EES4100: Installing the BACnet stack

Once you have the Modbus relevant code for your project working, it's time to start looking at the BACnet stack.

The BACnet stack that we will be using is hosted at:

http://bacnet.sourceforge.net/.

The developers have chosen to use *Subversion* for their version control, rather than *git*. They provide instructions on their web page on how to check out the latest version, however I'll also repeat the steps here.

The first thing that you'll notice after checking out their code is that they don't use the *autotools* build system. Integrating with other build systems can be challenging so I've provided an *autotools* version of the BACnet stack for you to use. From here on I'll refer to the *autotools* version of the BACnet stack as *libbacnet*. (Similar to *libmodbus*).

Using *libbacnet* will allow you to integrate the BACnet stack into your project using the same PKG_CHECK_MODULES macro that you used for *libmodbus*.

Installing *libbacnet*

In order to actually use libbacnet, you first need to install it on your system. This is similar to how you installed *libmodbus* using *emerge*, however in this case you will manually install the package. (This is necessary as the package hasn't been integrated into *Gentoo*). The following command list may be executed from your home directory.

Here is the complete list of commands necessary to install the *libbacnet* package:

`svn` is the interface to *Subversion*

```
svn checkout https://svn.code.sf.net/p/bacnet/code/trunk/bacnet-stack/
```

Change directory into bacnet-stack:

```
cd bacnet-stack
```

Clone the *libbacnet* package:

```
git clone https://github.com/kmtaylor/libbacnet.git
```

Patch the BACnet stack for use as a system library:

```
patch -p0 < libbacnet/patches/datalink.patch
```

Change directory into *libbacnet*:

```
cd libbacnet
```

Bootstrap *libbacnet*:

```
./bootstrap.sh
```

Configure *libbacnet*: (we provide a prefix argument to the configure strip so that the package will be installed as a system package. The default prefix is /usr/local. You can use the default prefix if you like, however, this will mean that you will have to inform your project of the non-standard location using the environment variables PKG_CONFIG_PATH and LD_LIBRARY_PATH)

```
./configure --prefix=/usr
```

Build *libbacnet*:

```
make
```

Install *libbacnet* as the root user:

```
su
make install
exit
```

Congratulations, you now have *libbacnet* installed as a system package on your machine. The next thing to do is link it to your project.

Linking *libbacnet* to your project

The *libbacnet* package provides you with a pkg-config metadata file. This means that you can link it to your project using the same method as *libmodbus*. The process is as follows:

1. Add the line `PKG_CHECK_MODULES([BACNET], [libbacnet])` to your configure.ac
2. You now have the variables `BACNET_LIBS` and `BACNET_CFLAGS` available for use in your Makefile.am files.
3. Add the relevant variables to your `prog_name_CFLAGS` and `prog_name_LDADD` lists in your Makefile.am files, where `prog_name` needs to be replaced with the name of your project's binary executable.

Example code

Unfortunately, unlike *libmodbus*, the documentation for the BACnet stack is fairly scant. For example, there are no man pages documenting the API. The best way to understand how the library works is to look at the library's source code, and to view the example code that I've provided at:

https://github.com/kmtaylor/EES4100_Testbench

This repository will build three executables, *modbus_server*, *bacnet_client* and *bacnet_server* from the source files modbus_server.c, bacnet_client.c and bacnet_server.c

modbus_server is the server running on a machine with IP address: 140.159.153.159. *bacnet_client* will run on a machine with IP address: 140.159.160.7

modbus_server and *bacnet_client* are provided in order for you to understand what is expected of your project. They will be updated in the future with a custom set of registers for each student.

bacnet_server may be used as example code for your application. It contains the necessary BACnet device objects for you to be able to pass the Modbus data that you obtain onto the BACnet client.

I strongly encourage you to look at all three modules in order to improve your understanding of the project and it's test bench as a whole. I also strongly encourage you to have a look at the *libbacnet* package, in order to improve your understating of the *autotools* build system and how it can be used to ease integration of 3rd party libraries.

Avoiding namespace pollution

You may have noticed that all functions provided by the *libmodbus* library are prefixed with `modbus_`. Name prefixing is typically used by 3rd party libraries so that the reader is immediately aware of which library is in use at each function call.

The BACnet stack does not provide a prefix for its API.

In order to work around this, the example code at

https://github.com/kmtaylor/EES4100_Testbench uses some helper scripts and the C preprocessor to enforce API name prefixes (using the prefix `bacnet_`)

The two scripts involved are `scripts/build_namespace.sh` and `scripts/check_namespace.sh`. Have a look at `src/Makefile.am` and you will notice that these scripts get called whenever a file in the list `BACNET_DEPENDENT_SRC` is modified.

The scripts are fed a list of names provided in the file `src/bacnet_api_names`.

In turn the file `src/bacnet_namespace.h` is generated, containing preprocessor macros renaming the BACnet API. Since `src/bacnet_namespace.h` is generated during the build process, it must be included in the *automake* list `BUILT_SOURCES`.

If you find yourself using any additional functions or macros from the BACnet API, ensure that you also add those names to `src/bacnet_api_names` to enforce the `bacnet_` prefix. This will greatly improve your code's readability.