How to make CBS datawarehouse work on Computerome

Peter Wad Sackett

DTU Bioinformatics
Department of Bio and Health Informatics

V 1.4

Overview of content

Necessary accounts

Benefits and advice

Initial setup – can’t be skipped

- SSH keys
- MySQL Config file
- Script

Using the MySQL client

Using Perl and MySQL

Using Python and MySQL

The general idea is to create a SSH tunnel between Computerome and CBS MySQL server where the data can flow. This means that you need a password-less SSH connection, i.e. SSH keys, and a unique port for the tunnel. A (SSH) tunnel is a kind of a VPN connection for a very specific purpose.

Setup - accounts

It is required to have an account on

- The CBS network – for the MySQL
- Computerome – for the compute

The guide only works for normal CBS accounts, not student accounts. It is possible to get student accounts to work, if the account is set up for it.

If you reach a deep enough understanding of the method, then your MySQL server can be any place of your choosing, just like the compute server is not necessarily Computerome.

You still need accounts on both systems.

Benefits and Advice

When you have successfully have set up SSH keys, you can SSH to CBS without given your password. You can also securecopy (scp) from Computerome to CBS without a password.

Please note that the IP of login.cbs.dtu.dk is 192.38.86.2

When setting up SSH keys this can verify that you are doing it correctly on the right server. Look at the messages.

Warning: It needs to noted that the CBS MySQL server is not dimensioned to serve a compute unit the size of Computerome.

There is a substantial (time) cost in setting up a MySQL connection. This means that in order to work effectively, you should set up the connection and have your program work with it for a long time. Setting up the connection and tearing it down after 5 seconds is not cost-effective. Design your jobs according to this.

Due to limitations both in the method and the hardware it is best to have only a few jobs in your workflow perform the MySQL access. Having hundreds of jobs accessing the MySQL server simultaneously is not a good idea.

Setup – SSH keys

There is a directory **.ssh** in your home on both on Computerome and at CBS.

```
cd ~/.ssh
```

At Computerome there is a file called **id_rsa.pub** or **id_dsa.pub** in this directory. In the following id_rsa.pub is used as an example.

If the id_rsa.pub file is not present at Computerome, run

```
ssh-keygen -t rsa
```

and it will be there. Don't enter a passphrase.

This file needs to be copied from Computerome to CBS. This is easiest to do from Computerome (you are in the .ssh directory).

```
scp id_rsa.pub CBSLOGIN@login.cbs.dtu.dk:.ssh/tmp
```

Log in to CBS and go to the .ssh dir. **Add** the file to authorized_keys and remove it afterwards. Ensure correct permissions.

```
cat tmp >> authorized_keys; rm tmp;
chmod 600 authorized_keys
```

Setup – MySQL configuration file; .my.cnf

In your home on Computerome you must create a file called

.my.cnf

with rw------- permissions: `chmod 600 .my.cnf`

The content must be (the spaces are essential):

```
[client]
user       =  YOUR_CBS_LOGIN
password   =  YOUR_CBS_MYSQL_PASSWORD
host       =  127.0.0.1
port       =  RANDOM_NUMBER_BETWEEN_4000_AND_14000
[mysql]
pager      =  less
```

The capitalized parameters must be replaced appropiately.

You can find your CBS mysql password in the .my.cnf file in your home on the CBS network.

Choose a good random number or you will conflict with other users.

Using a script for SSH tunnel administration

On a technical level there can only be **one tunnel per node per user**. The tunnel also have to be **stopped** when you are done with accessing the database. To ensure this is handled correctly also when scheduling jobs that uses the database, load the modules

```
module load tools perl/5.20.2 cbsperltools/1.0
```

This will give access to the script `mysqltunnel.pl`, which uses the data in the `.my.cnf` configuration file and keeps count of the processes that might use the database.

Before you start your database access, either in a job script or interactively executed on a node, type

```
mysqltunnel.pl begin
```

When you are done – interactively or in a job script, type

```
mysqltunnel.pl end
```

There are some more options and possibilities, which can be seen by just executing `mysqltunnel.pl`.

Using MySQL client

In order to use the MySQL client, load this module on Computerome.

```
module load tools mariadb/10.0.21
```

Assuming the tunnel has been started, all you do is type

```
mysql
```

and you will have access to all the databases you have at CBS.

The `mysql` command is to be used exactly as you do at CBS.

Remember to **start** the tunnel as shown on previous page before using the mysql client. Also remember to **stop** the tunnel.

How to use the mysql command is outside the scope of this guide.

Consult online resources for this, like

https://linux.die.net/man/1/mysql

Programming against the database

It is possible - and actually a superior method - to program in Perl or Python against the MySQL database. You need to load some modules as shown on subsequent pages.

When you do this, you **do not** have to stop and start the tunnel using the script. This is done dynamically by using the correct Perl/Python modules. These modules uses the data in `.my.cnf` and ensure no conflict between several programs using the database can arise due to tunneling issues.

Both the `mysqltunnel.pl` script and the Perl/Python modules have been developed in a general way, such that they can be used from any computer, that needs to access a database that lies behind a firewall in a completely system/network.

Contact the author, Peter Wad Sackett, for details.

Using Perl and MySQL

The following modules have to be loaded on Computerome.

```
module load tools mariadb/10.0.21
module load tools perl/5.20.2
module load tools cbsperltools/1.0
```

Perl uses DBI for database access. A Perl module (’man MysqlTunnel’ for more info) has been created that will dynamically set up the ssh tunnel without the shell script previously mentioned.

```
use MysqlTunnel;
use DBI;
 # Open tunnel, then find connection data from mysql config file .my.cnf
my $tunnel = new MysqlTunnel;
my ($dsn, $user, $password) = $tunnel->getparm(’DATABASE');
# Create connection
 my $dbh = DBI->connect($dsn, $user, $password, { RaiseError => 1, AutoCommit => 0 });
# Do your database queries here
 # Close MySQL db connection and your tunnel afterwards
$dbh->disconnect;
$tunnel->close;
```

You must replace DATABASE with the name of the database you want to use on the CBS MySQL server.

Using Python and MySQL

The following modules have to be loaded on Computerome. You have to choose between Python 2 or Python 3 (anaconda2/3).

```
module load tools mariadb/10.0.21
module load tools anaconda2/4.0.0
module load tools anaconda3/4.0.0
module load tools cbspythontools/1.0
```

or

In Python, you use MySQLdb for handling database access. A python module ('man MysqlTunnel' for more info) which sets up the SSH tunnel exists. The code for the essential **connect** statement looks like (replace DATABASE with the database you want to use):

```
import MySQLdb
import MysqlTunnel
# Create tunnel and connect to database
host, port, user, passwd = MysqlTunnel.create()
dbh = MySQLdb.connect(host=host, port=port, user=user, passwd=passwd, db="DATABASE")
# Do database queries here
# Now close the connections
dbh.close()
MysqlTunnel.close()
```

Deprecated bash script for SSH tunnel administration

On the next page is a strongly deprecated bash script for the administration of the SSH tunnel. It is included for educational purposes. It uses some of the data in the `.my.cnf` configuration file.

Save the script on Computerome under the name **`mysqltunnel`**.

It must be started on each node that wants to use the database.

```
mysqltunnel start
```

When the node (your program) is done using the database, the tunnel must stopped again. This is VERY IMPORTANT.

```
mysqltunnel stop
```

This means that if you have several batch jobs accessing the database the jobs must start with creating the tunnel and end with stopping it.

There can only be ONE tunnel per node per user.

This limitation severely and silently enforced. It is a consequence of making the solution this easy and it also helps in keeping the connections to the database server down.

The deprecated bash script itself

```
#!/bin/tcsh
set mac = (`ip link | gawk '/link\/ether/ {gsub(”:”,””); print $2; exit;}'`)
if ( $1 == 'start' ) then
   if ( -e $HOME/.mysqltun.pid.$mac ) then
      echo "The pid file already exists, tunnel is probably running"
   else
      set param = (`gawk '/^user/{u=$3;}/^port/{p=$3;}END{print u, p;}' $HOME/.my.cnf`)
      ssh -N -L {$param[2]}:mysql.cbs.dtu.dk:3306 {$param[1]}@login.cbs.dtu.dk &
      echo $! > $HOME/.mysqltun.pid.$mac
   endif
else if ( $1 == 'stop' ) then
   if ( -e $HOME/.mysqltun.pid.$mac ) then
      set pid = `cat $HOME/.mysqltun.pid.$mac`
      kill $pid
      rm $HOME/.mysqltun.pid.$mac
   else
      echo "No pid file, tunnel is probably not running"
   endif
else
   echo "Usage: mysqltunnel [start|stop]"
endif
```