FREE eBook

LEARNING rabbitmq

Free unaffiliated eBook created from
Stack Overflow contributors.

#rabbitmq

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: rabbitmq

It is an unofficial and free rabbitmq ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official rabbitmq.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with rabbitmq

Remarks

This section provides an overview of what rabbitmq is, and why a developer might want to use it.

It should also mention any large subjects within rabbitmq, and link out to the related topics. Since the Documentation for rabbitmq is new, you may need to create initial versions of those related topics.

Versions

Version	Release notes	Release date
3.6.5	Release notes	2016-08-05
3.6.4	Release notes	2016-07-29
3.6.3	Release notes	2016-07-06
3.6.2	Release notes	2016-05-19
3.6.1	Release notes	2016-03-01
3.6.0	Release notes	2015-12-22
3.5.7	Release notes	2015-12-15
3.5.6	Release notes	2015-10-07
3.5.5	Release notes	2015-09-24
3.5.4	Release notes	2015-07-22
3.5.3	Release notes	2015-05-22
3.5.2	Release notes	2015-05-12
3.5.1	Release notes	2015-04-03
3.5.0	Release notes	2015-03-11
3.4.4	Release notes	2015-02-11
3.4.3	Release notes	2015-01-07
3.4.2	Release notes	2014-11-26

Version	Release notes	Release date
3.4.1	Release notes	2014-10-29
3.4.0	Release notes	2014-10-21
3.3.5	Release notes	2014-08-11
3.3.4	Release notes	2014-06-24
3.3.3	Release notes	2014-06-17
3.3.2	Release notes	2014-06-09
3.3.1	Release notes	2014-04-29
3.3.0	Release notes	2014-04-02
3.2.4	Release notes	2014-03-04
3.2.3	Release notes	2014-01-23
3.2.2	Release notes	2013-12-11
3.2.1	Release notes	2013-11-07
3.2.0	Release notes	2013-10-23
3.1.5	Release notes	2013-08-15
3.1.4	Release notes	2013-08-07
3.1.3	Release notes	2013-06-26
3.1.2	Release notes	2013-06-24
3.1.1	Release notes	2013-05-21
3.1.0	Release notes	2013-05-01

Examples

Installing RabbitMQ on Ubuntu Server

A quick note before actually installing RabbitMQ: Ubuntu 14.04's Erlang packages have issues if you are using SSL with RabbitMQ, so you'll need to install a newer version than what the Ubuntu package maintainers provide, so use the binaries at https://www.erlang-solutions.com/resources/download.html, for Erlang 17.0 or higher.

Add RabbitMQ to the package repositories list:

```
echo 'deb http://www.rabbitmq.com/debian/ testing main' |
    sudo tee /etc/apt/sources.list.d/rabbitmq.list
```

And then add the signing key:

```
wget -O- https://www.rabbitmq.com/rabbitmq-release-signing-key.asc |
    sudo apt-key add -
```

Then update and install:

```
sudo apt-get update && sudo apt-get install rabbitmq-server
```

RabbitMQ 'Hello World'

This code create a producer which send two messages to a queue, and a consumer which receives all the messages from that queue.

Code for producer.py (using the pika 0.10.0 Python client):

```
import pika

connection = pika.BlockingConnection(pika.ConnectionParameters(
        host='localhost'))
channel = connection.channel()

channel.queue_declare(queue='queueName')

channel.basic_publish(exchange='',
                      routing_key='queueName',
                      body='Hello')
channel.basic_publish(exchange='',
                      routing_key='queueName',
                      body='World!')
print("Message sent")
connection.close()
```

Code for consumer.py:

```
import pika

connection = pika.BlockingConnection(pika.ConnectionParameters(
        host='localhost'))
channel = connection.channel()

channel.queue_declare(queue='queueName')

def callback(ch, method, properties, body):
    print("Received message: %r" % body)

channel.basic_consume(callback,
                      queue='queueName',
                      no_ack=True)

print('Waiting for messages...')
```

```
channel.start_consuming()
```

The output is:

```
$ python receive.py
Waiting for messages...
Received message: 'Hello'
Received message: 'World!'
```

Other examples are available in the RabbitMQ tutorial page for other languages.

Read Getting started with rabbitmq online: https://riptutorial.com/rabbitmq/topic/2816/getting-started-with-rabbitmq

Chapter 2: Configuring RabbitMQ

Examples

Configuring RabbitMQ on *nix Systems

Configuring your environment

RabbitMQ looks for an set of environment variables in `/etc/rabbitmq/rabbitmq-env.conf`. If it does not exist, it assumes default values. All values in `rabbitmq-env.conf` get exported to the environment the RabbitMQ server runs in with a `RABBITMQ_` prefix; this prefix is not included in the configuration file. The following variables can be set (in a `key=value` syntax):

`NODENAME`: Sets the name of the RabbitMQ node.

`CONFIG_FILE`: Sets the location of the RabbitMQ configuration file (not this environment file)

`NODE_IP_ADDRESS`: Sets the specific IP address to listen on.

`NODE_PORT`: Sets the port to listen on

`DIST_PORT`: Sets the port to listen on for clustering

`USE_LONGNAME`: Boolean value indicating whether or not to use fully qualified names to identify nodes. Useful when in environments with identical short names.

`CTL_ERL_ARGS`: Sets arguments for the `erl` command invoked when running `rabbitmqctl`. Useful for debugging.

`SERVER_ERL_ARGS`: Sets arguments for the `erl` command invoked when starting RabbitMQ. Setting this value overrides the default (see example config below).

`SERVER_ADDITIONAL_ERL_ARGS`: Sets additional arguments for the `erl` command invoked when starting RabbitMQ. Setting this value appends to the `SERVER_ERL_ARGS` variable, and is empty by default.

`SERVER_START_ARGS`: Also set for the `erl` command invoked when starting RabbitMQ.

An example configuration file (all values are actually defaults):

```
NODENAME=rabbit@localhost
CONFIG_FILE=/etc/rabbitmq/rabbitmq.config
NODE_IP_ADDRESS=""
NODE_PORT=5672
DIST_PORT=25672
USE_LONGNAME=false
CTL_ERL_ARGS=""
SERVER_ERL_ARGS="+K true +A30 +P 1048576 -kernel inet_default_connect_options
[{nodelay,true}]"
```

```
SERVER_ADDITIONAL_ERL_ARGS=""
SERVER_START_ARGS=""
```

Configuring RabbitMQ

Note: RabbitMQ's configuration file is in the standard Erlang configuration syntax. If you're not familiar with Erlang, this may be confusing at first. It follows the following format:

```
[
    {rabbit,
        [
            {config_value_1, []},
            {config_value_2, []}
        ]
    },
    {additional_rabbitmq_plugins,
        [...]
    }
]
```

RabbitMQ's section of the configuration has the following keys (pulled from https://www.rabbitmq.com/configure.html):

- `tcp_listeners`: List of ports on which to listen for AMQP connections (without SSL). Can contain integers (meaning "listen on all interfaces") or tuples such as `{"127.0.0.1", 5672}` to listen on one or more interfaces.

  Default: `[5672]`

- `num_tcp_acceptors`: Number of Erlang processes that will accept connections for the TCP listeners.

  Default: `10`

- `handshake_timeout`: Maximum time for AMQP 0-8/0-9/0-9-1 handshake (after socket connection and SSL handshake), in milliseconds.

  Default: `10000`

- `ssl_listeners`: As above, for SSL connections.

  Default: `[]`

- `num_ssl_acceptors`: Number of Erlang processes that will accept connections for the SSL listeners.

  Default: `1`

- `ssl_options`: SSL configuration.

  Default: `[]`

- `ssl_handshake_timeout`: SSL handshake timeout, in milliseconds.

  Default: `5000`

- `vm_memory_high_watermark`: Memory threshold at which the flow control is triggered. See the memory-based flow control documentation.

  Default: `0.4`

- `vm_memory_high_watermark_paging_ratio`: Fraction of the high watermark limit at which queues start to page messages out to disc to free up memory. See the memory-based flow control documentation.

  Default: `0.5`

- `disk_free_limit`: Disk free space limit of the partition on which RabbitMQ is storing data. When available disk space falls below this limit, flow control is triggered. The value may be set relative to the total amount of RAM (e.g. `{mem_relative, 1.0}`). The value may also be set to an integer number of bytes. Or, alternatively, in information units (e.g `"50MB"`). By default free disk space must exceed 50MB. See the Disk Alarms documentation.

  Default: `50000000`

- `log_levels`: Controls the granularity of logging. The value is a list of log event category and log level pairs.

  The level can be one of `none` (no events are logged), `error` (only errors are logged), `warning` (only errors and warning are logged), `info` (errors, warnings and informational messages are logged), or `debug` (errors, warnings, informational messages and debugging messages are logged).

  At present there are four categories defined. Other, currently uncategorised, events are always logged.

  The categories are:

  - `channel` - for all events relating to AMQP channels
  - `connection` - for all events relating to network connections
  - `federation` - for all events relating to federation
  - `mirroring` - for all events relating to mirrored queues

  Default: `[{connection, info}]`

- `frame_max`: Maximum permissible size of a frame (in bytes) to negotiate with clients. Setting to 0 means "unlimited" but will trigger a bug in some QPid clients. Setting a larger value may improve throughput; setting a smaller value may improve latency.

  Default: `131072`

- `channel_max`: Maximum permissible number of channels to negotiate with clients. Setting to 0 means "unlimited". Using more channels increases memory footprint of the broker.

  Default: `0`

- `channel_operation_timeout`: Channel operation timeout in milliseconds (used internally, not directly exposed to clients due to messaging protocol differences and limitations).

  Default: `15000`

- `heartbeat`: Value representing the heartbeat delay, in seconds, that the server sends in the connection.tune frame. If set to 0, heartbeats are disabled. Clients might not follow the server suggestion, see the AMQP reference for more details. Disabling heartbeats might improve performance in situations with a great number of connections, but might lead to connections dropping in the presence of network devices that close inactive connections.

  Default: `60` (`580` prior to release 3.5.5)

- `default_vhost`: Virtual host to create when RabbitMQ creates a new database from scratch. The exchange `amq.rabbitmq.log` will exist in this virtual host.

  Default: `<<"/">>`

- `default_user`: User name to create when RabbitMQ creates a new database from scratch.

  Default: `<<"guest">>`

- `default_pass`: Password for the default user.

  Default: `<<"guest">>`

- `default_user_tags`: Tags for the default user.

  Default: `[administrator]`

- `default_permissions`: Permissions to assign to the default user when creating it.

  Default: `[<<".*">>, <<".*">>, <<".*">>]`

- `loopback_users`: List of users which are only permitted to connect to the broker via a loopback interface (i.e. localhost).

  If you wish to allow the default guest user to connect remotely, you need to change this to `[]`.

  Default: `[<<"guest">>]`

- `cluster_nodes`: Set this to cause clustering to happen automatically when a node starts for the very first time. The first element of the tuple is the nodes that the node will try to cluster to. The second element is either disc or ram and determines the node type.

  Default: `{[], disc}`

- `server_properties`: List of key-value pairs to announce to clients on connection.

  Default: `[]`

- `collect_statistics`: Statistics collection mode. Primarily relevant for the management plugin. Options are:

  - `none` (do not emit statistics events)
  - `coarse` (emit per-queue / per-channel / per-connection statistics)
  - `fine` (also emit per-message statistics)

  You probably don't want to change this yourself.

  Default: `none`

- `collect_statistics_interval`: Statistics collection interval in milliseconds. Primarily relevant for the management plugin.

  Default: `5000`

- `auth_mechanisms`: SASL authentication mechanisms to offer to clients.

  Default: `['PLAIN', 'AMQPLAIN']`

- `auth_backends`: List of authentication / authorization backends to use. This list can contain names of modules (in which case the same module is used for both authentication and authorization), or 2-tuples like `{ModN, ModZ}` in which case ModN is used for authentication and ModZ is used for authoriZation.

  In the 2-tuple case, ModZ can be replaced by a list, all the elements of which must confirm each authorisation query, e.g. `{ModN, [ModZ1, ModZ2]}`. This allows authorisation plugins to mix-in and provide additional security constraints.

  Other databases than `rabbit_auth_backend_internal` are available through plugins.

  Default: `[rabbit_auth_backend_internal]`

- `reverse_dns_lookups`: Set to true to have RabbitMQ perform a reverse DNS lookup on client connections, and present that information through `rabbitmqctl` and the management plugin.

  Default: `false`

- `delegate_count`: Number of delegate processes to use for intra-cluster communication. On a machine which has a very large number of cores and is also part of a cluster, you may wish to increase this value.

  Default: `16`

- `trace_vhosts`: Used internally by the tracer. You shouldn't change this.

Default: `[]`

- `tcp_listen_options`: Default socket options. You probably don't want to change this.

  Default:

  ```
  [{backlog,       128},
   {nodelay,       true},
   {exit_on_close, false}]
  ```

- `hipe_compile`: Set to true to precompile parts of RabbitMQ with HiPE, a just-in-time compiler for Erlang. This will increase server throughput at the cost of increased startup time.

  You might see 20-50% better performance at the cost of a few minutes delay at startup. These figures are highly workload- and hardware-dependent.

  HiPE support may not be compiled into your Erlang installation. If it is not, enabling this option will just cause a warning message to be displayed and startup will proceed as normal. For example, Debian / Ubuntu users will need to install the `erlang-base-hipe` package.

  HiPE is not available at all on some platforms, notably including Windows.

  HiPE has known issues in Erlang/OTP versions prior to 17.5. Using a recent Erlang/OTP version is highly recommended for HiPE.

  Default: `false`

- `cluster_partition_handling`: How to handle network partitions. Available modes are:

  - `ignore`
  - `pause_minority`
  - `{pause_if_all_down, [nodes], ignore | autoheal}` where `[nodes]` is a list of node names (ex: `['rabbit@node1', 'rabbit@node2']`)
  - `autoheal`

  See the documentation on partitions for more information.

  Default: `ignore`

- `cluster_keepalive_interval`: How frequently nodes should send keepalive messages to other nodes (in milliseconds). Note that this is not the same thing as `net_ticktime`; missed keepalive messages will not cause nodes to be considered down.

  Default: `10000`

- `queue_index_embed_msgs_below`: Size in bytes of message below which messages will be embedded directly in the queue index. You are advised to read the persister tuning documentation before changing this.

  Default: `4096`

- `msg_store_index_module`

: Implementation module for queue indexing. You are advised to read the persister tuning documentation before changing this.

Default: `rabbit_msg_store_ets_index`

- `backing_queue_module`: Implementation module for queue contents. You probably don't want to change this.

  Default: `rabbit_variable_queue`

- `msg_store_file_size_limit`: Tunable value for the persister. You almost certainly should not change this.

  Default: `16777216`

- `mnesia_table_loading_timeout`: Timeout used when waiting for Mnesia tables in a cluster to become available.

  Default: `30000`

- `queue_index_max_journal_entries`: Tunable value for the persister. You almost certainly should not change this.

  Default: `65536`

- `queue_master_locator`: Queue master location strategy. Available strategies are:

  - `<<"min-masters">>`
  - `<<"client-local">>`
  - `<<"random">>`

  See the documentation on queue master location for more information.

  Default: `<<"client-local">>`

Read Configuring RabbitMQ online: https://riptutorial.com/rabbitmq/topic/4018/configuring-rabbitmq

Credits

S. No	Chapters	Contributors
1	Getting started with rabbitmq	Community, drheart, legoscia, Lorenzo Belli
2	Configuring RabbitMQ	drheart, legoscia