VMS Software

Mosquitto V2.0-22 for OpenVMS

Release Notes

Publication Date: January 2026

Operating Systems: VSI OpenVMS Alpha V8.4-2L1 or later
VSI OpenVMS IA-64 V8.4-2L1 or later
VSI OpenVMS x86-64 V9.2-2 or later

Kit Names: VSI-AXPVMS-MOSQUITTO-V0200-22-1.PCSI
VSI-I64VMS-MOSQUITTO-V0200-22-1.PCSI
VSI-X86VMS-MOSQUITTO-V0200-22-1.PCSI

VMS Software, Inc. (VSI)
Boston, Massachusetts, USA

Table of Contents

1. Introduction

Thank your for your interest in this port of the Mosquitto MQTT broker to OpenVMS. The current release of Mosquitto for OpenVMS is based on the Mosquitto V2.0-22 distribution.

Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol versions 5.0, 3.1.1, and 3.1. MQTT is a lightweight messaging protocol based on the publish–subscribe model that operates over TCP/IP. 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.

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. acknowledges the Mosquitto development team for their ongoing efforts in developing and supporting this software, which demands considerable time and dedication.

3. What is New in this Release

For a detailed description of the features and bug fixes included in this release, please read https://mosquitto.org/blog/2025/07/version-2-0-22-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).

SSL/TLS support is statically linked into the broker and uses OpenSSL V3.0.

4. Requirements

The kit you are receiving has been compiled and built using the operating system and compiler versions listed below. Although the kit is expected to perform correctly on systems with newer versions of the operating system or products listed, VSI cannot guarantee successful installation or operation on older versions.

- OpenVMS 8.4-2L1 or higher (IA-64), OpenVMS V8.4-2L1 or higher (Alpha), OpenVMS 9.2-3 or higher (x86-64)
- VSI TCP/IP
- C compiler for application development (optional)
- 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 solid understanding of OpenVMS and its software development 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 that can be installed by a suitably privileged user with the following command:

```
$ PRODUCT INSTALL MOSQUITTO
```

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 MOSQUITTO V2.0-22           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 V2.0-22: Mosquitto for OpenVMS is based on
 Mosquitto Version 2.0-22

    © Copyright 2025 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 V2.0-22           DISK$I64SYS:[VMS$COMMON.]

Portion done: 0%...10%...20%...30%...50%...70%...80%...90%...100%

The following product has been installed:
    VSI I64VMS MOSQUITTO V2.0-22           Layered Product

VSI I64VMS MOSQUITTO V2.0-22: Mosquitto for OpenVMS is based on Mosquitto
 Version 2.0-22

    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:SYSHUTDWN.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 SYSHUTDWN.COM. This ensures that the Mosquitto broker starts when OpenVMS is booted and stops during shutdown.

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

- Attempt to start the broker

  Although you may modify the default configuration; you should first verify the installation by starting the broker with the default settings using the 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 client utilities MOSQUITTO_PUB.EXE and MOSQUITTO_SUB.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 website.

- 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, the Mosquitto broker will create log files named MOSQUITTO.LOG in the directory MOSQUITTO$ROOT:[LOGS] with a new version of the log file created each time the broker is started. If you prefer the log files to be written to another location, modify the value of the **/OUTPUT** qualifier for the **RUN/DETACHED** command in MOSQUITTO$STARTUP.COM as appropriate.

If you intend to use only the client components of the Mosquitto software distribution and do not want to start the broker, run MOSQUITTO$STARTUP.COM as follows.

```
$ @SYS$STARTUP:MOSQUITO$STARTUP CLIENT
```

Running the start-up procedure in this manner defines all necessary system logical names; however, the broker will not be started.

6.2. Privileges and Quotas

The privileges TMPMBX, NETMBX, BYPASS, SYSPRV, and DETACH are required to run the Mosquitto start-up and shutdown scripts. The Mosquitto broker, executed as a detached process, inherits the default privileges of the user account that initiates it.

The Mosquitto broker can require considerable resources 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  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
```

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).

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). Note that the plugin presently verifies only credential validation, ensuring that a valid username and password are supplied. It does not provide authorization or access control for authenticated users once connected to the broker; such functionality will be introduced 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 may also want to set `allow_anonymous` to false to disable anonymous (unauthenticated) logins.

8. Example Client Programs

The directory MOSQUITTO$ROOT:[EXAMPLES] contains simple example programs written in C that serve to illustrate use of the Mosquitto MQTT client API. These example programs that can be used to learn about the API or as a source of inspiration for the development of new applications. The examples can be compiled and linked using the provided build procedure (SAMPLE.COM), and once built, these programs are simple to run, requiring the specification of no command line arguments.

Note

The examples are hard coded to connect to a Mosquitto broker running at test.mosquitto.org. To connect to your local Mosquitto broker, test.mosquitto.org should be replaced in the example code with the name or address of the OpenVMS server on which you are running broker.

The example program code is clearly organized and well-commented, allowing you to adapt it to your needs with minimal reference to the API documentation.

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 listener 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 simplest solution to this problem is to restrict the broker to IPv4 only by specifying an appropriate value (host name or address) for the listener parameter.

Note

This parameter is explicitly defined as `listener 1883` in the configuration file included with this kit, where `1883` is the standard MQTT port number. The configuration file included with this kit also specifies `allow_anonymous true`, enabling clients to connect to the broker without providing a username. This has been done to allow users to quickly get started with the software, however in general it is not recommended to leave this setting enabled in a production environment.