MySQL Router

REST API

Frédéric Descamps
Community Manager
MySQL
February 2021
Who am I ?

about.me/lefred
Frédéric Descamps

- @lefred
- MySQL Evangelist
- Managing MySQL since 3.20
- devops believer
- living in Belgium 🇧🇪
- https://lefred.be
What is it?

MySQL Router
MySQL Router

MySQL Router is a building block for high availability (HA) solutions. It simplifies application development by intelligently routing connections to MySQL servers for increased performance and reliability.

MySQL Router is part of MySQL InnoDB Cluster and MySQL InnoDB ReplicaSet.

MySQL Router is written in C++ and is part of MySQL's trunk.
MySQL Router

REST API
MySQL Router REST API - Why?

When a problem occurs, it's not always obvious to understand why? For example if for a given route, the amount of max_connections is reached, it's important to know it and to know that value before we reach it.

MySQL Router exposes data (statistics, settings, ...) as REST endpoints via HTTP methods as JSON payload.

This is explained in WL#8965.
MySQL Router REST API - How?

Since MySQL 8.0.16, MySQL Router has the possibility to launch an internal http server. At that time it could only serve static files.

Then, in MySQL 8.0.17 we added the REST API to MySQL Router.

With MySQL 8.0.20 the authentication credentials to access the REST API could also be stored on MySQL in a metadata table, before it was only using a file. See WL#12952.

And finally with MySQL 8.0.22, bootstrapping a MySQL Router also configures the REST API functionality into the generated mysqlrouter.conf configuration file.
Test it and get the paths

To query `swagger.json` no authentication is required:

```
[fred@imac ~] $ curl -s -k https://example.com:8443/api/20190715/swagger.json | jq '.paths | keys'
```

```
["/metadata",
 "/metadata/{metadataName}/config",
 "/metadata/{metadataName}/status",
 "/router/status",
 "/routes",
 "/routes/{routeName}/blockedHosts",
 "/routes/{routeName}/config",
 "/routes/{routeName}/connections",
 "/routes/{routeName}/destinations",
 "/routes/{routeName}/health",
 "/routes/{routeName}/status"
]
```
MySQL Router REST API - Authentication

MySQL Router uses realms for authentication. The backend can be a file or a record in a metadata table:

```
[http_auth_backend:default_auth_backend]
backend=metadata_cache
```

If you use a file, the `mysqlrouter_passwd` command-line utility must be used to generate and manage the users.
MySQL Router REST API - Authentication (metadata)

So there is a table in the metadata that we can use to connect to the REST API. Let's have a look...
So there is a table in the metadata that we can use to connect to the REST API. Let's have a look...

```
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Null</th>
<th>Key</th>
<th>Default</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster_id</td>
<td>char(36)</td>
<td>NO</td>
<td>PRI</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>user</td>
<td>varchar(256)</td>
<td>NO</td>
<td>PRI</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>authentication_method</td>
<td>varchar(64)</td>
<td>NO</td>
<td></td>
<td></td>
<td>modular_crypt_format</td>
</tr>
<tr>
<td>authentication_string</td>
<td>text</td>
<td>YES</td>
<td></td>
<td></td>
<td>NULL</td>
</tr>
<tr>
<td>description</td>
<td>varchar(255)</td>
<td>YES</td>
<td></td>
<td></td>
<td>NULL</td>
</tr>
<tr>
<td>privileges</td>
<td>json</td>
<td>YES</td>
<td></td>
<td></td>
<td>NULL</td>
</tr>
<tr>
<td>attributes</td>
<td>json</td>
<td>YES</td>
<td></td>
<td></td>
<td>NULL</td>
</tr>
</tbody>
</table>

7 rows in set (0.0017 sec)
MySQL Router REST API - Authentication (metadata)

So there is a table in the metadata that we can use to connect to the REST API. Let’s have a look...

```
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Null</th>
<th>Key</th>
<th>Default</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster_id</td>
<td>char(36)</td>
<td>NO</td>
<td>PRI</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>user</td>
<td>varchar(256)</td>
<td>NO</td>
<td>PRI</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>authentication_method</td>
<td>varchar(64)</td>
<td>NO</td>
<td></td>
<td>modular_crypt_format</td>
<td>NULL</td>
</tr>
<tr>
<td>authentication_string</td>
<td>text</td>
<td>YES</td>
<td></td>
<td></td>
<td>NULL</td>
</tr>
<tr>
<td>description</td>
<td>varchar(255)</td>
<td>YES</td>
<td></td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>privileges</td>
<td>json</td>
<td>YES</td>
<td></td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>attributes</td>
<td>json</td>
<td>YES</td>
<td></td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>
```

7 rows in set (0.0017 sec)

So when we bootstrap a Router against a MySQL InnoDB Cluster or a MySQL InnoDB ReplicaSet, we need to add a user in this table.
My journey to add a user in `router_rest_accounts`

Let me share with you my journey to insert a user into that table...
My journey to add a user in `router_rest_accounts`

Let me share with you my journey to insert a user into that table...
Adding a user in router_rest_accounts - Attempt #1

Let's try:

```sql
INSERT INTO router_rest_accounts VALUES (  
(SELECT cluster_id FROM mysql_innodb_cluster_metadata.v2_clusters LIMIT 1),  
"lefred", "modular_crypt_format", "fosdem", NULL, NULL, NULL);
```
Adding a user in `router_rest_accounts` - Attempt #1

Let's try:

```sql
INSERT INTO router_rest_accounts VALUES (
(SELECT cluster_id FROM mysql_innodb_cluster_metadata.v2_clusters LIMIT 1),
"lefred", "modular_crypt_format", "fosdem", NULL, NULL, NULL);
```

```bash
[fred@imac ~]$ curl -s -k -u lefred:fosdem https://:8443/api/20190715/routes | jq
```

Copyright @ 2021 Oracle and/or its affiliates.
Adding a user in `router_rest_accounts` - Attempt #1

Let's try:

```sql
INSERT INTO router_rest_accounts VALUES (
  (SELECT cluster_id FROM mysql_innodb_cluster_metadata.v2_clusters LIMIT 1),
  "lefred", "modular_crypt_format", "fosdem", NULL, NULL, NULL);
```

```
[fred@imac ~] $ curl -s -k -u lefred:fosdem https://...:8443/api/20190715/routes | jq
[fred@imac ~] $
```
lefred - security : 0 - 1
Adding a user in *router_rest_accounts* - Attempt #2

For my second attempt, I asked to the router dev team for an example of string I could use.

The reply was simple 'copy' the value of your `authentication_string` column in the `mysql.user`:

```sql
select user, plugin, authentication_string from mysql.user where user='clusteradmin';
```

<table>
<thead>
<tr>
<th>user</th>
<th>plugin</th>
<th>authentication_string</th>
</tr>
</thead>
<tbody>
<tr>
<td>clusteradmin</td>
<td>caching_sha2_password</td>
<td>$6$A$005$ct/_qP0Y7gsp&quot;KZGfT0g/gUuQRfRbR1w59C//wG01xDWypNaEjYT62</td>
</tr>
</tbody>
</table>
```
Adding a user in `router_rest_accounts` - Attempt #2

For my second attempt, I asked to the router dev team for an example of string I could use.

The reply was simple 'copy' the value of your `authentication_string` column in the `mysql.user`:

```sql
SELECT user, plugin, authentication_string FROM mysql.user WHERE user='clusteradmin';
```

```
<table>
<thead>
<tr>
<th>user</th>
<th>plugin</th>
<th>authentication_string</th>
</tr>
</thead>
<tbody>
<tr>
<td>clusteradmin</td>
<td>caching_sha2_password</td>
<td>$A$005$ct/_q0P0Y7gsp&quot;KZGft0g/guQTNURfbRT1w59C/\VGo1xDwYpNaEjYT62</td>
</tr>
</tbody>
</table>
```

```
REPLACE INTO router_rest_accounts VALUES (  (SELECT cluster_id FROM mysql_innodb_cluster_metadata.v2_clusters LIMIT 1),  "lefred", "modular_crypt_format",  (SELECT authentication_string FROM mysql.user WHERE user='clusteradmin'),  NULL, NULL, NULL);
```
Adding a user in *router_rest_accounts* - Attempt #2

Let's test it:

```
[fred@imac ~] $ curl -s -k -u lefred:password https://example.com:8443/api/20190715/routes | jq '.items'
```

```
[ 
  {"name": "mycluster_ro"},
  {"name": "mycluster_rw"},
  {"name": "mycluster_x_ro"},
  {"name": "mycluster_x_rw"}
]
```
Adding a user in `router_rest_accounts` - Attempt #2

Let's test it:

```bash
[fred@imac ~] $ curl -s -k -u lefred:password https://[host]/:8443/api/20190715/routes | jq '.items'
```

```
[  
  {  
    "name": "mycluster_ro"  
  },  
  {  
    "name": "mycluster_rw"  
  },  
  {  
    "name": "mycluster_x_ro"  
  },  
  {  
    "name": "mycluster_x_rw"  
  }
]
```

It works... but I don't want to use a MySQL user to monitor Router!
Adding a user in *router_rest_accounts* - Attempt #2

Let's test it:

```bash
$ curl -s -k -u lefred:password https://[redacted]:8443/api/20190715/routes | jq '.items'
```

```
[{
  "name": "mycluster_ro"
},
  "name": "mycluster_rw"
},
  "name": "mycluster_x_ro"
},
  "name": "mycluster_x_rw"
}
```

It works... but I don't want to use a MySQL user to monitor Router!

It's important to provide accurate information and context when you ask something...
lefred - developer : 0 - 1
Adding a user in *router_rest_accounts* - Attempt #3

My goal was to manage the credential for the REST API using a MySQL Shell Plugin...

I've been said that I should use the same string as a MySQL user... let's generate it!
Adding a user in *router_rest_accounts* - Attempt #3

My goal was to manage the credential for the REST API using a MySQL Shell Plugin...

I've been said that I should use the same string as a MySQL user... let's generate it!

I created a program that does it:

```
[fred@imac ~/workspace/genauth_string] (main) $ ./myauth fosdem
MySQL caching_sha2_password authentication_string generator

The authentication string to use is:
$A$005$zwM4!A72%s](-ed8FN-euE7cwRygTogySEB0TQY46UbIU483McwXJYioyTbML2
```
Adding a user in `router_rest_accounts` - Attempt #3

My goal was to manage the credential for the REST API using a MySQL Shell Plugin...

I've been said that I should use the same string as a MySQL user... let's generate it!

I created a program that does it:

```bash
[fred@imac ~/workspace/genauth_string] (main) $ ./myauth fosdem
MySQL caching_sha2_password authentication_string generator
The authentication string to use is:
$A$005$zwM4!A72%s](-ed8FN-euE7cwRygTogySEBOTQY46UbIU483McwXJYiovTbML2
```

```sql
REPLACE INTO router_rest_accounts VALUES (  
(SELECT cluster_id FROM mysql_innodb_cluster_metadata.v2_clusters LIMIT 1),  
"lefred", "modular_crypt_format",  
'$A$005$zwM4!A72%s](-ed8FN-euE7cwRygTogySEBOTQY46UbIU483McwXJYiovTbML2',  
NULL, NULL, NULL);
```
Adding a user in `router_rest_accounts` - Attempt #3

Let's test it:

```bash
[fred@lmac ~] $ curl -s -k -u lefred:fosdem https://[[[MAC Nh 21] 8:443/api/20190715/router/status | jq

```
Adding a user in `router_rest_accounts` - Attempt #3

Let's test it:

```
[fred@lmac ~] $ curl -s -k -u lefred:fosdem https://10.1.1.1:8443/api/20190715/router/status | jq
{
  "processId": 17446,
  "productEdition": "MySQL Community - GPL",
  "timeStarted": "2021-01-07T13:37:53.982719Z",
  "version": "8.0.22",
  "hostname": "single-mysql"
}
```

It works... but I don't want to use an external program in my plugin, that's not very portable.
lefred - usability : 0 - 1
Adding a user in router_rest_accounts - Attempt #4

Mmm... let's try something else...
Adding a user in `router_rest_accounts` - Attempt #4

Mmm... let's try something else...

A component!
Adding a user in *router_rest_accounts* - Attempt #4

Mmm... let's try something else...

A component!

```sql
SELECT generate_auth_string('fosdem2', 1);
```

<table>
<thead>
<tr>
<th>password</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
Adding a user in `router_rest_accounts` - Attempt #4

Mmm... let's try something else...

A component!

```
SELECT generate_auth_string('fodsem2', 1);
```

```
REPLACE INTO router_rest_accounts VALUES
  (SELECT cluster_id FROM mysql_innodb_cluster_metadata.v2_clusters LIMIT 1),
   "lefred", "modular_crypt_format",
   "$A$005$op^'U#}b)!n*~3E<{+Pzrz8/Rg9jU0.b4iMY56vET9E3rt2bws/BBw43zpTW8",
    NULL, NULL, NULL);
```
Adding a user in *router_rest_accounts* - Attempt #4

Let's test it again:

```bash
[fred@imac ~] $ curl -s -k -u lefred:fosdem2 https://localhost:8443/api/20190715/router/status | jq
{
  "processId": 17446,
  "productEdition": "MySQL Community - GPL",
  "timeStarted": "2021-01-07T13:37:53.982719Z",
  "version": "8.0.22",
  "hostname": "single-mysql"
}
```
Adding a user in *router_rest_accounts* - Attempt #4

Let's test it again:

```
[fred@imac ~] $ curl -s -k -u lefred:fosdem2 https://127.0.0.1:8443/api/20190715/router/status | jq
{
  "processId": 17446,
  "productEdition": "MySQL Community - GPL",
  "timeStarted": "2021-01-07T13:37:53.982719Z",
  "version": "8.0.22",
  "hostname": "single-mysql"
}
```

It works... but again, another component needs to be installed and maintained...
lefred - usability : 0 - 2
Adding a user in `router_rest_accounts` - Attempt #5

What else?
Adding a user in *router_rest_accounts* - Attempt #5

What else?

Let me ask again with more context to the MySQL Router Development Team...
Adding a user in router_rest_accounts - Attempt #5

What else?

Let me ask again with more context to the MySQL Router Development Team.

Summary of the answer: we do support MySQL 8.0’s default authentication string but also modular_crypt_format for MCF style password hashes as specified in the WL...
Adding a user in `router_rest_accounts` - Attempt #5

What else?

Let me ask again with more context to the MySQL Router Development Team...

Summary of the answer: we do support MySQL 8.0’s default authentication string but also modular_crypt_format for MCF style password hashes as specified in the WL...

Oups...
Adding a user in `router_rest_accounts` - Attempt #5

What else?

Let me ask again with more context to the MySQL Router Development Team...

Summary of the answer: we do support MySQL 8.0's default authentication string but also modular_crypt_format for MCF style password hashes as specified in the WL...

Oups...

This means that the standard Python `crypt` module is what I need!

```python
>>> import crypt
>>> crypt.crypt("fosdem3", crypt.mksalt(method=crypt.METHOD_SHA256))
'$5$tzRKNfEyehq1B5/Q$/PLKjs6PCEjNFiSDtdVZY2aL666SZrvKvrMWFyKI082'
```
lefred - developer: 0 - 2
Adding a user in `router_rest_accounts` - Attempt #5

```sql
REPLACE INTO router_rest_accounts VALUES (
    (SELECT cluster_id FROM mysql_innodb_cluster_metadata.v2_clusters LIMIT 1),
    "lefred", "modular_crypt_format",
    "$5$tzRKNfEYehq1B5/Q$/PlKjs6PCEjNFiSDtdVZV2aL666SZrvKvrMWfYk1082",
    NULL, NULL, NULL);
```
Adding a user in `router_rest_accounts` - Attempt #5

```sql
REPLACE INTO router_rest_accounts VALUES (  
(SELECT cluster_id FROM mysql_innodb_cluster_metadata.v2_clusters LIMIT 1),  
"lefred", "modular_crypt_format",  
"$5$tzRKNFhEeyhq1B5/Q$/P1Kjs6PCEjNFisDtdVZv2al6665zIvKvRMWfYkIO82",  
NULL, NULL, NULL);
```

```bash
[fred@imac ~]$ curl -s -k -u lefred:fosdem3 https://example.com:8443/api/20190715/router/status | jq
{
  "productId": 17446,
  "productEdition": "MySQL Community - GPL",
  "timeStarted": "2021-01-07T13:37:53.982719Z",
  "version": "8.0.22",
  "hostname": "single-mysql"
}
```
Woohoooo \o/
Examples

MySQL Router REST API
Some examples

We can now use the REST API with curl and include that in any monitoring tool like Sensu, Icinga...
MySQL Router & MySQL Shell

With MySQL Shell’s router plugin (https://github.com/lefred/mysqlshell-plugins), you can take benefit of all this.
MySQL Router & MySQL Shell

With MySQL Shell’s router plugin (https://github.com/lefred/mysqlshell-plugins), you can take benefit of all this.

Creating a user for the REST API:

```
MySQL> router.createRestUser()
JS> router.createRestUser()
Enter the username: fosdemuser
Enter the password: ******
Enter the password again: ******
You can now use 'fosdemuser' to authenticate to MySQL Router's REST API.
Use myrouter=router.create("fosdemuser@<router IP>:8443") to create an object to monitor.
```
MySQL Router & MySQL Shell

With MySQL Shell's router plugin (https://github.com/lefred/mysqlshell-plugins), you can take benefit of all this.

Creating a user for the REST API:

```
MySQL> router.createRestUser()
Enter the username: fosdemuser
Enter the password: ******
Enter the password again: ******
You can now use 'fosdemuser' to authenticate to MySQL Router's REST API.
Use myrouter=router.create("fosdemuser@<router IP>:8443") to create an object to monitor.
```

Much easier ;) Convenient++
Creating the **Router** object:

```javascript
JS
myrouter=router.create("fosdemuser@192.168.0.100:8443")
Password: *******
```

```json
{
  "api": "20190715",
  "blockedHosts": <Function:<lambda>>,
  "connections": <Function:<lambda>>,
  "status": <Function:<lambda>>
}
```
MySQL Router & MySQL Shell

```javascript
myrouter.status()
+-----------------------------+
| Cluster name: mycluster |
+-----------------------------+
    Refresh Succeeded: 301
    Refresh Failed: 0
Last Refresh Hostname: single-mysql:3306

+--------+
| routes |
+--------+

* mycluster_ro (alive):
  Total Connections: 6  Active Connections: 1  Blocked Hosts: 0
  ---> mysql-2: 3306
  ---> mysql-3: 3306

* mycluster_rw (alive):
  Routing Strategy: first-available  Protocol: classic
  Total Connections: 533  Active Connections: 4  Blocked Hosts: 1
  ---> single-mysql: 3306

* mycluster_x_ro (alive):
  Total Connections: 0  Active Connections: 0  Blocked Hosts: 0
  ---> mysql-2: 33060
  ---> mysql-3: 33060

* mycluster_x_rw (alive):
  Routing Strategy: first-available  Protocol: x
  Total Connections: 0  Active Connections: 0  Blocked Hosts: 0
  ---> single-mysql: 33060
```
MySQL Router & MySQL Shell

We have an host blocked by MySQL Router, which one is it?
My SQL Router & My SQL Shell

We have an host blocked by MySQL Router, which one is it?

<table>
<thead>
<tr>
<th>Route</th>
<th>Blocked Host(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>mycluster_ro</td>
<td></td>
</tr>
<tr>
<td>mycluster_rw</td>
<td>10.0.1.2</td>
</tr>
<tr>
<td>mycluster_x_ro</td>
<td></td>
</tr>
<tr>
<td>mycluster_x_rw</td>
<td></td>
</tr>
</tbody>
</table>
MySQL Router & MySQL Shell

And finally the routing statistics:

<table>
<thead>
<tr>
<th>Route</th>
<th>Source</th>
<th>Destination</th>
<th>From Server</th>
<th>To Server</th>
<th>Connection Started</th>
</tr>
</thead>
<tbody>
<tr>
<td>mycluster_ro</td>
<td>127.0.0.1:36464</td>
<td>10.0.1.3:3306</td>
<td>109 kb</td>
<td>12 kb</td>
<td>2021-01-07T15:29:30.109637Z</td>
</tr>
<tr>
<td>mycluster_rw</td>
<td>10.0.1.2:53108</td>
<td>10.0.0.2:3306</td>
<td>21 kb</td>
<td>771 bytes</td>
<td>2021-01-07T15:35:12.786968Z</td>
</tr>
<tr>
<td></td>
<td>109.128.188.66:44432</td>
<td>10.0.0.2:3306</td>
<td>2 kb</td>
<td>865 bytes</td>
<td>2021-01-07T17:18:45.192253Z</td>
</tr>
<tr>
<td></td>
<td>127.0.0.1:38154</td>
<td>10.0.0.2:3306</td>
<td>3 kb</td>
<td>1018 bytes</td>
<td>2021-01-07T15:38:17.023359Z</td>
</tr>
<tr>
<td></td>
<td>109.128.188.66:44962</td>
<td>10.0.0.2:3306</td>
<td>2 kb</td>
<td>865 bytes</td>
<td>2021-01-07T17:46:11.911282Z</td>
</tr>
<tr>
<td></td>
<td>127.0.0.1:38130</td>
<td>10.0.0.2:3306</td>
<td>25 kb</td>
<td>3 kb</td>
<td>2021-01-07T15:23:03.941722Z</td>
</tr>
<tr>
<td>mycluster_x_ro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mycluster_x_rw</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Questions ?