# Config file for mosquitto

#

# See mosquitto.conf(5) for more information.

#

# Default values are shown, uncomment to change.

#

# Use the # character to indicate a comment, but only if it is the

# very first character on the line.

# =================================================================

# General configuration

# =================================================================

# Time in seconds to wait before resending an outgoing QoS=1 or

# QoS=2 message.

#retry\_interval 20

# Time in seconds between updates of the $SYS tree.

# Set to 0 to disable the publishing of the $SYS tree.

#sys\_interval 10

# Time in seconds between cleaning the internal message store of

# unreferenced messages. Lower values will result in lower memory

# usage but more processor time, higher values will have the

# opposite effect.

# Setting a value of 0 means the unreferenced messages will be

# disposed of as quickly as possible.

#store\_clean\_interval 10

# Write process id to a file. Default is a blank string which means

# a pid file shouldn't be written.

# This should be set to /var/run/mosquitto.pid if mosquitto is

# being run automatically on boot with an init script and

# start-stop-daemon or similar.

#pid\_file

# When run as root, drop privileges to this user and its primary

# group.

# Leave blank to stay as root, but this is not recommended.

# If run as a non-root user, this setting has no effect.

# Note that on Windows this has no effect and so mosquitto should

# be started by the user you wish it to run as.

#user mosquitto

# The maximum number of QoS 1 and 2 messages currently inflight per

# client.

# This includes messages that are partway through handshakes and

# those that are being retried. Defaults to 20. Set to 0 for no

# maximum. Setting to 1 will guarantee in-order delivery of QoS 1

# and 2 messages.

#max\_inflight\_messages 20

# The maximum number of QoS 1 and 2 messages to hold in a queue

# above those that are currently in-flight. Defaults to 100. Set

# to 0 for no maximum (not recommended).

# See also queue\_qos0\_messages.

#max\_queued\_messages 100

# Set to true to queue messages with QoS 0 when a persistent client is

# disconnected. These messages are included in the limit imposed by

# max\_queued\_messages.

# Defaults to false.

# This is a non-standard option for the MQTT v3.1 spec but is allowed in

# v3.1.1.

#queue\_qos0\_messages false

# This option sets the maximum publish payload size that the broker will allow.

# Received messages that exceed this size will not be accepted by the broker.

# The default value is 0, which means that all valid MQTT messages are

# accepted. MQTT imposes a maximum payload size of 268435455 bytes.

#message\_size\_limit 0

# This option controls whether a client is allowed to connect with a zero

# length client id or not. This option only affects clients using MQTT v3.1.1

# and later. If set to false, clients connecting with a zero length client id

# are disconnected. If set to true, clients will be allocated a client id by

# the broker. This means it is only useful for clients with clean session set

# to true.

#allow\_zero\_length\_clientid true

# If allow\_zero\_length\_clientid is true, this option allows you to set a prefix

# to automatically generated client ids to aid visibility in logs.

#auto\_id\_prefix

# This option allows persistent clients (those with clean session set to false)

# to be removed if they do not reconnect within a certain time frame.

#

# This is a non-standard option in MQTT V3.1 but allowed in MQTT v3.1.1.

#

# Badly designed clients may set clean session to false whilst using a randomly

# generated client id. This leads to persistent clients that will never

# reconnect. This option allows these clients to be removed.

#

# The expiration period should be an integer followed by one of h d w m y for

# hour, day, week, month and year respectively. For example

#

# persistent\_client\_expiration 2m

# persistent\_client\_expiration 14d

# persistent\_client\_expiration 1y

#

# The default if not set is to never expire persistent clients.

#persistent\_client\_expiration

# If a client is subscribed to multiple subscriptions that overlap, e.g. foo/#

# and foo/+/baz , then MQTT expects that when the broker receives a message on

# a topic that matches both subscriptions, such as foo/bar/baz, then the client

# should only receive the message once.

# Mosquitto keeps track of which clients a message has been sent to in order to

# meet this requirement. The allow\_duplicate\_messages option allows this

# behaviour to be disabled, which may be useful if you have a large number of

# clients subscribed to the same set of topics and are very concerned about

# minimising memory usage.

# It can be safely set to true if you know in advance that your clients will

# never have overlapping subscriptions, otherwise your clients must be able to

# correctly deal with duplicate messages even when then have QoS=2.

#allow\_duplicate\_messages false

# The MQTT specification requires that the QoS of a message delivered to a

# subscriber is never upgraded to match the QoS of the subscription. Enabling

# this option changes this behaviour. If upgrade\_outgoing\_qos is set true,

# messages sent to a subscriber will always match the QoS of its subscription.

# This is a non-standard option explicitly disallowed by the spec.

#upgrade\_outgoing\_qos false

# =================================================================

# Default listener

# =================================================================

# IP address/hostname to bind the default listener to. If not

# given, the default listener will not be bound to a specific

# address and so will be accessible to all network interfaces.

# bind\_address ip-address/host name

#bind\_address

# Port to use for the default listener.

port 9001

# The maximum number of client connections to allow. This is

# a per listener setting.

# Default is -1, which means unlimited connections.

# Note that other process limits mean that unlimited connections

# are not really possible. Typically the default maximum number of

# connections possible is around 1024.

#max\_connections -1

# Choose the protocol to use when listening.

# This can be either mqtt or websockets.

# Websockets support is currently disabled by default at compile time.

# Certificate based TLS may be used with websockets, except that

# only the cafile, certfile, keyfile and ciphers options are supported.

protocol websockets

# When a listener is using the websockets protocol, it is possible to serve

# http data as well. Set http\_dir to a directory which contains the files you

# wish to serve. If this option is not specified, then no normal http

# connections will be possible.

#http\_dir

# Set use\_username\_as\_clientid to true to replace the clientid that a client

# connected with with its username. This allows authentication to be tied to

# the clientid, which means that it is possible to prevent one client

# disconnecting another by using the same clientid.

# If a client connects with no username it will be disconnected as not

# authorised when this option is set to true.

# Do not use in conjunction with clientid\_prefixes.

# See also use\_identity\_as\_username.

#use\_username\_as\_clientid

# -----------------------------------------------------------------

# Certificate based SSL/TLS support

# -----------------------------------------------------------------

# The following options can be used to enable SSL/TLS support for

# this listener. Note that the recommended port for MQTT over TLS

# is 8883, but this must be set manually.

#

# See also the mosquitto-tls man page.

# At least one of cafile or capath must be defined. They both

# define methods of accessing the PEM encoded Certificate

# Authority certificates that have signed your server certificate

# and that you wish to trust.

# cafile defines the path to a file containing the CA certificates.

# capath defines a directory that will be searched for files

# containing the CA certificates. For capath to work correctly, the

# certificate files must have ".crt" as the file ending and you must run

# "c\_rehash <path to capath>" each time you add/remove a certificate.

#cafile

#capath

# Path to the PEM encoded server certificate.

#certfile

# Path to the PEM encoded keyfile.

#keyfile

# This option defines the version of the TLS protocol to use for this listener.

# The default value allows v1.2, v1.1 and v1.0, if they are all supported by

# the version of openssl that the broker was compiled against. For openssl >=

# 1.0.1 the valid values are tlsv1.2 tlsv1.1 and tlsv1. For openssl < 1.0.1 the

# valid values are tlsv1.

#tls\_version

# By default a TLS enabled listener will operate in a similar fashion to a

# https enabled web server, in that the server has a certificate signed by a CA

# and the client will verify that it is a trusted certificate. The overall aim

# is encryption of the network traffic. By setting require\_certificate to true,

# the client must provide a valid certificate in order for the network

# connection to proceed. This allows access to the broker to be controlled

# outside of the mechanisms provided by MQTT.

#require\_certificate false

# If require\_certificate is true, you may set use\_identity\_as\_username to true

# to use the CN value from the client certificate as a username. If this is

# true, the password\_file option will not be used for this listener.

#use\_identity\_as\_username false

# If you have require\_certificate set to true, you can create a certificate

# revocation list file to revoke access to particular client certificates. If

# you have done this, use crlfile to point to the PEM encoded revocation file.

#crlfile

# If you wish to control which encryption ciphers are used, use the ciphers

# option. The list of available ciphers can be optained using the "openssl

# ciphers" command and should be provided in the same format as the output of

# that command.

# If unset defaults to DEFAULT:!aNULL:!eNULL:!LOW:!EXPORT:!SSLv2:@STRENGTH

#ciphers DEFAULT:!aNULL:!eNULL:!LOW:!EXPORT:!SSLv2:@STRENGTH

# -----------------------------------------------------------------

# Pre-shared-key based SSL/TLS support

# -----------------------------------------------------------------

# The following options can be used to enable PSK based SSL/TLS support for

# this listener. Note that the recommended port for MQTT over TLS is 8883, but

# this must be set manually.

#

# See also the mosquitto-tls man page and the "Certificate based SSL/TLS

# support" section. Only one of certificate or PSK encryption support can be

# enabled for any listener.

# The psk\_hint option enables pre-shared-key support for this listener and also

# acts as an identifier for this listener. The hint is sent to clients and may

# be used locally to aid authentication. The hint is a free form string that

# doesn't have much meaning in itself, so feel free to be creative.

# If this option is provided, see psk\_file to define the pre-shared keys to be

# used or create a security plugin to handle them.

#psk\_hint

# Set use\_identity\_as\_username to have the psk identity sent by the client used

# as its username. Authentication will be carried out using the PSK rather than

# the MQTT username/password and so password\_file will not be used for this

# listener.

#use\_identity\_as\_username false

# When using PSK, the encryption ciphers used will be chosen from the list of

# available PSK ciphers. If you want to control which ciphers are available,

# use the "ciphers" option. The list of available ciphers can be optained

# using the "openssl ciphers" command and should be provided in the same format

# as the output of that command.

#ciphers

# =================================================================

# Extra listeners

# =================================================================

# Listen on a port/ip address combination. By using this variable

# multiple times, mosquitto can listen on more than one port. If

# this variable is used and neither bind\_address nor port given,

# then the default listener will not be started.

# The port number to listen on must be given. Optionally, an ip

# address or host name may be supplied as a second argument. In

# this case, mosquitto will attempt to bind the listener to that

# address and so restrict access to the associated network and

# interface. By default, mosquitto will listen on all interfaces.

# Note that for a websockets listener it is not possible to bind to a host

# name.

# listener port-number [ip address/host name]

#listener

# The maximum number of client connections to allow. This is

# a per listener setting.

# Default is -1, which means unlimited connections.

# Note that other process limits mean that unlimited connections

# are not really possible. Typically the default maximum number of

# connections possible is around 1024.

#max\_connections -1

# The listener can be restricted to operating within a topic hierarchy using

# the mount\_point option. This is achieved be prefixing the mount\_point string

# to all topics for any clients connected to this listener. This prefixing only

# happens internally to the broker; the client will not see the prefix.

#mount\_point

# Choose the protocol to use when listening.

# This can be either mqtt or websockets.

# Certificate based TLS may be used with websockets, except that only the

# cafile, certfile, keyfile and ciphers options are supported.

#protocol mqtt

# When a listener is using the websockets protocol, it is possible to serve

# http data as well. Set http\_dir to a directory which contains the files you

# wish to serve. If this option is not specified, then no normal http

# connections will be possible.

#http\_dir

# Set use\_username\_as\_clientid to true to replace the clientid that a client

# connected with with its username. This allows authentication to be tied to

# the clientid, which means that it is possible to prevent one client

# disconnecting another by using the same clientid.

# If a client connects with no username it will be disconnected as not

# authorised when this option is set to true.

# Do not use in conjunction with clientid\_prefixes.

# See also use\_identity\_as\_username.

#use\_username\_as\_clientid

# -----------------------------------------------------------------

# Certificate based SSL/TLS support

# -----------------------------------------------------------------

# The following options can be used to enable certificate based SSL/TLS support

# for this listener. Note that the recommended port for MQTT over TLS is 8883,

# but this must be set manually.

#

# See also the mosquitto-tls man page and the "Pre-shared-key based SSL/TLS

# support" section. Only one of certificate or PSK encryption support can be

# enabled for any listener.

# At least one of cafile or capath must be defined to enable certificate based

# TLS encryption. They both define methods of accessing the PEM encoded

# Certificate Authority certificates that have signed your server certificate

# and that you wish to trust.

# cafile defines the path to a file containing the CA certificates.

# capath defines a directory that will be searched for files

# containing the CA certificates. For capath to work correctly, the

# certificate files must have ".crt" as the file ending and you must run

# "c\_rehash <path to capath>" each time you add/remove a certificate.

#cafile

#capath

# Path to the PEM encoded server certificate.

#certfile

# Path to the PEM encoded keyfile.

#keyfile

# By default an TLS enabled listener will operate in a similar fashion to a

# https enabled web server, in that the server has a certificate signed by a CA

# and the client will verify that it is a trusted certificate. The overall aim

# is encryption of the network traffic. By setting require\_certificate to true,

# the client must provide a valid certificate in order for the network

# connection to proceed. This allows access to the broker to be controlled

# outside of the mechanisms provided by MQTT.

#require\_certificate false

# If require\_certificate is true, you may set use\_identity\_as\_username to true

# to use the CN value from the client certificate as a username. If this is

# true, the password\_file option will not be used for this listener.

#use\_identity\_as\_username false

# If you have require\_certificate set to true, you can create a certificate

# revocation list file to revoke access to particular client certificates. If

# you have done this, use crlfile to point to the PEM encoded revocation file.

#crlfile

# If you wish to control which encryption ciphers are used, use the ciphers

# option. The list of available ciphers can be optained using the "openssl

# ciphers" command and should be provided in the same format as the output of

# that command.

#ciphers

# -----------------------------------------------------------------

# Pre-shared-key based SSL/TLS support

# -----------------------------------------------------------------

# The following options can be used to enable PSK based SSL/TLS support for

# this listener. Note that the recommended port for MQTT over TLS is 8883, but

# this must be set manually.

#

# See also the mosquitto-tls man page and the "Certificate based SSL/TLS

# support" section. Only one of certificate or PSK encryption support can be

# enabled for any listener.

# The psk\_hint option enables pre-shared-key support for this listener and also

# acts as an identifier for this listener. The hint is sent to clients and may

# be used locally to aid authentication. The hint is a free form string that

# doesn't have much meaning in itself, so feel free to be creative.

# If this option is provided, see psk\_file to define the pre-shared keys to be

# used or create a security plugin to handle them.

#psk\_hint

# Set use\_identity\_as\_username to have the psk identity sent by the client used

# as its username. Authentication will be carried out using the PSK rather than

# the MQTT username/password and so password\_file will not be used for this

# listener.

#use\_identity\_as\_username false

# When using PSK, the encryption ciphers used will be chosen from the list of

# available PSK ciphers. If you want to control which ciphers are available,

# use the "ciphers" option. The list of available ciphers can be optained

# using the "openssl ciphers" command and should be provided in the same format

# as the output of that command.

#ciphers

# =================================================================

# Persistence

# =================================================================

# If persistence is enabled, save the in-memory database to disk

# every autosave\_interval seconds. If set to 0, the persistence

# database will only be written when mosquitto exits. See also

# autosave\_on\_changes.

# Note that writing of the persistence database can be forced by

# sending mosquitto a SIGUSR1 signal.

#autosave\_interval 1800

# If true, mosquitto will count the number of subscription changes, retained

# messages received and queued messages and if the total exceeds

# autosave\_interval then the in-memory database will be saved to disk.

# If false, mosquitto will save the in-memory database to disk by treating

# autosave\_interval as a time in seconds.

#autosave\_on\_changes false

# Save persistent message data to disk (true/false).

# This saves information about all messages, including

# subscriptions, currently in-flight messages and retained

# messages.

# retained\_persistence is a synonym for this option.

persistence true

# The filename to use for the persistent database, not including

# the path.

#persistence\_file mosquitto.db

# Location for persistent database. Must include trailing /

# Default is an empty string (current directory).

# Set to e.g. /var/lib/mosquitto/ if running as a proper service on Linux or

# similar.

persistence\_location /mosquitto/data/

# =================================================================

# Logging

# =================================================================

# Places to log to. Use multiple log\_dest lines for multiple

# logging destinations.

# Possible destinations are: stdout stderr syslog topic file

#

# stdout and stderr log to the console on the named output.

#

# syslog uses the userspace syslog facility which usually ends up

# in /var/log/messages or similar.

#

# topic logs to the broker topic '$SYS/broker/log/<severity>',

# where severity is one of D, E, W, N, I, M which are debug, error,

# warning, notice, information and message. Message type severity is used by

# the subscribe/unsubscribe log\_types and publishes log messages to

# $SYS/broker/log/M/susbcribe or $SYS/broker/log/M/unsubscribe.

#

# The file destination requires an additional parameter which is the file to be

# logged to, e.g. "log\_dest file /var/log/mosquitto.log". The file will be

# closed and reopened when the broker receives a HUP signal. Only a single file

# destination may be configured.

#

# Note that if the broker is running as a Windows service it will default to

# "log\_dest none" and neither stdout nor stderr logging is available.

# Use "log\_dest none" if you wish to disable logging.

log\_dest file /mosquitto/log/mosquitto.log

# If using syslog logging (not on Windows), messages will be logged to the

# "daemon" facility by default. Use the log\_facility option to choose which of

# local0 to local7 to log to instead. The option value should be an integer

# value, e.g. "log\_facility 5" to use local5.

#log\_facility

# Types of messages to log. Use multiple log\_type lines for logging

# multiple types of messages.

# Possible types are: debug, error, warning, notice, information,

# none, subscribe, unsubscribe, websockets, all.

# Note that debug type messages are for decoding the incoming/outgoing

# network packets. They are not logged in "topics".

log\_type error

log\_type warning

log\_type notice

log\_type information

log\_type all

# Change the websockets logging level. This is a global option, it is not

# possible to set per listener. This is an integer that is interpreted by

# libwebsockets as a bit mask for its lws\_log\_levels enum. See the

# libwebsockets documentation for more details. "log\_type websockets" must also

# be enabled.

#websockets\_log\_level 0

# If set to true, client connection and disconnection messages will be included

# in the log.

#connection\_messages true

# If set to true, add a timestamp value to each log message.

#log\_timestamp true

# =================================================================

# Security

# =================================================================

# If set, only clients that have a matching prefix on their

# clientid will be allowed to connect to the broker. By default,

# all clients may connect.

# For example, setting "secure-" here would mean a client "secure-

# client" could connect but another with clientid "mqtt" couldn't.

#clientid\_prefixes

# Boolean value that determines whether clients that connect

# without providing a username are allowed to connect. If set to

# false then a password file should be created (see the

# password\_file option) to control authenticated client access.

# Defaults to true.

#allow\_anonymous true

# In addition to the clientid\_prefixes, allow\_anonymous and TLS

# authentication options, username based authentication is also

# possible. The default support is described in "Default

# authentication and topic access control" below. The auth\_plugin

# allows another authentication method to be used.

# Specify the path to the loadable plugin and see the

# "Authentication and topic access plugin options" section below.

#auth\_plugin

# If auth\_plugin\_deny\_special\_chars is true, the default, then before an ACL

# check is made, the username/client id of the client needing the check is

# searched for the presence of either a '+' or '#' character. If either of

# these characters is found in either the username or client id, then the ACL

# check is denied before it is sent to the plugin.o

#

# This check prevents the case where a malicious user could circumvent an ACL

# check by using one of these characters as their username or client id. This

# is the same issue as was reported with mosquitto itself as CVE-2017-7650.

#

# If you are entirely sure that the plugin you are using is not vulnerable to

# this attack (i.e. if you never use usernames or client ids in topics) then

# you can disable this extra check and have all ACL checks delivered to your

# plugin by setting auth\_plugin\_deny\_special\_chars to false.

#auth\_plugin\_deny\_special\_chars true

# -----------------------------------------------------------------

# Default authentication and topic access control

# -----------------------------------------------------------------

# Control access to the broker using a password file. This file can be

# generated using the mosquitto\_passwd utility. If TLS support is not compiled

# into mosquitto (it is recommended that TLS support should be included) then

# plain text passwords are used, in which case the file should be a text file

# with lines in the format:

# username:password

# The password (and colon) may be omitted if desired, although this

# offers very little in the way of security.

#

# See the TLS client require\_certificate and use\_identity\_as\_username options

# for alternative authentication options.

#password\_file

# Access may also be controlled using a pre-shared-key file. This requires

# TLS-PSK support and a listener configured to use it. The file should be text

# lines in the format:

# identity:key

# The key should be in hexadecimal format without a leading "0x".

#psk\_file

# Control access to topics on the broker using an access control list

# file. If this parameter is defined then only the topics listed will

# have access.

# If the first character of a line of the ACL file is a # it is treated as a

# comment.

# Topic access is added with lines of the format:

#

# topic [read|write|readwrite] <topic>

#

# The access type is controlled using "read", "write" or "readwrite". This

# parameter is optional (unless <topic> contains a space character) - if not

# given then the access is read/write. <topic> can contain the + or #

# wildcards as in subscriptions.

#

# The first set of topics are applied to anonymous clients, assuming

# allow\_anonymous is true. User specific topic ACLs are added after a

# user line as follows:

#

# user <username>

#

# The username referred to here is the same as in password\_file. It is

# not the clientid.

#

#

# If is also possible to define ACLs based on pattern substitution within the

# topic. The patterns available for substition are:

#

# %c to match the client id of the client

# %u to match the username of the client

#

# The substitution pattern must be the only text for that level of hierarchy.

#

# The form is the same as for the topic keyword, but using pattern as the

# keyword.

# Pattern ACLs apply to all users even if the "user" keyword has previously

# been given.

#

# If using bridges with usernames and ACLs, connection messages can be allowed

# with the following pattern:

# pattern write $SYS/broker/connection/%c/state

#

# pattern [read|write|readwrite] <topic>

#

# Example:

#

# pattern write sensor/%u/data

#

#acl\_file

# -----------------------------------------------------------------

# Authentication and topic access plugin options

# -----------------------------------------------------------------

# If the auth\_plugin option above is used, define options to pass to the

# plugin here as described by the plugin instructions. All options named

# using the format auth\_opt\_\* will be passed to the plugin, for example:

#

# auth\_opt\_db\_host

# auth\_opt\_db\_port

# auth\_opt\_db\_username

# auth\_opt\_db\_password

# =================================================================

# Bridges

# =================================================================

# A bridge is a way of connecting multiple MQTT brokers together.

# Create a new bridge using the "connection" option as described below. Set

# options for the bridges using the remaining parameters. You must specify the

# address and at least one topic to subscribe to.

# Each connection must have a unique name.

# The address line may have multiple host address and ports specified. See

# below in the round\_robin description for more details on bridge behaviour if

# multiple addresses are used.

# The direction that the topic will be shared can be chosen by

# specifying out, in or both, where the default value is out.

# The QoS level of the bridged communication can be specified with the next

# topic option. The default QoS level is 0, to change the QoS the topic

# direction must also be given.

# The local and remote prefix options allow a topic to be remapped when it is

# bridged to/from the remote broker. This provides the ability to place a topic

# tree in an appropriate location.

# For more details see the mosquitto.conf man page.

# Multiple topics can be specified per connection, but be careful

# not to create any loops.

# If you are using bridges with cleansession set to false (the default), then

# you may get unexpected behaviour from incoming topics if you change what

# topics you are subscribing to. This is because the remote broker keeps the

# subscription for the old topic. If you have this problem, connect your bridge

# with cleansession set to true, then reconnect with cleansession set to false

# as normal.

#connection <name>

#address <host>[:<port>] [<host>[:<port>]]

#topic <topic> [[[out | in | both] qos-level] local-prefix remote-prefix]

# Set the version of the MQTT protocol to use with for this bridge. Can be one

# of mqttv31 or mqttv311. Defaults to mqttv31.

#bridge\_protocol\_version mqttv31

# If a bridge has topics that have "out" direction, the default behaviour is to

# send an unsubscribe request to the remote broker on that topic. This means

# that changing a topic direction from "in" to "out" will not keep receiving

# incoming messages. Sending these unsubscribe requests is not always

# desirable, setting bridge\_attempt\_unsubscribe to false will disable sending

# the unsubscribe request.

#bridge\_attempt\_unsubscribe true

# If the bridge has more than one address given in the address/addresses

# configuration, the round\_robin option defines the behaviour of the bridge on

# a failure of the bridge connection. If round\_robin is false, the default

# value, then the first address is treated as the main bridge connection. If

# the connection fails, the other secondary addresses will be attempted in

# turn. Whilst connected to a secondary bridge, the bridge will periodically

# attempt to reconnect to the main bridge until successful.

# If round\_robin is true, then all addresses are treated as equals. If a

# connection fails, the next address will be tried and if successful will

# remain connected until it fails

#round\_robin false

# Set the client id to use on the remote end of this bridge connection. If not

# defined, this defaults to 'name.hostname' where name is the connection name

# and hostname is the hostname of this computer.

# This replaces the old "clientid" option to avoid confusion. "clientid"

# remains valid for the time being.

#remote\_clientid

# Set the clientid to use on the local broker. If not defined, this defaults to

# 'local.<clientid>'. If you are bridging a broker to itself, it is important

# that local\_clientid and clientid do not match.

#local\_clientid

# Set the clean session variable for this bridge.

# When set to true, when the bridge disconnects for any reason, all

# messages and subscriptions will be cleaned up on the remote

# broker. Note that with cleansession set to true, there may be a

# significant amount of retained messages sent when the bridge

# reconnects after losing its connection.

# When set to false, the subscriptions and messages are kept on the

# remote broker, and delivered when the bridge reconnects.

#cleansession false

# If set to true, publish notification messages to the local and remote brokers

# giving information about the state of the bridge connection. Retained

# messages are published to the topic $SYS/broker/connection/<clientid>/state

# unless the notification\_topic option is used.

# If the message is 1 then the connection is active, or 0 if the connection has

# failed.

#notifications true

# Choose the topic on which notification messages for this bridge are

# published. If not set, messages are published on the topic

# $SYS/broker/connection/<clientid>/state

#notification\_topic

# Set the keepalive interval for this bridge connection, in

# seconds.

#keepalive\_interval 60

# Set the start type of the bridge. This controls how the bridge starts and

# can be one of three types: automatic, lazy and once. Note that RSMB provides

# a fourth start type "manual" which isn't currently supported by mosquitto.

#

# "automatic" is the default start type and means that the bridge connection

# will be started automatically when the broker starts and also restarted

# after a short delay (30 seconds) if the connection fails.

#

# Bridges using the "lazy" start type will be started automatically when the

# number of queued messages exceeds the number set with the "threshold"

# parameter. It will be stopped automatically after the time set by the

# "idle\_timeout" parameter. Use this start type if you wish the connection to

# only be active when it is needed.

#

# A bridge using the "once" start type will be started automatically when the

# broker starts but will not be restarted if the connection fails.

#start\_type automatic

# Set the amount of time a bridge using the automatic start type will wait

# until attempting to reconnect. Defaults to 30 seconds.

#restart\_timeout 30

# Set the amount of time a bridge using the lazy start type must be idle before

# it will be stopped. Defaults to 60 seconds.

#idle\_timeout 60

# Set the number of messages that need to be queued for a bridge with lazy

# start type to be restarted. Defaults to 10 messages.

# Must be less than max\_queued\_messages.

#threshold 10

# If try\_private is set to true, the bridge will attempt to indicate to the

# remote broker that it is a bridge not an ordinary client. If successful, this

# means that loop detection will be more effective and that retained messages

# will be propagated correctly. Not all brokers support this feature so it may

# be necessary to set try\_private to false if your bridge does not connect

# properly.

#try\_private true

# Set the username to use when connecting to a broker that requires

# authentication.

# This replaces the old "username" option to avoid confusion. "username"

# remains valid for the time being.

#remote\_username

# Set the password to use when connecting to a broker that requires

# authentication. This option is only used if remote\_username is also set.

# This replaces the old "password" option to avoid confusion. "password"

# remains valid for the time being.

#remote\_password

# -----------------------------------------------------------------

# Certificate based SSL/TLS support

# -----------------------------------------------------------------

# Either bridge\_cafile or bridge\_capath must be defined to enable TLS support

# for this bridge.

# bridge\_cafile defines the path to a file containing the

# Certificate Authority certificates that have signed the remote broker

# certificate.

# bridge\_capath defines a directory that will be searched for files containing

# the CA certificates. For bridge\_capath to work correctly, the certificate

# files must have ".crt" as the file ending and you must run "c\_rehash <path to

# capath>" each time you add/remove a certificate.

#bridge\_cafile

#bridge\_capath

# Path to the PEM encoded client certificate, if required by the remote broker.

#bridge\_certfile

# Path to the PEM encoded client private key, if required by the remote broker.

#bridge\_keyfile

# When using certificate based encryption, bridge\_insecure disables

# verification of the server hostname in the server certificate. This can be

# useful when testing initial server configurations, but makes it possible for

# a malicious third party to impersonate your server through DNS spoofing, for

# example. Use this option in testing only. If you need to resort to using this

# option in a production environment, your setup is at fault and there is no

# point using encryption.

#bridge\_insecure false

# -----------------------------------------------------------------

# PSK based SSL/TLS support

# -----------------------------------------------------------------

# Pre-shared-key encryption provides an alternative to certificate based

# encryption. A bridge can be configured to use PSK with the bridge\_identity

# and bridge\_psk options. These are the client PSK identity, and pre-shared-key

# in hexadecimal format with no "0x". Only one of certificate and PSK based

# encryption can be used on one

# bridge at once.

#bridge\_identity

#bridge\_psk

# =================================================================

# External config files

# =================================================================

# External configuration files may be included by using the

# include\_dir option. This defines a directory that will be searched

# for config files. All files that end in '.conf' will be loaded as

# a configuration file. It is best to have this as the last option

# in the main file. This option will only be processed from the main

# configuration file. The directory specified must not contain the

# main configuration file.

#include\_dir

# =================================================================

# rsmb options - unlikely to ever be supported

# =================================================================

#ffdc\_output

#max\_log\_entries

#trace\_level

#trace\_output