service

```
[Test Service]
type=service
```

In order for MaxScale to forward any requests it must have at least one service defined within the configuration file. The definition of a service alone is not enough to allow MaxScale to forward requests however, the service is merely present to link together the other configuration elements.

Router

The router parameter of a service defines the name of the router module that will be used to implement the routing algorithm between the client of MaxScale and the backend databases. Additionally routers may also be passed a comma separated list of options that are used to control the behaviour of the routing algorithm. The two parameters that control the routing choice are `router` and `router_options`. The router options are specific to a particular router and are used to modify the behaviour of the router. The read connection router can be passed options of master, slave or synced, an example of configuring a service to use this router and limiting the choice of servers to those in slave state would be as follows.

```
router=readconnroute
router_options=slave
```

To change the router to connect on to servers in the master state as well as slave servers the router options can be modified to include the master state.

```
router=radconnroute
```

```
	router_options=master,slave
```

A more complete description of router options and what is available for a given router is included with the documentation of the router itself.

Servers

The servers parameter in a service definition provides a comma separated list of the backend servers that comprise the service. The server names are those used in the name section of a block with a type parameter of server (see below).

```
	servers=server1,server2,server3
```

User

The `user` parameter, along with the `passwd` parameter are used to define the credentials used to connect to the backend servers to extract the list of database users from the backend database that is used for the client authentication.

```
	user=maxscale
	passwd=maxpassword
```

Authentication of incoming connections is performed by MaxScale itself rather than by the database server to which the client is connected. The client will authenticate itself with MaxScale, using the username, hostname and password information that MaxScale has extracted from the backend database servers. A detailed discussion of how this impacts the authentication process please see the “Authentication” section below.

The host matching criteria is restricted to IPv4, IPv6 will be added in a future release.

Existing user configuration in the backend databases must be checked and may be updated before successful MaxScale authentication:

In order for MaxScale to obtain all the data it must be given a username it can use to connect to the database and retrieve that data. This is the parameter that gives MaxScale the username to use for this purpose.

The account used must be able to select from the mysql.user table, the following is an example showing how to create this user.

```
	MariaDB [mysql]> create user 'maxscale'@'maxscalehost'
	identified by 'Mhu87p2D';
	Query OK, 0 rows affected (0.01 sec)
```

```
MariaDB [mysql]> grant SELECT on mysql.user to
'maxscalemon'@'maxscalehost';
Query OK, 0 rows affected (0.00 sec)
```

Passwd

The auth parameter provides the password information for the above user and may be either a plain text password or it may be an encrypted password. See the section on encrypting passwords for use in the MaxScale.cnf file. This user must be capable of connecting to the backend database and executing the SQL statement “SELECT user, host, password FROM mysql.user”.

enable_root_user

This parameter controls the ability of the root user to connect to MaxScale and hence onwards to the backend servers via MaxScale.

The default value is 0, disabling the ability of the root user to connect to MaxScale.

Example for enabling root user:

```
enable_root_user=1
```

version_string

This parameter sets a custom version string that is sent in the MySQL Handshake from MaxScale to clients.

Example:

```
version_string=5.5.37-MariaDB-RWsplit
```

If not set the default value is the server version of the embedded MySQL/MariaDB library.
Example: `5.5.35-MariaDB`

Server

Server sections are used to define the backend database servers that can be formed into a service. A server may be a member of one or more services within MaxScale. Servers are identified by a server name which is the section name in the configuration file. Servers have a type parameter of server, plus address port and protocol parameters.

```
[server1]
```

```
type=server
address=127.0.0.1
port=3000
protocol=MySQLBackend
```

Address

The IP address or hostname of the machine running the database server that is being defined. MaxScale will use this address to connect to the backend database server.

Port

The port on which the database listens for incoming connections. MaxScale will use this port to connect to the database server.

Protocol

The name for the protocol module to use to connect MaxScale to the database. Currently only one backend protocol is supported, the MySQLBackend module.

Monitoruser

The monitor has a username and password that is used to connect to all servers for monitoring purposes, this may be overridden by supplying a monitoruser statement for each individual server

```
monitoruser=mymonitoruser
```

MonitorPw

The monitor has a username and password that is used to connect to all servers for monitoring purposes, this may be overridden by supplying a monpasswd statement for the individual servers

```
monitorpw=mymonitorpasswd
```

The monpasswd parameter may be either a plain text password or it may be an encrypted password. See the section on encrypting passwords for use in the MaxScale.cnf file.

Listener

The listener defines a port and protocol pair that is used to listen for connections to a service. A service may have multiple listeners associated with it, either to support multiple protocols or multiple ports. As with other elements of the configuration the section name is the listener name and a type parameter is used to identify the section as a listener definition.

```
[Test Listener]
type=listener
service=Test Service
protocol=MySQLClient
```

SkySQL

MaxScale

Configuration & Usage Scenarios

Mark Riddoch

Last Updated: 8th May 2014