前言

概述

本文详尽阐述了在Windows平台上使用MySQL源码安装的过程，并深入探讨了基于aarch64-none-linux-gnu工具链的MySQL移植技术。此外，文章还提供了对常见的基于C语言的MySQL API的详细使用指南。通过本文的学习，您将能够更高效地安装、移植和使用MySQL，为您的开发工作带来极大的便利。

版本

V1.0.0

修改说明

无

Mysql安装(Windos 源码)

前言

概述

不知道写什么

环境

- Windows 10 专业版

主要参考资料

- MySQL 5.7详细下载安装配置教程-CSDN博客

安装流程

- 下载Mysql5.7源码
- 配置环境变量
- 创建my.ini文件
- 安装Mysql5.7
- 设置Mysql root用户密码

安装Mysql

下载源码

- 下载Mysql

  进入Mysql官网选择DOWNLOADS中的MySQL Comm server

进入下载界面选择Archives选项选择Product Version为对应版本5.7.36，选择Operating System为Windos平台，选择对应平台对应位数版本，选择Windos(X86,64-bit).ZIP Archive。

- 解压源码

在D盘创建文件夹Mysql5.7，并将之前下载的压缩包移动到Mysql5.7文件夹中，并解压到对应文件夹中。

设置环境变量

- 创建MYSQL_HOME环境变量

右击此电脑->选择属性->点击高级系统设置->点击环境变量->在系统环境变量中点击新建。变量名输入"**MYSQL_HOME**",变量值为之前下载Mysql源码解压的文件目录，点击确定。

- 编辑Path

选中系统变量中的Path->点击编辑选项->点击新建->"**%MYSQL_HOME%\bin**"->点击确定。

配置my.ini文件

在之前解压目录中创建my.ini文件，并输入如下配置信息：

```
[mysqld]
#端口号
port = 3306
#mysql-5.7.27-winx64的路径
basedir=D:\Mysql5.7\mysql-5.7.36-winx64
#mysql-5.7.27-winx64的路径+\data
datadir=D:\Mysql5.7\mysql-5.7.36-winx64\data
#最大连接数
max_connections=200
#编码
character-set-server=utf8

default-storage-engine=INNODB

sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES

[mysql]
#编码
default-character-set=utf8
```

安装Mysql

打开CMD，运行Mysql源码中的安装命令。

- 以管理员权限打开CMD
- cd进入Mysql解压目录的bin文件夹中
- 执行mysqld -install
- 执行mysqld --initialize
- 执行net start mysql

设置用户密码

- 执行命令关闭Mysql服务

```
        net stop mysql
```

- 修改my.ini文件

mysqld中任意位置添加skip-grant-tables

```
[mysqld]
#端口号
port = 3306
#mysql-5.7.27-winx64的路径
basedir=D:\Mysql5.7\mysql-5.7.36-winx64
#mysql-5.7.27-winx64的路径+\data
datadir=D:\Mysql5.7\mysql-5.7.36-winx64\data
#最大连接数
max_connections=200
#编码
character-set-server=utf8

default-storage-engine=INNODB

sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES

skip-grant-tables

[mysql]
#编码
default-character-set=utf8
```

- Mysql服务

```
        net start mysql
```

- 连接数据库

```
        mysql -u root -p
```

- 进入数据库

```
1          use mysql
```

- 修改密码

```
1          update user set authentication_string=password("xxxxxx") where
  user="root";    xxxxxx为设置的密码
```

修改成功会提示“Query OK”

- 手动停止Mysql服务

  打开服务

  选中MySQL并停止此服务

- 修改my.ini **删除或屏蔽skip-grant-tables**

```
[mysqld]
#端口号
port = 3306
#mysql-5.7.27-winx64的路径
basedir=D:\Mysql5.7\mysql-5.7.36-winx64
#mysql-5.7.27-winx64的路径+\data
datadir=D:\Mysql5.7\mysql-5.7.36-winx64\data
#最大连接数
max_connections=200
#编码
character-set-server=utf8

default-storage-engine=INNODB

sql_mode=NO_ENGINE_SUBSTITUTION,STRICT_TRANS_TABLES

#skip-grant-tables

[mysql]
#编码
default-character-set=utf8
```

- 再次打开一个cmd
- 启动Mysql服务

```
        net start mysql
```

- 连接数据库

  密码为之前自己设置的密码

```
        mysql -u root -p
```

- 进入数据库

```
use mysql
```

通过密码连接数据库，成功会提示"Database changed"并使前缀变为"mysql>"

安装如果遇到错误常考常见安装错误章节。

Mysql移植

前言

概述

项目需要使用数据库完成数据存储且要求具有数据库同步功能。由于Mysql是最大的开源数据库故数据库选择位Mysql。数据库需要部署到本地(ARM)上，Mysql没有对应的安装包，需要自己移植。且Mysql官方最低版本只有5.7才有官方手册，Mysql8以上的版本需要gcc10、cmake3.75版本太高交叉编译器不支持，于是选用Mysql5.7版本。

Mysql编译依赖与boost和openssl库，openssl库依赖与ncurses库，Mysql交叉编译生成的编译脚本无法在主机上运行，所以还需要主机编译的脚本文件。所以Mysql移植一共需要boost库、ncurses库、openssl库以及Mysql Linux版本脚本，所以移植Mysql前应该先移植boost、ncurses、openssl以及编译主机Mysql(linux)。移植步骤均是下载源码、配置对应设置、编译、安装、查看编译库文件格式。移植方案根据mysql源码一共有两种一种是下载带boost源码的mysql源码，另一种是下载不带boost源码自己下载boost源码编译后通过mysql源码调用。本次采用方案一移植(第二种不会配置boost)。

- 注意
  - 由于历史原因问题导致Mysql交叉编译常见问题没有参考文章(但是编译完下班第二天找不到了)
  - Mysql5.7.36版本依赖的openssl版本为openssl1.1.1、boost版本为boost1.59.0(其他版本看对应Mysql脚本文件)
  - Mysql主机编译和交叉编译用的Mysql源码是同一份

环境

- 虚拟机版本 VMware® Workstation 16 Pro 16.1.2 build-17966106
- Ubuntu版本 Ubuntu 18.04.6 LTS 64位
- Ubuntu编译工具链版本 gcc version 7.5.0
- 交叉编译工具链 aarch64-none-linux-gnu 64位
- 交叉编译工具链版本 gcc version 9.2.1 20191025 (GNU Toolchain for the A-profile Architecture 9.2-2019.12 (arm-9.10))

- vim版本 vim version 8.0.3741
- cmake版本 cmake version 3.10.2

主要参考文献

- 网友资料
  - MySQL 5.7详细下载安装配置教程-CSDN博客
  - mysql交叉编译方式
  - mysql交叉编译到arm平台（hasi）
- 官方手册
  - MySql5.7参考手册
  - C API参考手册
  - boost1.59参考手册
- 源码配置文件
  - ncurses配置文件 命令 ./configure --help
  - boost配置文件 命令 ./bootstrap.sh --help ./b2 --help
  - openssl配置文件 命令 ./config --help

移植流程

- 移植ncurses
- 移植boost
- 移植openssl
- 编译主机mysql
- 交叉编译mysql

Mysql移植

移植ncurses

下载源码

ncurses官方下载ncurses6.1版本源码，复制到虚拟机中解压到ncurses-6.1文件夹，并在同级目录下创建用于存放交叉编译文件的ncurses-6.1-arm文件夹。

- 下载官方源码
- 复制压缩包到虚拟机中
- 解压到ncurses-6.1文件夹中

```
tar -zvxf ncurses-6.1.tar.gz
```

- 创建ncurses-6.1-arm文件夹作为ncurses-6.1交叉编译安装目录

```
mkdir ncurses-6.1-arm
```

修改配置

- 编译选项说明

名称	作用
host	交叉编译器
prefix	编译完成的绝对路径
CC	指定交叉编译器
with-shared	生成共享库
without-progs	禁止使用程序如(tic)构建/安装

- 进入ncurses-6.1目录，设置交叉编译器和安装目录

```
cd ncurses-6.1
./configure --host=aarch64-none-linux-gnu --prefix=/home/xiang/Mysql/ncurses-6.1-arm CC=aarch64-none-linux-gnu-gcc --with-shared --without-progs
```

- **注意：ncurses安装路径需要使用绝对路径且路径末尾没有"/"。**

编译

- 执行make编译文件

```
make

make install //将编译文件安装到configure指定的文件夹
cd ncursea_6.1-arm
ls
```

- 查看安装文件格式是否和交叉编译器一致

```
cd lib
file libcurses.so.1
```

移植boost

下载源码

在boost官方下载boost_1_59_0版本源码，复制到虚拟机中解压到boost_1_59_0文件夹，并在同级目录下创建用于存放交叉编译文件的boost_1_59_0-arm文件夹。

- 官网下载boost_1_59_0
- 复制到虚拟机
- 解压到boost_1_59_0到boost_1_59_0文件夹

```
tar -vxf boost_1_59_0.tar.gz
```

- 创建boost_1_59_0-arm文件夹作为boost交叉编译安装目录

```
mkdir boost_1_59_0
ls
```

修改配置

- 配置选项说明

名称	作用
prefix	安装路径

- 进入解压的boost_1_59_0文件夹输入配置命令

```
cd boost_1_59_0
./bootstrap.sh  --prefix=/home/xiang/Mysql/boost_1_59_0-arm
```

编译

- 用vim打开project-config.jam配置文件，修改文件中的编译器选项
- 将using gcc 修改为 交叉编译器路径
- 路径应该有空格
- 编译

```
./b2              //编译
./b2 install      //安装
```

- 进入编译文件路径并用file命令查看编译文件是否和交叉编译器格式相匹配。

```
cd ../boost_1_59_0-arm/lib
file libboost_atomic.so.1.59.0
```

移植openssl

下载源码

在openssl官方下载openssl3.2.0版本源码，复制到虚拟机中解压到openssl3.2.0文件夹，并在同级目录下创建用于存放交叉编译文件的openssl3.2.0-arm文件夹。

- 官方下载openssl3.2.0(Mysql5.7需要openssl最低版本为1.01)

- 将下载文件复制到虚拟机
- 将openssl-3.2.tar.gz解压到openssl-3.2文件夹

```
tar -zxvf openssl-3.2.tar.gz
ls
```

- 创建openssl-3.2.0-arm文件夹作为交叉编译安装

```
mkdir open-3.2.0-arm
ls
```

修改配置

- 配置说明

名称	说明
prefix	安装路径(绝对路径))
cross-compile-prefix	编译器前缀
no-asm	不需要汇编
no-shared	静态编译
shared	共享库编译

- 进入openssl-3.2.0后，通过config配置Makefile选项

```
cd openssl-3.2.0
./config --prefix=/home/xiang/Mysql/openssl-3.2.0-arm --cross-compile-prefix=aarch64-none-linux-gnu- no-asm no-shared
```

编译

- 执行make编译和安装命令

```
make
make install
```

- 查看安装文件格式是否和交叉编译器一致

```
cd ../openssl-arm/bin
file openssl
```

编译主机mysql

安装依赖

主机编译也需要 boost 、openssl 和 ncurse 库，在Ubuntu系统中openssl和ncurse库可以通过apt-get 命令下载，具体命令如下：

```
apt-get install openssl
apt-get install libssl-dev

 openssl version    //查看安装的openssl版本

/* 通过dpgk安装工具查看版本号 */
dpgk -l libssl-dev openssl
```

下载源码

官网下载Mysql5.7.36源码，并复制到虚拟机中的Ubuntu系统中解压到mysql-5.7.36，在同级目录创建mysql-5.7.36-pc文件夹。

将压缩包复制到Ubuntu系统中

解压两份mysql-5.7.36，一份用于主机编译，一份用于交叉编译

```
#Ubuntu> tar -zxvf mysql-5.7.36.tar.gz
#Ubuntu> cp mysql-5.7.36 mysql-5.7.36-pc
#Ubuntu> mv mysql-5.7.36 mysql-5.7.36-arm
```

修改配置

DDOWNLOAD_BOOST	是否下载 Boost 库
DWITH_BOOST	boost路径
DCMAKE_INSTALL_PREFIX	Mysql安装路径

```
cd mysql-5.7.36-pc
sudo cmake . -DDOWNLOAD_BOOST=1 -DWITH_BOOST=/home/xiang/Mysql/mysql-5.7.36-pc/boost -DCMAKE_INSTALL_PREFIX=/home/xiang/Mysql/mysql-5.7.36-pc/lintall
 ls    //查看boost是否安装完成
```

编译

执行make安装命令

```
sudo make
sudo make install
```

通过file查看安装文件属性

```
cd ../mysql-5.7.36-pc/
ls
cd bin/
ls
file mysql
```

交叉编译mysql

下载源码

使用之前主机编译的源码，或者重新下载参考主机编译篇。

修改配置

进入Mysql解压的文件夹，修改CMakelists.txt。直接通过配置参数设置也是一样的，主要是为了设置ncurses、boost、openssl等依赖路径和编译器选择。

- 修改CMakelists.txt配置文件

```
vim CMakelists.txt
```

```
# this is required
SET(CMAKE_SYSTEM_NAME Linux)
SET(CMAKE_CROSSCOMPILING TRUE)

# specify the cross compiler
SET(CMAKE_C_COMPILER /home/xiang/ti-processor-sdk-linux-rt-am62xx-evm-08.06.00.42/linux-devkit/sysroots/x86_64-arago-linux/usr/bin/aarch64-none-linux-gnu-gcc)
SET(CMAKE_CXX_COMPILER /home/xiang/ti-processor-sdk-linux-rt-am62xx-evm-08.06.00.42/linux-devkit/sysroots/x86_64-arago-linux/usr/bin/aarch64-none-linux-gnu-g++)

# where is the target environment
SET(CMAKE_FIND_ROOT_PATH  /home/xiang/ti-processor-sdk-linux-rt-am62xx-evm-08.06.00.42/linux-devkit/sysroots/x86_64-arago-linux/usr/bin/aarch64-none-linux-gnu)

# search for programs in the build host directories (not necessary)
SET(CMAKE_FIND_ROOT_PATH_MODE_PROGRAM NEVER)
# for libraries and headers in the target directories
SET(CMAKE_FIND_ROOT_PATH_MODE_LIBRARY ONLY)
SET(CMAKE_FIND_ROOT_PATH_MODE_INCLUDE ONLY)
SET(CMAKE_FIND_ROOT_PATH_MODE_PACKAGE ONLY)

# configure Boost
SET(BOOST_ROOT /home/xiang/Mysql/boost_1_59_0-arm)
SET(BOOST_INCLUDE_DIR /home/xiang/Mysql/boost_1_59_0-arm/include)
SET(BOOST_LIBRARY_DIR /home/xiang/Mysql/boost_1_59_0-arm/lib)

# openssl configuration
SET(OPENSSL_INCLUDE_DIR /home/xiang/Mysql/openssl-3.2.0-arm/include)
SET(OPENSSL_LIBRARY /home/xiang/Mysql/openssl-3.2.0-arm/lib/libssl.so)
SET(CRYPTO_LIBRARY /home/xiang/Mysql/openssl-3.2.0-arm/lib/libcrypto.so)

SET(CMAKE_CXX_LINK_FLAGS "-L/home/xiang/Mysql/openssl-3.2.0-arm/lib -lssl -lcrypto")

```

- 设置配置参数

```
cmake . -LH -DCMAKE_INSTALL_PREFIX=/usr/xiang/mysql -
DMYSQL_DATADIR=/usr/local/arm/mysql/data -DDEFAULT_CHARSET=utf8 -
DDEFAULT_COLLATION=utf8_general_ci  -DEXTRA_CHARSETS=all -DENABLED_LOCAL_INFILE=1  -
DCMAKE_CXX_COMPILER=arm-linux-gnueabihf-g++ -DCMAKE_C_COMPILER=arm-linux-gnueabihf-gcc
-DWITH_BOOST=/home/velarn/work/boost/boost_1_59_0 -
DCURSES_INCLUDE_PATH=/home/velarn/work/ncurses/build/include/ncurses -
DCURSES_LIBRARY=/home/velarn/work/ncurses/build/lib/libncurses.a
```

编译

执行make并根据问题提示和常见问题解决

```
make
```

编译完成后执行安装命令，将编译文件安装到对应位置

```
make install
```

Mysql API

概述

Mysql使用可以通过sql命令也可以通过Mysql官方提供的Mysql API接口使用，以C语言API为例(其它的不会)介绍API下载和常用的API。在官网就就可以下载对应的系统的API压缩包

API下载

- Linux C API下载

  在官方下载页面中，选择需要的版本和Linux - Generic操作系统，选择对应位数的压缩包。

- Windos C API下载

  在官方下载页面中，选择需要的版本和Microsoft Windows操作系统，选择对应位数的压缩包。

- 其他语言API下载

- 注意 Mysql没有提供arm版本的库，需要自己交叉编译，arm库在交叉编译完成的文件夹lib目录中。

常见API介绍

名称	介绍
my_init()	初始化客户端
mysql_close()	关闭客户端连接
mysql_query()	执行sql语句

常见错误

Mysql安装(Windos 源码)

问题1

问题描述

执行命令:**mysqld -install**，出现Install of the Service Denied。

解决办法

用管理员权限执行CMD

参考文献

MySQL 5.7详细下载安装配置教程-CSDN博客

问题2

问题描述

执行命令：**net start mysql**时出现服务启动失败。

解决办法

我的问题是之前安装了Mysql8，由于Mysql安装时默认端口为3306，Mysql启动时端口被Mysql8占用了，使用taskkill关闭对应端口。

- 执行命令：**netstat -ano**寻找对应3306端口 (Mysql服务器启动默认端口为3306)

```
netstat -ano
```

- 执行命令：**tasklist|findstr 端口号对应的PID号**查询对应对应端口程序

```
tasklist|findstr 18192
```

- 执行命令：**taskkill /pid 端口号对应的PID号 /f**关闭对应端口

```
tasklist|findstr 18192
```

关闭占用端口程序后，重新启动就可以了。

参考文献

最详细的解决方法：MySQL服务无法启动。服务没有报告任何错误。端口问题，配置问题。_mysql服务无法启动没有任何错误-CSDN博客

问题3

问题描述

cmd提示无法识别mysql命令

解决办法

使用管理员身份打开cmd,前缀为C:\Windos\system32>

参考文献

无

移植Mysql

移植ncurses

问题1

问题描述

找不到编译器

解决办法

由于版本兼容问题引起的缺少环境。

```
sudo apt-get install libtool
cp /usr/share/libtool/build-aux/config.sub .
cp /usr/share/libtool/build-aux/config.guess .
```

参考文献

configure: error: cannot run /bin/sh ./config.sub解决办法

问题2

问题描述

_32053.c:835:15: error: expected ')' before 'int'

```
/* 错误描述 */
arm-linux-gnueabi-gcc -DHAVE_CONFIG_H -I../ncurses -I. -
I/home/charles/code/build_systemd/_install/include -D_GNU_SOURCE -
D_FILE_OFFSET_BITS=64  -DNDEBUG -I. -I../include -
I/home/charles/code/build_systemd/_install/include/ncurses -
I/home/charles/code/build_systemd/_install/include  --param max-inline-insns-
single=1200 -c ../ncurses/lib_gen.c -o ../objects/lib_gen.o
In file included from ../ncurses/curses.priv.h:283:0,
                 from ../ncurses/lib_gen.c:19:
_32053.c:835:15: error: expected ')' before 'int'
../include/curses.h:1594:56: note: in definition of macro 'mouse_trafo'
 #define mouse_trafo(y,x,to_screen) wmouse_trafo(stdscr,y,x,to_screen)
                                                        ^
Makefile:785: recipe for target '../objects/lib_gen.o' failed

```

解决办法

- 清空配置

```
export CPPFLAGS="-P"
./configure --host=aarch64-none-linux-gnu --prefix=/home/xiang/MySql/ncu --without-cxx-
binding
make
```

- 导出配置

```
export CPPFLAGS="-P"    //导出配置
./configure --host=aarch64-none-linux-gnu --prefix=/home/xiang/MySql/ncu --without-cxx-binding
make
```

参考文献

Ncurses 5.9 Compilation Error - error: expected ')' before 'int'

移植boost

问题1

问题描述

编译器路径错误

解决办法

重新在project-config.jam配置文件中写正确路径和编译器名

参考文献

无

移植openssl

问题1

问题描述

aarch64-linux-gnu-gcc: error: unrecognized command line option '-m64'

解决办法

删除(屏蔽)Makefile中第所有-m64选项,本版本共有两处使用了-m64.分别是3509行和3510行.

参考文献

openwrt 交叉编译 unrecognized command line option -m64 错误

问题2

问题描述

没有权限创建文件夹

解决办法

- 修改安装目录到当前用户组当前用户目录下
- 使用**sudo**提升权限

```
sudo make install
```

参考文献

无

问题3

问题描述

找不到编译器

解决办法

- 改变root环境变量

```
su root
vim etc/profile    /* 打开 /etc/profile 文件 需要打开一次改变时间戳 */
source /etc/profile
make install
```

- 修改到当前用户组当前用户目录

```
make clean
--prefix=xxx/xxx/xxx
make
make install
```

参考文献

../libtool:line XXXX:arm-linux-ranlib command not found

问题4

问题描述

ossl_prov_digest_reset 未定义

解决办法

无

参考文献

无

编译Mysql(主机Linux)

问题1

问题描述

编译时没有文件读写权限

```
/*Cmake 打开文件的读写权限 */
CMake Error: Cannot open file for write: /home/xiang/MySql/mysql-5.7.36/CMakeFiles/abi_check.dir/depend.make.tmp
CMake Error: : System Error: Permission denied
CMakeFiles/abi_check.dir/build.make:74: recipe for target 'CMakeFiles/abi_check.dir/depend' failed
make[2]: *** [CMakeFiles/abi_check.dir/depend] Error 2
CMakeFiles/Makefile2:99: recipe for target 'CMakeFiles/abi_check.dir/all' failed
make[1]: *** [CMakeFiles/abi_check.dir/all] Error 2
Makefile:162: recipe for target 'all' failed
make: *** [all] Error 2
```

解决办法

安装Mysql时会下载相关依赖,安装时应该使用超级用户权限

参考文献

无

问题2

问题描述

Cmake已存在

解决办法

编译Mysql时会自己生成Cmake且无法自动覆盖,所以每次配置前应该应该删除CMakeCache.txt.

```
rm CMakeCache.txt
make
```

参考文献

无

编译Mysql(arm)

问题1

问题描述

无法找到SSL库

解决办法

Mysql5.7.36的Cmake匹配openssl版本号规则和openssl3.2的版本规则不一致。Mysq5.7Cmake匹配openssl版本号规则为匹配OPENSSL_VERSION_NUMBER宏而openssl3.2版本宏定义为主版本号宏OPENSSL_VERSION_MAJOR、副版本宏OPENSSL_VERSION_MINOR、补丁宏OPENSSL_VERSION_PATCH。将openssl版本替换为openssl-1.1.1在进行编译。

- openssl-3.2 版本宏定义(opnessl-3.2/include/openssl/opensslv.h)

```
/*
 * Base version macros
 *
 * These macros express version number MAJOR.MINOR.PATCH exactly
 */
# define OPENSSL_VERSION_MAJOR  3    //主版本号
# define OPENSSL_VERSION_MINOR  2    //副版本号
# define OPENSSL_VERSION_PATCH  0    //补丁
... ...
```

- opnessl-1.1.1版本宏定义(opnessl-1.1.1/include/openssl/opensslv.h)

```
/*-
 * Numeric release version identifier:
 * MNNFFPPS: major minor fix patch status
 * The status nibble has one of the values 0 for development, 1 to e for betas
 * 1 to 14, and f for release.  The patch level is exactly that.
 * For example:
 * 0.9.3-dev      0x00903000
 * 0.9.3-beta1    0x00903001
 * 0.9.3-beta2-dev 0x00903002
 * 0.9.3-beta2    0x00903002 (same as ...beta2-dev)
 * 0.9.3          0x0090300f
 * 0.9.3a         0x0090301f
 * 0.9.4          0x0090400f
 * 1.2.3z         0x102031af
 *
 * For continuity reasons (because 0.9.5 is already out, and is coded
 * 0x00905100), between 0.9.5 and 0.9.6 the coding of the patch level
 * part is slightly different, by setting the highest bit.  This means
 * that 0.9.5a looks like this: 0x0090581f.  At 0.9.6, we can start
 * with 0x0090600S...
 *
 * (Prior to 0.9.3-dev a different scheme was used: 0.9.2b is 0x0922.)
 * (Prior to 0.9.5a beta1, a different scheme was used: MMNNFFRBB for
 *  major minor fix final patch/beta)
 */
# define OPENSSL_VERSION_NUMBER  0x1010100fL
# define OPENSSL_VERSION_TEXT    "OpenSSL 1.1.1  11 Sep 2018"
... ...
```

- Mysql5.7.36匹配openssl版本规则(Mysql3.2/cmake/ssl.cmake)

```
    IF(OPENSSL_INCLUDE_DIR)
    MESSAGE(STATUS "this is ${OPENSSL_INCLUDE_DIR}")
      # Verify version number. Version information looks like:
      #   #define OPENSSL_VERSION_NUMBER 0x1000103fL
      # Encoded as MNNFFPPS: major minor fix patch status
      FILE(STRINGS "${OPENSSL_INCLUDE_DIR}/openssl/opensslv.h"
        OPENSSL_VERSION_NUMBER
        REGEX "^#[ ]*define[\t ]+OPENSSL_VERSION_NUMBER[\t ]+0x[0-9].*"
        )
      STRING(REGEX REPLACE
        "^.*OPENSSL_VERSION_NUMBER[\t ]+0x([0-9]).*$" "\\1"
        OPENSSL_MAJOR_VERSION "${OPENSSL_VERSION_NUMBER}"
        )
      STRING(REGEX REPLACE
        "^.*OPENSSL_VERSION_NUMBER[\t ]+0x[0-9]([0-9][0-9]).*$" "\\1"
        OPENSSL_MINOR_VERSION "${OPENSSL_VERSION_NUMBER}"
        )
      STRING(REGEX REPLACE
        "^.*OPENSSL_VERSION_NUMBER[\t ]+0x[0-9][0-9][0-9]([0-9][0-9]).*$" "\\1"
        OPENSSL_FIX_VERSION "${OPENSSL_VERSION_NUMBER}"
        )
    ENDIF()
```

参考文献

无

问题2 [11%]

问题描述

无法运行extra/comp_err脚本无法生成comp_err.h文件

解决办法

复制之前编译的x86主机的mysql/extra/comp_err文件到现在的extra目录中。

```
cp comp_err ../../mysql-5.7.36-arm/extra/
ls ../../mysql-5.7.36-arm/extra/comp_err
```

参考文献

忘记了 CSDN上某一篇文章

问题3 [11%]

问题描述

无法运行libmysql/libmysql_api_test脚本无法生成libmysql_api_test.h文件

解决办法

复制之前编译的x86主机的libmysql/libmysql_api_test文件到现在的libmysql目录中。

```
cp ./libmysql/libmysql_api_test ../mysql-5.7.36-arm/libmysql
ls ../mysql-5.7.36-arm/libmysql/libmysql_api_test
```

参考文献

忘记了 CSDN上某一篇文章

问题4 [41%]

问题描述

无法运行scripts/commp_sql脚本无法生成commp_sql.h文件

解决办法

复制之前编译的x86主机的scripts/commp_sql文件到现在的scripts目录中。

```
cp scripts/comp_sql ../mysql-5.7.36-arm/scripts/
ls -c ../mysql-5.7.36-arm/scripts/comp_sql
```

参考文献

忘记了 CSDN上某一篇文章

问题5 [41%]

问题描述

无法运行sql/gen_lex_hash脚本无法生成gen_lex_hash.h文件

解决办法

复制之前编译的x86主机的sql/gen_lex_hash文件到现在的sql目录中。

参考文献

忘记了 CSDN上某一篇文章

问题6 [41%]

问题描述

无法运行sql/gen_lex_token脚本无法生成gen_lex_token.h文件

解决办法

复制之前编译的x86主机的sql/gen_lex_token文件到现在的sql目录中。

```
cp ./sql/gen_lex_token ../mysql-5.7.36-arm/sql/
ls -c ../mysql-5.7.36-arm/sql/gen_lex_token
```

参考文献

忘记了 CSDN上某一篇文章

问题7 [41%]

问题描述

arm编译 os0atomic.ic 、os0atomic.h 中没有 HAVE_IB_GCC_ATOMIC_COMPARE_EXCHANGE 与 IB_STRONG_MEMORY_MODEL 这两个宏定义。

解决办法

- 修改os0atomic.h 在#define IB_STRONG_MEMORY_MODEL后添加宏HAVE_ATOMIC_BUILTINS

```
///home/xiang/Mysql/mysql-5.7.36-arm/storage/innobase/include/os0atomic.h
#if defined __i386__ || defined __x86_64__ || defined _M_IX86 \
    || defined _M_X64 || defined __WIN__

#define IB_STRONG_MEMORY_MODEL

#else

#define HAVE_ATOMIC_BUILTINS

#endif /* __i386__ || __x86_64__ || _M_IX86 || _M_X64 || __WIN__ */
```

- 修改os0atomic.ic 将#elif defined(IB_STRONG_MEMORY_MODEL)改为#elif defined(HAVE_ATOMIC_BUILTINS)

```
///home/xiang/Mysql/mysql-5.7.36-arm/storage/innobase/include/os0atomic.ic
#elif defined(HAVE_ATOMIC_BUILTINS)    //198行
```

参考文献

忘记了 某一论坛上的帖子

问题8 [69%]

问题描述

无法运行extra/protobuf/protoc脚本无法生成protoc.h文件

解决办法

复制之前编译的x86主机的extra/protobuf/protoc文件到现在的extra/protobuf目录中。

```
cp extra/protobuf/protoc ../mysql-5.7.36-arm/extra/protobuf
ls -c ../mysql-5.7.36-arm/extra/protobuf/protoc
```

参考文献

忘记了 CSDN上某一篇文章