Hentzenwerke Whitepaper Series

Installing MySQL via RPMs

By Whil Hentzen

Installing MySQL on Linux is a reasonably straightforward procedure, and the documentation is excellent - organized, detailed, and with a lot of examples. However, there are still a lot of variations that you can encounter, and, as with any documentation team, there are still places where they have to assume a minimal level of knowledge, and if you're missing that knowledge, the procedure can still be difficult. The purpose of this document is to provide a supplement to the online documentation, with a detailed description of how to install MySQL from RPMs onto a Linux machine from the perspective of an experienced Windows programmer who is relatively new to Linux, and to provide background and perspective behind the steps.

Hentzenwerke Publishing, Inc. • books@hentzenwerke.com • www.hentzenwerke.com

1. Preface

1.1 Copyright

1.2 Revisions

1.2.1 History

Version	Date	Synopsis	Author
1.0.0	2004/8/2	Original	WH

1.2.2 New version

The newest version of this document will be found at www.hentzenwerke.com.

1.2.3 Feedback and corrections

If you have questions, comments, or corrections about this document, please feel free to email me at 'books@hentzenwerke.com'. I also welcome suggestions for passages you find unclear.

1.3 References and acknowledgments

Thanks to the ProLinux irregulars, including Leland Jackson, Ed Leafe and Ted Roche, Paul DuBois of the MySQL team, Daniel Kasak, Fagyal Csongor, and Bruce Douglas on the MySQL list.

1.4 Disclaimer

No warranty! This material is provided as is, with no warranty of fitness for any particular purpose. Use the concepts, examples and other content at your own risk. There may be errors and inaccuracies that in some configurations may be damaging to your system. The author(s) disavows all liability for the contents of this document.

Before making any changes to your system, ensure that you have backups and other resources to restore the system to its state before making those changes.

All copyrights are held by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark. Naming of particular products or brands should not be seen as endorsements.

1.5 Prerequisites

This document was written using MySQL 4.0.20 running on Fedora Core 2.0 and assumes a beginner's familiarity with use of Linux via the GUI and the Command Window.

2. Overview

As servers go, MySQL is pretty straightforward. If you've not installed a server of any type before, however, there are some things you need to know.

MySQL runs as a service. This means that once it's installed and started, it waits in the background for a request, and delivers results to the requester, based on the request.

Similar to the way a Web server takes requests from a browser and delivers Web pages back to the browser, the MySQL server takes requests from a user, modifies or queries the database as a response to those requests, and, finally, returns those results to the user.

The server is just a program that can be started up automatically when the machine is booted, or manually by issuing a command in a command window. During startup, it looks for information about how to configure itself from a configuration file. It creates a socket file (provides an anchor for a connection) and a PID file (a placeholder that identifies which mysqld is connected to the databases). Once running, it looks for databases in a predefined location.

The MySQL service has to be run by a Linux user, just as when you execute a command or load a program when logged in with your account. The installation of MySQL can create a specific user, named mysql, under which the service runs. For example, if you were logged into the Linux machine as 'herman', started up MySQL, and then executed a 'ps' (process status) command, you'd see a listing like this (the lines have been truncated so they'd fit):

```
[root@indy ~] ps -aux
root      2420  0.0  0.3  5504  984 ?        S    19:30   0:00 /bin/sh /usr/bin/mysqld_sa...
mysql     2437  0.0  4.2 30436 10932 ?       S    19:30   0:00 /usr/sbin/mysqld --basedir...
mysql     2438  0.0  4.2 30436 10932 ?       S    19:30   0:00 /usr/sbin/mysqld --basedir...
mysql     2439  0.0  4.2 30436 10932 ?       S    19:30   0:00 /usr/sbin/mysqld --basedir...
herman    2554  0.0  0.3  4136  772 pts/3    R    22:20   0:00 ps -aux
```

This shows that the mysqld process was started by the root user, but then all subsequent mysqld processes were run by the mysql user, and that the user 'herman' executed the 'ps' command. This 'mysql' user also needs access to the various files that MySQL accesses while it's running. For example, the user 'mysql' needs access to the data directory and files.

However, this 'mysql' user doesn't have anything to do with what's happening inside mysql - it's just who is running this process as far as the operating system (Linux) is concerned.

Inside MySQL, there's another set of users who are defined via entries in a set of MySQL tables. When MySQL is installed, two users are created by default; the root user, who has permission to do anything, and the anonymous user, who has limited capabilities.

In the existing documentation, I've occasionally found it confusing which user is being talked about during the installation - the Linux user or one of the the MySQL users, so I'll be sure to be explicit about which one I'm talking about

3. The essential installation process

The basic process involves five steps. (1) Read the documentation on the MySQL site. (2) Download the appropriate file(s). (3) Install the files. (4) Test the installation. (5) Set passwords.

There are several ways to install MySQL. The source distribution method involves unpacking gzipped 'tar' files (that have extensions of .tar.gz) while the binary distribution method uses RPM files and the RPM installer. This document describes how to install the RPM files on a Fedora Core (or compatible) system, using only the command window. It also assumes you've got root access to the server (in these examples named 'indy'), and have a regular user account (in these examples named 'herman'.)

The reason we're going to use RPMs instead of, for example, installing from the tar files, is that the RPM installation includes a number of scripts that set permissions, create databases, and set up the Linux 'mysql' user account that are run automatically. With the source distribution, you have execute each of those steps yourself. While the source distribution option provides more flexibility and power, it's probably better to rely on the automated RPM installation your first time so that you don't miss any steps. Once you're acquainted with MySQL to some extent, you can decide if it's appropriate to try a source distribution installation.

3.1. Read the documentation

Since you're reading this document, you're not too adverse to reading the fine manual that comes with the product, and in this specific case, I urge you to take advantage of it. The online documentation is very good, and you shouldn't take this document as a substitute for it. However, the documentation has to cover all possible scenarios, such as installing various types of files (RPMs and tar files) on Windows, Linux and other operating systems, so it can be confusing when having to skip parts that are not related to your particular situation. Thus, I winnowed out the parts not relevant to this particular type of installation, but point you to specific sections in the documentation when the time comes. So look at this document as a supplement to the online documentation, not a replacement.

Go to www.mysql.com, select the Developer Zone tab, and then the Documentation link. There are a number of options provided under the "MySQL Reference Manual" link. I've found the "Searchable, with user comments" link, as shown in **Figure 1**, to be easy to navigate and very useful, as the user comments often provide additional tips and a variety of perspectives of real life usage.

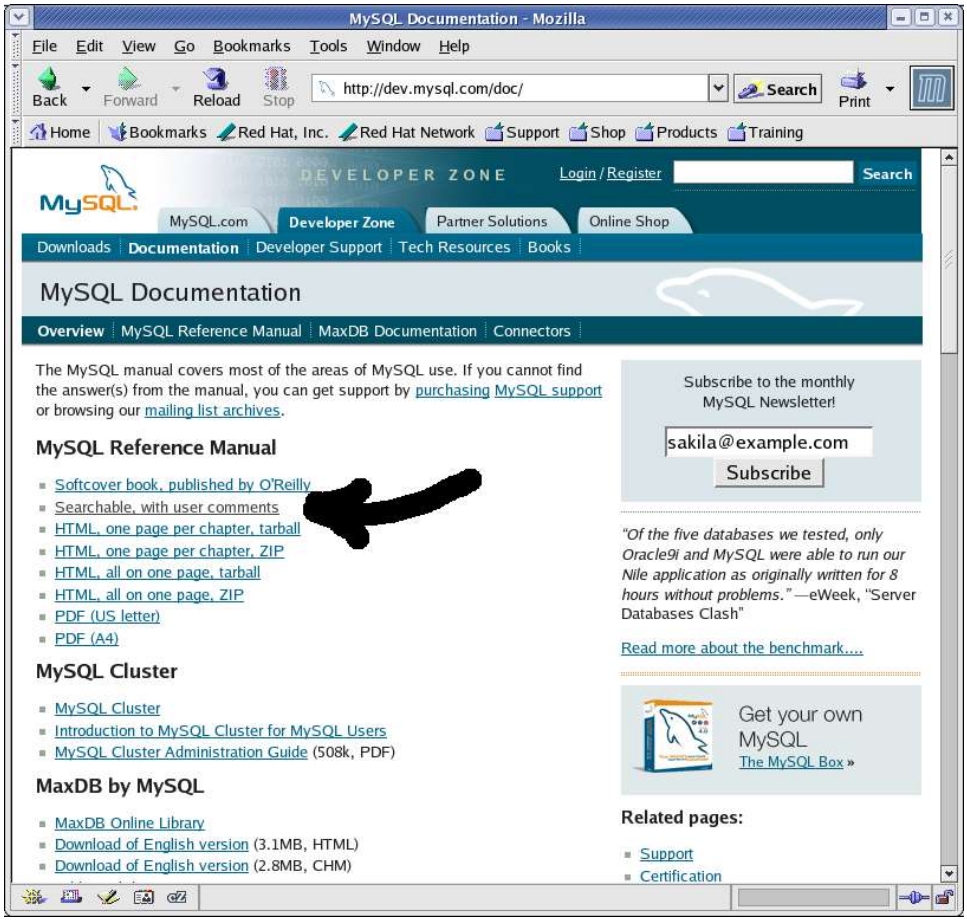

***Figure 1**. I've found the "Searchable, with user comments" online documentation to be very useful.*

Click on the "Installing MySQL" link on the left, and read away. The relevant sections you're going to need, at the very minimum, are:

```
2.1.4 Verifying Package Integrity Using MD5 Checksums or GnuPG
2.1.5 Installation layouts
2.2 Standard MySQL Installation using a Binary Distribution
2.2.2 Installing MySQL on Linux
2.2.5 Installing MySQL on Other Unix-Like Systems
2.4 Post-Installation Setup and Testing
```

I mention these specifically because I'll refer to each of these again in the following discussion, but don't assume that you don't need to read anything else. Depending on your situation, you may find other sections useful as well.

3.2. Download

Since this document assumes you're going to install the RPMs, let's get to it!

You'll be downloading the RPM files after logging into your regular user account on your Linux machine. But before you actually download the MySQL files, you should decide where to put them. Personally, I put all of the files I download into a directory called "zips" under my home directory to put, i.e. /home/whil/zips. (I've also got a directory on the file server that serves a similar purpose. Once I find that I'm actually using a program or tool, I'll archive the installation file(s) to the server for backup. It can be awfully inconvenient to want to reinstall something and find out that you can't get to the vendor's website.)

Once you've got your own storage location decided upon, go to www.mysql.com, click on Developer Zone, Downloads, and find the "MySQL 4.0 -- Generally Available (GA) release (recommended)" link as shown in **Figure 2**.

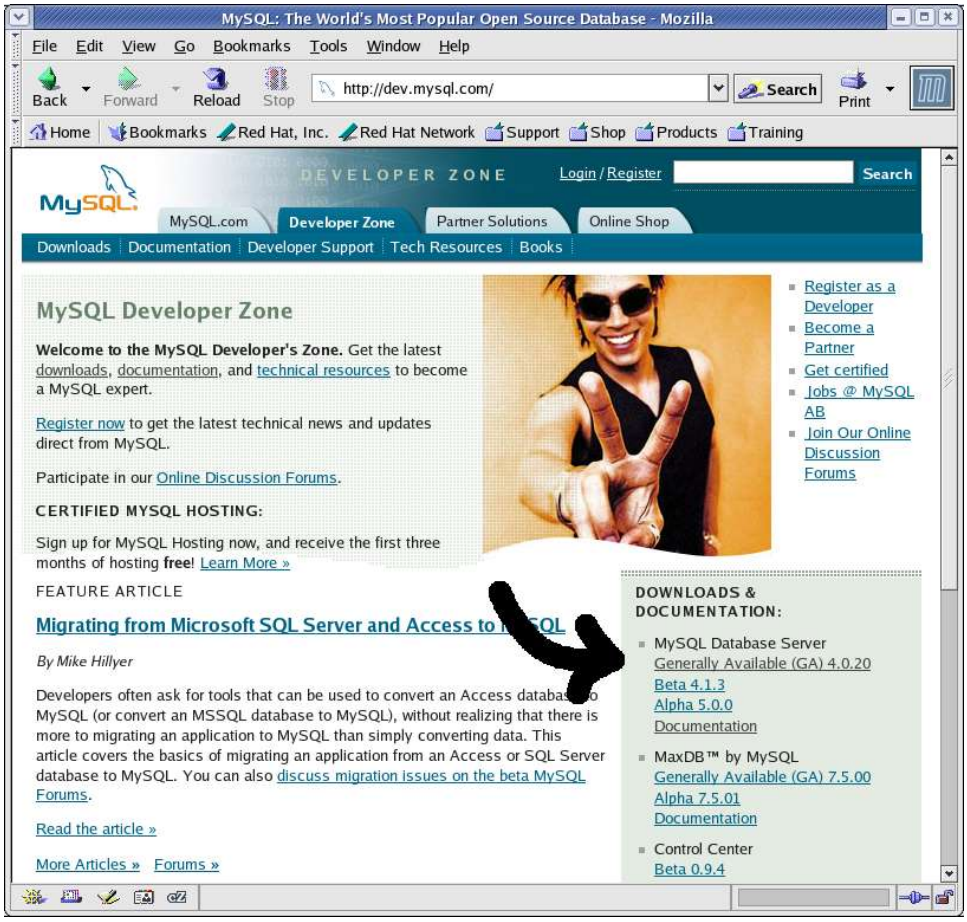

***Figure 2**. The Generally Available (GA) link is where the most current version of MySQL is located.*

Click on the link to get to the listing of downloads. Scroll down to the Linux x86 RPM downloads section, and download both the Server (9.8 MB) and Client programs (2.6 MB) files, as shown in **Figure 3**.

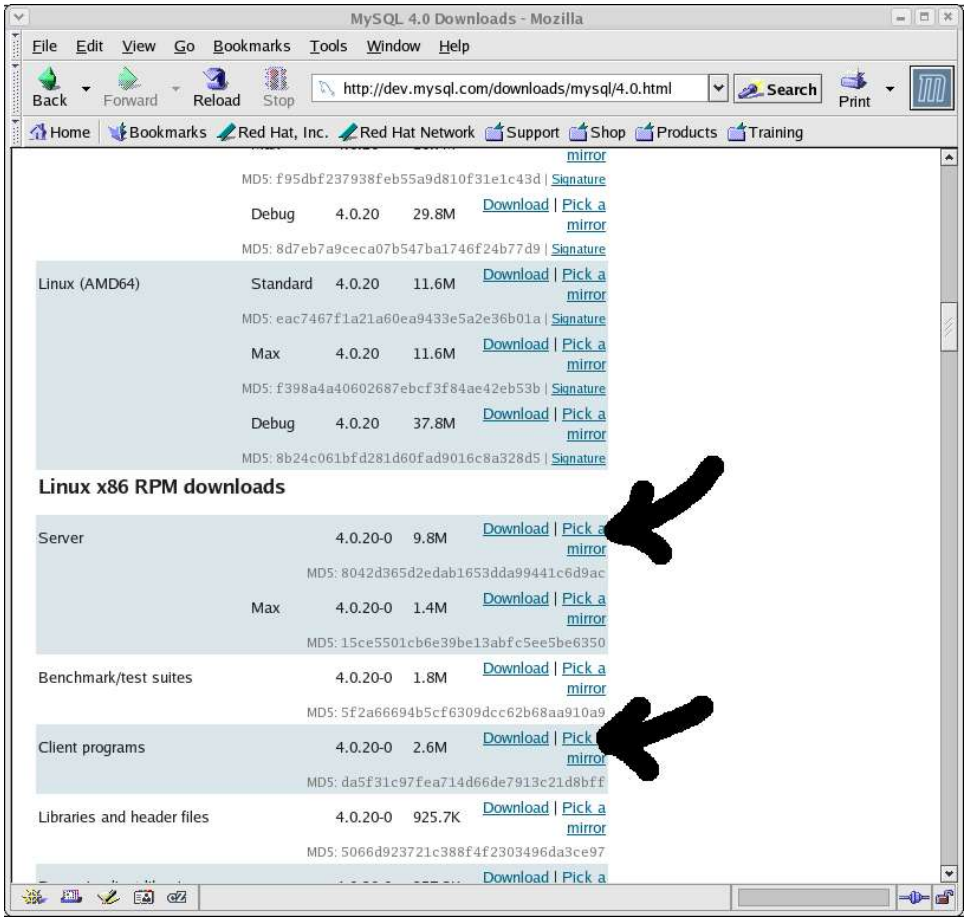

***Figure 3**. The Linux x86 RPM downloads section contains the links for the Server and Client programs.*

You'll end up with two files:

```
MySQL-server-4.0.20-0.i386.rpm
MySQL-client-4.0.20-0.i386.rpm
```

Note that the version (4.0.20.0 in this example) may change by the time you're reading this.

3.2.1. Verify the bits

It's a good idea to verify the integrity and authenticity of packages before installing them, as the bits can get scrambled while they're being downloaded. See the documentation's section 2.1.4 Verifying Package Integrity Using MD5 Checksums or GnuPG for details.

3.3. Install

Now that you've got genuine MySQL bits, switch to root with the 'su -' command, and then run the RPM commands as root. Note that although you're logged in as 'root', you're accessing the RPM files from within herman's home directory. The following listing shows what you might see when you install.

```
[herman@indy ~/zips] su -
Password?
[root@indy ~] cd /home/herman/zips
[root@indy /home/herman/zips] rpm -i MySQL-server-4.0.20.0.i386.rpm
warning: MySQL-server-4.0.20-0.i386.rpm: V3 DSA signature: NOKEY, key ID 5072e1f5
Preparing db table
Preparing host table
Preparing user table
Preparing func table
Preparing tables_priv table
Preparing columns_priv table
Installing all prepared tables
```

```
040729 16:20:21  Warning: Asked for 196608 thread stack, but got 126976
040729 16:20:21  /usr/sbin/mysqld: Shutdown Complete

[root@indy /home/herman/zips] rpm -i MySQL-client-4.0.20.0.i386.rpm
warning: MySQL-client-4.0.20-0.i386.rpm: V3 DSA signature: NOKEY, key ID 5072e1f5
```

If you run into a "error: can't create transaction lock" error, like so:

```
[herman@indy ~/zips] Dude? rpm -i MySQL-server-4.0.20-0.i386.rpm
warning: MySQL-server-4.0.20-0.i386.rpm: V3 DSA signature: NOKEY, key ID 5072e1f5
error: can't create transaction lock
```

you may be trying to install without switching to root first. In this listing, you can see that the command prompt shows 'herman' as the current user.

3.4. Installation results

If installation succeeds, a number of things have happened.

3.4.1. MySQL files have been installed

MySQL installs files in a number of directories. Which files are placed where depends on what type of installation is done. Installations created from Linux RPM distributions place files under the following system directories:

```
Directory  Contents of Directory
`/usr/bin'  Client programs and scripts
`/usr/sbin'    The mysqld server
`/var/lib/mysql'       Log files, databases
`/usr/share/doc/packages'     Documentation
`/usr/include/mysql'   Include (header) files
`/usr/lib/mysql'       Libraries
`/usr/share/mysql'     Error message and character set files
`/usr/share/sql-bench'
```

Section 2.1.5 in the online documentation also describes what files are placed where for other types of installations.

3.4.2 Linux user account is created

A Linux user account named "mysql' is created, if it doesn't already exist. This account is used for running MySQL but nothing else, and thus is set up to have limited rights.

Note: You normally don't need access to the Linux mysql user account, but if you want to log on as that user for some reason, you'll need to change the password to something you know by running the 'passwd' command as the Linux root user.

3.4.3 Data files are created

The server RPM places data under the `/var/lib/mysql' directory. Note that the Linux 'mysql' user has to have rights to this directory, or else MySQL will get frustrated trying to start up. You can check whether the data directory can be accessed by the 'mysql' user by changing to the /var/lib/mysql dir and then running the 'ls -al' command, like so:

```
[root@indy /var/lib/mysql] ls -al
total 20580
drwxr-xr-x   4 mysql root      4096 Jul 29 19:30 .
drwxr-xr-x  20 root  root      4096 Jul 29 16:20 ..
-rw-rw----   1 mysql mysql    25088 Jul 29 16:20 ib_arch_log_0000000000
-rw-rw----   1 mysql mysql 10485760 Jul 29 19:23 ibdata1
-rw-rw----   1 mysql mysql  5242880 Jul 29 19:30 ib_logfile0
-rw-rw----   1 mysql mysql  5242880 Jul 29 16:20 ib_logfile1
-rw-rw----   1 mysql root      1386 Jul 29 19:30 indy.hidbigo.com.err
-rw-rw----   1 mysql mysql        5 Jul 29 19:30 indy.hidbigo.com.pid
-rw-rw----   1 mysql mysql      291 Jul 29 19:23 indy.log
-rw-rw----   1 mysql mysql     2362 Jul 29 23:57 innodb.status.2437
drwx--x--x   2 mysql root      4096 Jul 29 16:20 mysql
srwxrwxrwx   1 mysql mysql        0 Jul 29 19:30 mysql.sock
drwxr-xr-x   2 mysql root      4096 Jul 29 16:20 test
```

The third column lists the user who owns the file or directory in question.

The RPM installation scripts automatically handle all of the ownership and permissions requirements. Details on what is required are covered in section 2.2.5, “Installing MySQL on Other Unix-Like Systems” of the online documentation.

3.4.4 Automatic startup entries are created

A 'mysql' entry is made in /etc/init.d.

Links to the mysql script in /etc/init.d are made in the various /etc/rcN.d directories for the appropriate runlevels. For example, on my machine, /etc/rc3.d contains a link named S90mysql while /etc/rc0.d contains a link named K20mysql. (Your machine may use different numbers, such as S80mysql or K10mysql.) The first script starts up MySQL in runlevel 3, while the second script shuts down MySQL when the shutdown procedure enters runlevel 0.

More information on this is in section 2.4.2.2, Starting and Stopping MySQL Automatically.

3.4.5 The mysql daemon is started

If the RPM files that you install include MySQL-server, the mysqld server daemon should be up and running after installation. You should now be able to begin using MySQL.

3.5. Test to make sure MySQL is running

It's a good idea to test to make sure that the server is running and that you can access it properly. Here are a few tests you can perform to do so, ranging from determining whether the server is up or not, to whether or not you can connect to it and perform various operations.

3.5.1. Check to see if the mysql daemon process running

First, you can check that mysql is running simply by using the 'ps' command to see if there are 'mysqld' processes running.

```
[root@indy ~] ps -aux
```

If you don't see any entries like the following (note that these lines spill over onto two lines),

```
root      3805  0.0  0.3  5972  980 pts/2    S    16:20   0:00 /bin/sh /usr/bin/mysqld_safe --
datadir=/var/lib/mysql --pid-file=/va
mysql     3826  0.6  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3827  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3828  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3829  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3830  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3831  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3832  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3833  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3834  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
mysql     3835  0.0  3.9 30360 10160 pts/2   S    16:20   0:00 /usr/sbin/mysqld --basedir=/ --
datadir=/var/lib/mysql --user=mysql -
root      3836  0.0  0.3  3724  772 pts/2    R    16:22   0:00 ps -aux
```

it means that the mysqld daemon is not running.

3.5.2. Use mysqladmin to see if the daemon is running

The MySQL client installation includes a tool called "mysqladmin" that is used for what it sounds like - administering MySQL. Obviously, you'll need to have installed the client RPM as well as the server in order to have mysqladmin available, a requirement determined the hard way by yours truly when, in the heat of the moment, he forgot to do the client install after the server install.

Running /etc/bin/mysqladmin with the 'version' parameter generates feedback like that shown in the following listing.

```
[herman@indy ~] /usr/bin/mysqladmin version
/usr/bin/mysqladmin  Ver 8.40 Distrib 4.0.20, for pc-linux on i686
Copyright (C) 2000 MySQL AB & MySQL Finland AB & TCX DataKonsult AB
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL license

Server version          4.0.20-standard
Protocol version        10
Connection              Localhost via UNIX socket
UNIX socket             /var/lib/mysql/mysql.sock
```

```
Uptime:                 4 hours 35 min 51 sec

Threads: 1  Questions: 23  Slow queries: 0  Opens: 8  Flush tables: 1  Open tables: 2  Queries per second
avg: 0.001
```

This test is also covered in Step 4 of section 2.4.2, “Unix Post-Installation Procedures”, in the online documentation.

3.5.3. See what MySQL variables are available

MySQL has a number of system variables that contain useful information about itself and how it interacts with the environment. Using the 'variables' parameter with the mysqladmin command generates a long, long list of these variables. A few of these variables are shown in the next listing.

```
[herman@indy ~] /usr/bin/mysqladmin variables
+---------------------------------+----------------------------------------+
| Variable_name                   |Value                                   |
+---------------------------------+----------------------------------------+
| back_log                        | 50                                     |
| basedir                         | /                                      |
| binlog_cache_size               | 32768                                  |
| datadir                         | /var/lib/mysql/                        |
| default_week_format             | 0                                      |
| pid_file                        | /var/lib/mysql/indy.hidbigo.com.pid    |
| port                            | 3306                                   |
| socket                          | /var/lib/mysql/mysql.sock              |
| version                         | 4.0.20-standard                        |
```

3.5.4. See if you can shut down the server

The next test is to make sure you can shut the server down. (Restarting will be covered next.) As either a regular or root user, issue the mysqladmin command as shown here:

```
[herman@indy ~] /usr/bin/mysqladmin -u root shutdown
```

The results should look something like the following.

```
[herman@indy ~] 040731 16:57:24  mysqld ended
```

The "-u root" parameters specify which mysql user will be used to shut the server down internally. This 'root' user is the mysql root user, not the Linux 'root' user. Hopefully, the 'shutdown' parameter is rather obvious.

This test is also covered in Step 5 of section 2.4.2, “Unix Post-Installation Procedures”, in the online documentation.

3.5.5. See if you can restart the server

It's darn handy to be able to restart the server after you've shut it down. The mysqld_safe command does this as shown. (Note that there's a 'd' after 'mysql' in the command!)

```
[root@indy ~] /usr/bin/mysqld_safe --user=mysql --log &
```

The "mysqld_safe" part is the name of a script that runs the mysqld command that starts MySQL. The first parameter, --user=mysql, identifies the Linux user whose account will be used to run the mysqld command. The second parameter, --log, logs connections and queries to the default log file located in /var/lib/mysql, which is particularly useful when you're getting started and might need to do some troubleshooting.

The ampersand, "&", is technically not part of the mysqld_safe command, but rather is a standard Linux/Unix command metacharacter. It runs the issued command as a background process so that the command window comes back. It's like the 'nowait' clause found in Visual FoxPro. If you don't use the "&" at the end, the mysqld_safe process will run in the foreground and will remain 'tied' to the command window you started it from, and so if you close the command window, you may kill the mysqld_safe process.

As noted earlier, mysql creates several of files in the /var/lib/mysql directory, including a .err, a .pid, and a .sock file. If you attempted to start mysql while logged onto the machine as a Linux user who doesn't have rights to that directory, your attempt will fail, and you'll get an error like this:

```
/usr/bin/mysqld_safe --user=mysql --log &
Starting mysqld daemon with databases from /var/lib/mysql
/usr/bin/mysqld_safe: line 308: /var/lib/mysql/indy.hidbigo.com.err: Permission denied
/usr/bin/mysqld_safe: line 1: /var/lib/mysql/indy.hidbigo.com.err: Permission denied
```

```
tee: /var/lib/mysql/indy.hidbigo.com.err: Permission denied
040731 17:21:09  mysqld ended
tee: /var/lib/mysql/indy.hidbigo.com.err: Permission denied
[1]+  Exit 1                  /usr/bin/mysqld_safe --user=mysql --log
```

The generally accepted practice is to switch to the root user with "su -" before executing the command. Once you do, you'll see something like the following listing if your startup is successful.

```
[root@indy ~] /usr/bin/mysqld_safe --user=mysql --log &
[1] 4245
[root@indy ~] Starting mysqld daemon with databases from /var/lib/mysql
[root@indy ~]
```

The number echoed in response to the command, in this case, 4245, is the process ID. You can see it if you run the ps -aux command.

This test is also covered in Step 6 of section 2.4.2, “Unix Post-Installation Procedures”, in the online documentation.

Note that section A.3.2, "How to Run MySQL as a Normal User", has a section that states "On Unix, the MySQL server mysqld can be started and run by any user. However, you should avoid running the server as the Unix root user for security reasons. In order to change mysqld to run as a normal unprivileged Unix user user_name, you must do the following..."

This can be confusing, in that if you try to execute the script that runs mysqld, like so:

```
/usr/bin/mysqld_safe
```

as a regular Linux user, you'll get the “Permission denied” error earlier. What the paragraph from section A.3.2 is saying is that you need to start the script as root, but the script should tell mysqld to run as a regular Linux user, such as the 'mysql' user that the RPM installation process creates. The script gets the MySQL daemon running, and the daemon runs under a regular Linux user account, as I demonstrated earlier with the 'ps -aux' command.

3.5.6. Run simple tests

At this point (if the previous tests have been successful), the MySQL daemon is running.

There are a number of commands that you can issue to MySQL via the Linux operating system, similar to the mysqladmin command. For example,

```
/usr/bin/mysqlshow
```

will display an ASCII graphic of the databases available to MySQL, like so:

```
[herman@indy ~] /usr/bin/mysqlshow
+-----------+
| Databases |
+-----------+
| test      |
+-----------+
```

If you're logged into your Linux machine as a regular user, you'll only see the test database (and any others that have wide open permissions, which is hopefully none.) If you're logged into your Linux machine as root, however, you'll see all of the databases, like so:

```
[root@indy ~] /usr/bin/mysqlshow
+-----------+
| Databases |
+-----------+
| mysql     |
| test      |
+-----------+
```

Including the name of a database name as a parameter to the mysqlshow command will display the tables in the database.

```
[root@indy ~] /usr/bin/mysqlshow mysql
Database: mysql
+--------------+
|    Tables    |
+--------------+
| columns_priv |
| db           |
| func         |
```

```
| host         |
| tables_priv  |
| user         |
+--------------+
```

You can even execute a single SQL command, as in this example:

```
[root@indy ~] /usr/bin/mysql -e "select host,db,user from db" mysql
+------+---------+------+
| host | db      | user |
+------+---------+------+
| %    | test    |      |
| %    | test\_% |      |
+------+---------+------+
```

3.5.7. Enter the MySQL interactive environment

Assuming that all of these tests went well, you can pretty much assume that MySQL is up and running and ready for you to take charge. The next thing you'll want to do is get into the MySQL interactive environment, called the MySQL monitor.

This environment offers a command prompt that allows you to interact with MySQL much like the Linux command prompt allows you to issue commands to the Linux operating system. In order to load the MySQL monitor, issue the command:

```
[herman@indy ~] mysql -u root
```

and you'll get a new prompt in your command window, like so:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7 to server version: 4.0.20-standard-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

At this point, you're logged into the MySQL environment as the MySQL root user (not the Linux root user.)

While it isn't the purpose of this document to describe how to use the MySQL monitor in detail, a few introductory instructions are in order. First, note that you've loaded the monitor while logged into the machine as a regular Linux user. However, the "-u root" clause after the 'mysql' command logs you into MySQL as the root user. If there was another user already set up in MySQL, such as 'alvin', you could have issued the command

```
[herman@indy ~] mysql -u alvin
```

and you would have been able to maneuver around inside of MySQL with all of the rights and permissions that alvin had been granted.

Second, as you saw in the opening response, you need to terminate a command with the semi-colon command (;) or \g in order to get MySQL to react to it. This means you can type a command on multiple lines, like so:

```
mysql> select NameFirst, NameLast
    -> from customers
    -> where State = "WI" ;
```

If you hit the Enter key at the end of a line without first typing one of the terminating characters, MySQL will provide a new line with a "->" prompt and the cursor wills imply jump to the beginning of the line.

In order to exit the monitor, type 'quit' or 'exit', and press Enter.

```
mysql> quit
Bye
[herman@indy ~]
```

You don't need to type a terminating character with either of these commands.

3.6. Set passwords

All the work we've done so far has been without the benefit of passwords on the accounts set up inside of MySQL. The next step, now that you've confirmed that you can start and stop the MySQL server and interact with it, is to set up passwords for the

MySQL users. If you're working on a test server behind a firewall, you may be tempted to wait for a bit before assigning passwords, but I'd encourage you not to.

It can be rather frustrating to spend a lot of time on a development project while using a test database without users and permissions set up, or to do the bulk of your work as 'root', only to find that you have to spend significant time reworking things because your system doesn't work like you thought it would when operated by regular users.

The first time you start the MySQL server, you're prompted like so:

```
PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
To do so, start the server, then issue the following commands:
/usr/bin/mysqladmin -u root password 'new-password'
/usr/bin/mysqladmin -u root -h indy.hidbigo.com password 'new-password'

See the manual for more instructions.

Please report any problems with the /usr/bin/mysqlbug script!

The latest information about MySQL is available on the web at
http://www.mysql.com
Support MySQL by buying support/licenses at https://order.mysql.com
```

You'll also read in the documentation that you can also set passwords inside MySQL with the 'set password' command and the password function, like so:

```
set password for ''@'localhost' = password('newpassword');
```

The password function, password('newpassword') encrypts the password before storing it int he database. The syntax of the part to the left of the equals sign, though, can be confusing at first. The first set of empty quotes, before the '@' sign, normally contains the name of the user. There are two users created in MySQL by the RPM installation routine. One is root and the other has no name - it's the anonymous user, ''. If you wanted to set the password for root using this syntax, you'd use

```
set password for 'root'@'localhost' = password('newpassword');
```

So if you want to set a password for the anonymous user, you simply don't enter a username, and you get "@'localhost'.

You can find out what users exist and if passwords, if any, exist for those users, with a SQL select statement, like so:

```
[herman@indy ~] mysql -u root -p
mysql> select host, user, password from mysql.user ;
+------------------+------+-------------------+
| host             | user | password          |
+------------------+------+-------------------+
| localhost        | root | 48bf4fd20c61a2f0 |
| indy.hidbigo.com | root | 48bf4fd20c61a2f0 |
| localhost        |      |                   |
| indy.hidbigo.com |      |                   |
+------------------+------+-------------------+
4 rows in set (0.07 sec)
```

This shows that the root user has had a password assigned but that the anonymous user has not. The password shown, 48bf4fd20c61a2f0, by the way, isn't the real 'root' password, but the encrypted version, due to the use of the password function in the 'set password' command.

This information is covered in section 2.4.3, "Securing the Initial MySQL Accounts".

3.7. How MySQL user passwords affect commands

After you assign a password to a MySQL user, you'll need to prompt for the password when issuing commands, like so:

After you assign a password to the root mysql user, you'll need to prompt for a password with the -p clause as well, like so:

```
[herman@indy ~] /usr/bin/mysqladmin -u root -p shutdown
Enter password:
[herman@indy ~] 040731 16:57:24  mysqld ended
```

Else, you'll get an error like so:

```
[herman@indy ~] /usr/bin/mysqladmin -u root shutdown
/usr/bin/mysqladmin: connect to server at 'localhost' failed
error: 'Access denied for user: 'root@localhost' (Using password: NO)'
```

Another example of failure is when trying to run the MySQL monitor:

```
[herman@indy ~] mysql -u root
ERROR 1045: Access denied for user: 'root@localhost' (Using password: NO)
```

Including the '-p' clause results in this exchange with the computer:

```
[herman@indy ~] mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11 to server version: 4.0.20-standard-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

And now you're connected to the MySQL database server via the monitor as the MySQL user named 'root'.

4. Summary

The documentation provided on the MySQL website is excellent, but there's a lot of it, some of which isn't applicable to an installation on a specific environment. In addition, there are scenarios where it can be helpful to hear something put a different way.

Subsequent whitepapers will cover other topics, such as connections to a MySQL server, converting data from other file formats, and connecting MySQL with development tools such as PHP and Python.

5. Where to go for more information

This free whitepaper is published and distributed by Hentzenwerke Publishing, Inc. We have the largest lists of “Moving to Linux”, OpenOffice.org, and Visual FoxPro books on the planet.

We also have oodles of free whitepapers on our website and more are being added regularly. Our Preferred Customer mailing list gets bi-monthly announcements of new whitepapers (and gets discounts on our books, first crack at special deals, and other stuff as we think of it.)

Click on “Your Account” at www.hentzenwerke.com to get on our Preferred Customer list.