

Mosquitto for OpenVMS I64

October 2015

1. Introduction

Thank you for your interest in this port of the Mosquitto MQTT broker to OpenVMS. The current release of Mosquitto on OpenVMS is based on the Mosquitto 1.3.5 distribution.

Mosquitto is an open source (BSD licensed) message broker that implements the MQ Telemetry Transport protocol versions 3.1 and 3.1.1. MQTT is a lightweight publish-subscribe based messaging 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 publish-subscribe messaging pattern requires a message broker, which is responsible for distributing messages to interested clients based on the topic of a message.

This OpenVMS port of the Mosquitto broker includes all functionality provided by the Open Source release, including SSL/TLS support, bridging, client authentication, and access controls. Additional information about the Mosquitto broker and MQTT in general can be found at <http://mosquitto.org> and <http://mqtt.org> respectively.

2. Acknowledgements

VMS Software Inc. would like to acknowledge the work of the Mosquitto development team for their ongoing efforts in developing and supporting this software, which takes a not an insignificant amount of time and effort.

3. What's new in this release

For a detailed description of the features and bug fixes included in this release, please read <http://mosquitto.org/2014/10/version-1-3-5-released>.

This release also includes a simple OpenVMS-specific plugin that provides the ability for the Mosquitto broker to authenticate users via standard OpenVMS means (namely checking the supplied credentials against those in SYSUAF).

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 operating system or products listed, we cannot say for sure that you will be so lucky if your system is running older versions.

- OpenVMS 8.4 I64

- HP TCP/IP Services V5.7

It has not been verified whether the kit works with the MultiNet TCP/IP stack, but there is a good chance that it will.

- C compiler - HP C V7.3-018
- OpenSSL 1.0.1j (statically linked)
- It is strongly recommended that the software is installed on an ODS-5-enabled file system

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

Before using the Mosquitto broker, it is recommended that users read the documentation available at <http://mosquitto.org/documentation/> in order to better understand how to configure and manage the broker.

6. Installing the kit

The kit is provided as an OpenVMS PCSI kit (VSI-I64VMS-MOSQUITTO-V0103-5-1.PCSI) that can be installed by a suitably privileged user using the following command:

```
$ PRODUCT INSTALL MOSQUITTO
```

The installation will then proceed as follows (output may differ slightly from that shown):

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

```
The following product has been selected:
```

```
VSI I64VMS MOSQUITTO V1.3-5          Layered Product
```

```
Do you want to continue? [YES]
```

```
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 MOSQUITTO V1.3-5: Mosquitto for OpenVMS is
based on Mosquitto Version 1.3.5
```

```
© Copyright 2015 VMS Software Inc.
```

```
VSI Software Inc.
```

```
* This product does not have any configuration options.
```

```
Execution phase starting ...
```

```
The following product will be installed to destination:
```

```
VSI I64VMS MOSQUITTO V1.3-5          DISK$I64SYS:[VMS$COMMON.]
```

```
Portion done: 0%...20%...30%...50%...70%...90%...100%
```

The following product has been installed:

VSI I64VMS MOSQUITTO V1.3-5 Layered Product

VSI I64VMS MOSQUITTO V1.3-5: Mosquitto for OpenVMS is based on Mosquitto Version 1.3.5

Post-installation tasks are required.

To start Mosquitto at system boot time, add the following lines to

SYS\$MANAGER:SYSTARTUP_VMS.COM:

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

To shutdown Mosquitto at system shutdown, add the following lines to SYS\$MANAGER:SYSHUTDOWN.COM:

```
$ file := SYS$STARTUP:MOSQUITTO$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` and `SYSHUTDOWN.COM` to ensure that the Mosquitto broker is started and stopped when OpenVMS is booted and shutdown.

Other steps that should be performed once the installation has completed are as follows:

- Attempt to start the broker

It is likely that you will want to modify the default configuration; however as an initial test to verify the installation, attempt to start the broker using the default configuration by running the start-up procedure `SYS$STARTUP:MOSQUITTO$STARTUP.COM`.

After running the start-up procedure, perform the following tasks:

- Verify that the logical name `MOSQUITTO$ROOT` is correctly defined. This system-level logical name points to the root of the Mosquitto installation tree.
- Verify that the broker is running by issuing a “`SHOW SYSTEM`” command and checking the displayed output for the existence of the process name “`MOSQUITTO`”. Note that if there was already another process running on your system with this process name, the broker will fail to start.
- Examine the log file `MOSQUITTO$ROOT:[LOGS]MOSQUITTO.LOG` and verify that there are no errors or warnings being reported. Note that a new version of this log file will be created whenever the broker is restarted, and it is recommended that appropriate processes are put in place to backup and purge old log files.
- Run one or more test programs (or use a generic MQTT client) to verify that the broker is functioning as expected. Note that the installation includes the utilities `PUB_CLIENT.EXE` and `SUB_CLIENT.EXE` (found in `MOSQUITTO$ROOT:[BIN]`), which can also be used for basic testing purposes (these programs can be run by defining appropriate foreign commands, and running the programs without any arguments will display the various options that are available).

- Modify the broker configuration as required

Once basic operation of the broker has been verified, you may wish to modify the default broker configuration to address specific requirements. The broker configuration file is `MOSQUITTO$ROOT:[CONF]MOSQUITTO.CONF` and can be modified as necessary in accordance with the notes that are included as comments in the file. Additional information regarding configuration of the broker can be found on the Mosquitto web site.

- Restart the broker

The broker must be restarted to pick up any configuration changes. After restarting, be sure to verify that the broker is operating correctly.

- Specify an alternative location for log files (optional)

By default Mosquitto will create log files named `mosquitto.log` in the directory `mosquitto$root:[logs]` (a new version of `mosquitto.log` will be created each time the broker is started). If you would prefer log files to be written to another location, simply modify the value of the `/output` qualifier for the `run/detached` command in `mosquitto$startup.com` as appropriate.

6.2. Privileges and quotas

The privileges `TMPMBX`, `NETMBX`, `BYPASS`, `SYSPRV`, and `DETACH` are required in order to run the Mosquitto start-up and shutdown scripts, and the Mosquitto broker process (run as a detached process) will inherit the default privileges for the username under which it is started.

The Mosquitto broker can require considerable resources in order to operate efficiently, depending on workload requirements. The following quotas should be adequate for most purposes; however resource usage should be carefully monitored, and quotas adjusted as necessary.

Maxjobs:	0	Fillm:	256	Byt1m:	128000
Maxacctjobs:	0	Shrfillm:	0	Pbyt1m:	0
Maxdetach:	0	BI01m:	150	JTquota:	4096
Prclm:	50	DI01m:	150	WSdef:	4096
Prio:	4	AST1m:	300	WSquo:	8192
Queprio:	4	TQE1m:	100	WSextent:	16384
CPU:	(none)	Enqlm:	4000	Pgflquo:	256000

If the broker is expected to support large numbers of connections then it may also be necessary to increase the `CHANNELCNT` system parameter (this parameter can usually be safely set to its maximum value of 65535).

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 (`MOSQUITTO$STARTUP.COM` and `MOSQUITTO$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. Configuring SYSUAF-based authentication

The shareable image `MOSQUITTO$ROOT:[LIB]MOSQUITTO$AUTH_PLUGIN_VMS.EXE` is a simple plugin for the Mosquitto MQTT broker that provides the ability for the broker to authenticate users via standard OpenVMS means (namely checking the supplied credentials against those in `SYSUAF`). It should be noted that in its present form the plugin verifies only that valid username and password credentials have been supplied; it does nothing to control what authenticated users can and cannot do once they are connected to the broker (the plugin will be enhanced to support such functionality in a future release).

7.1. Configuring the plugin

To configure the `SYSUAF` authentication plugin, edit the Mosquitto broker configuration file (`MOSQUITTO$ROOT:[CONF]MOSQUITTO.CONF`) to include the following details (the logical name `MOSQUITTO$AUTH_PLUGIN_VMS` is defined in the Mosquitto start-up procedure to point to the authentication plugin shareable image):

```
# Authentication plugin (specify path to OpenVMS shareable image)
auth_plugin mosquitto$auth_plugin_vms
```

While editing the configuration file you might also wish to set `allow_anonymous` to `false` to disable anonymous (unauthenticated) logins.

8. What's missing?

The supplied kit for OpenVMS includes all functionality supported by version 1.3.5 of the Open Source Mosquitto release. It is noted that this version is now somewhat old, and VMS Software Inc. are working to make available a newer release in the not too distant future.

Future releases will likely also include OpenVMS-specific functionality, such as support for `SYSUAF`-based authentication.

9. Known problems

Depending on your TCP/IP configuration it may be necessary to explicitly specify a value in `MOSQUITTO$ROOT:[CONF]MOSQUITTO.CONF` for the `bind_address` parameter.

A problem has been observed whereby the broker determines that IPv6 is available causing it to attempt to bind to an IPv6 address (in addition to IPv4); however if IPv6 is not enabled the broker will encounter an error and will fail to start. The solution to this problem is to restrict the broker to IPv4 only by specifying an appropriate value (host name or address) for the `bind_address` parameter.