Welcome!

Welcome to the official BlackBerry Support Community Forums.

This is your resource to discuss support topics with your peers, and learn from each other.

Getting Started
Official BlackBerry Support

Register Connect with Facebook Sign In Help

Board

All New Topics | All New Posts

Device Support Forums BES Support Forums Developer Forums English

Developer Support Application Platforms Native Development Porting libraries: GPGME (GnuPG Made Easy)

Native Development

Reply Topic Options Message Listing Previous Topic Next Topic

doturner
Retired

Posts: 35
Registered: 10-15-2013

My Device: BlackBerry Z30
My Carrier: O2 UK

Porting libraries: GPGME (GnuPG Made Easy) [Edited]

Options

10-03-2014 12:07 PM - edited 10-03-2014 12:10 PM

Library name: libgpgme
Website: https://www.gnupg.org/
Source code: https://www.gnupg.org/download/index.html
Version ported: 1.5.1
Build system: autotools
Dependencies: **libgpg** -error (1.16), libassuan (2.1.2) - both available from gnupg.org.
Build environment: OSX 10.9.5, BlackBerry SDK 10.2.0.1155

If you just want the binaries they're attached to this post.

1. Overview
Before starting make sure your environment is configured for command line development, and that you have autotools installed.

In order to build GPGME we need to build the following libraries in order: **libgpg** -error, libassuan, libgpgme.

2. Build libgpg -error
Download and extract the source. Then run the configure command for your target architecture.

Simulator (x86):

```
./configure --host=i486-pc-nto-qnx8.0.0 --prefix=`pwd`/build/x86-qnx
```

Device (arm):

```
./configure --host=arm-unknown-nto-qnx8.0.0eabi --prefix=`pwd`/build/arm-qnx
```

Also set the default compiler for qcc to your target architecture - without this the configure script doesn't correctly set the CC version for the src folder build.

```
qcc -set-default -V4.6.3,gcc_ntoarmv7le_cpp
```

Now, make the gen-posix-lock-obj binary:

```
cd src
make gen-posix-lock-obj
```

This binary is used to create a header file later in the build process. Now we have our first attempt at compiling:

```
cd ..
make
```

This will error with the following output:

```
./mkerrcodes | awk -f ./mkerrcodes2.awk >code-from-errno.h
/bin/sh: ./mkerrcodes: cannot execute binary file
...
./mkheader nto-qnx8.0.0 i486-pc-nto-qnx8.0.0 ./gpg-error.h.in \
../config.h 1.16 0x011000 >gpg-error.h
/bin/sh: ./mkheader: cannot execute binary file
```

This is because the binaries which have been built will only run on the QNX architecture. To run them we need to copy them to a QNX target. I use Momentics for this although you could use SCP if you prefer.

Copy the following files to the target device (simulator or physical device):

```
src/err-codes.h.in
src/err-sources.h.in
src/errnos.in
src/gpg-error.h.in
src/gen-posix-lock-obj
src/mkerrcodes
src/mkerrcodes2.awk
src/mkheader
config.h
```

Now start an SSH session on the target and run the commands which failed during the first attempt at make:

```
./gen-posix-lock-obj > tmp.h
mkdir syscfg
mv tmp.h "syscfg/$(awk 'NR==1 {print $2}' tmp.h)"
./mkerrcodes | awk -f ./mkerrcodes2.awk >code-from-errno.h
./mkheader nto-qnx8.0.0 i486-pc-nto-qnx8.0.0 ./gpg-error.h.in ./config.h 1.16 0x011000
>gpg-error.h
```

The final command needs to be copied from the error output as it will vary depening on whether you're building for x86 or arm. Also note that config.h is in the same directory so it needs the path updating.

Copy the newly created files back to the src folder:

```
code-from-errno.h
gpg-error.h
```

Now you can run make again and **libgpg**-error should build correctly. Headers and binaries will be installed in the build/<arch>-qnx folder as specified in the --prefix configure flag.

```
make
make install
```

3. Build libassuan
Again, make sure you have set the qcc default compiler to your target architecture, then run configure:

For sim:

```
./configure --host=i486-pc-nto-qnx8.0.0 --prefix=`pwd`/build/x86-qnx --with-libgpg-error-prefix=`pwd`/../libgpg-error-1.16/build/x86-qnx
```

For device:

```
./configure --host=arm-unknown-nto-qnx8.0.0eabi --prefix=`pwd`/build/arm-qnx --with-libgpg-error-prefix=`pwd`/../libgpg-error-1.16/build/arm-qnx
```

Now attempt to build the library:

```
make
```

As with **libgpg**-error, this results in an error as the mkheader file cannot be executed as it is built for the QNX architecture. Copy the following files to your target:

```
src/mkheader
src/assuan.h.in
src/posix*
```

Now **from the target** run the command which errored:

```
./mkheader nto-qnx8.0.0 ./assuan.h.in 2.1.2 0x020102 >assuan.h
```

And copy assuan.h back to the src folder. Now you should be able to run make successfully:

```
make
make install
```

4. Finally build gpgme
Run configure for your architecture. Note that we have to include the **--disable-gpg-test** flag to stop the build from building the tests, without this the build will fail.

For sim:

```
./configure --host=i486-pc-nto-qnx8.0.0 --prefix=`pwd`/build/x86-qnx --with-libgpg-error-
prefix=`pwd`/../libgpg-error-1.16/build/x86-qnx --with-libassuan-
prefix=`pwd`/../libassuan-2.1.2/build/x86-qnx --disable-gpg-test
```

For device:

```
./configure --host=arm-unknown-nto-qnx8.0.0eabi --prefix=`pwd`/build/arm-qnx --with-
libgpg-error-prefix=`pwd`/../libgpg-error-1.16/build/arm-qnx --with-libassuan-
prefix=`pwd`/../libassuan-2.1.2/build/arm-qnx --disable-gpg-test
```

This time it should build first time using make:

```
make
make install
```

For convenience I have built the above 3 libraries for both the x86 and arm architectures. They are attached to this post.

If you have any problems building or using these libraries please post within this thread and I'll do my best to answer.

Problems

Problem 1: When running make on **libgpg**-error you get the following error:

```
code-from-errno.h:3:1: error: expected identifier or '(' before '}' token
```

Cause: Using an old version (1.9) of **libgpg**-error
Solution: Use the latest version (1.16).

Problem 2: When building libgpgme you receive the following error:

```
make[3]: *** No rule to make target `../../src/libgpgme-pthread.la', needed by `t-
thread1'.
```

Cause: It looks like the configure script checks for the existence of the pthread library and uses that to determine whether to build libgpgme-pthread.la. On QNX systems the pthread library doesn't exist, instead pthread is part of libc. This means that libgpgme-pthread.la is not built and the thread test fails to be built.
Solution: Add the --disable-gpg-test flag to the configure command

Attachments:
libgpg-error-1.6.zip 502 KB
libassuan-2.1.2.zip 346 KB
gpgme-1.5.1.zip 975 KB

1 Like Reply

Report Inappropriate Content
Message 1 of 1 (236 Views)

Message Listing Previous Topic Next Topic

powered by Lithium

BlackBerry.com/Support | Mobile Site | Full Site

Legal