Paho-C for OpenVMS Alpha and Integrity

September 2019

1. Introduction

Thank you for your interest in this port of the Paho-C MQTT client API to OpenVMS. The current release of Paho-C for OpenVMS is based on the Paho-C 1.3.0 distribution.

The Paho MQTT C API is a fully-featured MQTT client written in ANSI standard C. MQTT is a lightweight publish-subscribe protocol for use on top of the TCP/IP protocol. It is useful for connections with remote locations where a small code footprint is required and/or network bandwidth is at a premium. For example, it has been used in sensors communicating to a broker via satellite link, over occasional dial-up connections with healthcare providers, and in a range of home automation and small device scenarios. It is also ideal for mobile applications because of its small size, low power usage, minimized data packets, and efficient distribution of information to one or many receivers.

The MQTT publish-subscribe messaging pattern requires a message broker, which is responsible for distributing messages to interested clients based on the topic of a message. A comprehensive list of currently available MQTT brokers and the features they support can be found at https://github.com/mqtt/mqtt.github.io/wiki/Server%20support.

There are synchronous and asynchronous variants of the API. The synchronous API is generally simpler to use from a programming, with I/O-related calls blocking until the operation in question has completed. In contrast, no calls ever block in the asynchronous API, with notifications of call results or completion being provided asynchronously via callbacks. The choice of API (synchronous or asynchronous) depends on the requirements of the application in question. From an OpenVMS perspective, it should be noted that asynchronous behaviour is achieved through the use of POSIX threads as opposed to ASTs.

This OpenVMS port of the Paho-C MQTT API includes all functionality provided by the Open Source release, including SSL/TLS support. Additional information about the Paho-C MQTT API and about MQTT in general can be found at https://www.eclipse.org/paho and http://mqtt.org respectively.

2. Acknowledgements

VMS Software Inc. would like to acknowledge the work of the Paho API development team and the Eclipse Foundation for their ongoing efforts in developing and supporting this Open Source software.

3. What's new in this release

For details of new features and bug fixes included in this release, please read the project home page (https://www.eclipse.org/paho/) and links contained therein, and review the project repository change log (http://git.eclipse.org/c/paho/org.eclipse.paho.mqtt.c.git/log/).

SSL/TLS support is statically linked into the shareable images `libmqttv3cs$shr.exe` and `libmqttv3as$shr.exe` and uses OpenSSL 1.1.1b.

4. Requirements

The kit you are receiving has been compiled and built using the operating system and compiler versions listed below. While it is highly likely that you will have no problems installing and using the kit on systems running higher versions of the products listed, we cannot say for sure that you will be so lucky if your system is running older versions.

- OpenVMS 8.4-1H1 or above
- VSI TCP/IP, HPE TCP/IP Services for OpenVMS, or MultiNet TCP/IP stack for network communication
- C compiler - VSI C V7.4-001 or comparable
- OpenSSL 1.1.1 (statically linked into the supplied Paho-C shareable images)

  Note that if you wish to statically link application code requiring with the supplied object libraries and require SSL/TLS support, it will be necessary to link with a comparable OpenSSL distribution.

In addition to the above requirements, it is assumed that the reader has a good knowledge of OpenVMS and of software development in the OpenVMS environment.

5. Recommended reading

It is recommended that developers read the documentation on the Eclipse Foundation web site (http://wiki.eclipse.org/Paho) and carefully examine the provided sample programs before using the software.

6. Installing the kit

The kit is provided as an OpenVMS PCSI kit (`VSI-I64VMS-PAHO_C-V0103-0-1.PCSI` for i64 or `VSI-AXPVMS-PAHO_C-V0103-0-1.PCSI` for Alpha) that can be installed by a suitably privileged user using the following command:

```
$ PRODUCT INSTALL PAHO_C
```

The installation will then proceed as follows (output may differ slightly from that shown depending on platform and other factors):

```
Performing product kit validation of signed kits ...

The following product has been selected:
    VSI I64VMS PAHO_C V1.3-0               Layered Product

Do you want to continue? [YES] y

Configuration phase starting ...

You will be asked to choose options, if any, for each selected
product and for
any products that may be installed to satisfy software dependency
requirements.

Configuring VSI I64VMS PAHO_C V1.3-0
```

```
    VMS Software Inc. & the Eclipse Foundation

* This product does not have any configuration options.

Execution phase starting ...

The following product will be installed to destination:
    VSI I64VMS PAHO_C V1.3-0               DISK$I64SYS:[VMS$COMMON.]

Portion done: 0%...10%...40%...50%...90%...100%

The following product has been installed:
    VSI I64VMS PAHO_C V1.3-0               Layered Product

VSI I64VMS PAHO_C V1.3-0

    Post-installation tasks are required.

    To enable the Paho-C runtime at system boot, add the following
    lines to SYS$MANAGER:SYSTARTUP_VMS.COM:

        $ file := SYS$STARTUP:PAHO$STARTUP.COM
        $ if f$search("''file'") .nes. "" then @'file'

    To disable the Paho-C runtime at system shutdown, add the
    following lines to SYS$MANAGER:SYSHUTDWN.COM:

        $ file := SYS$STARTUP:PAHO$SHUTDOWN.COM
        $ if f$search("''file'") .nes. "" then @'file'
```

6.1. *Post-installation steps*

After the installation has successfully completed, include the commands displayed at the end of the installation procedure into `SYSTARTUP_VMS.COM` to ensure that the logical names required in order for developers to use the software are defined system-wide at start-up.

Note that in addition to the logical name `PAHO$ROOT` (which points to the root of the Paho-C software installation), the following logical names are defined:

Logical name	Description
`LIBMQTTV3A$SHR`	Paho-C synchronous API; no TLS/SSL support
`LIBMQTTV3AS$SHR`	Paho-C asynchronous API; includes TLS/SSL support
`LIBMQTTV3C$SHR`	Paho-C synchronous API; no TLS/SSL support
`LIBMQTTV3CS$SHR`	Paho-C synchronous API; includes TLS/SSL support

These logical names can be used by developers to link application code with the appropriate library. Alternatively developers can statically link their code with the corresponding object libraries found in `PAHO$ROOT:[LIB]`.

From a development perspective, it should be noted that symbols in the shareable images and object libraries are mixed-case, and developers should use the C compiler option `/NAMES=(AS_IS,SHORTENED)` or include in their code appropriate `#pragma` directives to ensure that symbols are correctly resolved when linking. Developers will also need to include in their code header files found in `PAHO$ROOT:[INCLUDE]`.

6.2. *Privileges and quotas*

Generally speaking there are no special quota or privilege requirements for applications developed using the Paho-C library, although a high `BYTLM` is recommended, and `SYSPRV`, `BYPASS`, or `OPER` privilege will be required if applications developed using the library need to utilise privileged ports (ports below 1024).

The following quotas should be more than adequate for most purposes:

```
Maxjobs:          0  Fillm:        256  Bytlm:        128000
Maxacctjobs:      0  Shrfillm:       0  Pbytlm:            0
Maxdetach:        0  BIOlm:        150  JTquota:        4096
Prclm:           50  DIOlm:        150  WSdef:          4096
Prio:             4  ASTlm:        300  WSquo:          8192
Queprio:          4  TQElm:        100  WSextent:      16384
CPU:         (none)  Enqlm:       4000  Pgflquo:      256000
```

6.3. *Installing in an alternative location*

By default the software will be installed in `SYS$SYSDEVICE:[VMS$COMMON]`. If you wish to install the software in an alternative location this can be achieved using the `/DESTINATION` qualifier with the `PRODUCT INSTALL` command to specify the desired location; however it is important to note that an additional manual step will then be required to complete the installation. Specifically, when an alternative destination is specified, the start-up and shutdown procedures (`PAHO$STARTUP.COM` and `PAHO$SHUTDOWN.COM`) will be placed into a subdirectory `[.SYS$STARTUP]` residing under the specified destination directory. If you wish to run these files from your standard `SYS$STARTUP` directory they will need to be copied from the destination subdirectory into your systems `SYS$STARTUP` directory.

7. Sample applications

The directory `PAHO$ROOT:[EXAMPLES]` contains several simple example programs that can be used to learn about the API or as a source of inspiration for the development of new applications. These examples can be compiled and linked using the provided build procedure (`SAMPLES.COM`). Once built, these programs are simple to run. In most cases it will be necessary to run the programs via an appropriately defined foreign command, and running the program without any additional command line arguments will display for some of the programs display usage information. In most cases there are also useful comments included in the code. In order to run the example programs it is necessary to have access to a MQTT broker.

It should be noted that the examples as currently built by `SAMPLES.COM` are statically linked to the non-TLS/SSL object libraries. To use TLS/SSL it is recommended that applications are linked with one of the shareable images `LIBMQTTV3AS$SHR.EXE` (asynchronous API) or `LIBMQTTV3CS$SHR.EXE` (synchronous API), which are statically linked with OpenSSL. These images reside in `PAHO$ROOT:[LIB]` and can be linked with application code using the logical names `LIBMQTTV3AS$SHR` and `LIBMQTTV3CS$SHR` respectively.

8. What’s missing?

The supplied kit for OpenVMS includes all functionality supported by version 1.3.0 of the Paho-C API. Future releases will likely also include OpenVMS-specific functionality, such as a simple abstraction layer that will make it easier for developers to use the underlying APIs from languages other than C, such as COBOL, FORTRAN, and BASIC.