编译MongoDB C Driver

使用cmake和vs可以生成mongoc的驱动dll，vs仅支持2010及后续版本。其中vs2013+1.2.1由于编码问题可能会修改一些文件后才能正常编译，vs2015+1.6.2不存在该问题。生成的include目录中libbson和mongoc是分目录存放的，可以合并到同一目录下，方便使用。

# VS2015 x64

## 编译环境

cmake-3.8.0-win64-x64

mongo-c-driver-1.6.2.tar.gz

win7 x64 sp1

vs2015 w update 1

2017.4.21

## 步骤

参考 <http://mongoc.org/libmongoc/current/installing.html>，未做任何调整即可编译成功。默认生成的是debug版，需要debug的vc库。

Building on Windows requires Windows Vista or newer and Visual Studio 2010 or newer. Additionally, cmake is required to generate Visual Studio project files.

Let's start by generating Visual Studio project files for libbson, a dependency of the C driver. The following assumes we are compiling for 64-bit Windows using Visual Studio 2015 Express, which can be freely downloaded from Microsoft.

cd mongo-c-driver-1.6.2\src\libbson

cmake -G "Visual Studio 14 2015 Win64" "-DCMAKE\_INSTALL\_PREFIX=C:\mongo-c-driver"

(Run cmake -LH . for a list of other options.)

Now that we have project files generated, we can either open the project in Visual Studio or compile from the command line. Let's build using the command line program msbuild.exe

msbuild.exe ALL\_BUILD.vcxproj

Now that libbson is compiled, let's install it using msbuild. It will be installed to the path specified by CMAKE\_INSTALL\_PREFIX.

msbuild.exe INSTALL.vcxproj

You should now see libbson installed in C:\mongo-c-driver

Now let's do the same for the MongoDB C driver.

cd mongo-c-driver-1.6.2

cmake -G "Visual Studio 14 2015 Win64" \

"-DCMAKE\_INSTALL\_PREFIX=C:\mongo-c-driver" \

"-DBSON\_ROOT\_DIR=C:\mongo-c-driver"

msbuild.exe ALL\_BUILD.vcxproj

msbuild.exe INSTALL.vcxproj

All of the MongoDB C Driver's components will now be found in C:\mongo-c-driver.

Native TLS Support on Windows (Secure Channel)

The MongoDB C Driver supports the Windows native TLS and crypto libraries. Using the native libraries there is no need to install OpenSSL. By default however, the driver will compile against OpenSSL if it detects it being available. If OpenSSL is not available, it will fallback on the native libraries.

To compile against the Windows native TLS and crypto libraries, even when OpenSSL is available, configure the driver like so:

cmake -G "Visual Studio 14 2015 Win64" \

"-DENABLE\_SSL=WINDOWS" \

"-DCMAKE\_INSTALL\_PREFIX=C:\mongo-c-driver" \

"-DBSON\_ROOT\_DIR=C:\mongo-c-driver"

## 调试信息和符号表

使用vs2015打开生成的解决方案文件，选择相应配置，直接生成install项目即可。建议选择RelWithDebInfo。如果是本机编译的，并且将生成的dll和pdb放入其他测试程序中，可以直接跟入相应Mongoc函数，跳转到驱动的源代码中。

$(mongocdrv)src\libbson\libbson.sln

$(mongocdrv)\libmongoc.sln

# VS2013 x64

## 编译环境

cmake-3.3.2-win32-x86

mongo-c-driver-1.2.0.tar

win7 x64 sp1

VS2013 w update 4

## 步骤

参照$(mongocdrv)\readme

### 环境准备

解压cmake到

C:\lib\mongocdrv\cmake-3.3.2-win32-x86

解压mongo-c-driver到

C:\lib\mongo-c-driver-1.2.0

启动vs命令行，添加path路径

path=C:\lib\mongocdrv\cmake-3.3.2-win32-x86\bin;%path%

运行cmake

cmake –help查看参数

### 修改文件

C:\lib\mongo-c-driver-1.2.0\CMakeLists.txt

删除第16行

include(FindSASL2)

推测是SSL/SASL支持。不需加密时可以忽略。否则编译时会报错，未找到其他解决方案。

$(mongo-c-driver)\src\libbson\tests

test-bson.c, test-json.c

两个文件含有特殊编码，使用VS另存为含有bom头的unicode文件(unicode – 代码页1200)

### 编译libbson

生成vs解决方案文件：在命令行下

cd C:\lib\mongo-c-driver-1.2.0\src\libbson

cmake -G " Visual Studio 12 2013 Win64"

使用VS2013打开libbson.sln并编译。或使用msbuild编译vcxproj文件

如果提示编译错误，可能是因为.c文件编码格式问题。将出问题的文件另存为（编码保存）unicode – 代码页1200重试。一般可以解决。

### 编译driver

生成解决方案

cd C:\lib\mongo-c-driver-1.2.0

cmake –G " Visual Studio 12 Win64"

### 完整编译命令：

path=C:\lib\mongocdrv\cmake-3.3.2-win32-x86\bin;%path%

cd C:\lib\mongo-c-driver-1.2.0

cd src\libbson

cmake -DCMAKE\_INSTALL\_PREFIX=C:\lib\mongoc -G "Visual Studio 12 Win64"

msbuild.exe ALL\_BUILD.vcxproj

msbuild.exe INSTALL.vcxproj

cd ..\..

cmake -DCMAKE\_INSTALL\_PREFIX=C:\lib\mongoc -DBSON\_ROOT\_DIR=C:\lib\mongoc -G "Visual Studio 12 Win64"

msbuild.exe ALL\_BUILD.vcxproj

msbuild.exe INSTALL.vcxproj

## 备注

官方说明中，1.2.0的C驱动仅支持VISTAL + VS2010以后的windows。

VS2008无法静态链接vs2013的lib库，因为VS2012及以后的版本使用了\_\_report\_rangecheckfailure检查堆栈缓存溢出，该函数在之前版本的VC运行库中不存在。

可以使用动态链接lib的方法解决。

#pragma comment(lib, "bson-1.0.lib")

#pragma comment(lib, "mongoc-1.0.lib")

## 使用

### 新建VS项目

VS2013,新建win32控制台项目，使用MFC

配置管理器，新建x64配置（从win32复制）

### 添加include目录

C:\lib\mongoc\include\libbson-1.0

C:\lib\mongoc\include\libmongoc-1.0

### 添加lib目录

C:\lib\mongoc\lib

### 测试代码

参考cmake生成的示例项目example-client.vcxproj

C:\lib\mongo-c-driver-1.2.0\examples\example-client.c

Debug, x64, 无内存泄露

#include <mongoc.h>

#include <stdio.h>

#include <stdlib.h>

#pragma comment(lib, "bson-static-1.0.lib")

#pragma comment(lib, "mongoc-static-1.0.lib")

mongoc\_client\_t \*client;

mongoc\_collection\_t \*collection;

mongoc\_cursor\_t \*cursor;

bson\_error\_t error;

const bson\_t \*doc;

const char \*uristr = "mongodb://192.168.1.120/";

const char \*collection\_name = "TRecord";

bson\_t query;

char \*str;

mongoc\_init();

client = mongoc\_client\_new(uristr);

if (!client) {

fprintf(stderr, "Failed to parse URI.\n");

return EXIT\_FAILURE;

}

bson\_init(&query);

bool bRet = 0;

// append condition, empty for all

// bRet = bson\_append\_int32(&query, "w\_nProcessModeBits", -1, 127);

// bRet = bson\_append\_int32(&query, "w\_nAudioEncodingTag", -1, 1);

#if 0

bson\_append\_utf8(&query, "hello", -1, "world", -1);

#endif

collection = mongoc\_client\_get\_collection(client, "DrVoice", "TVprObject");

cursor = mongoc\_collection\_find(collection,

MONGOC\_QUERY\_NONE,

0,

0,

0,

&query,

NULL, /\* Fields, NULL for all. \*/

NULL); /\* Read Prefs, NULL for default \*/

while (mongoc\_cursor\_next(cursor, &doc)) {

const bson\_t\* b = doc;

bson\_iter\_t iter;

// get all fileld name

if (bson\_iter\_init(&iter, b))

{

while (bson\_iter\_next(&iter))

{

cout << bson\_iter\_key(&iter) << endl;

}

}

bson\_iter\_t val;

bson\_iter\_t child;

const char\* name;

uint32\_t u8len = 0;

if (bson\_iter\_init(&iter, b))

{

name = "\_id";

if (bson\_iter\_find\_descendant(&iter, name, &val) &&

BSON\_ITER\_HOLDS\_UTF8(&val))

{

cout << name << " : " << bson\_iter\_utf8(&val, &u8len) << endl;

}

name = "w\_strVprId";

if (bson\_iter\_find\_descendant(&iter, name, &val) &&

BSON\_ITER\_HOLDS\_UTF8(&val))

{

cout << name << " : " << bson\_iter\_utf8(&val, &u8len) << endl;

}

name = "w\_bIsHighTh";

if (bson\_iter\_find\_descendant(&iter, name, &val) &&

BSON\_ITER\_HOLDS\_BOOL(&val))

{

cout << name << " : " << bson\_iter\_bool(&val) << endl;

}

name = "w\_fThreshold";

if (bson\_iter\_find\_descendant(&iter, name, &val) &&

BSON\_ITER\_HOLDS\_DOUBLE(&val))

{

cout << name << " : " << bson\_iter\_double(&val) << endl;

}

name = "w\_nVprState";

if (bson\_iter\_find\_descendant(&iter, name, &val) &&

BSON\_ITER\_HOLDS\_INT32(&val))

{

cout << name << " : " << bson\_iter\_int32(&val) << endl;

}

// name = "subdoc";

// if (bson\_iter\_find\_descendant(&iter, name, &val) &&

// BSON\_ITER\_HOLDS\_DOCUMENT(&val) &&

// bson\_iter\_recurse(&val, &child))

// {

// cout << name << " : " << endl;

// while (bson\_iter\_next(&child))

// {

// // PROC DOC

// }

// cout << endl;

// }

name = "w\_vtPosFileNames";

if (bson\_iter\_find\_descendant(&iter, name, &val) &&

BSON\_ITER\_HOLDS\_ARRAY(&val) &&

bson\_iter\_recurse(&val, &child))

{

cout << name << " : " << endl;

while (bson\_iter\_next(&child))

{

if (BSON\_ITER\_HOLDS\_UTF8(&child))

{

cout << bson\_iter\_utf8(&child, &u8len) << endl;

}

}

cout << endl;

}

}

str = bson\_as\_json(doc, NULL);

fprintf(stdout, "%s\n", str);

bson\_free(str);

}

if (mongoc\_cursor\_error(cursor, &error)) {

fprintf(stderr, "Cursor Failure: %s\n", error.message);

return EXIT\_FAILURE;

}

bson\_destroy(&query);

mongoc\_cursor\_destroy(cursor);

mongoc\_collection\_destroy(collection);

mongoc\_client\_destroy(client);

mongoc\_cleanup();

return EXIT\_SUCCESS;

# VS2013 x86

## 步骤

可以直接使用cmake生成x86项目。

path=C:\lib\mongocdrv\cmake-3.3.2-win32-x86\bin;%path%

cd C:\lib\mongo-c-driver-1.2.0

cd src\libbson

cmake -DCMAKE\_INSTALL\_PREFIX=C:\lib\libmongoc32 -G "Visual Studio 12"

msbuild.exe ALL\_BUILD.vcxproj /p:Configuration=Release

msbuild.exe INSTALL.vcxproj /p:Configuration=Release

cd ..\..

cmake -DCMAKE\_INSTALL\_PREFIX=C:\lib\libmongoc32 -DBSON\_ROOT\_DIR=C:\lib\libmongoc32 -G "Visual Studio 12"

msbuild.exe ALL\_BUILD.vcxproj /p:Configuration=Release

msbuild.exe INSTALL.vcxproj /p:Configuration=Release

以下方法不再使用。

## 间接生成步骤（不建议）

参考vs2013 x64 mongoclib编译过程，先用cmake生成x64解决方案，然后使用VS2013配置管理器生成win32配置。+k

### 生成win32 libbson

#### 生成vs2013 x64 libbson解决方案:

path=C:\lib\mongocdrv\cmake-3.3.2-win32-x86\bin;%path%

cd C:\lib\mongo-c-driver-1.2.0

cd src\libbson

cmake -DCMAKE\_INSTALL\_PREFIX=C:\lib\mongoclib -G "Visual Studio 12 Win64"

#### 从x64复制生成win32平台配置

使用vs2013打开src\libbson\libbson.sln，配置管理器，新建win32平台（从x64复制）。

#### 项目配置中移除/machine:x64选项

项目属性，连接器，命令行，其它选项中，删除 /machine:x64

包括：

bson\_shared，test-libbson

#### 编译项目

分别编译bson\_shared, test-libbson, test-libbson, ZERO\_CHECK项目，成功后编译ALL\_BUILD。直接编译ALL\_BUILD会导致不同项目写入同一个中间文件产生编译错误。

编译INSTALL将lib库文件及头文件复制到CMAKE\_INSTALL\_PREFIX

### 生成win32 libmongoc

由于cmake生成libmongoc解决方案时会使用libbson，所以需要先成功生成libbson的库之后才能正确生成libmongoc

#### 生成vs2013 x64 libmongoc解决方案:

cd ..\..

cmake -DCMAKE\_INSTALL\_PREFIX=C:\lib\mongoclib -DBSON\_ROOT\_DIR=C:\lib\mongoclib -G "Visual Studio 12 Win64"

#### 从x64复制生成win32平台配置

使用vs2013打开libmongoc.sln，配置管理器，新建win32平台（从x64复制）。

#### 项目配置中移除/machine:x64选项

检查各项目，项目属性，连接器，命令行，其它选项，删除 /machine:x64。

包括example\*, mongoc\*, test\*项目。

项目比较多，可以参考错误处理部分使用UE批量编辑vcxproj文件。

#### 编译项目

分别编译mongoc\*, example\*, test\*项目，完成后编译ALL\_BUILD

编译INSTALL将lib库文件及头文件复制到CMAKE\_INSTALL\_PREFIX

### 错误处理

编译错误：

fatal error LNK1112: 模块计算机类型“X86”与目标计算机类型“x64”冲突

项目属性，连接器，命令行，其它选项，强制指定了 /machine:x64 参数。

解决：

删除该选项即可。

由于项目比较多，使用ultraedit32简化操作：  
vs2013, 配置管理器，活动解决方案平台，编辑，移除x64平台，保存。

使用ue打开所有相关vcxproj文件，替换，/machine:x64为空格，选项选中所有打开的文件。替换后保存所有文件。完成

## 测试

直接在VS2008新建win32项目。类似VS2013 X64测试代码。测试通过。

# VS2008 x86

msbuid不支持VS2008项目，因此生成解决方案文件后，需要使用VS2008打开解决方案并编译。另外，libmongoc默认不支持vs2008，所以需要修改一些文件。

path=C:\lib\mongocdrv\cmake-3.3.2-win32-x86\bin;%path%

cd C:\lib\mongo-c-driver-1.2.0

cd src\libbson

cmake -DCMAKE\_INSTALL\_PREFIX=C:\lib\mongoc -G "Visual Studio 9 2008"

cd ..\..

cmake -DCMAKE\_INSTALL\_PREFIX=C:\lib\mongoc -DBSON\_ROOT\_DIR=C:\lib\mongoc -G "Visual Studio 9 2008"

编译libmongoc时，提示未定义 ETIMEDOUT

可以在mongoc-thread-private.h头部添加定义

const int ETIMEDOUT = 138;

test-libmongoc项目无法编译通过。

错误位置 const read\_prefs\_test tests []

可以不管，mongoc-\*项目编译通过后，编译install即可。