MySQL HackFest Basics

Presented at

"MySQL Users Group"

Brisbane September 2006

Ronald Bradford

ronald.bradford@arabx.com.au

"Independent Consultant specialising in Database Modelling, Large Systems Design and Web Development Technologies."

MySQL HackFest Basics

The plan of attack

- Baseline
- Debugging
- Source Structure
- Your first change
- Documentation

Author: Ronald Bradford. Doc Version - 1.1

MySQL HackFest Basics

See Appendix Commands

Creating a Baseline

- 1. Getting the source
  - Downloads
  - Snapshots
  - Bit Keeper
- 2. Compiling
  - BUILD scripts
  - configure/make
- 3. Configuration
  - make install
  - mysql_install_db
- 4. Confirm Build
  - mysqld_safe
  - mysql
  - mysql-test

Author: Ronald Bradford. Doc Version - 1.1

MySQL HackFest Basics

Troubleshooting

- Is the code base known to compile?
- BUILD/compile-pentium-debug **-n**

MySQL HackFest Basics

Source Structure

- Major Directories
  - BUILD, client, vio, sql, mysys
  - storage/* (myisam, csv, blackhole etc)
  - dbug, pstack, regex, strings, zlib

client
|
vio
|
sql
|
storage/*
|
mysys

MySQL HackFest Basics

Debugging

- --debug (uses DBUG Package)

http://mysql.com/doc/refman/5.1/en/the-dbug-package.html
http://mysql.com/doc/refman/5.1/en/making-trace-files.html

- gdb - Command Line
- ddd – Graphical Debugger

MySQL HackFest Basics

Your First Change

- 1. New mysql SHOW command
  - SHOW USERGROUPS;
- 2. Add Acknowledgement
  - SHOW AUTHORS;
- 3. INFORMATION_SCHEMA
  - SELECT * FROM INFORMATION_SCHEMA.USER_GROUPS

MySQL HackFest Basics

2. Add Acknowledgement

See Appendix Commands

We will cheat and add this first (as it's easier, and explains 1.)

- sql_yacc.yy (Line 8195)
- sql_parse.cc (Line 3532)
- sql_show.cc (Line 204)
- authors.h (Line 37)

MySQL HackFest Basics

1. New SHOW Command

See Appendix Commands

- lex.h (1 line)
- sql_yacc.yy (2 lines + 1 chunk)

- sql_lex.h (1 line)
- sql_parse.cc (3 lines)

make clean required

- mysql_priv.h (1 line)
- sql_show.cc (1 chunk)
- usergroups.h (new)

MySQL HackFest Basics

Documentation

- Internals Manual (5.0) http://mysql.com/doc/internals
  - MySQL Manual snippets (Sections 2.9, 2.9.4, 2.13.1.3)
- Read the Source
- Internals Lists http://lists.mysql.com/internals
- Doxygen http://dev.mysql.com/sources/doxygen/mysql-5.1/index.html
  http://forge.mysql.com/wiki/CommunityDoxygenProject

MySQL HackFest Basics

Conclusion

- It's not trivial
- What is your objective?
- True Open Source is about contribution
  - Are your a leecher or a seeder?

MySQL HackFest Basics

References

- My Blogs
  - http://blog.arabx.com.au/?cat=31
- MySQL Farm
  - http://forge.mysql.com/wiki/MySQL_Build_Farm_Initiative
- MySQL Internals Manual
  - http://mysql.com/doc/internals

MySQL HackFest Basics

Appendix - Baseline Commands

```
# Pre-Configure
su -
useradd mysqldev
su - mysqldev
mkdir -p $HOME/src
cd $HOME/src

# Download
SNAPSHOT="mysql-5.1.11-beta";export SNAPSHOT
#wget http://mysql.planetmirror.com/Downloads/MySQL-5.1/${SNAPSHOT}.tar.gz
cp /tmp/${SNAPSHOT}.tar.gz .
md5sum ${SNAPSHOT}.tar.gz
# MD5: f5e807425247f7b726faefa232367fbd

# Compile/Install
tar xfz ${SNAPSHOT}.tar.gz
cd $SNAPSHOT
./BUILD/compile-pentium-debug  --prefix=$HOME/deploy
make install

# Configure Installation
cd $HOME/deploy
bin/mysql_install_db

# Confirm Installaton
bin/mysqld_safe --port=9306 --socket=/tmp/mysqldev.sock &
bin/mysql -P9306 -S/tmp/mysqldev.sock -e "SELECT VERSION()"
bin/mysqladmin -uroot --port=9306 --socket=/tmp/mysqldev.sock shutdown

# Test
cd mysql-test
./mysql-test-run --force
```

MySQL HackFest Basics

Appendix – Arjen's Baseline Commands

```
## ARJEN'S ALTERNATIVE

SNAPSHOT="mysql-5.1.11-beta";export SNAPSHOT
cd ~/src/$SNAPSHOT

./BUILD/compile-pentium-debug
cd mysql-test
./mysql-test-run alias

../sql/mysqld --skip-innodb --skip-grant-tables --basedir=. --datadir=./var/master-
data --language=../sql/share/english/ --socket=/tmp/mysqldev.sock - port=9050 &
sleep 5
../client/mysql -uroot --socket=/tmp/mysqldev.sock  -e "SELECT VERSION()"
../client/mysqladmin -uroot --socket=/tmp/mysqldev.sock shutdown
```

MySQL HackFest Basics

Appendix Debugging --debug

```
T@9190304: >dispatch_command
T@9190304: | >alloc_root
..
@9190304: | query: SHOW AUTHORS
T@9190304: | >mysql_parse
T@9190304: | | >mysql_init_query
T@9190304: | | | >lex_start
T@9190304: | | | | >alloc_root
T@9190304: | | | | | enter: root: 0x9590b38
T@9190304: | | | | | exit: ptr: 0x95c1c70
T@9190304: | | | | <alloc_root
T@9190304: | | | <lex_start
T@9190304: | | | >mysql_reset_thd_for_next_command
T@9190304: | | | <mysql_reset_thd_for_next_command
T@9190304: | | <mysql_init_query
T@9190304: | | >Query_cache::send_result_to_client
T@9190304: | | <Query_cache::send_result_to_client
T@9190304: | | >mysql_execute_command
T@9190304: | | | >mysqld_show_authors
T@9190304: | | | | >alloc_root
T@9190304: | | | | | enter: root: 0x9590b38
T@9190304: | | | | | exit: ptr: 0x95c1c78
T@9190304: | | | | <alloc_root
...
T@9190304: | | | | >send_fields
T@9190304: | | | | | packet_header: Memory: 0x8c2480  Bytes: (4)
01 00 00 01
T@9190304: | | | | | >alloc_root
T@9190304: | | | | >Protocol::write
T@9190304: | | | | <Protocol::write
..
T@9190304: | | | | >send_eof
```

MySQL HackFest Basics

Appendix Debugging --debug

```
...
T@9190304: | | | | >send_eof
T@9190304: | | | | | packet_header: Memory: 0x8c2550  Bytes: (4)
05 00 00 51
T@9190304: | | | | | >net_flush
T@9190304: | | | | | | >vio_is_blocking
T@9190304: | | | | | | | exit: 0
T@9190304: | | | | | | <vio_is_blocking
T@9190304: | | | | | | >net_real_write
T@9190304: | | | | | | | >vio_write
T@9190304: | | | | | | | | enter: sd: 23, buf: 0x0x95b9bd0, size: 5107
T@9190304: | | | | | | | | exit: 5107
T@9190304: | | | | | | | <vio_write
T@9190304: | | | | | | <net_real_write
T@9190304: | | | | | <net_flush
T@9190304: | | | | | info: EOF sent, so no more error sending allowed
T@9190304: | | | | <send_eof
T@9190304: | | | <mysqld_show_authors
T@9190304: | | <mysql_execute_command
T@9190304: | | >query_cache_end_of_result
T@9190304: | | <query_cache_end_of_result
...
T@9190304: | <mysql_parse
T@9190304: | info: query ready
T@9190304: | >log_slow_statement
T@9190304: | <log_slow_statement
T@9190304: <dispatch_command
```


Author: Ronald Bradford. Doc Version - 1.1

MySQL HackFest Basics

Appendix Debugging --debug=d,info,error,query,general,where:O,/tmp/mysqld.trace)

```
create_new_thread: info: creating thread 3
create_new_thread: info: Thread created
?func: info: handle_one_connection called by thread 3
?func: info: New connection received on socket (23)
?func: info: Host: localhost
?func: info: vio_read returned -1,  errno: 11
thr_alarm: info: reschedule
process_alarm: info: sig: 14  active alarms: 1
?func: info: client_character_set: 8
check_user: info: Capabilities: 238213  packet_length: 16777216  Host: 'localhost'  Login user: 'rbradfor' Priv_user: ''
Using password: no Access: 0  db: '*none*'
send_ok: info: affected_rows: 0  id: 0  status: 2  warning_count: 0
send_ok: info: OK sent, so no more error sending allowed
do_command: info: vio_read returned -1,  errno: 11
thr_alarm: info: reschedule
process_alarm: info: sig: 14  active alarms: 1
do_command: info: Command on socket (23) = 3 (Query)
dispatch_command: query: SHOW AUTHORS
send_eof: info: EOF sent, so no more error sending allowed
dispatch_command: info: query ready
do_command: info: vio_read returned -1,  errno: 11
thr_alarm: info: reschedule
process_alarm: info: sig: 14  active alarms: 1
do_command: info: Command on socket (23) = 1 (Quit)
~THD(): info: freeing security context
end_thread: info: sending a broadcast
end_thread: info: unlocked thread_count mutex
```

MySQL HackFest Basics

Appendix Debugging ---debug=d:N:F:L:O,/tmp/mysqld.trace

```
    49:    sql_parse.cc:  1536: do_command: info: Command on socket (23) = 3 (Query)
    50:      my_alloc.c:   172: alloc_root: enter: root: 0x9068b38
    51:      my_alloc.c:   219: alloc_root: exit: ptr: 0x9099c30
    52:    safemalloc.c:   128: _mymalloc: enter: Size: 16384
    53:    safemalloc.c:   197: _mymalloc: exit: ptr: 0x909cc80
    54:    safemalloc.c:   262: _myfree: enter: ptr: 0x9095bf8
    55:    sql_parse.cc:  1757: dispatch_command: query: SHOW AUTHORS
..
    88:     sql_show.cc:   385: mysqld_show_authors: packet_header: Memory: 0x940590  Bytes: (4)
..
   166:     protocol.cc:   385: send_eof: packet_header: Memory: 0x940550  Bytes: (4)
05 00 00 51
   167:     viosocket.c:   184: vio_is_blocking: exit: 0
   168:     viosocket.c:   105: vio_write: enter: sd: 23, buf: 0x0x9091bd0, size: 5107
   169:     viosocket.c:   117: vio_write: exit: 5107
   170:     protocol.cc:   342: send_eof: info: EOF sent, so no more error sending allowed
   171:      sql_lex.cc:   199: lex_end: enter: lex: 0x9068b58
   172:    sql_parse.cc:  1796: dispatch_command: info: query ready
   173:      my_alloc.c:   331: free_root: enter: root: 0x9068b38  flags: 1
   174:     viosocket.c:   184: vio_is_blocking: exit: 0
   175:     viosocket.c:    36: vio_read: enter: sd: 23, buf: 0x0x9091bd0, size: 4
   176:     viosocket.c:    49: vio_read: vio_error: Got error 11 during read
   177:     viosocket.c:    52: vio_read: exit: -1
   178:    sql_parse.cc:   809: do_command: info: vio_read returned -1,  errno: 11

   190:    sql_parse.cc:  1536: do_command: info: Command on socket (23) = 1 (Quit)
   191:      my_alloc.c:   331: free_root: enter: root: 0x9068b38  flags: 1
   192:       mysqld.cc:  1681: close_connection: enter: fd: socket (23)  error: ''
   193: sql_handler.cc:   628: mysql_ha_flush: enter: tables: (nil)  mode_flags: 0x02
```

MySQL HackFest Basics

Appendix Source

sql_yacc.yy (lex.h, sql_lex.h)

```
8192        | AUTHORS_SYM
8193          {
8194            LEX *lex=Lex;
8195            lex->sql_command= SQLCOM_SHOW_AUTHORS;
8196          }
```

sql_parse.cc

```
3531   case SQLCOM_SHOW_AUTHORS:
3532     res= mysqld_show_authors(thd);
3533     break;
```

MySQL HackFest Basics

Appendix Source sql_show.cc

```
204 bool mysqld_show_authors(THD *thd)
205 {
206   List<Item> field_list;
207   Protocol *protocol= thd->protocol;
208   DBUG_ENTER("mysqld_show_authors");
209
210   field_list.push_back(new Item_empty_string("Name",40));
211   field_list.push_back(new Item_empty_string("Location",40));
212   field_list.push_back(new Item_empty_string("Comment",80));
213
214   if (protocol->send_fields(&field_list,
215                             Protocol::SEND_NUM_ROWS | Protocol::SEND_EOF))
216     DBUG_RETURN(TRUE);
217
218   show_table_authors_st *authors;
219   for (authors= show_table_authors; authors->name; authors++)
220   {
221     protocol->prepare_for_resend();
222     protocol->store(authors->name, system_charset_info);
223     protocol->store(authors->location, system_charset_info);
224     protocol->store(authors->comment, system_charset_info);
225     if (protocol->write())
226       DBUG_RETURN(TRUE);
227   }
228   send_eof(thd);
229   DBUG_RETURN(FALSE);
230 }
```


Author: Ronald Bradford. Doc Version - 1.1

MySQL HackFest Basics

Appendix Source

authors.h

```
 37 struct show_table_authors_st show_table_authors[]= {
 38   { "Brian (Krow) Aker", "Seattle, WA, USA",
 39     "Architecture, archive, federated, bunch of little stuff :)" },
 40   { "Venu Anuganti", "", "Client/server protocol (4.1)" },
 41   { "David Axmark", "Uppsala, Sweden",
 42     "Small stuff long time ago, Monty ripped it out!" },
...
147   { "Brisbane Users Group", "Brisbane, QLD, Australia",
148     "New Command SHOW USERGROUPS" },
149   {NULL, NULL, NULL}
150 };
```

MySQL HackFest Basics

Appendix Debugging with gdb

```
$ cd $HOME/deploy
# libexec/mysqld --port=9306 --socket=/tmp/mysqldev.sock
$ gdb libexec/mysqld
(gdb) set args --basedir=/home/mysqldev/deploy --port=9306 --socket=/tmp/mysqldev.sock
(gdb) run
```