Ubuntu 16.04 cross compile mysql 5.7.16 to arm platform

Topping November 12, 2016 00:45:27
Reading number: 5013
Ubuntu 16.04 cross compile mysql 5.7.16 to arm platform

1 Platform:
(1) ubuntu 16.04 32bit
(2) arm: 210 arm cortex-A8
(3) mysql 5.7.16 32bit

2 Prepare two sets of mysql source code, decompress mysql-5.7.16.tar.gz, copy two copies of mysql-arm, mysql-x86, and source file directories respectively in ubuntu #/opt directory.
(1) mysql-arm (arm-linux-gcc of arm platform, compilation of arm-linux-g++)
(2) mysql-x86 (gcc for x86 linux platform, g++ compilation)

3 tools and libraries used
1.cmake-3.5.1.tar.gz --- The new version of MySQL is compiled and installed with CMAKE
2.ncurses-5.9.tar.gz --- Need to install MySQL http://ftp.gnu.org/gnu/ncurses/
3.bison-3.0.4.tar.gz --- need to install MySQL
4.boost_1_59_0.tar.gz ---Requires http://www.boost.org/users/news/ when installing MySQL

Starting from MySQL 5.7.5, the Boost library is required. Download the Boost library and upload it to /usr/local/src. After unzipping, copy it to the /usr/local/boost directory, then re-cmake and add it in the options that follow. Upper option -
DWITH_BOOST=/usr/local/boost
5. git
There is an error when installing git without compiling. Could NOT find Git (missing: GIT_EXECUTABLE)
6. mysql-5.7.16.tar.gz

/** *************/
An installation related tool and inventory first $ sudo apt-get update
1. Install cmake. Build a cross-platform installation (compilation tool), the new version of MySQL is compiled with CMAKE $ apt-get install cmake -y

2. Install the C/C++ compiler
$ sudo apt-get install gcc g++ -y (generally we use UBUNTU comes with)

3. Install the cross tool chain
3.1 Installing arm-linux-gcc 4.4.3
(1) Decompress $ tar zxvf arm-linux-gcc-4.4.3.tar.gz
(2) Copy the 4.4.3 directory to the /usr/local/arm directory $ cp 4.4.3 /usr/local/arm -radf
3.2 Configure the environment variables,
(1) Add export PATH=/usr/local/arm/4.4.1/bin:$PATH
$ vi /etc/bash.bashrc
(2) Make bash.bashrc effective $ source /etc/bash.bashrc
(3) Check if the /usr/local/arm/4.4.1/bin path is included. $ echo $PATH
(3) Test can see the version number until $ arm-linux-gcc -v

4. Install git
$ apt-get install git -y

5. Install LINUX common graphics library $ apt-get install libncurses5 libncurses5-dev -y

6 Install boost,
(1). Extract the boost_1_59_0
$ tar zxvf boost_1_59_0.tar.gz
(2). Enter the boost_1_59_0 directory $ cd /boost_1_59_0
(3). Run the script and configure boost
$ ./bootstrap.sh
(4). Compile and install, will be installed to /usr/local/include/boost directory $./b2 install

Second: compile .mysql-x86, (here is the mysql-x86 directory x86 platform gcc, g++ compiler)
(1) Enter the BUILD directory $ cd /root/mysql-x86/BUILD
(2) Compile, such as 64-bit ./compile-pentium64
$ ./compile-pentium

```
//---------------
...
[100%] Linking CXX executable my_safe_process
[100%] Built target my_safe_process
The compilation is successful and is reserved for backup.
//---------------
```

Error and resolution
1 Error when executing ./compile-pentium
/***
*** **
Error when compiling with other versions of boost

$./compile-pentium

```
-- Running cmake version 3.5.1
-- Found Git: /usr/bin/git (found version "2.7.4")
-- Configuring with MAX_INDEXES = 64U
-- SIZEOF_VOIDP 4
-- MySQL 5.7.16 //[MySQL version]
-- Packaging as: mysql-5.7.16-Linux-i686
-- Found /usr/local/include/boost/version.hpp
-- BOOST_VERSION_NUMBER is #define BOOST_VERSION 106200
CMake Warning at cmake/boost.cmake:266 (MESSAGE):
Boost minor version found is 62 we need 59 //boost version is 62, what is needed is 59
Call Stack (most recent call first):
CMakeLists.txt: 455 (INCLUDE)
-- BOOST_INCLUDE_DIR /usr/local/include
-- LOCAL_BOOST_DIR
-- LOCAL_BOOST_ZIP
-- Could not find (the correct version of) boost. //[key error message]
-- MySQL currently requires boost_1_59_0 //[Workaround] MySQL currently needs boost_1_59_0
CMake Error at cmake/boost.cmake:81 (MESSAGE): //[specific errors and solutions]
You can download it with -DDOWNLOAD_BOOST=1 -DWITH_BOOST=<directory>
This CMake script will look for boost in <directory>. If it is not there,
It will download and unpack it (in that directory) for you.
If you are inside a firewall, you may need to use an http proxy:
Export http_proxy=http://example.com:80
Call Stack (most recent call first):
Cmake/boost.cmake:269 (COULD_NOT_FIND_BOOST)
CMakeLists.txt: 455 (INCLUDE)
-- Configuring incomplete, errors occurred!
See also "/root/mysql-x86/BUILD/CMakeFiles/CMakeOutput.log".
Make: *** No targets specified and no makefile found. Stop.
```

```
********************************************* ************************************/
./compile-pentium
-- Running cmake version 3.5.1
-- Configuring with MAX_INDEXES = 64U
-- SIZEOF_VOIDP 4
-- MySQL 5.7.16
-- Packaging as: mysql-5.7.16-Linux-i686
-- Found /usr/local/include/boost/version.hpp
-- BOOST_VERSION_NUMBER is #define BOOST_VERSION 105900
-- BOOST_INCLUDE_DIR /usr/local/include
-- Could NOT find Curses (missing: CURSES_LIBRARY CURSES_INCLUDE_PATH) //curses is a graphical
function library widely used under Linux/Unix. It can be used to draw user interface and beautiful graphics under DOS.
// Did not find curses (missing CURSES_LIBRARY library, CURSES_INCLUDE_PATH header file path)
Solution:
Yum -y install ncurses-devel
Rm -f CMakeCache.txt
```

CMake Error at cmake/readline.cmake:64 (MESSAGE): //The reason for the error is that ncurses-devel is not installed.
Curses library not found. Please install appropriate package,

Remove CMakeCache.txt and rerun cmake.On Debian/Ubuntu, package name is libncurses5-dev, on Redhat and derivates it is ncurses-devel.
Call Stack (most recent call first):
Cmake/readline.cmake:107 (FIND_CURSES)
Cmake/readline.cmake:197 (MYSQL_USE_BUNDLED_EDITLINE)
CMakeLists.txt:483 (MYSQL_CHECK_EDITLINE)

-- Configuring incomplete, errors occurred! //The configuration is incomplete and an error occurs!
See also "/root/mysql-x86/BUILD/CMakeFiles/CMakeOutput.log". //See
See also "/root/mysql-x86/BUILD/CMakeFiles/CMakeError.log".
Make: *** No targets specified and no makefile found. Stop.
** *********************************/

Three. Cross compilation ncuses
(1) Decompress $ tar xvf ncurses-5.9.tar.gz
(2) Enter the directory $ cd ncurses-5.9
(3) Configuration $ CC=arm-linux-gcc CXX=arm-linux-g++ ./configure --host=arm-linux-gnu --prefix=/usr
(4) Compile make
$ make
(5) Install make install
$ make install DESTDIR=/usr/local/arm/libncurses
Normal should be very smooth.

Four cross-compilation myslq, (here is the mysql-arm directory arm platform arm-linux-gcc, arm-linux-g++ compilation)
Basic process
Take the Linux platform as an example. The process of generating a Makefile using CMake and compiling is as follows:
Write the CMake configuration file CMakeLists.txt.
Run the cmake Path-to-Cmakelist/CMakeLists.txt command to generate a Makefile.
Compile with the make command.

(1) Copy the comp_err in /root/mysql-x86/BUILD/extra to the /usr/bin directory.
(2) Configure another directory mysql source code /root/mysql-arm/CMakeLists.txt

Find "IF(WIN32)" at the beginning of /root/mysql-arm/CMakeLists.txt and add the following
#== ========

```
#1 tells the current use of cross-compilation mode, must be configured
SET (CMAKE_SYSTEM_NAME Linux)


#2 Specify C cross compiler, must be configured, or cross compiler to use absolute address
SET(CMAKE_C_COMPILER "arm-linux-gcc")
SET(CMAKE_CXX_COMPILER "arm-linux-g++")


SET(CMAKE_SYSTEM_PROCESSOR arm)
SET(CMAKE_SYSTEM_VERSION 1)
SET(CMAKE_CROSSCOMPILING 1)


#Add header file search path, add according to the actual path of the machine
INCLUDE_DIRECTORIES(/usr/local/include)
Set(BOOST_INCLUDE_DIR /usr/local/include)
Set(LOCAL_BOOST_DIR /usr/local/include/boost)


#Add library file search path, add according to the actual path of the machine
SET(CMAKE_LIBRARY_PATH /usr/local/arm/libncurses/usr/lib)
Set(BOOST_LIBRARY_DIR /usr/local/lib)
LINK_DIRECTORIES(/usr/local/lib)


SET(STACK_DIRECTION 1)
SET(WITH_UNIT_TESTS OFF)
SET(WITH_EMBEDDED_SERVER TRUE)


#1 Specify the cross-compiler environment installation directory... (compiler environment path)
SET(CMAKE_FIND_ROOT_PATH /usr/local/arm/4.4.3)


#2 Never find a utility under the specified directory
SET(CMAKE_FIND_ROOT_PATH_MODE_PROGRAM NEVER)


#3 Find library files only in the specified directory
SET(CMAKE_FIND_ROOT_PATH_MODE_LIBRARY ONLY)


#4 Find header files only in the specified directory
SET(CMAKE_FIND_ROOT_PATH_MODE_INCLUDE ONLY)


/ / Put the above content below IF (WIN32) ==============================================
=========
```

Description:
Three main settings

```
#1 tells the current use of cross-compilation mode, must be configured
SET (CMAKE_SYSTEM_NAME Linux)

#2 Specify C cross compiler, must be configured
#或交叉编译器 uses an absolute address
SET (CMAKE_C_COMPILER "arm-linux-gcc") or SET (CMAKE_C_COMPILER
${TOOLCHAIN_DIR}/usr/local/arm/4.4.3)
#3 Specify C++ Cross Compiler
SET (CMAKE_CXX_COMPILER arm-linux-g++) or SET (CMAKE_CXX_COMPILER
${TOOLCHAIN_DIR}/bin/arm_v5t_le-g++)
#1 ----------------------------------------------
#1 Before setting cross-compilation, you must add such a sentence before CMakeList.txt, so that CMake will think that
you want to cross-compile:
SET (CMAKE_SYSTEM_NAME Linux)


SET(CMAKE_SYSTEM_PROCESSOR arm)
SET(CMAKE_SYSTEM_VERSION 1)
SET(CMAKE_CROSSCOMPILING 1)


#2 Specify C cross compiler, must be configured
#或交叉编译器 uses an absolute address
SET(CMAKE_C_COMPILER "arm-linux-gcc")


#+++++
SET(TOOLCHAIN_DIR "/opt/mv_pro_4.0/montavista/pro/devkit/arm/v5t_le")


#Add header file search path
SET(CMAKE_LIBRARY_PATH /usr/local/arm/ncurses/usr/lib)
INCLUDE_DIRECTORIES(/usr/local/include)
Set(BOOST_INCLUDE_DIR /usr/local/include)
Set(BOOST_LIBRARY_DIR /usr/local/lib)
Set(LOCAL_BOOST_DIR /usr/local/boost)


SET(STACK_DIRECTION 1)
SET(WITH_UNIT_TESTS OFF)
SET(WITH_EMBEDDED_SERVER TRUE)


#Add library file search path
LINK_DIRECTORIES(/usr/local/lib)
```

```
#------------------------------------------------ --
# After telling CMake to cross-compile, tell CMake which path to go to find the library file.
# Because CMake does not automatically go to the system default directory to find library files and header files during cross-compilation:
# The first line is what tells CMake what the root directory is. The latter is to tell CMake how
# Find the location of the utility program at compile time, the location of the library, and the location of the header files. Set to NEVER to indicate no lookup,
#Set to ONLY means to find only in the directory set by CMAKE_FIND_ROOT_PATH, set to BOTH (this is the default option)
# indicates that it can be found in the system directory or in the CMAKE_FIND_ROOT_PATH. Because we are cross-compiled,
# So the last two settings ONLY, for the compile-time call tool, generally need to look in the system directory,
#However, I don't need to set it to NEVER.


#1 Specify the cross-compiler environment installation directory... (compiler environment path)
SET(CMAKE_FIND_ROOT_PATH "/usr/local/arm/4.4.3")


#2 Never find a utility under the specified directory
SET(CMAKE_FIND_ROOT_PATH_MODE_PROGRAM NEVER)


#3 Find library files only in the specified directory
SET(CMAKE_FIND_ROOT_PATH_MODE_LIBRARY ONLY)


#4 Find header files only in the specified directory
SET(CMAKE_FIND_ROOT_PATH_MODE_INCLUDE ONLY)
#上段This is a modification of /root/mysql-arm/CMakeLists.txt.
#----------------------------------------------- ---------------
```

(3) In the /root/mysql-arm directory: execute the following command:

```
$ cmake -DCMAKE_INSTALL_PREFIX=/usr/local/arm/mysql \
-DMYSQL_UNIX_ADDR=/usr/local/arm/tmp/mysql.sock \
-DDEFAULT_CHARSET=utf8 \
-DDEFAULT_COLLATION=utf8_general_ci \
-DWITH_EXTRA_CHARSETS: STRING=utf8, gbk \
-DWITH_INNOBASE_STORAGE_ENGINE=1 \
-DWITH_READLINE=1 \
-DENABLED_LOCAL_INFILE=1 \
-DMYSQL_DATADIR=/usr/local/arm/mysql/data/ \
-DSTACK_DIRECTION=1 \
-DCURSES_LIBRARY=/usr/local/arm/libncurses/usr/lib/libncurses.a \
-DCURSES_INCLUDE_PATH=/usr/local/arm/libncurses/usr/include \
-DWITH_MYISAM_STORAGE_ENGINE=1
```

/*** ******************************
When using the source code installation, CMAKE has a lot of parameters. Here are some explanations for some parameters.
Commonly used parameters:
● (Common) -DCMAKE_INSTALL_PREFIX: Used to specify the installation path of the software. The default is to install to the /usr/local/mysql directory. It doesn't matter if the path is not suitable after compiling and installing. As long as the MySQL process is not started, you can modify this directory at any time. The name and storage path.
● -DDEFAULT_CHARSET: specifies the default character set of the MySQL service. The default value of this parameter is latin1. The character set that MySQL can support is very much. For details, please refer to the value of the SET (CHARSETS_AVAILABLE) variable in the cmake/character_sets.cmake file in the MySQL source directory. This option can also be set by the character_set_server parameter when the MySQL service starts.
● -DDEFAULT_COLLATION: Specifies the default collation rules for the MySQL service. The default value of this parameter is latin1 swedish_ ci. This option can also be set by the collation_server parameter when the MySQL service starts.
● -DENABLED_LOCAL_INFILE: Does not allow data to be loaded locally from the client to the MySQL server, dedicated to the LOAD DATA INFILE statement, which is not allowed by default.
● -DENABLED_PROFILING: Whether to enable query profiling, dedicated to SHOW PROFILE and SHOW PROFILES statements, is enabled by default.
● (Common) -DMYSQL_DATADIR: Specifies the storage path of the MySQL database data file. This option can be set by the datadir parameter when the MySQL service starts.
● -DSYSCONFDIR: Specifies the default path of the MySQL parameter file. This option can be set by the defaults-file parameter when the MySQL service starts.
● -DWITH_<ENGINE>_STORAGE_ENGINE: Statically compile a storage engine. The optional storage engine keywords are ARCHIVE, BLACKHOLE, EXAMPLE, FEDERATED, INNOBASE, PARTITION, PERFSCHEMA. In fact, the storage engine supported by MySQL is more than this, but like MyISAM, MERGE. The four storage engines, MEMORY and CSV are compiled to the server by default, no need to specify. In addition, some of the keywords listed above are not all storage engines. For example, PARTITION refers to whether to allow partitions, and PERFSCHEMA is a Performance_schema library.
Such as:
-DWITH_INNOBASE_STORAGE_ENGINE=1
-DWITH_ARCHIVE_STORAGE_ENGINE=1
-DWITH_BLACKHOLE_STORAGE_ENGINE=1

● -DWITHOUT_<ENGINE>_STORAGE_ENGINE: Contrary to the function of the previous parameter, this parameter is used to specify the storage engine that is not compiled. For example, when you don't need to compile the example storage engine, you can specify -DWITHOUT_EXAMPLE_STORAGE_ENGINE=1.
Such as:
-DWITHOUT_EXAMPLE_STORAGE_ENGINE=1
-DWITHOUT_FEDERATED_STORAGE_ENGINE=1
-DWITHOUT_PARTITION_STORAGE_ENGINE=1

● -DWITH_EXTRA_CHARSETS: Specifies the additional supported character set. The default is all, which is all.

The parameters that may be used are as follows:
● -DINSTALL_BINDIR: Specifies the storage path of MySQL commands. The default is in the

CMAKE_INSTALL_PREFIX/bin directory.
● -DINSTALL_DOCDIR: Specifies the storage path of the MySQL document. The default is in the CMAKE_INSTALL_PREFIX/docs directory.
● -DINSTALL_INCLUDEDIR: specifies the storage path of the header file. The default is saved in the CMAKE_INSTALL_PREFIX/include directory.
● -DINSTALL_LIBDIR: Specifies the storage path of the linked file. The default is saved in the CMAKE_INSTALL_PREFIX/lib directory.
● -DINSTALL_MANDIR: Specify the storage path of the user manual. The default is to save it in the CMAKE_INSTALL_PREFIX/man directory.
● -DINSTALL_PLUGINDIR: Specifies the storage path of the Plugin. The default is saved in the CMAKE_INSTALL_PREFIX/lib/plugin directory.
● -DINSTALL_SBINDIR: Specifies the storage path of the server execution script. The default is saved in the CMAKE_INSTALL_PREFIX/bin directory.
● -DINSTALL_SCRIPTDIR: Specify the storage path of the MySQL_install_db script that comes with MySQL. The default is to save it in the CMAKE_INSTALL_PREFIX/scripts directory.
● -DINSTALL_SHAREDIR=share points to the aclocal/mysql.m4 installation directory (prefix/share)
● -DINSTALL_INFODIR=share/info points to the info file storage directory (prefix/share/info)
● -DINSTALL_SQLBENCHDIR: Specify the storage path of sql-bench. The default is to save it in the CMAKE_INSTALL_PREFIX directory.

● -DINSTALL_SUPPORTFILESDIR: Specifies the storage path of the additional support class files that come with MySQL. The default is to save it in the CMAKE_INSTALL_PREFIX/support-files directory.
● -DMYSQL_TCP_PORT: Specifies the TCP/IP port that the MySQL database provides. The default is 3306. This option can be set by the port parameter when the MySQL service starts.
● -DMYSQL_UNIX_ADDR: Specifies the storage path of the socket file. The default is in the /tmp/mysql.sock directory. This option can be set by the socket parameter when the MySQL service starts.
● -DWITH_COMMENT: Specifies the compilation information. This parameter is valid in 5.1 and earlier versions. It is invalid for 5.5 and later. If you need to specify the compilation information, you can use the -DCOMPILATION_COMMENT parameter instead.
● -DWITH_READLINE: Specify the processing method of input and output. It does not need to be processed separately in 5.1 and before. The default is to use the readline method. However, after entering version 5.5, MySQL compiles the default input and output using libedit, which may cause the current environment to log in to the mysql command. After the line mode, Chinese cannot be input (only for the current compilation environment, other clients are not affected), so you need to specify it to be readline when compiling.
prompt:
Regarding the -DWITH READLINE parameter, in the 5.6.10 version, even if -DWITH_READLINE is specified, libedit is still used to process the input and output by default. In this case, even if the installation is successful, the whole process is not reported, but you may encounter a "Segmentation fault" when you input Chinese. "Error message, the local connection mysql command line error aborted, the official will be defined as BUG, details can refer to http://bugs.mysql.com/bug.php?id=68231.

● -DWITH_SSL=system Enable ssl library support (Secure Sockets Layer)
● -DWITH_ZLIB=system Enable libz library support (zib, gzib related)
● -DWTIH_LIBWRAP=0 disables the libwrap library (implementing the functionality of the generic TCP wrapper for the web service daemon)
● -DWITH_EMBEDDED_SERVER=1 Compile embedded server support
● -DMYSQL_USER=mysql specifies mysql user (default is mysql)

● -DWITH_DEBUG=0 disable debug (default is disabled)
● -DENABLE_PROFILING=0 disables profiling analysis (default is on)
● -DWITH_COMMENT='string' A descriptive comment about the build environment
** ********************************/

If there is an error, follow the instructions in the execution process to modify it. Execute the following command to delete CMakeCache.txt before each modification.
Rm /root/mysql-arm/CMakeCache.txt
Compile into attack to generate Makefile (4) make
During the make process, there are several problems. One is that some intptr types have to be changed to be long types! The second problem is that the file format is incorrect. If ncurses is installed incorrectly, this problem will be reported.

(5) make install