

# RabbitMQ for PCF®

Version 1.9


## User's Guide

© Copyright Pivotal Software Inc, 2013-present

## Table of Contents

Table of Contents	2
RabbitMQ for PCF	3
RabbitMQ® for PCF Release Notes	6
On-Demand Service Architecture	15
Installing and Configuring RabbitMQ for PCF as an On-Demand Service	18
Monitoring and KPIs for On-Demand RabbitMQ for PCF	25
Troubleshooting On-Demand RabbitMQ for PCF	33
RabbitMQ for PCF Operations FAQ's	49
Using On-Demand RabbitMQ for PCF	52
RabbitMQ® Environment Variables	56
Troubleshooting Instances	58
Installing and Configuring RabbitMQ for PCF as a Pre-Provisioned Service	61
Creating Isolation with the RabbitMQ for PCF Replicator	70
Configuring RabbitMQ in an IPsec environment	73
Limitations & Risks	73
Installing the tile	73
Deploying the RabbitMQ® Service	74
RabbitMQ® for Pivotal Cloud Foundry	78
Upgrades	78
Default policies for the RabbitMQ® Service	80
RabbitMQ Policy	80
Monitoring and KPIs for Pre-Provisioned RabbitMQ for PCF	82
Clustering and Network Partitions	90
Managing the RabbitMQ® Service	92

## RabbitMQ for PCF

 **Note:** RabbitMQ for PCF v1.9 is no longer supported. The support period for v1.9 has expired. To stay up-to-date with the latest software and security updates, upgrade to RabbitMQ for PCF v1.10 or later.

## About RabbitMQ for PCF

RabbitMQ for Pivotal Cloud Foundry (PCF) enables PCF app developers to provision and use the RabbitMQ message broker with a single command.

As of v1.8, RabbitMQ for PCF supports two types of service, an *on-demand* service and a *pre-provisioned* service. This table summarizes the main differences between the two:

	Available Since	VMs it Runs On	How VMs are Created	Metrics Name Prefix
<b>On-Demand Service</b>	v1.8	Dedicated VM that serves a single service instance	PCF creates each VM on-demand when app developer creates service instance	<code>p.rabbitmq</code> (with a dot) in future versions. As of v1.8.2, emitted as <code>p-rabbitmq</code> , a <a href="#">known issue</a> .
<b>Pre-Provisioned Service</b>	v1.2	Multi-tenant VMs shared by apps across PCF deployment	PCF creates all VMs when operator deploys or updates service	<code>p-rabbitmq</code> (with a dash)

This RabbitMQ for PCF v1.9 documentation describes both service types. Documentation for RabbitMQ for PCF v1.7 and earlier only describes a pre-provisioned service.

## What are Dedicated Instances

In RabbitMQ for PCF versions before v1.8.0, the RabbitMQ service instances correspond to a unique RabbitMQ virtual host on the multi-tenant RabbitMQ cluster. RabbitMQ for PCF v1.8.0 introduced [On-Demand Broker \(ODB\)](#) [↗](#) support. That means that a new, single-tenant, cluster can be created and dedicated to a single app.

For more information, see [On-Demand Service Architecture](#).

## About RabbitMQ

RabbitMQ is a fast and dependable open-source message server, which supports a wide range of use cases including reliable integration, content-based routing and global data delivery, and high-volume monitoring and data ingestion.

Emerging as the de facto standard for cloud messaging, RabbitMQ is used for efficient communication between servers, apps and devices, and creates lasting value by enabling rapid development of modern decentralized app and data architectures that can scale with your business needs.

## Product Snapshot

The following table provides version and version-support information about RabbitMQ for PCF.

Element	Details
Version	v1.9.21
Release date	March 28, 2018
Software component version	RabbitMQ OSS v3.6.15
Compatible Ops Manager version(s)	v1.10.x, v1.11.x, and v1.12.x
Compatible Elastic Runtime version(s)	v1.10.3 and later
IaaS support	AWS, Azure, GCP, OpenStack, and vSphere
IPsec support	No

## Features

### On-Demand

- Provision on-demand single node dedicated instances of RabbitMQ with a unique RabbitMQ virtual host
- Bind apps to an instance of the plan, providing unique credentials for each binding
- Management dashboard access to app developers
- Deployment into an availability zone specified by the plan
- Automated upgrades of RabbitMQ for major, minor, and patch releases (see release notes for downtime requirements)
- RabbitMQ Syslog forwarding configuration inherited from the pre-provisioned configuration
- RabbitMQ metrics are exposed on the Firehose
- On-demand instances come with `shovel` and `federation` plugins installed

### Pre-Provisioned

- Provision an instance of the RabbitMQ service, which corresponds to a unique RabbitMQ virtual host
- Bind apps to an instance of the plan, providing unique credentials for each binding
- Management dashboard access to PCF Operators and app developers
- Deployment across multiple availability zones, with nodes striped across the AZs automatically
- Enable SSL (Secure Sockets Layer) for the AMQP, MQTT, STOMP protocols
- HAProxy load balancer across all nodes to balance connections
- Plugin configuration can be easily changed at any time and the cluster redeployed and updated
- The cluster topology can be changed and easily scaled out
- Automated upgrades of RabbitMQ for major, minor, and patch releases (see release notes for downtime requirements)
- Configure the end point for the RabbitMQ Syslog
- RabbitMQ and HAProxy metrics are exposed on the Firehose
- Syslog forwarding on by default

## Release Notes and Known Issues

Check the [release notes](#) for your release version for important information and known issues. To see release notes for another version, select the version from the dropdown list at the top of the page.

## RabbitMQ for PCF and Other PCF Services

Some PCF services offer *on-demand* service plans. These plans let developers provision service instances when they want.

These contrast with the more common *pre-provisioned* service plans, which require operators to provision the service instances during installation and configuration through the service tile UI.

The following PCF services offer on-demand service plans:

- MySQL for PCF v2.0 and later
- RabbitMQ for PCF
- Redis for PCF
- Pivotal Cloud Cache (PCC)

These services package and deliver their on-demand service offerings differently. For example, some services, like Redis for PCF, have one tile, and you configure the tile differently depending on whether you want on-demand service plans or pre-provisioned service plans.

For other services, like PCC, you install one tile for on-demand service plans and a different tile for pre-provisioned service plans.

The following table lists and contrasts the different ways that PCF services package on-demand and pre-provisioned service offerings.

PCF service tile	Standalone product related to the service	Versions supporting on demand	Versions supporting pre-provisioned
RabbitMQ for PCF	Pivotal RabbitMQ	v1.8 and later	All versions
Redis for PCF	Redis	v1.8 and later	All versions
MySQL for PCF	MySQL	v2.x (based on Percona Server)	v1.x (based on MariaDB and Galera)
PCC	Pivotal GemFire	All versions	<i>NA</i>
GemFire for PCF	Pivotal GemFire	<i>NA</i>	All versions

Please provide any bugs, feature requests, or questions to the [PCF Feedback list](#).

## RabbitMQ® for PCF Release Notes

### Upgrade to the Latest Version

Pivotal recommends that you upgrade to the latest version of your current minor line, then upgrade to the latest available version of the new minor line. For example, if you use an older v1.7.x version, upgrade to the latest v1.7.x version before upgrading to the latest v1.8.x version.

See the [Product Compatibility Matrix](#) for product versions and upgrade paths.

### View Release Notes for Another Version

To view the release notes for another product version, select the version from the drop-down list at the top of this page.

## v1.9.21

**Release Date:** March 28, 2018

Features:

- Requires stemcell [3421.44](#)
- Updates Java package to version 1.9

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.15
- Erlang v19.3.6.4
- HAProxy v1.6.13

## v1.9.17

**Release Date:** February 26, 2018

Features:

- **[ Security Fix ]** Requires stemcell [3421.42](#) Fixes the following bugs:
  - Upgrades failed when SYSLOG was not configured.
  - Certain special characters could not be used with the RabbitMQ administrator password. You can now use special characters, with the exception of ``` and `'`, in RabbitMQ usernames and passwords.
  - If you changed the RabbitMQ admin username, the user lost access to preexisting vhosts.

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).

- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.15
- Erlang v19.3.6.4
- HAProxy v1.6.13

## v1.9.16

**Release Date:** January 30, 2018

Features:

- Update RabbitMQ to v3.6.15. For more information, see [RabbitMQ 3.6.15 Release Notes](#).

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- The RabbitMQ administrator password must be a combination of uppercase and lowercase alphanumerics. Special characters such as `!@#$%^&*+,-./:;<=>"'` can lead to issues logging into the Management UI as that user.

Packages:

- OSS RabbitMQ v3.6.15
- Erlang v19.3.6.4
- HAProxy v1.6.13

## v1.9.15 - Withdrawn

**Release Date:** January 24, 2018

**Withdrawal Date:** February 5, 2018

This release was withdrawn due to an issue upgrading when there are special characters in the RabbitMQ administrator credentials. Skip this version and upgrade to [v1.9.16](#) or later.

Features:

- [ **Security Fix** ] Requires stemcell [3421.38](#) to address Spectre vulnerabilities.

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.14
- Erlang v19.3.6.4
- HAProxy v1.6.13

## v1.9.14

Release Date: January 19, 2018

Features:

- [ **Security Fix** ] Requires stemcell [3421.37](#) to address GNU C Library vulnerabilities.

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.14
- Erlang v19.3.6.4
- HAProxy v1.6.13

## v1.9.13

Release Date: January 11, 2018

Features:

- [ **Security Fix** ] Requires stemcell [3421.36](#) to address the Meltdown security issue. For more information about Meltdown, see [Pivotal Vulnerability Report: Meltdown and Spectre Attacks](#).

On-Demand

- `smoke-tests` errand is renamed to `on-demand-broker-smoke-tests`
- Some CF timeouts for smoke tests have increased

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.14
- Erlang v19.3.6.4
- HAProxy v1.6.13

## v1.9.11

Release Date: December 14, 2017

Features:

- Requires stemcell [3421.34](#)

Known issues:



- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.14
- Erlang v19.3.6.4
- HAProxy v1.6.13

## v1.9.10

**Release Date:** December 6, 2017

Features:

- Requires stemcell 3421.32
- Update to RabbitMQ v3.6.14 (release notes available [here](#))
- Update to Erlang 19.3.6.4
- Go update (security fixes)
- Minor updates in Pre-Provisioned service (multi-tenant broker):
  - Fix misspelling in log message
  - Improve service offering description

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.14
- Erlang v19.3.6.4
- HAProxy v1.6.13

## v1.9.9

**Release Date:** November 2, 2017

Features:

- Requires stemcell 3421.31
- Enables `shovel` and `federation` plugins by default on on-demand instances
- Enables the creation of administrator users for on-demand instances
- Increases max HAProxy connections to 64000 for pre-provisioned plan
- Removes the stats UI on HAProxy instances for pre-provisioned plan
- Allows for more granular TLS protocol version selection for pre-provisioned plan
- Fixes known issue where some invalid characters in the administrator password
- Fixes a known issue where on-demand metrics were named `/p-rabbitmq` instead of `/p.rabbitmq`

- Adds queue depth, memory alarm, and disk alarm metrics

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.12
- Erlang v19.3.6.2
- HAProxy v1.6.13

## v1.9.8

**Release Date:** September 20, 2017

Features:

- Requires stemcell 3421.26

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- Certain special characters cannot be used with the RabbitMQ administrator password and using them can lead to issues logging into the Management UI as the administrator.

Packages:

- OSS RabbitMQ v3.6.12
- Erlang v19.3.6
- HAProxy v1.6.13

## v1.9.7

**Release Date:** September 14, 2017

Features:

- Update to RabbitMQ 3.6.12 — this involves [major changes](#) to the memory calculations.

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand-run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.12
- Erlang v19.3.6
- HAProxy v1.6.13

## v1.9.6

**Release Date:** September 8, 2017

Features:

- Adds the [Partition Indicator](#) metric.
- Smoke tests are more resilient to `cf domain` re-ordering.
- Updates `cf-cli` to v6.30 for `cf` command execution.

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime and can result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand-run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.10
- Erlang v19.3.6
- HAProxy v1.6.13

## v1.9.5

**Release Date:** August 21, 2017

- Update to RabbitMQ 3.6.10
- Requires stemcell 3421.20
- This version of the tile can only be installed in an Ops Manager v1.10.3+ and v1.11 environments

Features:

- Stemcell upgrade from the 3363 line to the 3421 line. Primary reason for the upgrade is to include the changes in 3421.18 which contains a fix for an rsyslog hanging issue that occurs when the IPsec add-on is installed.

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and may result in failed deployments. For details see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For details, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

## v1.9.4

**Release Date:** August 15, 2017

- Update to RabbitMQ 3.6.10
- Requires stemcell 3363.30

- This version of the tile can only be installed in an Ops Manager v1.10.3+ and v1.11 environments

#### Features:

- This is a security release addressed by a new stemcell

#### Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and may result in failed deployments. For details see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#)
- Changing networks and/or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For details, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

#### Packages:

- OSS RabbitMQ v3.6.10
- Erlang v19.3.6
- HAProxy v1.6.11

## v1.9.3

**Release Date:** August 8, 2017

- Update to RabbitMQ 3.6.10
- Requires stemcell 3363.29
- This version of the tile can only be installed in an Ops Manager v1.10.3+ and v1.11 environments

#### Features:

- This is a security release addressed by a new stemcell

#### Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and may result in failed deployments. For details see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#)
- Changing networks and/or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For details, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

#### Packages:

- OSS RabbitMQ v3.6.10
- Erlang v19.3.6
- HAProxy v1.6.11

## v1.9.2

**Release Date:** August 7, 2017

- Update to RabbitMQ 3.6.10
- Requires stemcell 3363.26
- This version of the tile can only be installed in an Ops Manager v1.10.3+ and v1.11 environments

#### Features:

- This version of the tile will use RabbitMQ Native Cluster formation rather than the rabbitmq-clusterer plugin.

#### Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and may result in failed deployments. For details see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For details, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

Packages:

- OSS RabbitMQ v3.6.10
- Erlang v19.3.6
- HAProxy v1.6.11

## v1.9.1

Release Date: July 7, 2017

- Update to RabbitMQ 3.6.10
- Requires stemcell 3363.26
- This version of the tile can only be installed in an Ops Manager v1.10.3+ and v1.11 environments

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and may result in failed deployments. For details see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For details, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- RabbitMQ for PCF v1.9.1 and earlier has a bug that does not delete management users, `mu-*`, when a service is deleted. To automate deletion of these users, contact [Pivotal Support](#).

Packages:

- OSS RabbitMQ v3.6.10
- Erlang v19.3.6
- HAProxy v1.6.11

## v1.9.0

Release Date: June 30, 2017

- Requires stemcell 3363.26
- This version of the tile can only be installed in an Ops Manager v1.10.3+ and v1.11 environments

Known issues:

- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and may result in failed deployments. For details see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For details, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- RabbitMQ for PCF v1.9.1 and earlier has a bug that does not delete management users, `mu-*`, when a service is deleted. To automate deletion of these users, contact [Pivotal Support](#).

Packages:

- OSS RabbitMQ v3.6.9

- Erlang v19.3.6
- HAProxy v1.6.11

## On-Demand Service Architecture

This topic describes the architecture for on-demand RabbitMQ® for Pivotal Cloud Foundry (PCF).

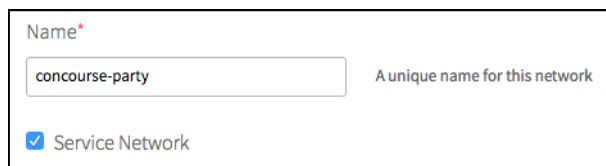
For information about architecture of the older, pre-provisioned service, see [Deploying the RabbitMQ® Service](#).

## BOSH 2.0 and Service Networks

Before BOSH 2.0, cloud operators pre-provisioned service instances from Ops Manager. In the Ops Manager Director **Networking** pane, they allocated a block of IP addresses for the service instance pool, and under **Resource Config** they provisioned pool VM resources, specifying the CPU, hard disk, and RAM they would use. All instances had to be provisioned at the same level. With each `create-service` request from a developer, Ops Manager handed out a static IP address from this block, and with each `delete-service` it cleaned up the VM and returned it to the available pool.

With BOSH 2.0 dynamic networking and Cloud Foundry asynchronous service provisioning, operators can now define a dynamically-provisioned service network that hosts instances more flexibly. The service network runs separate from the PCF default network. While the default network hosts VMs launched by Ops Manager, the VMs running in the service network are created and provisioned on-demand by BOSH, and BOSH lets the IaaS assign IP addresses to the service instance VMs. Each dynamic network attached to a job instance is typically represented as its own Network Interface Controller in the IaaS layer.

Operators enable on-demand services when they deploy PCF, by creating one or more service networks in the Ops Manager Director **Create Networks** pane and selecting the **Service Network** checkbox. Designating a network as a service network prevents Ops Manager from creating VMs in the network, leaving instance creation to the underlying BOSH.



The screenshot shows a form for creating a new network. It has a label 'Name\*' above a text input field containing 'concourse-party'. To the right of the input field is the text 'A unique name for this network'. Below the input field is a checkbox labeled 'Service Network' which is checked with a blue checkmark.

When they deploy an on-demand service, operators select the service network when configuring the tile for that on-demand service.

## Default Network and Service Network

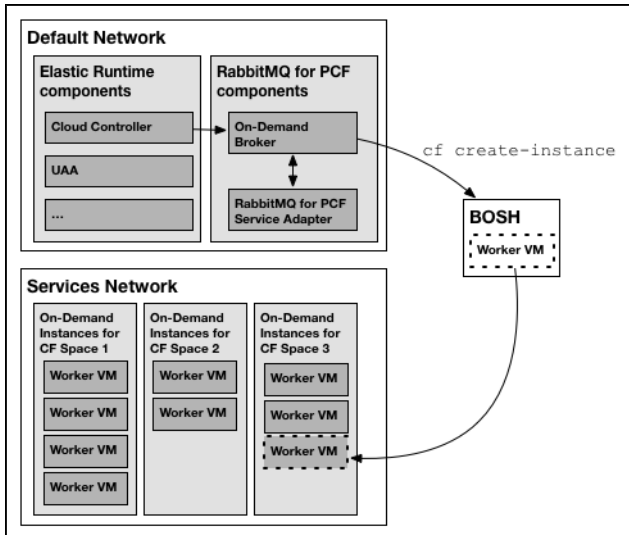
Like other on-demand PCF services, on-demand RabbitMQ for PCF relies on the BOSH 2.0 ability to dynamically deploy VMs in a dedicated network. The on-demand service broker uses this capability to create single-tenant service instances in a dedicated service network.

On-demand services use the dynamically-provisioned service network to host the single-tenant worker VMs that run as service instances within development spaces. This architecture lets developers provision IaaS resources for their service instances at creation time, rather than the operator pre-provisioning a fixed quantity of IaaS resources when they deploy the service broker.

By making services single-tenant, where each instance runs on a dedicated VM rather than sharing VMs with unrelated processes, on-demand services eliminate the “noisy neighbor” problem when one application hogs resources on a shared cluster. Single-tenant services can also support regulatory compliance where sensitive data must be compartmentalized across separate machines.

An on-demand service splits its operations between the default network and the service network. Shared components of the service, such as executive controllers and databases, run centrally on the default network along with the Cloud Controller, UAA, and other PCF components. The worker pool deployed to specific spaces runs on the service network.

The diagram below shows worker VMs in an on-demand service instance, such as RabbitMQ for PCF, running on a separate services network, while other components run on the default network.



## Required Networking Rules for On-Demand Services

Prior to deploying any service tile that uses the on-demand broker (ODB), the operator must request the network connections needed to allow various components of Pivotal Cloud Foundry (PCF) to communicate with ODB. The specifics of how to open those connections varies for each IaaS.

The following table shows the responsibilities of the key components in an on-demand architecture.

Key Components	Their Responsibility
BOSH Director	Creates and updates service instances as instructed by ODB
BOSH Agent	BOSH includes an Agent on every VM that it deploys. The Agent listens for instructions from the Director and carries out those instructions. The Agent receives job specifications from the Director and uses them to assign a role, or Job, to the VM.
BOSH UAA	As an OAuth2 provider, BOSH UAA issues tokens for clients to use when they act on behalf of BOSH users.
ERT	Contains the apps that are consuming services
ODB	Instructs BOSH to create and update services, and connects to services to create bindings
Deployed service instance	Runs the given data service (for example, the deployed Redis for PCF service instance runs the Redis for PCF data service)

Regardless of the specific network layout, the operator must ensure network rules are set up so that connections are open as described in the table below.


This component...	Must communicate with...	Default TCP Port	Communication direction(s)	Notes
ODB	<ul style="list-style-type: none"> <li>BOSH Director</li> <li>BOSH UAA</li> </ul>	<ul style="list-style-type: none"> <li>25555</li> <li>8443</li> </ul>	One-way	The default ports are not configurable.
ODB	Deployed service instances	Specific to the service (such as RabbitMQ for PCF). May be one or more ports.	One-way	This connection is for administrative tasks. Avoid opening general use, app-specific ports for this connection.
ODB	ERT	8443	One-way	The default port is not configurable.
Errand VMs	<ul style="list-style-type: none"> <li>ERT</li> <li>ODB</li> <li>Deployed Service Instances</li> </ul>	<ul style="list-style-type: none"> <li>8443</li> <li>8080</li> <li>Specific to the service. May be one or more ports.</li> </ul>	One-way	The default port is not configurable.



BOSH Agent	BOSH Director	4222	Two-way	The BOSH Agent runs on every VM in the system, including the BOSH Director VM. The BOSH Agent initiates the connection with the BOSH Director. The default port is not configurable.
Deployed apps on ERT	Deployed service instances	Specific to the service. May be one or more ports.	One-way	This connection is for general use, app-specific tasks. Avoid opening administrative ports for this connection.
ERT	ODB	8080	One-way	This port may be different for individual services. This port may also be configurable by the operator if allowed by the tile developer.

## Installing and Configuring RabbitMQ for PCF as an On-Demand Service

This topic provides instructions to Pivotal Cloud Foundry (PCF) operators about how to install, configure, and deploy the RabbitMQ for PCF tile to provide on-demand service.

 **Note:** For instructions on how to install, configure, and deploy the RabbitMQ for PCF tile as a pre-provisioned service, see [the documentation for pre-provisioned RabbitMQ for PCF](#).

## Prerequisites for Deploying the On-Demand Service

Before deploying RabbitMQ for PCF as an on-demand service, you must ensure that the required network rules are in place to allow various components to communicate.

See [Required Networking Rules for On-Demand Services](#) for details on the network connections that must be open to enable the on-demand service.

For information on the on-demand service architecture, see [this page](#).

## Download and Install RabbitMQ for PCF

1. Download the product file from [Pivotal Network](#).
2. Navigate to the Ops Manager Installation Dashboard and click **Import a Product** to upload the product file.
3. Under the **Import a Product** button, click + next to the version number of RabbitMQ for PCF. This adds the tile to your staging area.
4. Click the newly added **RabbitMQ for PCF** tile to configure the service.

## Configure RabbitMQ for PCF

The configuration screen appears when you click the RabbitMQ for PCF tile in Ops Manager:

[Installation Dashboard](#)

## RabbitMQ

Settings

Status

Credentials

Logs

Assign AZs and Networks

RabbitMQ

Networking

RabbitMQ Policy

Syslog

Dedicated Instance: Single Node Plan

Dedicated Instance: Global Settings

Errands

Resource Config

Stemcell

### Errands

Errands are scripts that run at designated points during an installation.

#### Post-Deploy Errands

Broker Registrar

Default (On)

Smoke Tests

Default (On)

Register On-Demand Service Broker

Default (On)

Upgrade All Service Instances

Default (On)

#### Pre-Delete Errands

Broker Deregistrar

Default (On)

Delete All Service Instances

Default (On)

Deregister On-Demand Service Broker


Default (On)

Save

## Configure AZs and Networks

Follow the steps below to configure the AZs and networks.

1. Click **Assign AZs and Networks**.

 **Important:** You cannot change the regions or networks after you have clicked **Apply Changes** in the [final step](#) below.

2. Configure the fields on the **Assign AZs and Networks** as follows:

Field	Instructions
Place singleton	Select the region that you want for singleton VMs. PCF creates the RabbitMQ broker in this AZ.

jobs in Balance other jobs in	Select additional region. This selection does not affect the on-demand RabbitMQ for PCF service.
Network	<p>Select a network for the RabbitMQ On-Demand Broker.</p> <p>This should be a separate network from the one you select for <b>Service Network</b>. This network is represented by the Default Network in this <a href="#">picture</a>. Typically, you select the network used for the Pivotal Elastic Runtime components.</p>
Service Network	<p>Select a separate network that the on-demand service instances run on.</p> <p>A typical practice is to put all on-demand services on a single network, separate from the network that ERT and the On-Demand Broker run on.</p> <p>This network is represented by the Services Network in this <a href="#">picture</a>.</p> <p>This field is also required for the pre-provisioned service, though in that case, it doesn't matter which network you select.</p>

**⚠ WARNING:** Changing the Network or Service Network, or changing their IP configurations, results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).

3. Click **Save**.

## Configure Global Settings

Follow the steps below to configure global settings.

1. Click **Dedicated Instance: Global Settings** and configure the following:

- **Service instance quota** min: 0, max: 50 set the total number of dedicated service instances which can be deployed.
- **VM options:**

**Allow outbound internet access (IaaS-dependent).** This checkbox must be ticked to allow external log forwarding, sending backup artifacts to external destinations, and communicating with an external BOSH blob store.

2. Click **Save**.

## Configure the Service Plan

In order to enable the on-demand service plan, you must configure the **Dedicated Instance: Single Node Plan**. The default name of this plan is `Solo`.

This plan deploys a single RabbitMQ node on a single virtual machine.

**💡 Note:** If the on demand plans are not enabled, the RabbitMQ On Demand Broker VM is deployed alongside the RabbitMQ installation, but the plans will not be available in the Marketplace.

1. Click **Dedicated Instance: Single Node Plan**.

Settings

Status

Credentials

Logs

Assign AZs and Networks

RabbitMQ

Networking

RabbitMQ Policy

Syslog

Dedicated Instance: Single Node Plan

Dedicated Instance: Global Settings

Errands

Resource Config

Stemcell

## Service Plan Configuration

☐ Enable this plan

Service instance quota \*

Plan name \*

Plan description \*

Plan features \*

RabbitMQ VM Type\*

Persistent disk type\*
 

The type of persistent disk that the RabbitMQ node will be created on

AZ placement \*
 

☐ europe-west1-b
 ☐ europe-west1-c
 ☐ europe-west1-d

Save

2. Configure the fields on the **Dedicated Instance: Single Node Plan** as follows:

Field	Instructions
Enable this plan	Select the checkbox.
Service instance quota	Enter the maximum number of dedicated service instances that can be available at one time.
Plan name	Accept the default or enter a name. This is the name that appears in the Marketplace.
Plan description	Accept the default or enter a description.
Plan features	Accept the default or enter information about the plan to help your app developers.
RabbitMQ VM Type	Select a large VM type. The plan will create a service instance of this size.
Persistent disk type	This is where RabbitMQ will page messages to disk. See <a href="#">Disk Size Concerns</a> for more details.
AZ placement	This field is available after you complete the <a href="#">Assign AZs and Networks</a> page. Select one AZ for the single node. The plan creates all the on-demand service instance VMs in this AZ.

3. Click **Save**.

## Resource Types and Disk Size Concerns

It is possible to configure the VM type for RabbitMQ and the size of the persistent disk that is going to be attached to the RabbitMQ instances.

RabbitMQ VM Type<sup>\*</sup>

medium (cpu: 2, ram: 4 GB, disk: 8 GB)

Persistent disk type<sup>\*</sup>

10 GB

RabbitMQ raises alarms when disk space drops below the configured limit. Incorrect disk sizes may cause the deployed instance not to start. RabbitMQ declines to start if there is not enough space available according to the threshold.

Dedicated instances are configured with a threshold set to the 40% of the memory (RAM) of the VM. Operators can take the following table as an example when selecting the size of the persistent disk.

The following table shows an example of possible RAM values, absolute minimal value below which RabbitMQ declines to start, and the disk size suggested for an average use case.

RAM	Free disk alarm threshold (0.4xRAM)	Suggested disk size (2xRAM)
10 GB	4 GB	20 GB
16 GB	6.4 GB	32 GB
32 GB	12.8 GB	64 GB

For more information, see the following documentation links:

- [Alarms overview](#)
- [Disk Alarms](#)

## Verify the Stemcell

1. Click **Stemcell**.
2. Verify and, if necessary, import a new stemcell version.  
For more information, see the information about importing the stemcell for your IaaS: [AWS](#), [Azure](#), [GCP](#), or [vSphere](#).

## Apply Changes from your Configuration

Your installation is not complete until you apply your configuration changes. Follow the steps below:

1. Return to the Ops Manager Installation Dashboard.
2. Click **Apply Changes**.

## Configure Logging to Monitor RabbitMQ for PCF

Pivotal recommends that you configure logging to monitor the health of RabbitMQ for PCF. Follow [this procedure](#) to configure logging.

## Errands

When deploying or updating RabbitMQ for PCF, Ops Manager can optionally run a series of [post-deploy errands](#). An example is the 

Smoke Tests

 errand,

which checks the health of the RabbitMQ cluster after a deploy or upgrade.

You can decide whether to run errands by toggling them on or off before an update. This is a one-time setting on the installation dashboard:

**IMPORTANT:** As of RabbitMQ for PCF v1.9.0, all post-deploy errands are on by default. Pivotal recommends keeping these defaults, because the smoke tests can encounter unexpected issues, and dedicated instances of RabbitMQ for PCF may fall behind if the **Upgrade All Service Instances** errand is not on by default.

However, if necessary, you can change these defaults by clicking **Errands** in the RabbitMQ for PCF **Settings** tab.

For more information on errand run rules, see the [Ops Manager documentation](#).

## Post-Deploy Errands

Errand	Description
Broker Registrar	Makes the pre-provisioned RabbitMQ service plans available in the Marketplace
Smoke Tests	Checks that a pre-provisioned RabbitMQ service instance can be bound to a Cloud Foundry app, and that the app can publish and subscribe to a RabbitMQ cluster ( <a href="#">details</a> below)
Register On-Demand Service Broker	Makes the on-demand RabbitMQ service plans available in the Marketplace. If you change the Service Plan Configuration, you must run this errand in order for the changes to be reflected in the Marketplace.
Upgrade All Service Instances	On-demand instances are updated and redeployed if there are changes to the Dedicated Instance settings or the tile is upgraded. If this errand is set to “Off” or “When Changed”, updates to Dedicated Instance settings will <b>not</b> be applied to existing service instances. Pivotal strongly recommends that this errand is configured to <b>always run</b> .

## Pre-Delete Errands

Errand	Description
Broker Deregistrar	Removes the pre-provisioned RabbitMQ service from the Marketplace and deletes all associated service instances and bindings
Delete All Service Instances	Unbinds and deletes existing dedicated service instances. The duration of this errand depends on the number of deployed dedicated instances.
Deregister On-Demand Service Broker	Removes the on-demand RabbitMQ service from the Marketplace

## Dedicated Instance Smoke Test Process

The smoke tests perform the following for each enabled service plan:

1. Create and target the org `smoke-test-org` and the space `smoke-test-space` to run the tests.
2. Deploy an instance of the [CF RabbitMQ Example App](#) in this cf space.
3. Create the RabbitMQ cluster and bind it to the CF RabbitMQ Example App.
4. Check that the CF RabbitMQ Example App can write to, and read from, the RabbitMQ service instance.
5. Clean up the deployed Example App and all its service bindings.
6. Delete the cf org and space created for the tests.

## Create an Admin User for a Service Instance

If you want to give app developers admin privileges to the RabbitMQ Management UI, you can create an admin user for a service instance and share the user credentials with app developers.

Both operators and app developers can use this procedure.

To create an admin user on a RabbitMQ instance do the following:

1. Run this command to create a service key:

```
cf create-service-key SERVICE_INSTANCE SERVICE_KEY -c '{"tags": "administrator"}'
```

where:

`SERVICE_INSTANCE` is the name you supplied when you ran `cf create-service`

`SERVICE_KEY` is a name you choose to identify the service key.

For example:

```
$ cf create-service-key my-instance my-admin-key -c '{"tags": "administrator"}'

Creating service key my-admin-key for service instance my-instance as user@example.com...
OK
```

2. Run this command to get the admin user credentials:

```
cf service-key SERVICE_INSTANCE SERVICE_KEY
```

where the variables are the same as above.

This returns a Dashboard URL containing the admin credentials, which can be used to access the management UI. For example:

```
$ cf service-key my-instance my-admin-key

Getting key my-admin-key for service instance my-instance as user@example.com...
{
  "dashboard_url": "https://my-instance.bosh-lite.com/#/login/admin-username/admin-password",
  "username": "admin-username",
  "password": "admin-password",
  ...
}
```



## Monitoring and KPIs for On-Demand RabbitMQ for PCF

This topic explains how to monitor the health of the on-demand version of the RabbitMQ for Pivotal Cloud Foundry (PCF) service using the logs, metrics, and Key Performance Indicators (KPIs) generated by RabbitMQ for PCF component VMs.

On-demand RabbitMQ for PCF components generate many of the [same metrics](#) as the pre-provisioned RabbitMQ service components.

See [Logging and Metrics](#) for general information about logging and metrics in PCF.

### Direct the Logs

As of RabbitMQ for PCF v1.9.0, syslog forwarding is preconfigured and set to on by default. Pivotal recommends that you keep the default setting because it is good operational practice. However, you can opt out by selecting **No** for **Do you want to configure syslog?** in the Ops Manager **Settings** tab.

To enable monitoring for RabbitMQ for PCF, operators designate an external syslog endpoint for RabbitMQ component log messages. The endpoint serves as the input to a monitoring platform such as Datadog, Papertrail, or SumoLogic.

To specify the destination for RabbitMQ for PCF log messages, do the following:

1. From the Ops Manager Installation Dashboard, click the RabbitMQ tile.
2. In the RabbitMQ tile, click the **Settings** tab.

The screenshot shows the 'RabbitMQ' settings page in the 'Installation Dashboard'. The 'Settings' tab is selected. On the left, a sidebar lists various configuration items with checkmarks: Assign AZs and Networks, RabbitMQ, Networking, RabbitMQ Policy, Syslog (highlighted), Dedicated Instance: Single Node Plan, Dedicated Instance: Global Settings, Errands, Resource Config, and Stemcell. The main content area is titled 'Forwards syslog messages via TCP to a syslog server'. It contains a question 'Do you want to configure syslog forwarding?' with radio buttons for 'No' and 'Yes' (selected). Below this are two text input fields: 'Syslog address' and 'Syslog port', both marked with an asterisk. A blue 'Save' button is at the bottom.

3. Click **Syslog**.

Forwards syslog messages via TCP to a syslog server

Syslog address

logmanager.example.com

Syslog port

29279

Save

4. Enter your syslog address and port.

5. Click **Save**.

6. Return to the Ops Manager Installation Dashboard and click **Apply Changes** to redeploy with the changes.

## Logging Format

With on-demand RabbitMQ for PCF logging configured, two types of components generate logs: the server nodes and the service broker:

- The logs for RabbitMQ server nodes follow the format `[job=rabbitmq-server-partition-GUID index=0]`
- The logs for the RabbitMQ service broker follow the format `[job=rabbitmq-broker-partition-GUID index=0]`

The RabbitMQ VMs log at the `info` level and capture errors, warnings, and informational messages.

The logging format does not change in v1.9.0. For users familiar with documentation for previous versions of the tile, the tag we used to call the `app_name` is now called the `program_name`. The generic log format is as follows:

```
<PRI>TIMESTAMP IP_ADDRESS PROGRAM_NAME [job=NAME index=JOB_INDEX id=JOB_ID] MESSAGE
```

The raw logs look similar to the following:

```
<7>2017-06-28T16:06:10.733560+00:00 10.244.16.133 vcap.agent [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] 2017/06/28 16:06:10 CEF:0(CloudFoundry|BOSH|1|agent_api|ssh|1|user=dir
<86>2017-06-28T16:06:16.704572+00:00 10.244.16.133 useradd [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] new group: name=bosh_ly0d2rbjr, GID=1003
<86>2017-06-28T16:06:16.704663+00:00 10.244.16.133 useradd [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] new user: name=bosh_ly0d2rbjr, UID=1001, GID=1003, home=/var/vcap/bosh_
<86>2017-06-28T16:06:16.736932+00:00 10.244.16.133 usermod [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] add 'bosh_ly0d2rbjr' to group 'admin'
<86>2017-06-28T16:06:16.736964+00:00 10.244.16.133 usermod [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] add 'bosh_ly0d2rbjr' to group 'vcap'
```

Logs sent to external logging tools such as Papertrail may be presented in a different format.

The following table describes the logging tags used in this template:

Tag	Description
PRI	This is a value which in future will be used to describe the severity of the log message and which facility it came from.
TIMESTAMP	This is the timestamp of when the log is forwarded, for example, <code>2016-08-24T05:14:15.000003Z</code> . The timestamp value is typically slightly after when the log message was generated.
IP_ADDRESS	The internal IP address of server on which the log message originated
PROGRAM_NAME	Process name of the program the generated the message. Same as <code>app_name</code> before v1.9.0. For more information about program name, see <a href="#">RabbitMQ Program Names</a> below.
NAME	The BOSH instance group name (for example, <code>rabbitmq_server</code> )
JOB_INDEX	BOSH job index. Used to distinguish between multiple instances of the same job.
JOB_ID	BOSH VM GUID. This is distinct from the CID displayed in the Ops Manager Status tab, which corresponds to the VM ID assigned by the infrastructure provider.
MESSAGE	The log message that appears

## RabbitMQ Program Names

Program Name	Description
rabbitmq_server_cluster_check	Checks that the RabbitMQ cluster is healthy. Runs after every deploy.
rabbitmq_server_node_check	Checks that the RabbitMQ node is healthy. Runs after every deploy.
rabbitmq_route_registrar_stderr	Registers the route for the management API with the Gorouter in your Elastic Runtime deployment.
rabbitmq_route_registrar_stdout	Registers the route for the management API with the Gorouter in your Elastic Runtime deployment.
rabbitmq_server	The Erlang VM and RabbitMQ apps. <i>Logs may span multiple lines</i>
rabbitmq_server_drain	Shuts down the Erlang VM and RabbitMQ apps. Runs as part of the BOSH lifecycle.
rabbitmq_server_http_api_access	Access to the RabbitMQ management UI.
rabbitmq_server_init	Starts the Erlang VM and RabbitMQ.
rabbitmq_server_post_deploy_stderr	Runs the node check and cluster check. Runs after every deploy.
rabbitmq_server_post_deploy_stdout	Runs the node check and cluster check. Runs after every deploy.
rabbitmq_server_pre_start	Runs before the rabbitmq-server job is started.
rabbitmq_server_sasl	Supervisor, progress, and crash reporting for the Erlang VM and RabbitMQ apps.
rabbitmq_server_shutdown_stderr	Stops the RabbitMQ app and Erlang VM.
rabbitmq_server_shutdown_stdout	Stops the RabbitMQ app and Erlang VM.
rabbitmq_server_startup_stderr	Starts the RabbitMQ app and Erlang VM, then configures users and permissions.
rabbitmq_server_startup_stdout	Starts the RabbitMQ app and Erlang VM, then configures users and permissions.
rabbitmq_server_upgrade	Shuts down Erlang VM and RabbitMQ app if required during an upgrade.

## Metrics

Metrics are regularly-generated log messages that report measured component states. The metrics polling interval defaults to 30 seconds. This interval is a configuration option on the RabbitMQ tile ( **Settings** > **RabbitMQ**). The interval setting applies to all components deployed by the tile.

Metrics are long, single lines of text that follow the format:

```
origin:"p.rabbitmq" eventType:ValueMetric timestamp:1441188462382091652 deployment:"cf-rabbitmq" job:"cf-rabbitmq-node" index:"0" ip:"10.244.3.46" valueMetric: < name:"/p.rabbitmq/rabbitmq/system/m
```

## Partition Indicator

A new metric has been introduced to help to identify network partitions. Essentially it exposes how many nodes each node knows. When a node is in partition the only node that it recognizes is itself and that is a good indication that that node might be in a partition.

An example of that metrics is:

```
origin:"p.rabbitmq" eventType:ValueMetric timestamp:1441188462382091652 deployment:"cf-rabbitmq" job:"cf-rabbitmq-node" index:"0" ip:"10.244.3.46" valueMetric: < name:"/p.rabbitmq/rabbitmq/erlang/rea
```

Monitors can be created to emit alerts in case a cluster seems to be in a partition. A metrics is emitted for each node in the cluster. For example: in a three-node cluster a monitor can expect to have a total of 9 (nine) since each node is expected to emit 3 (2 reachable nodes and itself). Otherwise, an alert can be sent to the team.

## Recovering from a network partition

Please refer to the oficial RabbitMQ guide to understand how to recover from a network partition: <https://www.rabbitmq.com/partitions.html> [↗](#)

## Key Performance Indicators

Key Performance Indicators (KPIs) for RabbitMQ for PCF are metrics that operators find most useful for monitoring their RabbitMQ service to ensure smooth operation. KPIs are high-signal-value metrics that can indicate emerging issues. KPIs can be raw component metrics or *derived* metrics generated by applying formulas to raw metrics.

Pivotal provides the following KPIs as general alerting and response guidance for typical RabbitMQ for PCF installations. Pivotal recommends that operators continue to fine-tune the alert measures to their installation by observing historical trends. Pivotal also recommends that operators expand beyond this guidance and create new, installation-specific monitoring metrics, thresholds, and alerts based on learning from their own installations.

For a list of all RabbitMQ for PCF raw component metrics, see [Component Metrics Reference](#).

## Component Heartbeats

Key RabbitMQ for PCF components periodically emit heartbeat metrics: the RabbitMQ server nodes, HAProxy nodes, and the Service Broker. The heartbeats are Boolean metrics, where `1` means the system is available, and `0` or the absence of a heartbeat metric means the service is not responding and should be investigated.

### Service Broker Heartbeat

p.rabbitmq.service_broker.heartbeat	
Description	<p>RabbitMQ Service Broker <code>is alive</code> poll, which indicates if the component is available and able to respond to requests.</p> <p><b>Use:</b> If the Service Broker does not emit heartbeats, this indicates that it is offline. The Service Broker is required to create, update, and delete service instances, which are critical for dependent tiles such as Spring Cloud Services and Spring Cloud Data Flow.</p> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> boolean  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> N/A  <b>Red critical:</b> &lt; 1</p>
Recommended response	<p>Check the RabbitMQ Service Broker logs for errors. You can find this VM by targeting your RabbitMQ deployment with BOSH and running one of the following commands depending on your Ops Manager Version:</p> <ul style="list-style-type: none"> <li><b>For Ops Manager v1.10 or earlier:</b>  <pre>bosh vms service-instance_GUID</pre></li> <li><b>For Ops Manager v1.11 or later:</b>  <pre>bosh2 -d service-instance_GUID vms</pre></li> </ul>

### Server Heartbeat


p.rabbitmq.rabbitmq.heartbeat	
Description	<p>RabbitMQ Server <code>is alive</code> poll, which indicates if the component is available and able to respond to requests.</p> <p><b>Use:</b> If the server does not emit heartbeats, this indicates that it is offline. To be functional, service instances require RabbitMQ Server.</p> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> boolean  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> N/A  <b>Red critical:</b> &lt; 1</p>
Recommended response	<p>Check the RabbitMQ Server logs for errors. You can find the VM by targeting your RabbitMQ deployment with BOSH and running one of the following commands, which lists <code>rabbitmq</code>:</p> <ul style="list-style-type: none"> <li><b>For Ops Manager v1.10 or earlier:</b>  <pre>bosh vms service-instance_GUID</pre></li> <li><b>For Ops Manager v1.11 or later:</b></li> </ul>

```
bosh2 -d service-instance_GUID vms
```


## RabbitMQ Server KPIs

The following KPIs from the RabbitMQ server component:

### File Descriptors

p.rabbitmq.rabbitmq.system.file_descriptors	
<b>Description</b>	<p>File descriptors consumed.</p> <p><b>Use:</b> If the number of file descriptors consumed becomes too large, the VM may lose the ability to perform disk IO, which can cause data loss.</p> <div>  <b>Note:</b> This assumes non-persistent messages are handled by retries or some other logic by the producers.         </div> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> count  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
<b>Recommended measurement</b>	Average over last 10 minutes
<b>Recommended alert thresholds</b>	<p><b>Yellow warning:</b> &gt; 50000</p> <p><b>Red critical:</b> &gt; 55000</p>
<b>Recommended response</b>	<p>The default <code>ulimit</code> for RabbitMQ for PCF v1.6 and later is 60000. If this metric is met or exceeded for an extended period of time, consider one of the following actions:</p> <ul style="list-style-type: none"> <li>Scaling the rabbit nodes in the tile <b>Resource Config</b> pane.</li> <li>Increasing the <code>ulimit</code></li> </ul>

### Erlang Processes

p.rabbitmq.rabbitmq.system.erlang_processes	
<b>Description</b>	<p><a href="#">Erlang</a>  processes consumed by RabbitMQ, which runs on an Erlang VM.</p> <p><b>Use:</b> This is the key indicator of the processing capability of a node.</p> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> count  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
<b>Recommended measurement</b>	Average over last 10 minutes
<b>Recommended alert thresholds</b>	<p><b>Yellow warning:</b> &gt; 900000</p> <p><b>Red critical:</b> &gt; 950000</p>
<b>Recommended response</b>	<p>The default Erlang process limit in RabbitMQ for PCF v1.6 and later is 1,048,816. If this metric meets or exceeds the recommended thresholds for extended periods of time, consider scaling the RabbitMQ nodes in the tile <b>Resource Config</b> pane.</p>

## BOSH System Metrics

All BOSH-deployed components generate the following system metrics. Coming from RabbitMQ for PCF components, these system metrics serve as KPIs for the RabbitMQ for PCF service.

## RAM

system.mem.percent	
Description	<p>RAM being consumed by the <code>p-rabbitmq</code> VM.</p> <p><b>Use:</b> RabbitMQ is considered to be in a good state when it has little or no messages. In other words, “an empty rabbit is a happy rabbit.” Alerting on this metric can indicate that there are too few consumers or apps that read messages from the queue.</p> <p>Healthmonitor reports when RabbitMQ uses more than 40% of its RAM for the past ten minutes.</p> <p><b>Origin:</b> JMX Bridge or BOSH HM  <b>Type:</b> percent  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> &gt; 40  <b>Red critical:</b> &gt; 50</p>
Recommended response	Add more consumers to drain the queue as fast as possible.

## CPU

system.cpu.percent	
Description	<p>CPU being consumed by the <code>p-rabbitmq</code> VM.</p> <p><b>Use:</b> A node that experiences context switching or high CPU usage will become unresponsive. This also affects the ability of the node to report metrics.</p> <p>Healthmonitor reports when RabbitMQ uses more than 40% of its CPU for the past ten minutes.</p> <p><b>Origin:</b> JMX Bridge or BOSH HM  <b>Type:</b> percent  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> &gt; 60  <b>Red critical:</b> &gt; 75</p>
Recommended response	Remember that “an empty rabbit is a happy rabbit”. Add more consumers to drain the queue as fast as possible.

## Ephemeral Disk

system.disk.percent	
Description	<p>Ephemeral Disk being consumed by the <code>p-rabbitmq</code> VM.</p> <p><b>Use:</b> If system disk fills up, there are too few consumers.</p> <p>Healthmonitor reports when RabbitMQ uses more than 40% of its CPU for the past ten minutes.</p> <p><b>Origin:</b> JMX Bridge or BOSH HM  <b>Type:</b> percent  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> &gt; 60  <b>Red critical:</b> &gt; 75</p>
Recommended response	Remember that “an empty rabbit is a happy rabbit”. Add more consumers to drain the queue as fast as possible.

## Persistent Disk

persistent.disk.percent	
Description	<p>Persistent Disk being consumed by the <code>p-rabbitmq</code> VM.</p> <p>Use: If system disk fills up, there are too few consumers.</p> <p>Healthmonitor reports when RabbitMQ uses more than 40% of its CPU for the past ten minutes.</p> <p>Origin: JMX Bridge or BOSH HM</p> <p>Type: percent</p> <p>Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p>Yellow warning: &gt; 60</p> <p>Red critical: &gt; 75</p>
Recommended response	Remember that “an empty rabbit is a happy rabbit”. Add more consumers to drain the queue as fast as possible.

## Component Metric Reference

RabbitMQ for PCF component VMs emit the following raw metrics. The full name of the metric follows the format: `/p-rabbitmq/COMPONENT/METRIC-NAME`

### RabbitMQ Server Metrics

RabbitMQ for PCF message server components emit the following metrics.

Full Name	Unit	Description
<code>/p.rabbitmq.rabbitmq.heartbeat</code>	boolean	Indicates whether the RabbitMQ server is available and able to respond to requests
<code>/p.rabbitmq/rabbitmq/erlang/erlang_processes</code>	count	The number of Erlang processes
<code>/p.rabbitmq/rabbitmq/system/memory</code>	MB	The memory in MB used by the node
<code>/p.rabbitmq/rabbitmq/system/mem_alarm</code>	boolean	Indicates if the memory alarm has gone off
<code>/p.rabbitmq/rabbitmq/system/disk_free_alarm</code>	boolean	Indicates if the disk free alarm has gone off
<code>/p.rabbitmq/rabbitmq/connections/count</code>	count	The total number of connections to the node
<code>/p.rabbitmq/rabbitmq/consumers/count</code>	count	The total number of consumers registered in the node
<code>/p.rabbitmq/rabbitmq/messages/delivered</code>	count	The total number of messages with the status <code>deliver_get</code> on the node
<code>/p.rabbitmq/rabbitmq/messages/delivered_no_ack</code>	count	The number of messages with the status <code>deliver_no_ack</code> on the node
<code>/p.rabbitmq/rabbitmq/messages/delivered_rate</code>	rate	The rate at which messages are being delivered to consumers or clients on the node
<code>/p.rabbitmq/rabbitmq/messages/published</code>	count	The total number of messages with the status <code>publish</code> on the node
<code>/p.rabbitmq/rabbitmq/messages/published_rate</code>	rate	The rate at which messages are being published by the node
<code>/p.rabbitmq/rabbitmq/messages/redelivered</code>	count	The total number of messages with the status <code>redeliver</code> on the node
<code>/p.rabbitmq/rabbitmq/messages/redelivered_rate</code>	rate	The rate at which messages are getting the status <code>redeliver</code> on the node
<code>/p.rabbitmq/rabbitmq/messages/got_no_ack</code>	count	The number of messages with the status <code>get_no_ack</code> on the node
<code>/p.rabbitmq/rabbitmq/messages/get_no_ack_rate</code>	rate	The rate at which messages get the status <code>get_no_ack</code> on the node
<code>/p.rabbitmq/rabbitmq/messages/pending</code>	count	The number of messages with the status <code>messages_unacknowledged</code> on the node
<code>/p.rabbitmq/rabbitmq/messages/depth</code>	count	The number of messages with the status <code>messages_unacknowledged</code> or <code>messages_ready</code> on the node

/p.rabbitmq/rabbitmq/system/file	count	The number of open file descriptors on the node
/p.rabbitmq/rabbitmq/exchanges/count	count	The total number of exchanges on the node
/p.rabbitmq/rabbitmq/messages/available	count	The total number of messages with the status <code>messages_ready</code> on the node
/p.rabbitmq/rabbitmq/queues/count	count	The number of queues on the node
/p.rabbitmq/rabbitmq/channels/count	count	The number of channels on the node
/p.rabbitmq/rabbitmq/queues/VHOST-NAME/QUEUE-NAME/consumers	count	The number of consumers per virtual host per queue
/p.rabbitmq/rabbitmq/queues/VHOST-NAME/QUEUE-NAME/depth	count	The number of messages with the status <code>messages_unacknowledged</code> or <code>messages_ready</code> per virtual host per queue



## Troubleshooting On-Demand RabbitMQ for PCF

In this topic:

- [About the BOSH CLI](#)
- [Troubleshooting Errors](#)
  - [Failed Install](#)
  - [Cannot Create or Delete Service Instances](#)
  - [Broker Request Timeouts](#)
  - [Cannot Bind to or Unbind from Service Instances](#)
  - [Cannot Connect to a Service Instance](#)
  - [Upgrade All Instances Fails](#)
  - [Missing Logs and Metrics](#)
  - [Failed Deployment on Upgrade or after Apply Changes](#)
- [Troubleshooting Components](#)
  - [BOSH problems](#)
  - [Configuration](#)
  - [Authentication](#)
  - [Networking](#)
  - [Quotas](#)
- [Techniques for Troubleshooting](#)
  - [Parse a Cloud Foundry \(CF\) Error Message](#)
  - [Access Broker and Instance Logs and VMs](#)
  - [Run Service Broker Errands to Manage Brokers and Instances](#)
  - [Select the BOSH Deployment for a Service Instance](#)
  - [Get Admin Credentials for a Service Instance](#)
  - [Reinstall a Tile](#)
  - [View Resource Saturation and Scaling](#)
  - [Identify Service Instance Owner](#)
  - [Monitor Quota Saturation and Service Instance Count](#)
- [Knowledge Base \(Community\)](#)
- [File a Support Ticket](#)

This topic provides operators with basic instructions for troubleshooting on-demand RabbitMQ® for Pivotal Cloud Foundry (PCF).

## About the BOSH CLI

The BOSH CLI is available in two major versions, v1 and v2. Pivotal recommends that you use the BOSH CLI v2 when possible.

This topic provides examples of using each version of the BOSH CLI. While all versions of the BOSH CLI work with RabbitMQ for PCF v1.9.x, your Ops Manager Version may affect which version of the BOSH CLI you can use. Consult the table below to determine which version of the CLI is supported for your installation.

Ops Manager Version	BOSH CLI Version
1.10	CLI v1
1.11	CLI v1 or CLI v2 (Pivotal recommends CLI v2)
1.12 and later	CLI v2

## Troubleshooting Errors

Start here if you're responding to a specific error or error messages.

## Failed Install

1. Certificate issues: The on-demand broker (ODB) requires valid certificates. Ensure that your certificates are valid and [generate new ones](#) if necessary.
2. Deploy fails: Deploys can fail for a variety of reasons. View the logs using Ops Manager to determine why the deploy is failing.
3. [Networking problems](#):
  - Cloud Foundry cannot reach the RabbitMQ for PCF service broker
  - Cloud Foundry cannot reach the service instances
  - The service network cannot access the BOSH director
4. [Register broker errand](#) fails.
5. The smoke test errand fails.
6. Resource sizing issues: These occur when the resource sizes selected for a given plan are less than the RabbitMQ for PCF service requires to function. Check your resource configuration in Ops Manager and ensure that the configuration matches that recommended by the service.
7. Other service-specific issues.

## Cannot Create or Delete Service Instances

If developers report errors such as the following:

Instance provisioning failed: There was a problem completing your request. Please contact your operations team providing the following information: service: redis-acceptance, service-instance-g

1. If the BOSH error shows a problem with the deployment manifest:

- a. Download the manifest for the on-demand service instance by running:

```
bosh download manifest service-instance_SERVICE-INSTANCE-GUID MY-SERVICE.yml.
```

- b. Check the manifest for configuration errors.



**Note:** This error does not apply if you are using BOSH CLI v2. In that case, to troubleshoot possible problems with the manifest, open it in a text editor and inspect the manifest there.

2. To continue troubleshooting, [Log in to BOSH](#) and target the RabbitMQ for PCF service instance using the instructions on [parsing a Cloud Foundry error message](#).
3. Retrieve the BOSH task ID from the error message and run one of the following commands depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh task TASK-ID</code>
1.11	<code>bosh2 task TASK-ID</code>
1.12 and later	<code>bosh task TASK-ID</code>

4. If you need more information, [access the broker logs](#) and use the `broker-request-id` from the error message above to search the logs for more information. Check for:
  - [Authentication errors](#)
  - [Network errors](#)
  - [Quota errors](#)

## Broker Request Timeouts

If developers report errors such as:

```
Server error, status code: 504, error code: 10001, message: The request to the service broker timed out: https://BROKER-URL/v2/service_instances/e34046d3-2379-40d0-a318-d54fc7a5b13f/ser
```

1. Confirm that Cloud Foundry (CF) is [connected to the service broker](#).
2. Check the BOSH queue size:
  - a. Log into BOSH as an admin.
  - b. Run one of these commands depending on your Ops Manager version:
    - 1.10 and earlier: `bosh tasks`
    - 1.11: `bosh2 tasks`
    - 1.12 and later: `bosh tasks`
3. If there are a large number of queued tasks, the system may be under too much load. BOSH is configured with two workers and one status worker, which may not be sufficient resources for the level of load. Advise app developers to try again once the system is under less load.

## Cannot Bind to or Unbind from Service Instances

### Instance Does Not Exist

If developers report errors such as:

```
Server error, status code: 502, error code: 10001, message: Service broker error: instance does not exist'
```

Follow these steps:

1. Type `cf service MY-INSTANCE --guid`. This confirms that the the RabbitMQ for PCF service instance exists in BOSH and CF, and returns a GUID.
2. Using the GUID obtained above, run one of the following BOSH CLI commands depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh vms service-instance_GUID</code>
1.11	<code>bosh2 -d service-instance_GUID vms</code>
1.12 and later	<code>bosh -d service-instance_GUID vms</code>

If the BOSH deployment is not found, it has been deleted from BOSH. Contact Pivotal support for further assistance.

### Other Errors

If developers report errors such as:

```
Server error, status code: 502, error code: 10001, message: Service broker error: There was a problem completing your request. Please contact your operations team providing the following inform
```

To find out the exact issue with the binding process:

1. [Access the service broker logs](#).
2. Search the logs for the `broker-request-id` string listed in the error message above.
3. Contact Pivotal support for further assistance if you are unable to resolve the problem.
4. Check for:
  - [Authentication errors](#)
  - [Network errors](#)

## Cannot Connect to a Service Instance

If developers report that their app cannot use service instances that they have successfully created and bound:

Ask the user to send application logs that show the connection error. If the error is originating from the service, then follow RabbitMQ for PCF-specific instructions. If the issue appears to be network-related, then:

1. Check that [application security groups](#) are configured correctly. Access should be configured for the service network that the tile is deployed to.
2. Ensure that the network the PCF Elastic Runtime tile is deployed to has network access to the service network. You can find the network definition for this service network in the Ops Manager Director tile.
3. In Ops Manager go into the service tile and see the service network that is configured in the networks tab.
4. In Ops Manager go into the ERT tile and see the network it is assigned to. Make sure that these networks can access each other.

## Upgrade All Instances Fails

If the `upgrade-all-service-instances` errand fails, look at the errand output in the Ops Manager log.

If an instance fails to upgrade, debug and fix it before running the errand again to prevent any failure issues from spreading to other on-demand instances.

Once the Ops Manager log no longer lists the deployment as `failing`, [re-run the errand](#) to upgrade the rest of the instances.

## Missing Logs and Metrics

If no logs are being emitted by the on-demand broker, check that your syslog forwarding address is correct in Ops Manager.

1. Ensure you have configured syslog for the tile.
2. Ensure that you have network connectivity between the networks that the tile is using and the syslog destination. If the destination is external, you need to use the [public ip](#) VM extension feature available in your Ops Manager tile configuration settings.
3. Verify that the Firehose is emitting metrics:
  - a. Install the `cf nozzle` [plugin](#)
  - b. Run `cf nozzle -f ValueMetric | grep --line-buffered "on-demand-broker/MY-SERVICE"` to find logs from your service in the `cf nozzle` output.

If no metrics appear within five minutes, verify that the broker network has access to the Loggregator system on all required ports.

[Contact Pivotal support](#) if you are unable to resolve the issue.

## Failed Deployment on Upgrade or after Apply Changes


If the deployment fails after editing the **\*\*Assign AZs and Networks\*\*** pane of the RabbitMQ for PCF tile, it might be due to a change to the IP addresses assigned to the ``RabbitMQ Server`` job. RabbitMQ for PCF requires that these IP addresses do not change once assigned. If you change them, the deployment fails. This includes changes made to your current installation or during an upgrade. To diagnose and solve this issue, see [\[Changing Network or IP Addresses Results in a Failed Deployment\]\(./changing-ips.html\)](#).

## Troubleshooting Components

Guidance on checking for and fixing issues in on-demand service components.

### BOSH problems

#### Missing BOSH Director UUID

 **Note:** This error does not occur if you are using BOSH CLI v2

If using the BOSH CLI v1, re-add the `director_uuid` to the manifest:

1. Run `bosh status --uuid` and record the `director_uuid` value from the output.
2. Edit the manifest and add the `director_uuid: DIRECTOR-UUID` from the last step at the top of the manifest.

For more, see [Deployment Identification](#) in the BOSH docs.

#### Large BOSH Queue

On-demand service brokers add tasks to the BOSH request queue, which can back up and cause delay under heavy loads. An app developer who requests a new RabbitMQ for PCF instance sees `create in progress` in the Cloud Foundry Command Line Interface (cf CLI) until BOSH processes the queued request.

Ops Manager currently deploys two BOSH workers to process its queue. Future versions of Ops Manager will let users configure the number of BOSH workers.

## Configuration

#### Service instances in failing state


You may have configured a VM / Disk type in tile plan page in Ops Manager that is insufficiently large for the RabbitMQ for PCF service instance to start. See tile-specific guidance on resource requirements.

## Authentication

#### UAA Changes

If you have rotated any UAA user credentials then you may see authentication issues in the service broker logs.

To resolve this, redeploy the RabbitMQ for PCF tile in Ops Manager. This provides the broker with the latest configuration.

 **Note:** You must ensure that any changes to UAA credentials are reflected in the Ops Manager `credentials` tab of the Elastic Runtime tile.

## Networking

Common issues include:

1. Network latency when connecting to the RabbitMQ for PCF service instance to create or delete a binding.
  - Solution: Try again or improve network performance
2. Network firewall rules are blocking connections from the RabbitMQ for PCF service broker to the service instance.
  - Solution: Open the RabbitMQ for PCF tile in Ops Manager and check the two networks configured in the **Networks** pane. Ensure that these networks allow access to each other.
3. Network firewall rules are blocking connections from the service network to the BOSH director network.
  - Solution: Ensure that service instances can access the Director so that the BOSH agents can report in.
4. Apps cannot access the service network.
  - Solution: Configure Cloud Foundry application security groups to allow runtime access to the service network.
5. Problems accessing BOSH's UAA or the BOSH director.
  - Solution: Follow network troubleshooting and check that the BOSH director is online.

### Validate Service Broker Connectivity to Service Instances

1. To validate you can `bosh2 ssh` onto the RabbitMQ for PCF service broker:
  - **With BOSH CLI v2:** Target the deployment, and reach the service instance.
  - **With BOSH CLI v1:** Download the broker manifest and target the deployment, then try to reach the service instance.
2. If no BOSH `task-id` appears in the error message, look in the broker log using the `broker-request-id` from the task.

### Validate App Access to Service Instance

Use `cf ssh` to access to the app container, then try connecting to the RabbitMQ for PCF service instance using the binding included in the `VCAP_SERVICES` environment variable.

## Quotas

### Plan Quota issues

If developers report errors such as:

```
Message: Service broker error: The quota for this service plan has been exceeded.  
Please contact your Operator for help.
```

1. Check your current plan quota.
2. Increase the plan quota.
3. Log into Ops Manager.
4. Reconfigure the quota on the plan page.

5. Deploy the tile.
6. Find who is using the plan quota and take the appropriate action.

## Global Quota Issues

If developers report errors such as:

Message: Service broker error: The quota for this service has been exceeded.  
Please contact your Operator for help.

1. Check your current global quota.
2. Increase the global quota.
3. Log into Ops Manager.
4. Reconfigure the quota on the on-demand settings page.
5. Deploy the tile.
6. Find out who is using the quota and take the appropriate action.

## Failing jobs and unhealthy instances

To determine whether there is an issue with the RabbitMQ for PCF service deployment, inspect the VMs. To do so, run one of the following commands:

Ops Manager Version	BOSH Command
1.10 or earlier	<code>bosh vms --vitals service-instance_GUID</code>
1.11	<code>bosh2 -d service-instance_GUID vms --vitals</code>
1.12 and later	<code>bosh -d service-instance_GUID vms --vitals</code>

For additional information, run one of the following commands:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh instances --ps --vitals</code>
1.11	<code>bosh2 instances --ps --vitals</code>
1.12 and later	<code>bosh instances --ps --vitals</code>

If the VM is failing, follow the service-specific information. Any unadvised corrective actions (such as running BOSH `restart` on a VM) can cause issues in the service instance.

## Techniques for Troubleshooting

Instructions on interacting with the on-demand service broker and on-demand service instance BOSH deployments, and on performing general maintenance and housekeeping tasks.

## Parse a Cloud Foundry (CF) Error Message

Failed operations (create, update, bind, unbind, delete) result in an error message. You can retrieve the error message later by running the cf CLI command `cf service INSTANCE-NAME`.

```
$ cf service myservice
```

```
Service instance: myservice
Service: super-db
Bound apps:
Tags:
Plan: dedicated-vm
Description: Dedicated Instance
Documentation url:
Dashboard:
```

```
Last Operation
```

```
Status: create failed
```

```
Message: Instance provisioning failed: There was a problem completing your request.
```

```
Please contact your operations team providing the following information:
```

```
service: redis-acceptance,
service-instance-guid: ae9e232c-0bd5-4684-af27-1b08b0c70089,
broker-request-id: 63da3a35-24aa-4183-aec6-db8294506bac,
task-id: 442,
operation: create
```

```
Started: 2017-03-13T10:16:55Z
```

```
Updated: 2017-03-13T10:17:58Z
```

Use the information in the `Message` field to debug further. Provide this information to Pivotal Support when filing a ticket.

The `task-id` field maps to the BOSH task id. For further information on a failed BOSH task, use the `bosh task TASK-ID` command in v1 of the BOSH CLI. For v2, use `bosh2 task TASK-ID`.

The `broker-request-guid` maps to the portion of the On-Demand Broker log containing the failed step. Access the broker log through your syslog aggregator, or access BOSH logs for the broker by typing `bosh logs broker 0`. If you have more than one broker instance, repeat this process for each instance.

## Access Broker and Instance Logs and VMs

Before following the procedures below, log into the [cf CLI](#) and the [BOSH CLI](#).

### Access Broker Logs and VM(s)

You can [access logs using Ops Manager](#) by clicking on the **Logs** tab in the tile and downloading the broker logs.

To access logs using the BOSH CLI, do the following:

1. Identify the on-demand broker (ODB) deployment by running one of the following commands, depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh deployments</code>
1.11	<code>bosh2 deployments</code>
	1.12 and later

2. For BOSH CLI v1 only:

- a. Run `bosh download manifest ODB-DEPLOYMENT-NAME odb.yml` to download the ODB manifest.
- b. Select the ODB deployment using `bosh deployment odb.yml`.

3. View VMs in the deployment using one of the following commands:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh instances</code>
1.11	<code>bosh2 -d DEPLOYMENT-NAME instances</code>
	1.12 and later

4. SSH onto the VM by running one of the following commands:



Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh ssh service-instance_GUID</code>
1.11	<code>bosh2 -d service-instance_GUID ssh</code>
1.12 and later	<code>bosh -d service-instance_GUID ssh</code>

5. Download the broker logs by running one of the following commands:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh logs service-instance_GUID</code>
1.11	<code>bosh2 -d service-instance_GUID logs</code>
1.12 and later	<code>bosh -d service-instance_GUID logs</code>

The archive generated by BOSH or Ops Manager includes the following logs:


Log Name	Description
broker.log	Requests to the on-demand broker and the actions the broker performs while orchestrating the request (e.g. generating a manifest and calling BOSH). Start here when troubleshooting.
broker_ctl.log	Control script logs for starting and stopping the on-demand broker.
post-start.stderr.log	Errors that occur during post-start verification.
post-start.stdout.log	Post-start verification.
drain.stderr.log	Errors that occur while running the drain script.

## Access Service Instance Logs and VMs

1. To target an individual service instance deployment, retrieve the GUID of your service instance with the cf CLI command `cf service MY-SERVICE --guid`.

2. For BOSH CLI v1 only:

a. Run `bosh status --uuid` to retrieve the BOSH Director GUID.

 **Note:** “GUID” and “UUID” mean the same thing.

b. To download your BOSH manifest for the service, run `bosh download manifest service-instance_BOSH-DIRECTOR-GUID MANIFEST.yml` using the GUID you just obtained and a filename you want to save the manifest as.

c. Edit the following line in the service instance manifest that you just saved, to include the current BOSH Director GUID:

```
director_uuid: BOSH-DIRECTOR-GUID
```

d. Run `bosh deployment MANIFEST.yml` to select the deployment using the Director UUID.

3. View VMs in the deployment using one of the following commands:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh instances</code>
1.11	<code>bosh2 -d DEPLOYMENT-NAME instances</code>
1.12 and later	<code>bosh -d DEPLOYMENT-NAME instances</code>

4. SSH onto a VM by running one of the following commands:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh ssh service-instance_GUID</code>
1.11	<code>bosh2 -d service-instance_GUID ssh</code>
1.12 and later	<code>bosh -d service-instance_GUID ssh</code>

5. Download the instance logs by running one of the following commands:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh logs service-instance_GUID</code>
1.11	<code>bosh2 -d service-instance_GUID logs</code>
1.12 and later	<code>bosh -d service-instance_GUID logs</code>

## Run Service Broker Errands to Manage Brokers and Instances

From the BOSH CLI, you can run service broker errands that manage the service brokers and perform mass operations on the service instances that the brokers created. These service broker errands include:

- `register-broker` registers a broker with the Cloud Controller and lists it in the Marketplace
- `deregister-broker` deregisters a broker with the Cloud Controller and removes it from the Marketplace
- `upgrade-all-service-instances` upgrades existing instances of a service to its latest installed version
- `delete-all-service-instances` deletes all instances of service
- `orphan-deployments` detects “orphan” instances that are running on BOSH but not registered with the Cloud Controller

To run errands:

1. **For BOSH CLI v1 only:** Select the broker deployment by running this command:

```
bosh deployment BOSH_MANIFEST.yml
```

2. Run one of the following commands depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh run errand ERRAND_NAME</code>
1.11	<code>bosh2 -d DEPLOYMENT_NAME run-errand ERRAND_NAME</code>
1.12 and later	<code>bosh -d DEPLOYMENT_NAME run-errand ERRAND_NAME</code>

Examples:

```
bosh run errand deregister-broker
```

```
bosh2 -d DEPLOYMENT-NAME run-errand deregister-broker
```

## Register Broker

The `register-broker` errand registers the broker with Cloud Foundry and enables access to plans in the service catalog. Run this errand whenever the broker is re-deployed with new catalog metadata to update the Cloud Foundry catalog.

Plans with disabled service access are not visible to non-admin Cloud Foundry users (including Org Managers and Space Managers). Admin Cloud Foundry users can see all plans including those with disabled service access.

The errand does the following:

- Registers the service broker with Cloud Controller.
- Enables service access for any plans that have the radio button set to `enabled` in the tile plan page.
- Disables service access for any plans that have the radio button set to `disabled` in the tile plan page.
- Does nothing for any for any plans that have the radio button set to `manual`.

To run the errand, do the following:

1. **For BOSH CLI v1 only:** Select the broker deployment by running this command:

```
bosh deployment BOSH_MANIFEST.yml
```

2. Run one of the following commands depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh run errand register-broker</code>
1.11	<code>bosh2 -d DEPLOYMENT-NAME run-errand register-broker</code>
1.12 and later	<code>bosh -d DEPLOYMENT-NAME run-errand register-broker</code>

## Deregister Broker

This errand deregisters a broker from Cloud Foundry.

The errand does the following:

- Deletes the service broker from Cloud Controller
- Fails if there are any service instances, with or without bindings

Use the [Delete All Service Instances errand](#) to delete any existing service instances.

To run the errand, do the following:

1. **For BOSH CLI v1 only:** Select the broker deployment by running the command:  
`bosh deployment BROKER_MANIFEST.yml`.
2. Run one of the following commands depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh run errand deregister-broker</code>
1.11	<code>bosh2 -d DEPLOYMENT-NAME run-errand deregister-broker</code>
1.12 and later	<code>bosh -d DEPLOYMENT-NAME run-errand deregister-broker</code>

## Upgrade All Service Instances

If you have made changes to the plan definition or uploaded a new tile into Ops Manager, you may want to upgrade all the RabbitMQ for PCF service instances to the latest software/plan definition.

The `upgrade-all-service-instances` errand does the following:

- Collects all of the service instances the on-demand broker has registered.
- For each instance the errand serially:
  - Issues an upgrade command to the on-demand broker.
  - Re-generates the service instance manifest based on its latest configuration from the tile.
  - Deploys the new manifest for the service instance.
  - Waits for this operation to complete, then proceeds to the next instance.
- Adds to a retry list any instances that have ongoing BOSH tasks at the time of upgrade.
- Retries any instances in the retry list until all are upgraded.

If any instance fails to upgrade, the errand fails immediately. This prevents systemic problems from spreading to the rest of your service instances. Run the errand by following either of the procedures below.

To run the errand, you can either select the errand through the Ops Manager UI and have it run when you click `Apply Changes`, or do the following:

1. **For BOSH CLI v1 only:** Select the broker deployment by running this command:  
`bosh deployment BOSH_MANIFEST.yml`
2. Run one of the following commands depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	

Ops Manager Version	BOSH Command
1.11	<code>bosh run errand upgrade-all-service-instances</code> <code>bosh2 -d DEPLOYMENT-NAME run-errand upgrade-all-service-instances</code>
1.12 and later	<code>bosh -d DEPLOYMENT-NAME run-errand upgrade-all-service-instances</code>

## Delete All Service Instances

This errand deletes all service instances of your broker's service offering in every org and space of Cloud Foundry. It uses the Cloud Controller API to do this, and therefore only deletes instances the Cloud Controller knows about. It will not delete orphan BOSH deployments.

Orphan BOSH deployments don't correspond to a known service instance. While rare, orphan deployments can occur. Use the `orphan-deployments` errand to identify them.

The errand does the following:

- Unbinds all applications from the service instances.
- Deletes all service instances sequentially.
- Checks if any instances have been created while the errand was running.
- If newly-created instances are detected, the errand fails.

**WARNING:** Use extreme caution when running this errand. You should only use it when you want to totally destroy all of the on-demand service instances in an environment.

To run the errand, do the following:

1. **For BOSH CLI v1 only:** Select the broker deployment by running the command:

```
bosh deployment BROKER_MANIFEST.yml
```

2. Run one of the following commands depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh run errand delete-all-service-instances</code>
1.11	<code>bosh2 -d service-instance_GUID delete-deployment</code>
1.12 and later	<code>bosh -d service-instance_GUID delete-deployment</code>

## Detect Orphaned Instances Service Instances

A service instance is defined as 'orphaned' when the BOSH deployment for the instance is still running, but the service is no longer registered in Cloud Foundry.

The `orphan-deployments` errand collates a list of service deployments that have no matching service instances in Cloud Foundry and return the list to the operator. It is then up to the operator to remove the orphaned BOSH deployments.

To run the errand, do the following:

1. **For BOSH CLI v1 only:** Select the broker deployment by running the command:

```
bosh deployment BROKER_MANIFEST.yml
```

2. Run the errand using one of the following commands depending on your Ops Manager version:

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh run errand orphan-deployments</code>
1.11	<code>bosh2 -d DEPLOYMENT-NAME run-errand orphan-deployments</code>
1.12 and later	<code>bosh -d DEPLOYMENT-NAME run-errand orphan-deployments</code>

If orphan deployments exist, the errand script will:

- Exit with exit code 10

- Output a list of deployment names under a `[stdout]` header
- Provide a detailed error message under a `[stderr]` header

For example:

```
[stdout]
[{"deployment_name":"service-instance_80e3c5a7-80be-49f0-8512-44840f3c4d1b"}]

[stderr]
Orphan BOSH deployments detected with no corresponding service instance in Cloud Foundry. Before deleting any deployment it is recommended to verify the service instance no longer exists in Cloud Foundry.

Errand 'orphan-deployments' completed with error (exit code 10)
```

These details will also be available through the BOSH `/tasks/` API endpoint for use in scripting:

```
$ curl 'https://bosh-user:bosh-password@bosh-url:25555/tasks/task-id/output?type=result' | jq .
{
  "exit_code": 10,
  "stdout": "[{"deployment_name":"service-instance_80e3c5a7-80be-49f0-8512-44840f3c4d1b"}]\n",
  "stderr": "Orphan BOSH deployments detected with no corresponding service instance in Cloud Foundry. Before deleting any deployment it is recommended to verify the service instance no longer exists in Cloud Foundry.",
  "logs": {
    "blobstore_id": "d830c4bf-8086-4bc2-8c1d-54d3a3c6d88d"
  }
}
```

If no orphan deployments exist, the errand script will:

- Exit with exit code 0
- Stdout will be an empty list of deployments
- Stderr will be `None`

```
[stdout]
[]

[stderr]
None

Errand 'orphan-deployments' completed successfully (exit code 0)
```

If the errand encounters an error during running it will:

- Exit with exit 1
- Stdout will be empty
- Any error messages will be under stderr

To clean up orphaned instances, run the following command on each instance:

**⚠ WARNING:** Running this command may leave IaaS resources in an unusable state.

Ops Manager Version	BOSH Command
1.10 and earlier	<code>bosh delete deployment service-instance_SERVICE-INSTANCE-GUID</code>
1.11	<code>bosh2 delete-deployment service-instance_SERVICE-INSTANCE-GUID</code>
1.12 and later	<code>bosh delete-deployment service-instance_SERVICE-INSTANCE-GUID</code>

## Select the BOSH Deployment for a Service Instance


This is an additional troubleshooting option for **BOSH CLI v1 only**. It does not apply to the BOSH CLI v2.

1. Retrieve the GUID of your service instance with the command `cf service YOUR-SERVICE-INSTANCE --guid`.
2. To download your BOSH manifest for the service, run `bosh download manifest service-instance_SERVICE-INSTANCE-GUID myservice.yml` using the GUID you just obtained and a file name you want to use when saving the manifest.
3. Run `bosh deployment MY-SERVICE.yml` to select the deployment.

## Get Admin Credentials for a Service Instance

To retrieve the admin and read-only admin credentials for a service instance, perform the following steps:

1. [Identify the service deployment by GUID.](#)
2. [Log into BOSH](#).
3. [Download the manifest for the service instance](#) and add the GUID if using the BOSH CLI v1.

 Skip this step if you are using the BOSH CLI v2. You cannot download the manifest with the BOSH CLI v2. Open it in a text editor instead.

4. Look in the manifest for the `admin` and `roadmin` credentials.

## Reinstall a Tile

To reinstall a tile in the same environment where it was previously uninstalled:

1. Ensure that the previous tile was correctly uninstalled as follows:
  - a. Log in as an admin with `cf login`.
  - b. Use `cf m` to confirm that the Marketplace does not list RabbitMQ for PCF.
  - c. Depending on which version of the BOSH CLI you are using, follow one of the steps below to log in to BOSH as an admin:
    - i. **For BOSH CLI v2:** Use `bosh2 log-in`.
    - ii. **For BOSH CLI v1:** Use `bosh login`.
  - d. Depending on which version of the BOSH CLI you are using, follow one of the steps below to display your BOSH deployments to confirm that the output does not show a RabbitMQ for PCF deployment:
    - i. **For BOSH CLI v2:** Use `bosh2 deployments`.
    - ii. **For BOSH CLI v1:** Use `bosh deployments`.
  - e. Run the `"delete-all-service-instances"` errand to delete every instance of the service.
  - f. Run the `"deregister-broker"` errand to delete the service broker.
  - g. Depending on which version of the BOSH CLI you are using, follow one of the steps below:
    - i. **For BOSH CLI v2:** Use `bosh2 delete-deployment BROKER-DEPLOYMENT-NAME` to delete the service broker BOSH deployment.
    - ii. **For BOSH CLI v1:** Use `bosh delete deployment BROKER-DEPLOYMENT-NAME` to delete the service broker BOSH deployment.
  - h. Reinstall the tile.

## View Resource Saturation and Scaling

### BOSH CLI v2: Viewing statistics

To view usage statistics for any service do the following:

1. For BOSH CLI v1 only: Select the broker deployment by running this command:

```
bosh deployment BOSH_MANIFEST.yml
```


2. Run the following commands depending on your Ops Manager version:

Ops Manager Version	BOSH Commands
v1.10 and earlier	Run the BOSH CLI v1 command <code>bosh vms --vitals</code> . To view process-level information, run <code>bosh instances --ps</code> .
v1.11	Run the BOSH CLI v2 command <code>bosh2 -d DEPLOYMENT-NAME vms --vitals</code> . To view process-level information, run <code>bosh2 -d DEPLOYMENT-NAME instances --ps</code> .
v1.12 and later	Run the BOSH CLI v2 command <code>bosh -d DEPLOYMENT-NAME vms --vitals</code> . To view process-level information, run <code>bosh2 -d DEPLOYMENT-NAME instances --ps</code> .

## Identify Service Instance Owner

If you want to identify which apps are using a specific service instance from the BOSH deployments name, you can run the following steps:


1. Take the deployment name and strip the `service-instance_` leaving you with the GUID.
2. Log in to CF as an admin.
3. Obtain a list of all service bindings by running the following: `cf curl /v2/service_instances/GUID/service_bindings`
4. The output from the above curl gives you a list of `resources`, with each item referencing a service binding, which contains the `APP-URL`. To find the name, org, and space for the app, run the following:
  - a. `cf curl APP-URL` and record the app name under `entity.name`
  - b. `cf curl SPACE-URL` to obtain the space, using the `entity.space_url` from the above curl. Record the space name under `entity.name`
  - c. `cf curl ORGANIZATION-URL` to obtain the org, using the `entity.organization_url` from the above curl. Record the organization name under `entity.name`

 **Note:** When running `cf curl` ensure that you query all pages, because the responses are limited to a certain number of bindings per page. The default is 50. To find the next page curl the value under `next_url`

## Monitor Quota Saturation and Service Instance Count

Quota saturation and total number of service instances are available through ODB metrics emitted to Loggregator. The metric names are shown below:


Metric Name	Description
<code>on-demand-broker/SERVICE-NAME-MARKETPLACE/quota_remaining</code>	global quota remaining for all instances across all plans
<code>on-demand-broker/SERVICE-NAME-MARKETPLACE/PLAN-NAME/quota_remaining</code>	quota remaining for a particular plan
<code>on-demand-broker/SERVICE-NAME-MARKETPLACE/total_instances</code>	total instances created across all plans
<code>on-demand-broker/SERVICE-NAME-MARKETPLACE/PLAN-NAME/total_instances</code>	total instances created for a given plan

 **Note:** Quota metrics are not emitted if no quota has been set.

## Knowledge Base (Community)

Find the answer to your question and browse product discussions and solutions by searching the [Pivotal Knowledge Base](#).

## File a Support Ticket

You can file a support ticket [here](#) . Be sure to provide the error message from `cf service YOUR-SERVICE-INSTANCE`.

To help expedite troubleshooting, also provide your service broker logs, your service instance logs and BOSH task output, if your `cf service YOUR-SERVICE-INSTANCE` output includes a `task-id`.



## RabbitMQ for PCF Operations FAQ's

This topic asks and answers some frequently asked questions (FAQs) about RabbitMQ for PCF.

### About the BOSH CLI

The BOSH CLI is available in two major versions, v1 and v2. Pivotal recommends that you use the BOSH CLI v2 when possible.

This topic provides examples of using each version of the BOSH CLI. While all versions of the BOSH CLI work with RabbitMQ for PCF v1.9.x, your Ops Manager version may affect which version of the BOSH CLI you can use. Consult the table below to determine which version of the CLI is supported for your installation.

Ops Manager Version	BOSH CLI Version
1.10	CLI v1
1.11	CLI v1 or CLI v2 (Pivotal recommends CLI v2)
1.12 and later	CLI v2

### What should I check before deploying a new version of the tile?

Ensure that all nodes in the cluster are healthy via the RabbitMQ Management UI, or health metrics exposed via the firehose.

Do not rely on BOSH `instances`, to execute this task correctly. That output reflects the state of the Erlang VM used by RabbitMQ, not the RabbitMQ application.

### What is the correct way to stop and start RabbitMQ in PCF?

Only BOSH commands should be used by the operator to interact with the RabbitMQ application. Examples for both versions of the CLI are given below.

- For Ops Manager v1.10 or earlier:

```
bosh stop rabbitmq-server bosh start rabbitmq-server
```

- For Ops Manager v1.11 or later:

```
bosh2 stop rabbitmq-server bosh2 start rabbitmq-server
```

There are BOSH job lifecycle hooks which are only fired when rabbitmq-server is stopped through BOSH. You can also stop individual instances by running one of the following commands:

- For Ops Manager v1.10 or earlier: `bosh stop JOB [index]`
- For Ops Manager v1.11 or later: `bosh2 stop JOB [index]`



**Note:** Do not use `monit stop rabbitmq-server`. This command does not call the drain scripts.

### What happens when I stop the RabbitMQ server?

You can stop the RabbitMQ server with BOSH `stop`.

BOSH starts the shutdown sequence from the bootstrap instance.

This command tells the RabbitMQ application to shut down and then shut down the Erlang VM in which it is running. If this succeeds, run the following checks to ensure that the RabbitMQ application and Erlang VM have stopped:

- If `/var/vcap/sys/run/rabbitmq-server/pid` exists, check that the PID inside this file does not point to a running Erlang VM process. This process checks the Erlang PID and not the RabbitMQ PID.
- Check that `rabbitmqctl` does not return an Erlang VM PID.

Once this completes on the bootstrap instance, BOSH will continue the same sequence on the next instance. All remaining RabbitMQ servers stop sequentially after that.

## What happens when the RabbitMQ server fails to stop?


If BOSH `stop` fails, you will likely get an error saying that the drain script failed with:

```
result: 1 of 1 drain scripts failed. Failed Jobs: rabbitmq-server.
```

## What do I do when the RabbitMQ server fails to stop?

The drain script logs to `/var/vcap/sys/log/rabbitmq-server/drain.log`. If you have a remote syslog configured, this will appear as the `rmq_server_drain` program.

1. BOSH `ssh` into the failing rabbitmq-server instance and start the rabbitmq-server job by running `monit start rabbitmq-server`.
2. When the rabbitmq-server job is running (confirm this via `monit status`), run `DEBUG=1 /var/vcap/jobs/rabbitmq-server/bin/drain`. This will tell you exactly why it's failing.

 **Note:** You will not be able to start the job with BOSH `start` as this command always runs the drain script first. It will fail since the drain script is failing.

## How can I manually back up the state of the RabbitMQ cluster?

You can back up the state of a RabbitMQ cluster for both the on-demand and pre-provisioned services using the RabbitMQ Management API. Backups include vhosts, exchanges, queues and users.

### Back up Manually

1. Log in to the RabbitMQ Management UI as the admin user you created.
2. Select **export definitions** from the main page.

### Back up and Restore with a Script

Use the API to run scripts with code similar to the following:

1. For the backup:

```
curl -u "$USERNAME:$PASSWORD" "http://$RABBIT_ADDRESS:15672/api/definitions"
-o "$BACKUP_FOLDER/rabbit-backup.json"
```

2. For the restore:

```
curl -u "$USERNAME:$PASSWORD" "http://$RABBIT_ADDRESS:15672/api/definitions"
-X POST -H "Content-Type: application/json" -d
"@$BACKUP_FOLDER/rabbit-backup.json"
```

## What pre-upgrade checks should I do?

Before doing any upgrade of RabbitMQ, Pivotal recommends checking the following:

1. In Operations Manager check that the status of all of the instances is healthy.

2. Log into the RabbitMQ Management UI and check that no alarms have been triggered and that all nodes are healthy, that is, they should display as green.
3. Check that the system is not close to hitting either the memory or disk alarm. Do this by looking at what has been consumed by each node in the RabbitMQ Management UI.

## Using On-Demand RabbitMQ for PCF

This topic provides instructions for developers using the on-demand RabbitMQ service for their Pivotal Cloud Foundry (PCF) apps. RabbitMQ enables messaging between cloud-based servers, apps and devices.

These procedures use the Cloud Foundry Command-Line Interface (cf CLI). You can also use [Apps Manager](#) to perform the same tasks using a graphical UI.

For general information, see [Managing Service Instances with the cf CLI](#).

## Prerequisites

To use on-demand RabbitMQ for PCF with your PCF apps, you need:

- A PCF installation with [RabbitMQ for PCF](#) installed and listed in the [Marketplace](#)
- A [Space Developer](#) or Admin account on the PCF installation
- A local machine with the following installed:
  - a browser
  - a shell
  - the [Cloud Foundry Command-Line Interface](#) (cf CLI)
  - the Linux [watch](#) command
- To [log into](#) the org and space containing your app

## Developer Guide

### Entries in the VCAP\_SERVICES Environment Variable

Applications running in Cloud Foundry gain access to the bound service instances via an environment variable credentials hash called `VCAP_SERVICES`. An example hash is show below:

```
{
  "p-rabbitmq": [{
    "credentials": {
      "dashboard_url": "http://pivotal-rabbitmq.your.pcf.example.com/#/login/b5d0ad14-4352-48e8-8982-d5b1d257029f/tavk86pnns1ddiqpsdtbchurn",
      "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
      "password": "tavk86pnns1ddiqpsdtbchurn",
      "protocols": {
        "amqp": {
          "password": "tavk86pnns1ddiqpsdtbchurn",
          "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
          "uris": [
            "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiqpsdtbchurn@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af",
            "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiqpsdtbchurn@10.0.0.51:5672/62e5ab21-7b38-44ac-b139-6aa97af"
          ]
        }
      }
    }
  ]
}
```

You can read more details about the environment variable `VCAP_SERVICES` [here](#).

## The Create-Bind Process

Because every app and service in PCF is scoped to a [space](#), an app can only use a service if an instance of the service exists in the same space.

To use RabbitMQ in a PCF app:

1. Use the [cf CLI](#) or [Apps Manager](#) to log into the org and space that contains the app.
2. Make sure an instance of the RabbitMQ for PCF service exists in the same space as the app.
  - If the space does not already have a RabbitMQ for PCF instance, [create](#) one.
  - If the space already has a Rabbit for PCF instance, you can [bind](#) your app to the existing instance or create a new instance to bind to your app.
3. [Bind](#) the app to the RabbitMQ for PCF service instance, to enable the app to use RabbitMQ.

## Confirm Service Availability

For an app to use a service, 1) the service must be available in the Marketplace for its space and 2) an instance of the service must exist in its space.

You can confirm both of these using the cf CLI as follows.

1. To find out if On-Demand RabbitMQ for PCF service is available in the Marketplace:
  - a. Enter `cf marketplace`
  - b. If the output lists `ondemand-rabbitmq` in the `service` column, on-demand RabbitMQ for PCF is available. If it is not available, ask your operator to install it.

```
$ cf marketplace
Getting services from marketplace in org my-org / space my-space as user@example.com...
OK
service      plans      description
[...]
ondemand-rabbitmq  Solo      RabbitMQ Service
[...]
```

2. To confirm that an On-Demand RabbitMQ for PCF instance is running in the space
  - a. Enter `cf services`
  - b. Any `ondemand-rabbitmq` listings in the `service` column are service instances of on-demand RabbitMQ in the space.

```
$ cf services
Getting services in org my-org / space my-space as user@example.com...
OK
name      service      plan  bound apps  last operation
my-instance  ondemand-rabbitmq  Solo      create succeeded
```

You can [bind](#) your app to an existing instance or [create](#) a new instance to bind to your app.

## Create a Service Instance

Unlike pre-provisioned services, on-demand services are created asynchronously, not immediately. The `watch` command shows you when your service instance is ready to bind and use.

To create an instance of the on-demand RabbitMQ for PCF service, run `cf create-service`:

1. Enter `cf create-service ondemand-rabbitmq Solo SERVICE_INSTANCE`  
Where `SERVICE_INSTANCE` is a name you choose to identify the service instance. This name will appear under `service` [sic] in output from `cf services`.
2. Enter `watch cf services` and wait for the `last operation` for your instance to show as `create succeeded`.

```
$ cf create-service ondemand-rabbitmq Solo my-instance
Creating service my-instance in org my-org / space my-space as user@example.com...
OK

$ watch cf services

Getting services in org my-org / space my-space as user@example.com...
OK
name      service      plan  bound apps  last operation
my-instance  ondemand-rabbitmq  Solo      create succeeded
```

If you get an error, see [Troubleshooting Instances](#).

## Bind a Service Instance to Your App

For an app to use a service, you must bind it to a service instance. Do this after you push or re-push the app using `cf push`.

To bind an app to a RabbitMQ instance run `$ cf bind-service`.

1. Enter `cf bind-service APP SERVICE_INSTANCE`

Where `APP` is the app you want to use the RabbitMQ service instance and `SERVICE_INSTANCE` is the name you supplied when you ran `cf create-service`.

```
$ cf bind-service my-app my-instance

Binding service mydb to my-app in org my-org / space test as user@example.com...
OK
TIP: Use 'cf push' to ensure your env variable changes take effect
```

## Use the RabbitMQ Service in Your App

To access the RabbitMQ service from your app:

1. Run `cf env APP_NAME` with the name of the app bound to the RabbitMQ for PCF instance.
2. In the output, note the connection strings listed in the `VCAP_SERVICES` > `credentials` object for the app.
3. In your app code, call the RabbitMQ service using the connection strings.

For how to code your app to use RabbitMQ messaging, see **About Using Pivotal RabbitMQ > Client Documentation** in the [RabbitMQ documentation](#).

## Restage an App with a New Service Instance

If a service instance has been updated based on a new version of the service, you need to run `cf restage` to restage your app to use the new instance.

1. Enter `cf restage-app APP`

Where `APP` is the app you want to use the updated service instance.

```
$ cf restage-app my-app
```

When a new version of a service obsoletes an older version, the platform operator may ask you to update your instances of the service and restage any apps bound to the service instances.

Pushing new version of an app automatically restages the app on any service instances it is bound to.

## Unbind a Service Instance to Your App

To stop an app from using a service it no longer needs, unbind it from the service instance using `cf unbind-service`.

1. Enter `cf unbind-service APP SERVICE_INSTANCE`

Where `APP` is the app you want to stop using the RabbitMQ service instance and `SERVICE_INSTANCE` is the name you supplied when you ran `cf create-service`.

```
$ cf unbind-service my-app my-instance

Unbinding app my-app from service my-instance in org my-org / space my-space as user@example.com...
OK
```

## Delete a Service Instance

To delete a service instance, run `cf delete-service`.

1. Enter `cf delete-service SERVICE_INSTANCE`

Where `SERVICE_INSTANCE` is the name of the service to delete.

```
$ cf delete-service my-instance

Are you sure you want to delete the service my-instance ? y
Deleting service my-service in org my-org / space my-space as user@example.com...
OK
```

2. Enter `watch cf service SERVICE_INSTANCE` and wait for a `Service instance not found` error indicating that the instance no longer exists.

You cannot delete a service instance that an app is bound to.

## RabbitMQ® Environment Variables

This topic provides a reference for the environment variables that Cloud Foundry stores for RabbitMQ for PCF service instances. These variables include the credentials that apps use to access the service instances.

### VCAP\_SERVICES

Applications running in Cloud Foundry gain access to the bound service instances via an environment variable credentials hash called `VCAP_SERVICES`. An example hash is show below:


```
{
  "p-rabbitmq": [{
    "label": "p-rabbitmq",
    "name": "my-rabbit-service-instance",
    "plan": "standard",
    "tags": ["rabbitmq", "messaging", "message-queue", "amqp", "pivotal"],
    "credentials": {
      "dashboard_url": "http://pivotal-rabbitmq.your.pcf.example.com/#/login/b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn",
      "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
      "vhost": "62e5ab21-7b38-44ac-b139-6aa97af01cd7",
      "password": "tavk86pnns1ddiypsdtbchurn",
      "ssl": false,
      "hostname": "10.0.0.41",
      "hostnames": [
        "10.0.0.41",
        "10.0.0.51"],
      "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.41:62e5ab21-7b38-44ac-b139-6aa97af01cd7",
      "uris": [
        "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.41:62e5ab21-7b38-44ac-b139-6aa97af01cd7",
        "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.51:62e5ab21-7b38-44ac-b139-6aa97af01cd7"],
      "http_api_uri": "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.41:15672/api",
      "http_api_uris": [
        "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.41:15672/api",
        "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.51:15672/api"],
      "protocols": {
        "amqp": {
          "password": "tavk86pnns1ddiypsdtbchurn",
          "port": 5672,
          "ssl": false,
          "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
          "vhost": "62e5ab21-7b38-44ac-b139-6aa97af01cd7",
          "host": "10.0.0.41",
          "hosts": [
            "10.0.0.41",
            "10.0.0.51"],
          "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
          "uris": [
            "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
            "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.51:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7"],
          "management": {
            "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
            "password": "tavk86pnns1ddiypsdtbchurn",
            "path": "/api",
            "port": 15672,
            "ssl": false,
            "host": "10.0.0.41",
            "hosts": [
              "10.0.0.41",
              "10.0.0.51"],
            "uri": "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.41:15672/api",
            "uris": [
              "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.41:15672/api",
              "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiypsdtbchurn@10.0.0.51:15672/api"]}]]}}}
```

You can search for your service by its `name`, given when creating the service instance, or dynamically via the `tags` or `label` properties. The `credentials` property can be used as follows:

- The top level properties `uri`, `uris`, `vhost`, `username`, `password`, `hostname` and `hostnames` provide access to the AMQP 0.9.1 protocol.
- A more flexible approach is provided by the `credentials.protocols` property, which has a key per enabled protocol. The possible keys are `amqp`, `management`, `mqtt`, and `stomp`. If SSL is enabled, then the keys will be `amqp+ssl`, `management+ssl`, `mqtt+ssl`, and `stomp+ssl` respectively.
- The values associated with each of these keys gives access credentials specific to each protocol. In all cases, URIs are provided, along with the individual components.



## Changing Enabled Plugins and Protocols

 **Note:** Removing or adding plugins/protocols may cause apps bound with RabbitMQ to break.

If you adjust the plugins and protocols enabled for RabbitMQ, you may need to force all app's `VCAP_SERVICES` environment variable to be regenerated. Adding and removing the following plugins require bound applications to be restaged:

- `rabbitmq_management`
- `rabbitmq_stomp`
- `rabbitmq_mqtt`
- `rabbitmq_amqp1_0`

In common with all services in [Pivotal Cloud Foundry](#) (PCF), the `VCAP_SERVICES` environment variable for an application is only modified when the application is bound to a service instance. Users will need to `cf unbind-service`, `cf bind-service` and `cf restage` their app in this scenario.

## Troubleshooting Instances

This topic provides basic instructions for app developers troubleshooting On-Demand RabbitMQ® for PCF.

### About the BOSH CLI

The BOSH CLI is available in two major versions, v1 and v2. Pivotal recommends that you use the BOSH CLI v2 when possible.

This topic provides examples of using each version of the BOSH CLI. While all versions of the BOSH CLI work with RabbitMQ for PCF v1.9.x, your Ops Manager Version may affect which version of the BOSH CLI you can use. Consult the table below to determine which version of the CLI is supported for your installation.

Ops Manager Version	BOSH CLI Version
1.10	CLI v1
1.11	CLI v1 or CLI v2 (Pivotal recommends CLI v2)
1.12 and later	CLI v2

### Errors

You may see an error when using the Cloud Foundry Command Line Interface (cf CLI) to perform basic operations on a RabbitMQ for PCF service instance:

- `cf create`
- `cf update`
- `cf bind`
- `cf unbind`
- `cf delete`

### Parse a Cloud Foundry (CF) Error Message

Failed operations (create, update, bind, unbind, delete) result in an error message. You can retrieve the error message later by running the cf CLI command `cf service INSTANCE-NAME`.

```
$ cf service myservice

Service instance: myservice
Service: super-db
Bound apps:
Tags:
Plan: dedicated-vm
Description: Dedicated Instance
Documentation url:
Dashboard:

Last Operation
Status: create failed
Message: Instance provisioning failed: There was a problem completing your request.
Please contact your operations team providing the following information:
service: redis-acceptance,
service-instance-guid: ae9e232c-0bd5-4684-af27-1b08b0e70089,
broker-request-id: 63da3a35-24aa-4183-aec6-db8294506bac,
task-id: 442,
operation: create
Started: 2017-03-13T10:16:55Z
Updated: 2017-03-13T10:17:58Z
```

Use the information in the `Message` field to debug further. Provide this information to Pivotal Support when filing a ticket.

The `task-id` field maps to the BOSH task id. For further information on a failed BOSH task, use the `bosh task TASK-ID` command in v1 of the BOSH CLI. For v2, use `bosh2 task TASK-ID`.

The `broker-request-guid` maps to the portion of the On-Demand Broker log containing the failed step. Access the broker log through your syslog aggregator,

or access BOSH logs for the broker by typing `bosh logs broker 0`. If you have more than one broker instance, repeat this process for each instance.

## Retrieve Service Instance Information

1. Log into the space containing the instance or failed instance.

```
$ cf login
```

2. If you do not know the name of the service instance, run `cf services` to see a listing of all service instances in the space. The service instances are listed in the `name` column.

```
$ cf services
Getting services in org my-org / space my-space as user@example.com...
OK
name      service      plan  bound apps  last operation
my-instance  ondemand-rabbitmq  Solo      create succeeded
```

3. Run `cf service SERVICE-INSTANCE-NAME` to retrieve more information about a specific instance.
4. Run `cf service SERVICE-INSTANCE-NAME --guid` to retrieve the GUID of the instance, which is useful for debugging.

## Retrieve RabbitMQ Instance Credentials

If you want to access the Management Dashboard or the RabbitMQ server for troubleshooting, you can create a new service-key to retrieve RabbitMQ instance credentials. Pivotal recommends that you use this key for troubleshooting only, and that you delete the key after troubleshooting. To retrieve the credentials, do the following:

1. Create a service-key for your RabbitMQ instance using the command `cf create-service-key INSTANCE-NAME SERVICE-KEY-NAME`.
2. Retrieve the credentials using the command `cf service-key INSTANCE-NAME SERVICE-KEY-NAME`.

For example:

```
$ cf create-service-key my-rmq-instance my-key
Creating service key my-key for service instance my-rmq-instance as admin...
OK
$ cf service-key my-rmq-instance my-key
Getting key my-key for service instance my-rmq-instance as admin...
{
  "host": "10.0.8.4",
  "password": "",
  "port": 6379
}
```

## Select the BOSH Deployment for a Service Instance

This is an additional troubleshooting option for **BOSH CLI v1 only**. It does not apply to the BOSH CLI v2.

1. Retrieve the GUID of your service instance with the command `cf service YOUR-SERVICE-INSTANCE --guid`.
2. To download your BOSH manifest for the service, run `bosh download manifest service-instance SERVICE-INSTANCE-GUID myservice.yml` using the GUID you just obtained and a file name you want to use when saving the manifest.
3. Run `bosh deployment MY-SERVICE.yml` to select the deployment.

## Knowledge Base (Community)

Find the answer to your question and browse product discussions and solutions by searching the [Pivotal Knowledge Base](#).

## File a Support Ticket


You can file a support ticket [here](#). Be sure to provide the error message from `cf service YOUR-SERVICE-INSTANCE`.

To help expedite troubleshooting, if possible also provide your service broker logs, service instance logs, and BOSH task output. Your cloud operator should be able to obtain these from your error message.

## Delete RabbitMQ Instance

On-Demand Broker provides a BOSH command to delete all the On-Demand Broker deployed instances. To delete the instances, do the following procedure:

1. **For BOSH CLI v1 only:** Set the deployment to the On-Demand Broker in BOSH.
2. Run one of the following commands to delete all instances of the On-Demand Broker, depending on your Ops Manager version:
  - **For v1.10 or earlier:** `bosh run errand delete-sub-deployments`
  - **For v1.11 or later:** `bosh2 run-errand delete-sub-deployments`

 **WARNING:** This command deletes deployment instances serially. It is very destructive and the operation cannot be undone.

## Installing and Configuring RabbitMQ for PCF as a Pre-Provisioned Service

This topic provides instructions to PCF operators about how to install, configure, and deploy the RabbitMQ for PCF tile to provide a pre-provisioned service.

**Note:** For instructions on how to install, configure, and deploy the RabbitMQ for PCF tile as an on-demand service, see [Installing and Configuring RabbitMQ for PCF as an On-Demand Service](#).

### About the BOSH CLI

The BOSH CLI is available in two major versions, v1 and v2. Pivotal recommends that you use the BOSH CLI v2 when possible.

This topic provides examples of using each version of the BOSH CLI. While all versions of the BOSH CLI work with RabbitMQ for PCF v1.9.x, your Ops Manager version may affect which version of the BOSH CLI you can use. Consult the table below to determine which version of the CLI is supported for your installation.

Ops Manager Version	BOSH CLI Version
1.10	CLI v1
1.11	CLI v1 or CLI v2 (Pivotal recommends CLI v2)
1.12 and later	CLI v2

### Download and Install the Tile

1. Download the product file from Pivotal Network.
2. Navigate to the Ops Manager Installation Dashboard and click **Import a Product** to upload the product file.
3. Under the **Import a Product** button, click + next to the version number of RabbitMQ for PCF. This adds the tile to your staging area.
4. Click the newly added **RabbitMQ for PCF** tile.

## Configure the Tile for Pre-Provisioned Service

### Management Dashboard

You must choose an admin username and password for RabbitMQ.

This will grant you full admin access to RabbitMQ through the Management UI.

RabbitMQ cluster configuration.

RabbitMQ admin user credentials \*

admin

\*\*\*\*\*

**Note:** To rotate your administrator credentials, enter a new username and password, save your options, and redeploy by returning to the Ops Manager Installation Dashboard and clicking **Apply Changes**.

### Plugins

You can choose which plugins you want to enable.

You must leave the management plugin enabled otherwise nothing will work.

RabbitMQ plugins \*

☐ rabbitmq\_amqp1\_0

☐ rabbitmq\_auth\_backend\_ldap

☐ rabbitmq\_auth\_mechanism\_ssl

☐ rabbitmq\_consistent\_hash\_exchange

☐ rabbitmq\_federation

☐ rabbitmq\_federation\_management

☒ rabbitmq\_management

☐ rabbitmq\_management\_visualiser

☐ rabbitmq\_mqtt

☐ rabbitmq\_shovel

☐ rabbitmq\_shovel\_management

☐ rabbitmq\_stomp

☐ rabbitmq\_tracing

☐ rabbitmq\_web\_stomp

☐ rabbitmq\_web\_stomp\_examples

☐ rabbitmq\_event\_exchange

☐ rabbitmq\_jms\_topic\_exchange

[Click here for more information about RabbitMQ plugins](#)

## HAProxy Ports

You can choose which ports HAProxy should load balance to the RabbitMQ nodes.

RabbitMQ cluster HAProxy ports \*

15672, 5672, 5671, 1883, 8883, 61613, 61614

By default, all the default ports of all the available plugins will be load-balanced.

However, if you install extra protocol plugins, or provide a custom configuration which changes the ports that RabbitMQ listens on then you must update the list of load-balanced ports.

Note that you must always leave the management plugin listening on port `15672` and load balance that port.

If you change the topology of your RabbitMQ cluster, the HAProxy is automatically reconfigured during the deployment.

## Disk free alarm limit

You can choose how much disk space rabbit should attempt to keep free at any given time. For more information about how this works ‘under the hood’, see [here](#). In summary, rabbit will periodically check if there is sufficient free space on disk. If there is not, rabbit will temporarily stop accepting new messages. This gives your apps time to consume existing messages, and thus free up some disk space. In the RabbitMQ Tile, we provide four options for this value:

Disk free alarm limit \*

☐ 50MB (legacy)

☒ 100% Memory (default)

☐ 150% Memory (recommended)

☐ 200% Memory (conservative)

Select the disk free limit after which RabbitMQ will raise an alarm. Value can be absolute or relative to the vm memory.

- `50MB` is a bare minimum value.

- `100% Memory` ensures that at the time when rabbit checks the available disk, there must be enough space for rabbit to page all memory-based messages out to disk.
- `150% Memory` is recommended. This is because it is possible that in between disk-space-checks, rabbit may:
  - write persistent messages to disk (using up some disk space)
  - accept more memory-based messages into various queues
  - page all memory-based messages to disk. In these circumstances, rabbit may require more free disk than it has memory.
- `200% Memory` is a conservative value, for when the operator wants higher confidence that rabbit will never run out of disk.

Selecting `50MB` is not recommended and could cause data loss. See below.

## What are the dangers of setting this value too low?

If the disk of a given rabbit node completely fills while rabbit is running, that node will crash. This can lead to data loss, and loss of availability.

Rabbit reserves the right to page any and all messages in memory (even transient messages) to disk at any time. You should set your `disk free alarm limit` high enough to ensure that rabbit always has at least enough space to do this.

## What are the disadvantages of setting this value too high?

If you set your `disk free alarm limit` to a value larger than the size of your persistent disk, then rabbit will never be able to free up enough disk space to accept new messages. You should ensure that you have a large enough disk to persist all the messages you intend to persist while *also* leaving enough space free to satisfy the `disk free alarm limit` that you have chosen.

## When should I use the `50MB` value?

This is not recommended in production. However, if you are experimenting with a development environment you might want to use a small disk to keep down costs, at the expense of increasing the chances that rabbit might crash and lose data.

## Connecting to a Highly Available RabbitMQ Cluster

The RabbitMQ tile, allows for a highly available cluster through multiple HAProxy nodes. The `hostnames`, `uris` and `hosts` properties have been added and should be used in preference over the equivalent singular properties. The singular properties are maintained for backwards compatibility and will always contain the first value from the equivalent plural property. The singular properties will eventually be deprecated.

For example with two HAProxy jobs deployed the following properties will be present:

```
"hostname": "10.0.0.41",
"hostnames": [
  "10.0.0.41",
  "10.0.0.51"]
```

## Port to protocol mappings

- 15672 = Management dashboard
- 5672 = RabbitMQ
- 5671 = RabbitMQ SSL
- 1883 = MQTT
- 8883 = MQTT SSL
- 61613 = STOMP
- 61614 = STOMP SSL
- 15674 = Web STOMP
- 4567 = RabbitMQ Service Broker
- 3457 - 3459 = CF Loggregator
- 4001 = CF Loggregator - Doppler

- 8300 - 8301 = Consul

## Security Groups

To enable access to the RabbitMQ tile service, you must ensure your security group allows access to the HAProxy and RabbitMQ Service Broker VMs configured in your deployment. You can obtain the IP addresses for these from the Ops Manager **Status** page for the RabbitMQ tile. Ensure the following ports are enabled for those VMs:

- Inbound

Port(s)	Protocol(s)	Source	Reason
15672	tcp	Broker and internet(*)	Allowing access to the RabbitMQ Management Dashboard & API
5671 - 5672	tcp	All AMQP clients	RabbitMQ will listen on those ports for AMQP
1883, 8883	tcp	All MQTT clients	RabbitMQ will listen on those ports for MQTT
61613, 61614	tcp	All STOMP clients	RabbitMQ will listen on those ports for STOMP
15674	tcp	All Web STOMP clients	RabbitMQ will listen on this port for STOMP-over-WebSockets
4567	tcp	ERT	ERT sends commands to the Service Broker for RabbitMQ
8080	tcp	ERT	ERT sends commands to the On Demand Service Broker for RabbitMQ
3457 - 3459	tcp	ERT	Between RabbitMQ and ERT network for Metrics
8300 - 8301	tcp, udp	ERT	Between RabbitMQ and ERT network for Consul

(\*) Everyone that needs to access the RabbitMQ Management Dashboard & API externally

- Outbound


Port(s)	Protocol(s)	Destination	Reason
3457 - 3459	tcp	ERT	Between RabbitMQ and ERT network for Metrics
4001	tcp	ERT	From RabbitMQ to ERT (etcd) for Metron
8300 - 8301	tcp, udp	ERT	Between RabbitMQ and ERT network for Consul

The following is a template for configuring your Cloud Foundry security groups:

```
[ {"protocol":"tcp","destination":"<haproxy-node-IP-addresses>","ports":"5671,5672,1883,8883,61613,61614,15672,15674"}, {"protocol":"tcp","destination":"<service-broker-node-IP-addresses>","ports":"4567"} ]
```

## Application Security Groups

To allow this service to have network access you must create [Application Security Groups](#) (ASGs).

 **Note:** The service is unusable without Application Security Groups.

## Application Container Network Connections

Application containers that use instances of the RabbitMQ service require the following outbound network connections:

Destination	Ports	Protocol	Reason
<a href="#">HAProxy IPs</a>	5672	tcp	Application containers using AMQP
<a href="#">HAProxy IPs</a>	5671	tcp	Application containers using AMQP over SSL
<a href="#">HAProxy IPs</a>	1883	tcp	Application containers using MQTT
<a href="#">HAProxy IPs</a>	8883	tcp	Application containers using MQTT over SSL
<a href="#">HAProxy IPs</a>	61613	tcp	Application containers using STOMP
<a href="#">HAProxy IPs</a>	61614	tcp	Application containers using STOMP over SSL
<a href="#">HAProxy IPs</a>	61613	tcp	Application containers using Web STOMP



Create an ASG name `rabbitmq-app-containers` with the above configuration and bind it to the appropriate space, or, to provide access to all started apps, bind it to the `default-running` ASG set and restart your apps. If you are using an external load balancer or have more than one IP address for HAProxy, you must also create egress rules for these. Example:

```
[
  {
    "ports": "5671-5672",
    "protocol": "tcp",
    "destination": "10.10.10.10/32"
  }
]
```

## SSL

You can provide SSL certificates and keys for use by the RabbitMQ cluster.

RabbitMQ server private key

RabbitMQ server certificate

RabbitMQ server CA certificate

SSL is simultaneously provided on the AMQPS port (5671) and the management port (15672).

If you provide SSL keys and certificates, you disable non-SSL support.

No other plugins are automatically configured for use with SSL.

SSL settings are applied equally across all machines in the cluster.

For more information about SSL support, see <https://www.rabbitmq.com/ssl.html>.

## Erlang Cookie

You can provide an Erlang cookie to be used by the cluster. This can be useful if you want to connect directly to the RabbitMQ cluster, such as with `rabbitmqctl`, or to connect other machines running Erlang.

Erlang cookie used by RabbitMQ nodes and rabbitmqctl

## Cluster Scaling Known Issue

If you have not set the Erlang cookie and you want to scale out your cluster size without downtime, follow these steps:

1. Follow the steps in one of the links below, up to and including the section *Log in to the BOSH Director*:

- For Ops Manager v1.10 or earlier: [Advanced Troubleshooting with the BOSH CLI](#)
- For Ops Manager v1.11 or later: [Prepare to Use the BOSH CLI](#)

2. Run one of the following commands depending on your Ops Manager version:

- For Ops Manager v1.10 or earlier: `bosh ssh rabbitmq-server/0`
- For Ops Manager v1.11 or later: `bosh2 ssh rabbitmq-server/0`

3. Run the following commands:

```
sudo -i
echo $(cat /var/vcap/store/rabbitmq/.erlang.cookie)
```

4. Paste the value returned from the last command into the Erlang cookie field in the **RabbitMQ** tab. This field is shown [above](#).

5. To increase the size of your cluster, navigate to the **Resource Config** tab and, in the first row, raise the value for the number of **Instances** of the **RabbitMQ node**.

6. Return to the Ops Manager Installation Dashboard, and click **Apply Changes**.

**Note:** BOSH tells you that the cookie has changed—this is because the default value in the manifest is empty, which results in an auto-generated cookie. However, the value of the cookie on the server remains the same, so the known issue below does **not** apply.

## Changing the Erlang Cookie Value Known Issue

If you want to change your Erlang cookie value, this will require cluster downtime. It is also *strongly* recommended that you do not change anything else, as it is possible for configuration to be inconsistently applied during this process.

The deployment may also fail—if it does, redeploying will fix the issue.

## RabbitMQ Config

You can optionally provide a full `rabbitmq.config` file by pasting its contents in the **RabbitMQ configuration** field in the **RabbitMQ** tab.

RabbitMQ configuration
RabbitMQ config file contents, can be blank

This `rabbitmq.config` file is then provided to all the nodes in the cluster. You can see an example `rabbitmq.config` file [here](#). For more information about the RabbitMQ configuration, see the [RabbitMQ documentation](#).

The input must be Base64 encoded.

For example, to configure the `rates_mode` of the `rabbitmq_management` stats, which looks like this:

```
[
  {rabbitmq_management, [
    {rates_mode, detailed}
  ]}
].
```

1. Encode the file into Base64:

```
WwogIHtyYWJiaXRicV9tYW5hZ2VtZW50LCBbCiAgICB7cmF0ZXNfbW9kZSwgZGV0YWlsZW9CigAgXX0KXS4K
```

2. Paste it into the **RabbitMQ configuration** field:

RabbitMQ configuration

WwogIHtyYWJiaXRtcV9tYW5hZ2VtZW50LCBbCiAgICB7cmF0ZXNfbW9kZSwgZGV0YWlsZWRR9CiAgXX0KXS4K

RabbitMQ config file contents, can be blank

## TLS Support

TLS v1.0 is disabled by default, due to security issues.

RabbitMQ TLS Versions \*

☐ TLS v1.0 (required for JDK 6.0)
   
☒ TLS v1.1
   
☒ TLS v1.2

TLS v1.1 and v1.2 are enabled by default and can be turned on and off.

## External load balancer

External load balancer DNS name

You can configure a DNS name or IP address of an external load balancer to be returned in the binding credentials ( `VCAP_SERVICES` ) to application developers.

## Assigned IPs

RabbitMQ for PCF does not support changing the IP addresses which have been assigned to the RabbitMQ deployments. For example, you cannot change the subnet into which the RabbitMQ cluster was originally provisioned. Doing so causes the deployment to fail. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).

## Static IPs

### Switching from dynamic IPs to static IPs (Upgrading)

It is not possible to switch from dynamic IPs to a different set of static IPs, but you can set up Ops Manager so the current set of dynamically assigned IPs will always continue to be used.

1. Go to the Status page on the RabbitMQ product.
2. Note the IPs for the RabbitMQ Server and HAProxy for RabbitMQ jobs, in the order nodes appear in the UI.
3. Go to the Settings tab, and navigate to the Networking page.
4. Fill the IP addresses you got from the Status page. IP addresses should be in a comma-separated list.

HAProxy Static IPs

RabbitMQ Server Static IPs

## RabbitMQ Server settings that cannot be overwritten

- `rabbit halt_on_upgrade_failure false`
- `rabbitmq_mqtt subscription_ttl 1800000`
- `log_levels [{connection,info}]`
- `halt_on_upgrade_failure false`
- `{rabbit, [ {collect_statistics_interval, 60000} ] }`
- `{rabbitmq_management, [ {rates_mode, none} ] }`

When SSL is enabled:

- `rabbit tcp_listeners []`
- `rabbit ssl_listeners [5671]`
- `rabbitmq_management listener [{port,15672},{ssl,false}]`
- `rabbitmq_mqtt ssl_listeners [8883]`
- `rabbitmq_stomp ssl_listeners [61614]`

## Configuring Metrics Polling Interval

The default polling interval is 30 seconds for all deployed components and it is recommended not to change it. The lowest interval setting is 10 seconds to avoid overwhelming components. This change will effect all deployed instances.

Metrics polling interval ( min: 10 ) \*

## Apply Configuration and Complete the Installation

1. Return to the Ops Manager Installation Dashboard and click **Apply Changes** to install RabbitMQ for PCF tile.
2. Q: How does one follow the progress of the tile, what to check? how long does it take?

## Smoke Test Process

Smoke tests run as a post-deployment errand. The smoke tests perform the following for each available service plan:

1. Target the org `system` and create a cf space to run the tests.
2. Deploy an instance of the [CF RabbitMQ Example App](#) in this cf space.
3. Create a RabbitMQ service instance and bind it to the CF RabbitMQ Example App.

4. Check that the CF RabbitMQ Example App can write to, and read from, the RabbitMQ service instance.
5. Clean up the deployed Example App and all its service bindings.
6. Delete the cf space created for the tests.

For more information, see [Errands](#).

## Creating Isolation with the RabbitMQ for PCF Replicator

### Overview

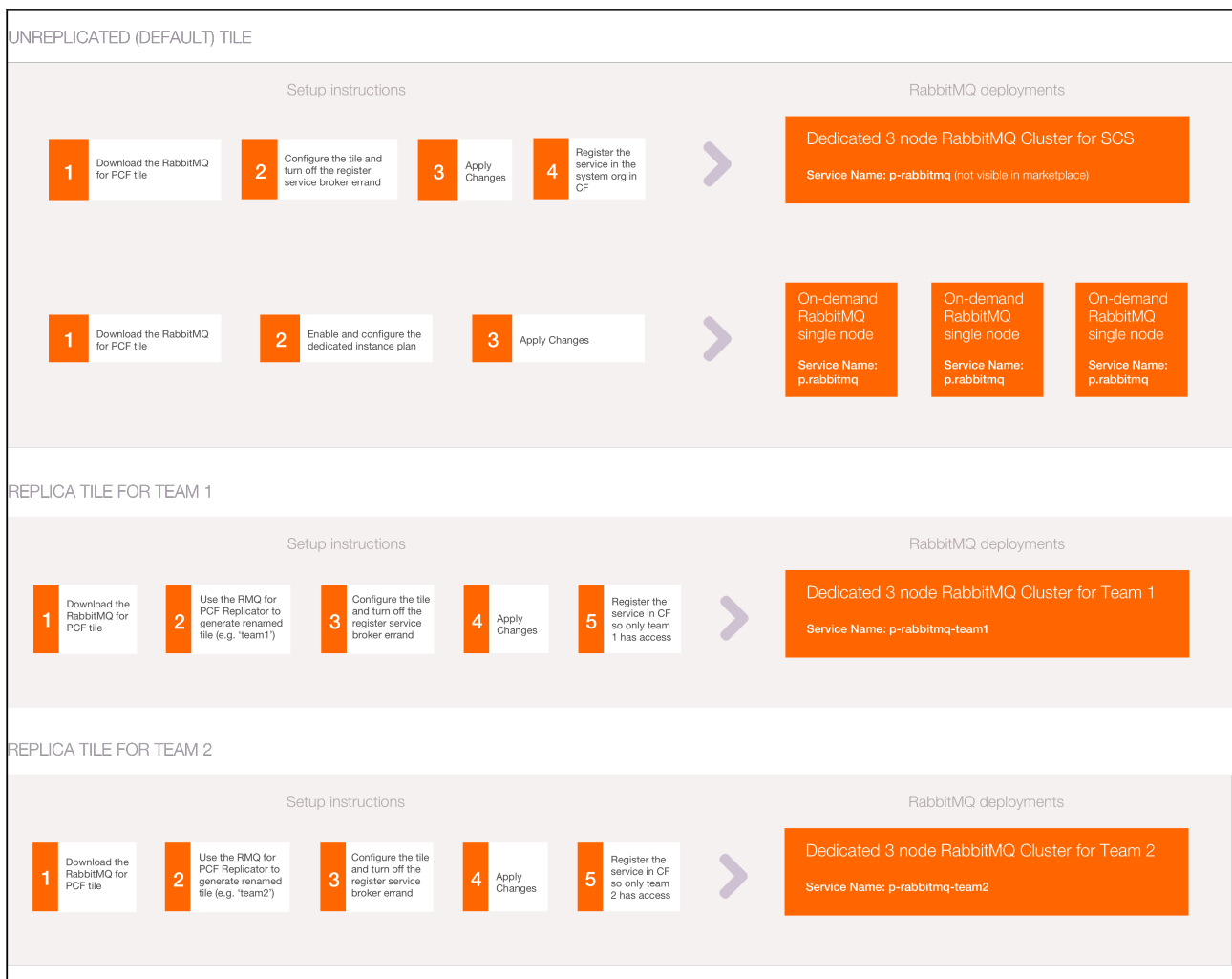
RabbitMQ for PCF Replicator is a tool that allows you to install multiple RabbitMQ for Pivotal Cloud Foundry (PCF) tiles in a single Ops Manager environment. This lets you run multiple pre-provisioned RabbitMQ clusters that are isolated from each other.

For example, you may want to isolate the cluster serving Spring Cloud Services (SCS) from the cluster serving apps in the Marketplace. Or you may want to give a certain team their own dedicated, pre-provisioned cluster that you manage for them.

### Common Use Cases

The image below illustrates how to isolate SCS on RabbitMQ for PCF from clustered and single node service instances dedicated to different teams. In this use case:

- The unreplicated RabbitMQ for PCF tile deploys two types of services:
  - A pre-provisioned service is used as a backing service for SCS
  - An on-demand service is used to create three completely isolated single node service instances
- Two replica tiles are used to create two dedicated pre-provisioned clusters, with each one dedicated to a specific team.



These and other common use cases are explained below.

## Running SCS on a Dedicated RabbitMQ Cluster

Replica RabbitMQ tiles cannot be used to provide a backing service for Spring Cloud Services (SCS) because SCS expects that the service is called `p-rabbitmq`. Therefore, if you want to isolate the RabbitMQ cluster that is used by SCS from other tenants, you can reserve the unreplicated RabbitMQ for PCF tile for SCS, as shown in the diagram below. You can then add replica RabbitMQ clusters for use by apps in the Marketplace.

Pivotal recommends that you use the unreplicated RabbitMQ for PCF tile solely for SCS to avoid contention between apps using SCS, and apps using RabbitMQ for PCF in the Marketplace.

To reserve the unreplicated tile for SCS, turn off the **Broker Registrar** errand to prevent the broker from being exposed in the Marketplace. For more information, see [Errands](#).

To offer RabbitMQ as a cloud messaging service in the Marketplace, create one or several replicas, install them in Ops Manager, and either allow the broker registrar errand to run, or [register the service manually using CF](#).

## Providing a Pre-Provisioned Dedicated Cluster

To reserve a RabbitMQ cluster for use by a specific team, all you need to do is [disable the broker registrar errand in Ops Manager](#). This prevents service registration in the Marketplace. After you deploy the tile, manually expose the service broker to your desired orgs and spaces. For instructions, see [Register a Broker](#).

## Using Replicas While Offering the On-Demand RabbitMQ Service

The On-Demand service is not offered in replica tiles, since the purpose of the replicator is to create additional pre-provisioned clusters. If you want to offer on-demand service plans, use the unreplicated RabbitMQ for PCF tile as shown in the diagram above.

## Blue-Green Upgrades (Advanced)

In order to do blue-green style upgrades to minimize downtime, you can stand up a new cluster and migrate data and users over to the new cluster over a period of time. Speak with your Platform Architect about how to enable this workflow.

## Generating Replica Tiles

This topic describes how to install the replicator and generate replica tiles of RabbitMQ for PCF.

### Prerequisites

- RabbitMQ for PCF v1.8.x or v1.9.x
- 2.5 GB of free disk space

### Download the Replicator

The RabbitMQ for PCF Replicator is currently available from [Pivotal Network](#). Search for and download this archive, and then run the enclosed binary.

### Generate Replica Tiles

The following is the syntax for generating a replica tile:

```
./rabbitmq-replicator-darwin\
--name YOUR_DESIRED_TILE_NAME\
--path PATH_TO_TILE\
--output DESIRED_FILE_NAME.pivotal
```

The following are the parameters expected in the above syntax:

Parameter	Description
<code>name</code>	The desired unique identifier for the replica tile, which is used to generate manifests, deployment names, and Marketplace name for the service. Only alphanumeric characters and <code>-</code> are accepted by the tool.
<code>path</code>	The location of the original, unreplicated, RabbitMQ for PCF source tile that you downloaded
<code>output</code>	The desired file name and path for the replica tile

## Installing Replica Tiles

After you have generated a replica tile, you can upload it to Ops Manager as you would any other tile. After you have uploaded it, follow the instructions for [Installing and Configuring RabbitMQ for PCF as a Pre-Provisioned Service](#). The On-Demand service is not offered on replica tiles.

## Upgrading Replica Tiles

You can upgrade replica tiles like regular tiles with one important difference. You must generate a replica of the newer version of the RabbitMQ for PCF tile, using the replicator, and give the new replica the same `name` as the existing replica. This is shown in the example workflow below.

### Example of an In-Place Upgrade of a Replica

Suppose you used the replicator to generate a replica of v1 of the RabbitMQ for PCF tile, with the `name` **trading-team**, and you installed it in Ops Manager. Here is the sample replicator command you used for the initial installation:

```
./rabbitmq-replicator-darwin\
--name trading-team\
--path /download/p-rabbitmq-v1.pivotal\
--output /output/p-rabbitmq-v1-trading-team.pivotal
```

To upgrade to v2, follow these steps:

1. Download the new RabbitMQ for PCF v2.
2. Run the replicator command, using the path to the new RabbitMQ for PCF v2 tile, and supply the same `name`, **trading-team**, as shown below.

```
./rabbitmq-replicator-darwin\
--name trading-team\
--path /download/p-rabbitmq-v2.pivotal\
--output /output/p-rabbitmq-v2-trading-team.pivotal
```

3. After you have the replica tile **p-rabbitmq-v2-trading-team.pivotal**, upload it to Ops Manager. This upgrades the v1 replica tile in place.

You can then proceed with upgrading the cluster.


If you want to do a blue-green style upgrade, see [Blue-Green Upgrades](#).

## Limitations

- The On-Demand service is not offered on replica tiles.



## Configuring RabbitMQ in an IPsec environment

 **Note:** If deploying RabbitMQ for PCF in an IPsec environment, the following steps **must** be performed

This option will configure and deploy RabbitMQ for PCF in a way that the machines in the deployment will not be in an IPsec network. This is necessary for cluster formation of RabbitMQ. As a result, we recommend you [configure RabbitMQ for PCF to use TLS](#).

### Limitations & Risks

- You can only deploy RabbitMQ to a single AZ.
- It is not possible to add a node to an existing RabbitMQ cluster when IPsec is enabled on the RabbitMQ nodes.
- Once IPs have been dynamically assigned (from a prior deployment) you cannot assign different static IPs.

### Installing the tile

1. Import and configure the RabbitMQ tile as usual, ensuring that you select only a single AZ.
2. On the Networking page enter the correct number of static IPs required for the number of HAProxy and RabbitMQ Server nodes you have configured on the resources page. These must be in a subnet on the AZ that you've configured the product to use.
3. Do not click Apply Changes until completing the following step.
4. Following the [IPsec Add-On documentation](#) add the static IPs you have configured to the `no_ipsec_subnets` list and update your runtime-config as the guide recommends.
5. Go back to the Installation Dashboard and click Apply Changes to deploy the changes. This will cause an update of all products, as the runtime-config has to be applied to all products.
6. Optionally, you can verify that traffic to the RabbitMQ Server and HAProxy nodes is unencrypted:
  - a. SSH into a node which should not be encrypted
  - b. Run `sudo tcpdump -i eth0 "ip proto 50"`, you should see no packets logged. This verifies there are no IPsec encrypted packets on that network interface. An IPsec encrypted packet will look like this:

```
11:13:07.801761 IP cloud-controller-0.node.dc1.cf.internal > ip-10-0-48-12.eu-west-1.compute.internal:
ESP(spi=0xcbb4206d,seq=0x2e4), length 232
```

## Deploying the RabbitMQ® Service

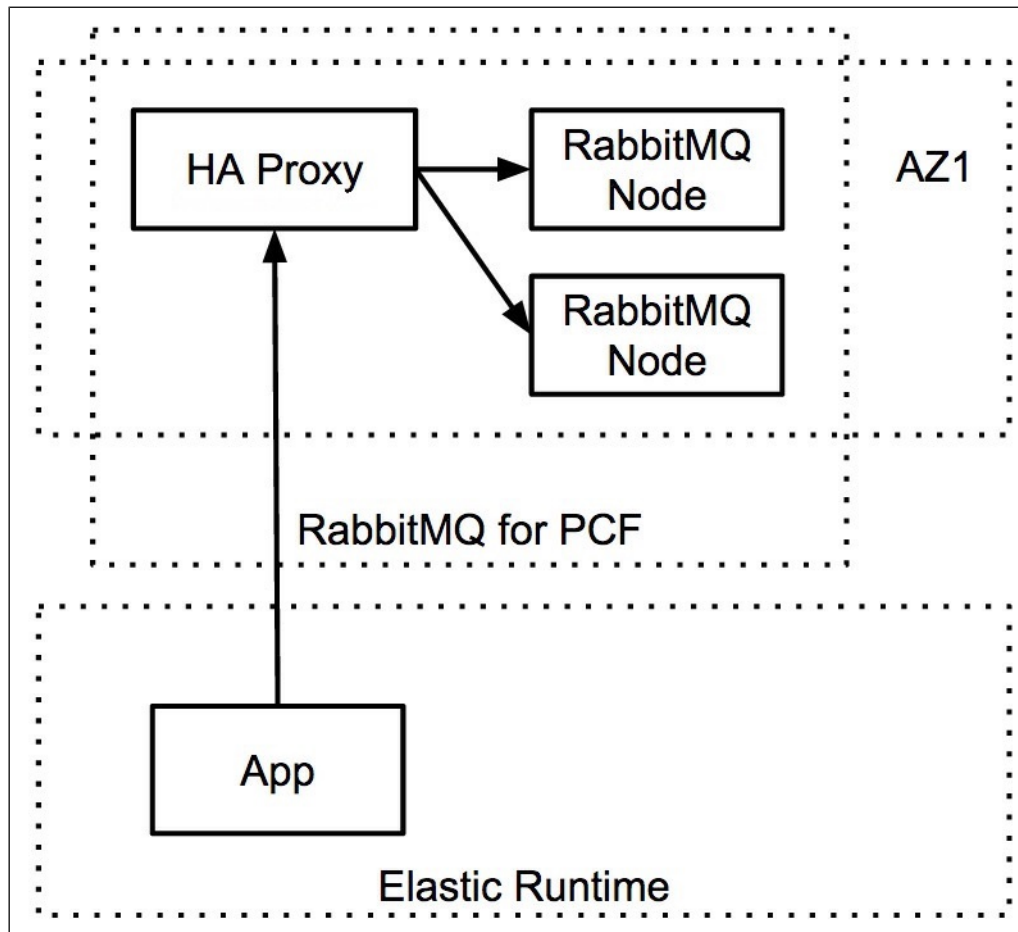
### Default Deployment

Deploying RabbitMQ for [Pivotal Cloud Foundry](#) (PCF) through Ops Manager will deploy a RabbitMQ cluster of **3 nodes** by default.

The deployment includes a single load balancer `haproxy` which spreads connections on all of the default ports, for all of the shipped plugins across all of the machines within the cluster.

The deployment will occur in a single availability zone (AZ).

The default configuration is for testing purposes only and it is recommended that customers have a minimum of **3 RabbitMQ nodes** and **2 HAProxy nodes**



### Considerations for this deployment

- Provides HA for the RabbitMQ cluster
- Queues must be judiciously configured to be HA as they are placed on one node by default
- Customers should decide on which partition behaviour is best suited to their use case. For two nodes 'automatic' is preferred
- HAProxy is a single point of failure (SPOF)
- The entire deployment is in a single AZ, which does not protect against external failures from failures in hardware, networking, etc.

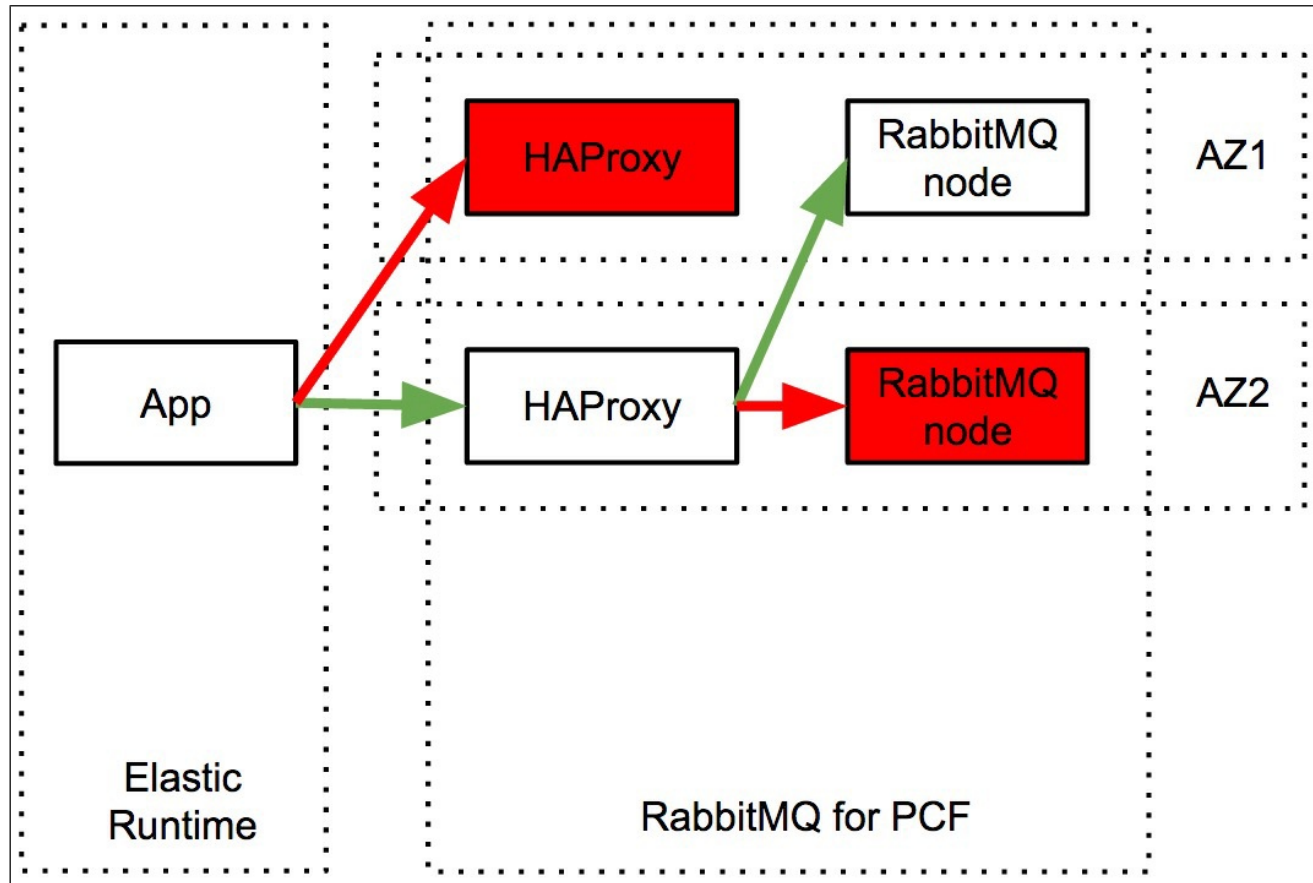
### Recommended Deployment

We recommend that RabbitMQ is deployed across at least two availability zones.

RabbitMQ server nodes should be scaled to an odd number and should be greater than 3.

Replication of queues should only be used where required as it can have a big impact on system performance.

The HAProxy job instance count should also be increased to match the number of AZs to ensure there is a HAProxy located in each AZ. This removes the HAProxy SPOF and provides further redundancy.



In the above diagram, you can see that you can now suffer the failure of a single HAProxy and single RabbitMQ node and still keep your cluster online and applications connected.

It is also recommend that customers chooses an odd number of RabbitMQ server nodes of three or more.

## Upgrading to this deployment from a single AZ deployment

It is **not** possible to upgrade to this setup from the default deployment across a single AZ.

This is because the AZ setup cannot be changed once the tile has being deployed for the first time, this is to protect against data loss when moving jobs between AZs.

## Upgrading to this deployment from a multi AZ deployment

If you have deployed the tile across two AZs, but with a single HAProxy instance you can migrate to this setup as follows:

1. Deploy an additional HAProxy instance through Ops Manager
2. New or re-bound applications to the RabbitMQ service will see the IPs of both HAProxys immediately
3. Existing bound applications will continue to work, but only using the previously deployed HAProxy IP Address. They can be re-bound as required at your discretion.

## Considerations for this deployment

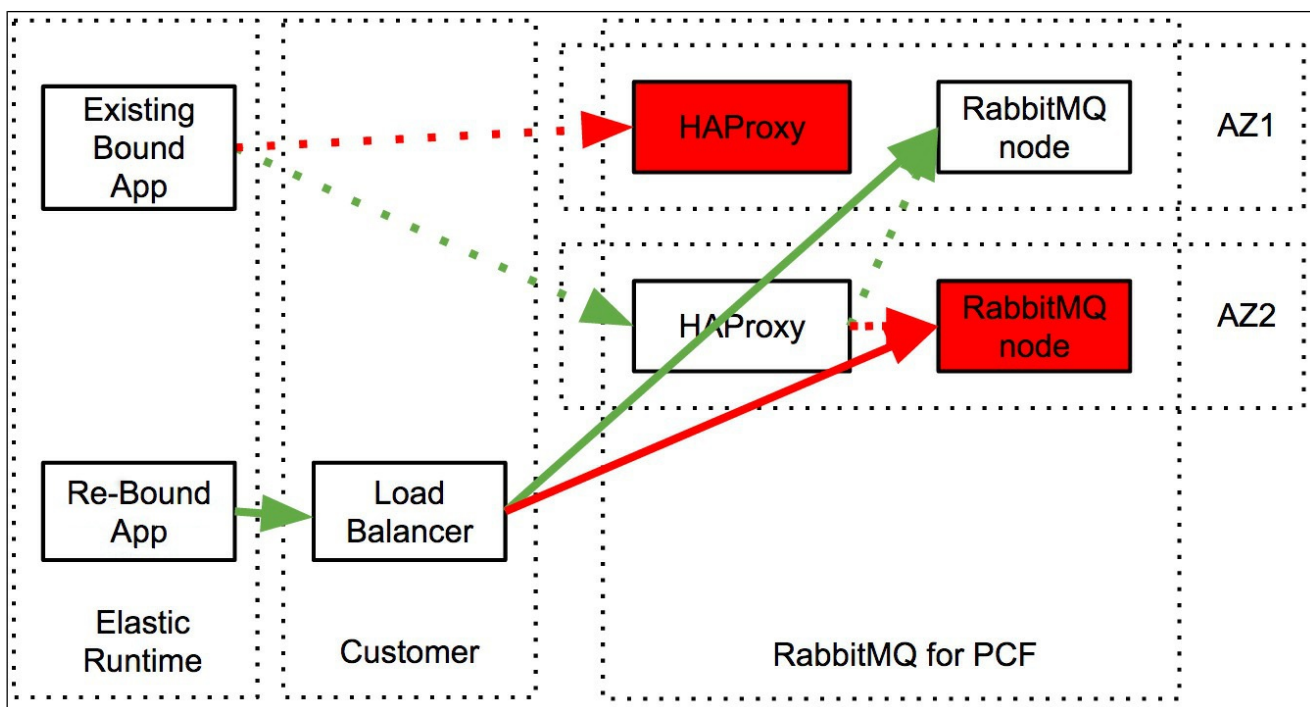
- Requires IaaS configuration for availability zones ahead of deploying the RabbitMQ tile
- Application developers will be handed the IPs of each deployed HAProxy in their environment variables
- Queues must be judiciously configured to be HA as they are placed on one node by default
- Customers should decide on which partition behaviour is best suited to their use case. For 3 or more nodes 'pause\_minority' is preferred

## Advanced Deployment

This deployment builds upon the above recommended deployment, so follows the same upgrade paths.

This allows you to replace the use of HAProxy with your own external load balancer.

You may choose to do this to remove any knowledge of the topology of the RabbitMQ setup from application developers.



### Advantages

- Application developers do not need to handle multiple IPs for the HAProxy jobs in their applications

### Disadvantages

- The load balancer needs to be configured with the IPs of the RabbitMQ Nodes. These will only be known once the deployment has finished. The IPs should remain the same during subsequent deployments but there is a risk they can change.

## Upgrading to this deployment from the recommended deployment

It is possible to first deploy with multiple HAProxy jobs, as per the recommended deployment and decided to later use your own external load balancer.

This can be achieved without downtime to your applications.

This can be achieved as follows:

1. Configure your external load balancer to point to the RabbitMQ Node IPs
2. Configure the DNS name or IP address for the external load balancer (ELB) on the RabbitMQ tile in Ops Manager
3. Deploy the changes

4. Any new instances of the RabbitMQ service or any re-bound connections will use the DNS name or IP address of the ELB in their `VCAP_SERVICES`
5. Any existing instances will continue to use the HAProxy IP addresses in their `VCAP_SERVICES`
6. Phase the re-binding of existing applications to have their environment variables updated
7. Once all applications are updated
8. Reduce the instance count of the `HAProxy` job in Ops Manager to 1
9. Deploy the changes

This approach works as any existing bound applications have their `VCAP_SERVICES` information cached in the cloud controller and are only updated by a re-bind request.

## Downgrading from this deployment to the recommended deployment

If you are currently using an external load balancer, then you can move back to using HAProxys instead.

You can achieve this by following the above steps in reverse order and re-instating the HAProxy jobs.

## Resource requirements

The following table shows the default resource and IP requirements for installing the tile:

Product	Resource	Instances	CPU	Ram	Ephemeral	Persistent	Static IP	Dynamic IP
RabbitMQ	RabbitMQ node	3	2	8192	16384	30720	1	0
RabbitMQ	HAProxy for RabbitMQ	1	1	2048	4096	0	1	0
RabbitMQ	RabbitMQ service broker	1	1	2048	4096	0	1	0
RabbitMQ	Broker Registrar	1	1	1024	2048	0	0	1
RabbitMQ	Broker Deregistrar	1	1	1024	2048	0	0	1
RabbitMQ	Smoke Tests	1	1	1024	2048	0	0	1
RabbitMQ	RabbitMQ on-demand broker	1	1	1024	8192	1024	0	1
RabbitMQ	Register On-Demand Service Broker	1	1	1024	2048	0	0	1
RabbitMQ	Deregister On-Demand Service Broker	1	1	1024	2048	0	0	1
RabbitMQ	Delete All Service Instances	1	1	1024	2048	0	0	1
RabbitMQ	Upgrade All Service Instances	1	1	1024	2048	0	0	1

### Notes:

- The number of `RabbitMQ Node` can be increased if required.
- Changing the number of RabbitMQ nodes when the erlang cookie is not defined will restart the cluster. Check [here](#) for more information.

## RabbitMQ® for Pivotal Cloud Foundry

### Upgrades


This product enables automated upgrades between versions of the product and is deployed through Ops Manager, and due to RabbitMQ product limitations may require the cluster to be taken offline. When this is necessary it is clearly noted in the release notes for that version.


Note there is a difference between the cluster remaining available during a tile upgrade/update, and an individual queue placed on nodes in a cluster. The upgrade paths are detailed [here](#) for each released version.

A reference guide for deployments is shown the table below. Please be aware that this is a guide only and that the release notes for the version you are updating to must be checked before upgrading.

Operations Manager Action	Will Downtime Be Required For This Upgrade / Update
Major Tile Version Upgrade	<ul style="list-style-type: none"> <li>The RabbitMQ cluster will be taken offline for the duration of the upgrade</li> </ul>
Minor Tile Version Upgrade	<ul style="list-style-type: none"> <li>The RabbitMQ cluster will be taken offline for the duration of the upgrade</li> </ul>
Patch Tile Version Upgrades	<ul style="list-style-type: none"> <li>Normally these are rolling deployments with each node being updated in turn. In these cases the cluster will remain available but individual queues may be taken offline, as each node is restarted. There are specific migration paths which will require downtime which will be identified in the release notes for that version.</li> </ul>
Stemcell Only - Patch Tile Version Upgrades	<ul style="list-style-type: none"> <li>Where the patch update is only a new stemcell version these are rolling deployments with each node being updated in turn. In these cases the cluster will remain available but individual queues may be taken offline, as each node is restarted.</li> </ul>

The specific upgrade paths are detailed [here](#) for each released version.

 **Note:** For specific information about updating RabbitMQ for PCF from v1.6.0–v1.6.4, see [Updating RabbitMQ for PCF from versions v1.6.x to v1.6.6](#).

 **Note:** For specific information about updating RabbitMQ for PCF from v1.6.0–v1.6.4, see [Updating RabbitMQ for PCF from versions v1.6.x to v1.6.6](#).

To upgrade the product:

- The Operator should download the latest version of the product from [Pivotal Network](#)
- Upload the new .pivotal file to Ops Manager
- Upload the stemcell associated with the update (*if required*)
- Update any new mandatory configuration parameters (*if required*)
- Press “Apply changes” and the rest of the process is automated

Depending on the network partition handling that you have configured and the resource config, you should benefit from the product’s stemcell rolling upgrades. The default HAProxy instance count has changed from 1 to 2 and the RabbitMQ node count from 2 to 3. This is required for rolling stemcell upgrades. This is the minimum deployment size that we recommend for the default `pause_minority` RabbitMQ network partition behaviour. See [Clustering and Network Partitions](#) for more information.

Only when upgrading between specific versions of Erlang or RabbitMQ will the cluster become unavailable. This will be clearly stated on the release notes for that version, should this be the case.

The length of the downtime depends on whether there is a stemcell update to replace the operating system image or whether the existing VM can simply have the RabbitMQ software updated. Stemcell updates incur additional downtime while the IaaS creates the new VM.

Ops Manager ensures the instances are updated with the new packages and any configuration changes are applied automatically.

Upgrading to a newer version of the product should not cause any loss of data or configuration.

Please note that it may take busy RabbitMQ nodes a long time to shutdown during the upgrade and this process must not be interrupted.

For any issues upgrading RabbitMQ, please refer to the [Troubleshooting](#) section.

## What do I need to do before upgrading to a newer version of RabbitMQ or Erlang?

Ensure the cluster is healthy via the RabbitMQ Management UI. You cannot rely on the BOSH `instances` output, that reflects the state of Erlang VM, not

## Why do I want to stop RabbitMQ?

We have to stop RabbitMQ when we are upgrading RabbitMQ package or Erlang VM version.

## Release policy

When a new version of RabbitMQ is released we aim to release a new version of the product containing this soon after.

Where there is a new version of RabbitMQ or another dependent software component such as the stemcell released due to a critical CVE, Pivotal's goal is to release a new version of the product within 48 hours.

## Default policies for the RabbitMQ® Service

### RabbitMQ Policy

The configuration box shown below can be used by Operators to set a default queue and an exchange policy to be applied to their RabbitMQ cluster. We recommend that you use the RabbitMQ Management Interface to make configuration changes after deployment.

An example policy is provided below. It ensures messages are mirrored on two nodes. Operators should consider some of the performance implications of making queues and exchanges highly available, and refer to the following documentation for more information: <https://www.rabbitmq.com/ha.html>

```
{
  "ha-mode": "exactly",
  "ha-params": 2,
  "ha-sync-mode": "automatic"
}
```

The following rules apply to policies set through this configuration box:

- The policy is only applied to new instances
- Any existing instances will not have the policy applied
- The policy can be updated in Ops Manager, and will be applied only to any new instances
- The policy can only be deleted manually from the RabbitMQ nodes
- Policies can be added dynamically using the RabbitMQ Management Interface

## Viewing or changing the policy

In Ops Manager on the RabbitMQ tile is a left-hand menu item named **RabbitMQ Policy**. The checkbox is not enabled by default, and so no policy will be applied.

RabbitMQ Policy definition applied to new instances

☒ Enable custom policy on new instances

Policy for new instances \*

```
{
  "ha-mode": "exactly", "ha-params": 2,
  "ha-sync-mode": "automatic"
}
```

The policy must be valid JSON and should meet valid RabbitMQ policy criteria. No validation occurs during the deployment, and errors can cause the deployment to fail or policies to be applied incorrectly.

For more information, view [RabbitMQ Policies](#).

## RabbitMQ dashboard

You can view the policy on the RabbitMQ Dashboard. You can obtain the URL can be obtained from the your `VCAP_SERVICES` for application developers.

The example policy is applied to all queues and given a rank of `50`. This allows you to override it by defining your own policy with a higher rank.



## Policies

### ▼ All policies

Filter:  ☐ Regex (?)

Name	Pattern	Apply to	Definition	Priority
<b>operator_set_policy</b>	.*	all	ha-mode: exactly ha-params: 2 ha-sync-mode: automatic	50

You can see any new queues created have the policy automatically applied.

## Queues

### ▼ All queues

Filter:  ☐ Regex (?)

Overview			Messages			Message rates			+/-
Name	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack	
<b>test</b>	<span>D</span> operator_set_policy	<span>running</span>	0	0	0				

## Network partition behavior

You can change how RabbitMQ acts once it discovers there has been a network partition. The two options are `pause_minority` and `autoheal`, and more detail on these settings can be found here: <https://www.rabbitmq.com/partitions.html> [↗](#)

You must choose the option you want before deploying, or the default `pause_minority` will be used. For production purposes, we recommend that customers have at least three RabbitMQ server nodes and two HAProxies spread across low latency availability zones.

## Monitoring and KPIs for Pre-Provisioned RabbitMQ for PCF

This topic explains how to monitor the health of the pre-provisioned version of the RabbitMQ for Pivotal Cloud Foundry (PCF) service using the logs, metrics, and Key Performance Indicators (KPIs) generated by RabbitMQ for PCF component VMs.

Pre-provisioned RabbitMQ for PCF components generate many of the [same metrics](#) as the on-demand RabbitMQ service components.

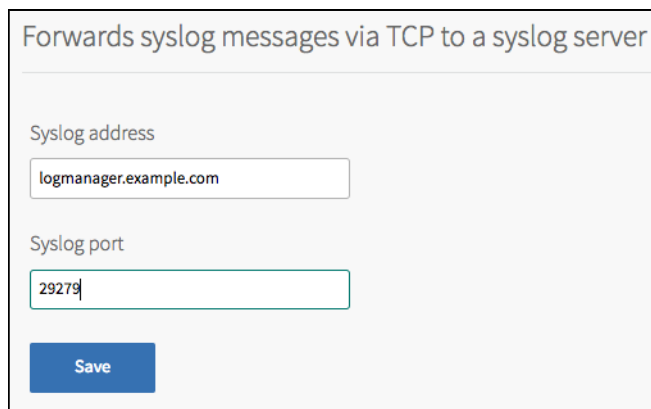
See [Logging and Metrics](#) for general information about logging and metrics in PCF.

### Direct the Logs

To enable monitoring for RabbitMQ for PCF, operators designate an external syslog endpoint for RabbitMQ component log messages. The endpoint serves as the input to a monitoring platform such as Datadog, Papertrail, or SumoLogic.

To specify the destination for RabbitMQ for PCF log messages, do the following:

1. From the Ops Manager Installation Dashboard, click the RabbitMQ tile.
2. In the RabbitMQ tile, click the **Settings** tab.
3. Click **Syslog**.



4. Enter your syslog address and port.
5. Click **Save**.
6. Return to the Ops Manager Installation Dashboard and click **Apply Changes** to redeploy with the changes.

### Logging Formats

With pre-provisioned RabbitMQ for PCF logging configured, three types of component generate logs: the RabbitMQ message server nodes, the service broker, and HAProxy. If you have multiple server or HAProxy nodes, you can identify logs from individual nodes by their index, which corresponds to the index of the RabbitMQ VM instances displayed in Ops Manager:

- The logs for RabbitMQ server nodes follow the format `[job=rabbitmq-server-partition-GUID index=X]`
- The logs for HAProxy nodes follow the format `[job=rabbitmq-haproxy-partition-GUID index=X]`
- The logs for the RabbitMQ service broker follow the format `[job=rabbitmq-broker-partition-GUID index=X]`

RabbitMQ and HAProxy servers log at the `info` level and capture errors, warnings, and informational messages.

The logging format does not change in v1.9.0. For users familiar with documentation for previous versions of the tile, the tag we used to call the `app_name` is now called the `program_name`. The generic log format is as follows:

```
<PRI>TIMESTAMP IP_ADDRESS PROGRAM_NAME [job=NAME index=JOB_INDEX id=JOB_ID] MESSAGE
```

The raw logs look similar to the following:

```
<7>2017-06-28T16:06:10.733560+00:00 10.244.16.133 vcap.agent [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] 2017/06/28 16:06:10 CEF:0|CloudFoundry|BOSH|1|agent_api|ssh|1|duser=dir
<8>2017-06-28T16:06:16.704572+00:00 10.244.16.133 useradd [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] new group: name=bosh_ly0d2rbjr, GID=1003
<8>2017-06-28T16:06:16.704663+00:00 10.244.16.133 useradd [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] new user: name=bosh_ly0d2rbjr, UID=1001, GID=1003, home=/var/vcap/bosh_
<8>2017-06-28T16:06:16.736932+00:00 10.244.16.133 usermod [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] add 'bosh_ly0d2rbjr' to group 'admin'
<8>2017-06-28T16:06:16.736964+00:00 10.244.16.133 usermod [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] add 'bosh_ly0d2rbjr' to group 'vcap'
```

Logs sent to external logging tools such as Papertrail may be presented in a different format.

The following table describes the logging tags used in this template:

Tag	Description
PRI	This is a value which in future will be used to describe the severity of the log message and which facility it came from.
TIMESTAMP	This is the timestamp of when the log is forwarded, for example, <code>2016-08-24T05:14:15.000003Z</code> . The timestamp value is typically slightly after when the log message was generated.
IP_ADDRESS	The internal IP address of server on which the log message originated
PROGRAM_NAME	Process name of the program the generated the message. Same as <code>app_name</code> before v1.9.0. For more information about program name, see <a href="#">RabbitMQ Program Names</a> below.
NAME	The BOSH instance group name (for example, <code>rabbitmq_server</code> )
JOB_INDEX	BOSH job index. Used to distinguish between multiple instances of the same job.
JOB_ID	BOSH VM GUID. This is distinct from the CID displayed in the Ops Manager Status tab, which corresponds to the VM ID assigned by the infrastructure provider.
MESSAGE	The log message that appears

## RabbitMQ Program Names

Program Name	Description
rabbitmq_server_cluster_check	Checks that the RabbitMQ cluster is healthy. Runs after every deploy.
rabbitmq_server_node_check	Checks that the RabbitMQ node is healthy. Runs after every deploy.
rabbitmq_route_registrar_stderr	Registers the route for the management API with the Gorouter in your Elastic Runtime deployment.
rabbitmq_route_registrar_stdout	Registers the route for the management API with the Gorouter in your Elastic Runtime deployment.
rabbitmq_server	The Erlang VM and RabbitMQ apps. <i>Logs may span multiple lines</i>
rabbitmq_server_drain	Shuts down the Erlang VM and RabbitMQ apps. Runs as part of the BOSH lifecycle.
rabbitmq_server_http_api_access	Access to the RabbitMQ management UI.
rabbitmq_server_init	Starts the Erlang VM and RabbitMQ.
rabbitmq_server_post_deploy_stderr	Runs the node check and cluster check. Runs after every deploy.
rabbitmq_server_post_deploy_stdout	Runs the node check and cluster check. Runs after every deploy.
rabbitmq_server_pre_start	Runs before the rabbitmq-server job is started.
rabbitmq_server_sasl	Supervisor, progress, and crash reporting for the Erlang VM and RabbitMQ apps.
rabbitmq_server_shutdown_stderr	Stops the RabbitMQ app and Erlang VM.
rabbitmq_server_shutdown_stdout	Stops the RabbitMQ app and Erlang VM.
rabbitmq_server_startup_stderr	Starts the RabbitMQ app and Erlang VM, then configures users and permissions.
rabbitmq_server_startup_stdout	Starts the RabbitMQ app and Erlang VM, then configures users and permissions.
rabbitmq_server_upgrade	Shuts down Erlang VM and RabbitMQ app if required during an upgrade.

## Metrics

Metrics are regularly-generated log messages that report measured component states. The metrics polling interval defaults to 30 seconds. This interval is a configuration option on the RabbitMQ tile ( **Settings** > **RabbitMQ**). The interval setting applies to all components deployed by the tile.

Metrics are long, single lines of text that follow the format:

```
origin:"p-rabbitmq" eventType:ValueMetric timestamp:1441188462382091652 deployment:"cf-rabbitmq" job:"cf-rabbitmq-node" index:"0" ip:"10.244.3.46" valueMetric: < name:"/p-rabbitmq/rabbitmq/system/m
```

## Partition Indicator

A new metric has been introduced to help to identify network partitions. Essentially it exposes how many nodes each node knows. When a node is in partition the only node that it recognizes is itself and that is a good indication that that node might be in a partition.

An example of that metrics is:

```
origin:"p-rabbitmq" eventType:ValueMetric timestamp:1441188462382091652 deployment:"cf-rabbitmq" job:"cf-rabbitmq-node" index:"0" ip:"10.244.3.46" valueMetric: < name:"/p-rabbitmq/rabbitmq/erlang/re
```

Monitors can be created to emit alerts in case a cluster seems to be in a partition. A metrics is emitted for each node in the cluster. For example: in a three-node cluster a monitor can expect to have a total of 9 (nine) since each node is expected to emit 3 (2 reachable nodes and itself). Otherwise, an alert can be sent to the team.

## Recovering from a network partition

Please refer to the official RabbitMQ guide to understand how to recover from a network partition: <https://www.rabbitmq.com/partitions.html>

## Key Performance Indicators

Key Performance Indicators (KPIs) for RabbitMQ for PCF are metrics that operators find most useful for monitoring their RabbitMQ service to ensure smooth operation. KPIs are high-signal-value metrics that can indicate emerging issues. KPIs can be raw component metrics or *derived* metrics generated by applying formulas to raw metrics.

Pivotal provides the following KPIs as general alerting and response guidance for typical RabbitMQ for PCF installations. Pivotal recommends that operators continue to fine-tune the alert measures to their installation by observing historical trends. Pivotal also recommends that operators expand beyond this guidance and create new, installation-specific monitoring metrics, thresholds, and alerts based on learning from their own installations.

For a list of all RabbitMQ for PCF raw component metrics, see [Component Metrics Reference](#).

## Component Heartbeats

Key RabbitMQ for PCF components periodically emit heartbeat metrics: the RabbitMQ server nodes, HAProxy nodes, and the Service Broker. The heartbeats are Boolean metrics, where `1` means the system is available, and `0` or the absence of a heartbeat metric means the service is not responding and should be investigated.

### Service Broker Heartbeat

p-rabbitmq.service_broker.heartbeat	
<b>Description</b>	<p>RabbitMQ Service Broker <code>is alive</code> poll, which indicates if the component is available and able to respond to requests.</p> <p><b>Use:</b> If the Service Broker does not emit heartbeats, this indicates that it is offline. The Service Broker is required to create, update, and delete service instances, which are critical for dependent tiles such as Spring Cloud Services and Spring Cloud Data Flow.</p> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> boolean  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
<b>Recommended measurement</b>	Average over last 5 minutes
<b>Recommended alert thresholds</b>	<p><b>Yellow warning:</b> N/A  <b>Red critical:</b> &lt; 1</p>
<b>Recommended response</b>	<p>Check the RabbitMQ Service Broker logs for errors. You can find this VM by targeting your RabbitMQ deployment with BOSH and running one of the following commands depending on your Ops Manager Version:</p> <ul style="list-style-type: none"> <li><b>For Ops Manager v1.10 or earlier:</b>  <code>bosh vms service-instance_GUID</code></li> </ul>

- For Ops Manager v1.11 or later:

```
bosh2 -d service-instance_GUID vms
```

## HAProxy Heartbeat

p-rabbitmq.haproxy.heartbeat	
Description	<p>RabbitMQ HAProxy <code>is alive</code> poll, which indicates if the component is available and able to respond to requests.</p> <p><b>Use:</b> If the HAProxy does not emit heartbeats, this indicates that it is offline. To be functional, service instances require HAProxy.</p> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> boolean  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> N/A  <b>Red critical:</b> &lt; 1</p>
Recommended response	<p>Check the RabbitMQ HAProxy logs for errors. You can find the VM by targeting your RabbitMQ deployment with BOSH and running one of the following commands, which lists <code>HAProxy_GUID</code>:</p> <ul style="list-style-type: none"> <li>• For Ops Manager v1.10 or earlier:  <pre>bosh vms service-instance_GUID</pre></li> <li>• For Ops Manager v1.11 or later:  <pre>bosh2 -d service-instance_GUID vms</pre></li> </ul>


## Server Heartbeat

p-rabbitmq.rabbitmq.heartbeat	
Description	<p>RabbitMQ Server <code>is alive</code> poll, which indicates if the component is available and able to respond to requests.</p> <p><b>Use:</b> If the server does not emit heartbeats, this indicates that it is offline. To be functional, service instances require RabbitMQ Server.</p> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> boolean  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> N/A  <b>Red critical:</b> &lt; 1</p>
Recommended response	<p>Check the RabbitMQ Server logs for errors. You can find the VM by targeting your RabbitMQ deployment with BOSH and running one of the following commands, which lists <code>rabbitmq</code>:</p> <ul style="list-style-type: none"> <li>• For Ops Manager v1.10 or earlier:  <pre>bosh vms service-instance_GUID</pre></li> <li>• For Ops Manager v1.11 or later:  <pre>bosh2 -d service-instance_GUID vms</pre></li> </ul>

## RabbitMQ Server KPIs

The following KPIs from the RabbitMQ server component:

## File Descriptors

p-rabbitmq.rabbitmq.system.file_descriptors	
Description	<p>File descriptors consumed.</p> <p><b>Use:</b> If the number of file descriptors consumed becomes too large, the VM may lose the ability to perform disk IO, which can cause data loss.</p> <div>  <b>Note:</b> This assumes non-persistent messages are handled by retries or some other logic by the producers.         </div> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> count  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> &gt; 50000  <b>Red critical:</b> &gt; 55000</p>
Recommended response	<p>The default <code>ulimit</code> for RabbitMQ for PCF v1.6 and later is 60000. If this metric is met or exceeded for an extended period of time, consider one of the following actions:</p> <ul style="list-style-type: none"> <li>Scaling the rabbit nodes in the tile <b>Resource Config</b> pane.</li> <li>Increasing the <code>ulimit</code></li> </ul>

## Erlang Processes

p-rabbitmq.rabbitmq.system.erlang_processes	
Description	<p><a href="#">Erlang</a> processes consumed by RabbitMQ, which runs on an Erlang VM.</p> <p><b>Use:</b> This is the key indicator of the processing capability of a node.</p> <p><b>Origin:</b> Doppler/Firehose  <b>Type:</b> count  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> &gt; 900000  <b>Red critical:</b> &gt; 950000</p>
Recommended response	<p>The default Erlang process limit in RabbitMQ for PCF v1.6 and later is 1,048,816. If this metric meets or exceeds the recommended thresholds for extended periods of time, consider scaling the RabbitMQ nodes in the tile <b>Resource Config</b> pane.</p>

## BOSH System Health Metrics

The BOSH layer that underlies PCF generates `healthmonitor` metrics for all VMs in the deployment. However, these metrics are not included in the Loggregator Firehose by default. To get these metrics, do either of the following:

- To send BOSH HM metrics through the Firehose, install the open-source [HM Forwarder](#).
- To retrieve BOSH health metrics outside of the Firehose, install the [JMX Bridge](#) for PCF tile.

In a future release the BOSH system health metrics will be available directly from the Firehose.

All BOSH-deployed components generate the following system metrics. Coming from RabbitMQ for PCF components, these system metrics serve as KPIs for the RabbitMQ for PCF service.

## RAM

system.mem.percent	
Description	<p>RAM being consumed by the <code>p-rabbitmq</code> VM.</p> <p><b>Use:</b> RabbitMQ is considered to be in a good state when it has little or no messages. In other words, “an empty rabbit is a happy rabbit.” Alerting on this metric can indicate that there are too few consumers or apps that read messages from the queue.</p> <p>Healthmonitor reports when RabbitMQ uses more than 40% of its RAM for the past ten minutes.</p> <p><b>Origin:</b> JMX Bridge or BOSH HM  <b>Type:</b> percent  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> &gt; 40  <b>Red critical:</b> &gt; 50</p>
Recommended response	Add more consumers to drain the queue as fast as possible.

## CPU

system.cpu.percent	
Description	<p>CPU being consumed by the <code>p-rabbitmq</code> VM.</p> <p><b>Use:</b> A node that experiences context switching or high CPU usage will become unresponsive. This also affects the ability of the node to report metrics.</p> <p>Healthmonitor reports when RabbitMQ uses more than 40% of its CPU for the past ten minutes.</p> <p><b>Origin:</b> JMX Bridge or BOSH HM  <b>Type:</b> percent  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> &gt; 60  <b>Red critical:</b> &gt; 75</p>
Recommended response	Remember that “an empty rabbit is a happy rabbit”. Add more consumers to drain the queue as fast as possible.

## Ephemeral Disk

system.disk.percent	
Description	<p>Ephemeral Disk being consumed by the <code>p-rabbitmq</code> VM.</p> <p><b>Use:</b> If system disk fills up, there are too few consumers.</p> <p>Healthmonitor reports when RabbitMQ uses more than 40% of its CPU for the past ten minutes.</p> <p><b>Origin:</b> JMX Bridge or BOSH HM  <b>Type:</b> percent  <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p><b>Yellow warning:</b> &gt; 60  <b>Red critical:</b> &gt; 75</p>
Recommended response	Remember that “an empty rabbit is a happy rabbit”. Add more consumers to drain the queue as fast as possible.

## Persistent Disk

persistent.disk.percent	
<b>Description</b>	<p>Persistent Disk being consumed by the <code>p-rabbitmq</code> VM.</p> <p><b>Use:</b> If system disk fills up, there are too few consumers.</p> <p>Healthmonitor reports when RabbitMQ uses more than 40% of its CPU for the past ten minutes.</p> <p><b>Origin:</b> JMX Bridge or BOSH HM</p> <p><b>Type:</b> percent</p> <p><b>Frequency:</b> 30 s (default), 10 s (configurable minimum)</p>
<b>Recommended measurement</b>	Average over last 10 minutes
<b>Recommended alert thresholds</b>	<p><b>Yellow warning:</b> &gt; 60</p> <p><b>Red critical:</b> &gt; 75</p>
<b>Recommended response</b>	Remember that “an empty rabbit is a happy rabbit”. Add more consumers to drain the queue as fast as possible.

## Component Metric Reference

RabbitMQ for PCF component VMs emit the following raw metrics. The full name of the metric follows the format: `/p-rabbitmq/COMPONENT/METRIC-NAME`

### RabbitMQ Server Metrics

RabbitMQ for PCF message server components emit the following metrics.

Full Name	Unit	Description
<code>/p-rabbitmq.rabbitmq.heartbeat</code>	boolean	Indicates whether the RabbitMQ server is available and able to respond to requests
<code>/p-rabbitmq/rabbitmq/erlang/erlang_processes</code>	count	The number of Erlang processes
<code>/p-rabbitmq/rabbitmq/system/memory</code>	MB	The memory in MB used by the node
<code>/p-rabbitmq/rabbitmq/system/mem_alarm</code>	boolean	Indicates if the memory alarm has gone off
<code>/p-rabbitmq/rabbitmq/system/disk_free_alarm</code>	boolean	Indicates if the disk free alarm has gone off
<code>/p-rabbitmq/rabbitmq/connections/count</code>	count	The total number of connections to the node
<code>/p-rabbitmq/rabbitmq/consumers/count</code>	count	The total number of consumers registered in the node
<code>/p-rabbitmq/rabbitmq/messages/delivered</code>	count	The total number of messages with the status <code>deliver_get</code> on the node
<code>/p-rabbitmq/rabbitmq/messages/delivered_no_ack</code>	count	The number of messages with the status <code>deliver_no_ack</code> on the node
<code>/p-rabbitmq/rabbitmq/messages/delivered_rate</code>	rate	The rate at which messages are being delivered to consumers or clients on the node
<code>/p-rabbitmq/rabbitmq/messages/published</code>	count	The total number of messages with the status <code>publish</code> on the node
<code>/p-rabbitmq/rabbitmq/messages/published_rate</code>	rate	The rate at which messages are being published by the node
<code>/p-rabbitmq/rabbitmq/messages/redelivered</code>	count	The total number of messages with the status <code>redeliver</code> on the node
<code>/p-rabbitmq/rabbitmq/messages/redelivered_rate</code>	rate	The rate at which messages are getting the status <code>redeliver</code> on the node
<code>/p-rabbitmq/rabbitmq/messages/got_no_ack</code>	count	The number of messages with the status <code>get_no_ack</code> on the node
<code>/p-rabbitmq/rabbitmq/messages/get_no_ack_rate</code>	rate	The rate at which messages get the status <code>get_no_ack</code> on the node
<code>/p-rabbitmq/rabbitmq/messages/pending</code>	count	The number of messages with the status <code>messages_unacknowledged</code> on the node
<code>/p-rabbitmq/rabbitmq/messages/depth</code>	count	The number of messages with the status <code>messages_unacknowledged</code> or <code>messages_ready</code> on the node
<code>/p-rabbitmq/rabbitmq/system/file_descriptors</code>	count	The number of open file descriptors on the node



/p-rabbitmq/rabbitmq/exchanges/count	count	The total number of exchanges on the node
/p-rabbitmq/rabbitmq/messages/available	count	The total number of messages with the status <code>messages_ready</code> on the node
/p-rabbitmq/rabbitmq/queues/count	count	The number of queues on the node
/p-rabbitmq/rabbitmq/channels/count	count	The number of channels on the node
/p-rabbitmq/rabbitmq/queues/VHOST-NAME/QUEUE-NAME/consumers	count	The number of consumers per virtual host per queue
/p-rabbitmq/rabbitmq/queues/VHOST-NAME/QUEUE-NAME/depth	count	The number of messages with the status <code>messages_unacknowledged</code> or <code>messages_ready</code> per virtual host per queue

## HAProxy Metrics

RabbitMQ for PCF HAProxy components emit the following metrics.

Name Space	Unit	Description
/p-rabbitmq.haproxy.heartbeat	boolean	Indicates whether the RabbitMQ HAProxy component is available and able to respond to requests
/p-rabbitmq/haproxy/health/connections	count	The total number of concurrent front-end connections to the server
/p-rabbitmq/haproxy/backend/qsize/amqp	size	The total size of the AMQP queue on the server
/p-rabbitmq/haproxy/backend/retries/amqp	count	The number of AMQP retries to the server
/p-rabbitmq/haproxy/backend/ctime/amqp	time	The total time to establish the TCP AMQP connection to the server

## Clustering and Network Partitions

### Clustering in RabbitMQ for PCF

In RabbitMQ for PCF, the RabbitMQ® broker is always deployed as a cluster of one or more virtual machines (nodes). A RabbitMQ broker is a logical grouping of one or several Erlang nodes, each running the RabbitMQ application and sharing users, virtual hosts, queues, exchanges, bindings, and runtime parameters.

#### What is Replicated between nodes in a RabbitMQ cluster?

All data/state required for the operation of a RabbitMQ broker is replicated across all nodes. An exception to this are message queues, which by default reside on one node, though they are visible and reachable from all nodes. This means that the RabbitMQ cluster may be available and serving requests, while an individual queue residing on a single node is offline.

Replicating message queues across nodes is an expensive operation and should only be done to the extent needed by the application. To understand more about replicating queues across nodes in a cluster, see the [documentation](#) on high availability.

### Automatic Network Partition Behaviors in RabbitMQ Clusters

The RabbitMQ® tile uses the `pause_minority` option for handling cluster partitions by default. This ensures data integrity by pausing the partition of the cluster in the minority, and resumes it with the data from the majority partition. You must maintain more than two nodes. If there is a partition when you only have two nodes, both nodes immediately pause.

You can also choose the `autoheal` option in the **RabbitMQ Policy** tab. In this mode, if a partition occurs, RabbitMQ automatically decides on a winning partition, and restarts all nodes that are not in the winning partition. This option allows you to continue to receive connections to both parts of partitions.

### Detecting a Network Partition

When a network partition occurs, a log message is written to the RabbitMQ node log:

```
=ERROR REPORT==== 15-Oct-2012::18:02:30 ===
Mnesia(rabbit@da3be74c053640fe92c6a39e2d7a5e46): ** ERROR ** mnesia_event got
{inconsistent_database, running_partitioned_network, rabbit@21b6557b73f343201277dbf290ae8b79}
```

You can also run the `rabbitmqctl cluster_status` command on any of the RabbitMQ nodes to see the network partition. To run `rabbitmqctl cluster_status`, do the following:

1. `$ sudo su -`
2. `$ cd /var/vcap/packages`
3. `$ export ERL_DIR=$PWD/erlang/bin/`
4. `$ cd rabbitmq-server/bin/`
5. `$ ./rabbitmqctl cluster_status`

```
[...
{partitions,
 [{rabbit@da3be74c053640fe92c6a39e2d7a5e46,
  [rabbit@21b6557b73f343201277dbf290ae8b79]}]}
```

### Recovering

Because the RabbitMQ tile uses the `pause_minority` option, minority nodes recover automatically after the partition is resolved. After a node recovers, it resumes accessing the queue along with data from the queues on the other nodes. However, if your queues use `ha-mode: all`, they only synchronize fully after consuming all the messages created while the node was down. This is similar to how messages synchronize when you create a new queue.

## Manually Synchronizing after a Partition

After a network partition, a queue on a minority node synchronizes after consuming all the messages created while it was down. You can also run the `sync_queue` command to synchronize a queue manually. To run `sync_queue`, do the following on each node:

1. `$ sudo su -`
2. `$ cd /var/vcap/packages`
3. `$ export ERL_DIR=$PWD/erlang/bin/`
4. `$ cd rabbitmq-server/bin/`
5. `$ ./rabbitmqctl list_queues`
6. `$ ./rabbitmqctl sync_queue name`

## Managing the RabbitMQ® Service

### RabbitMQ Management Dashboard

#### Admin User

To gain access to the management dashboard as the `admin` user, visit `http://pivotal-rabbitmq.<cf sys domain>`, where can be found at “Pivotal Elastic Runtime” configuration on “Domains” section: “System Domain” field.

The username and password is the username and password you provided in the RabbitMQ configuration in Ops Manager, which is also shown in the Credentials tab.

**RabbitMQ**

**Overview** | Connections | Channels | Exchanges | Queues | Admin

### Overview

**Totals**

Queued messages (chart: last minute) (?)

1.0  
0.0

15:35:20 15:35:30 15:35:40 15:35:50 15:36:00 15:36:10

Ready 0 msg  
Unacked 0 msg  
Total 0 msg

Message rates (chart: last minute) (?)

Currently idle

Global counts (?)

Connections: 0 | Channels: 0 | Exchanges: 15 | Queues: 0 | Consumers: 0

**Nodes**

Name	File descriptors (?)	Socket descriptors (?)	Erlang processes	Memory	Disk space	Info
rabbit@6031f2ce968ce856e9f684c504f6aecd	20 1024 available	1 829 available	177 1048576 available	40MB 3.1GB high watermark	7.3GB 977kB low watermark	Disc 2 Stats
rabbit@623a32cce88bc994de244cec4e6c6085	20 1024 available	1 829 available	175 1048576 available	39MB 3.1GB high watermark	7.3GB 977kB low watermark	Disc 2

**Ports and contexts**

Listening ports

Protocol	Node	Bound to	Port
amqp	6031f2ce968ce856e9f684c504f6aecd	::	5672
amqp	623a32cce88bc994de244cec4e6c6085	::	5672
clustering	6031f2ce968ce856e9f684c504f6aecd	::	25672
clustering	623a32cce88bc994de244cec4e6c6085	::	25672

Web contexts

Context	Node	Bound to	Port	SSL	Path
RabbitMQ Management	6031f2ce968ce856e9f684c504f6aecd	0.0.0.0	15672	o	/
RabbitMQ Management	623a32cce88bc994de244cec4e6c6085	0.0.0.0	15672	o	/

#### Application Developer

Users of Cloud Foundry who create instances via the Apps Manager or the cf CLI also get access to the Management UI. This is done using credentials that provide access only to their particular virtual host.

The appropriate URL is accessible via the Manage button within the Apps Manager.

SERVICES <span>Add Service</span>		
SERVICE INSTANCE	SERVICE PLAN	BOUND APPS
rabbit <a href="#">Manage</a>   <a href="#">Documentation</a>   <a href="#">Support</a>   <a href="#">Delete</a>	RabbitMQ for Pivotal CF Production	0

Or it is also injected into the `VCAP_SERVICES` environment variable provided to apps running on Cloud Foundry. This can also be found via the CLI using

```
cf env <your app name>
```

```
➔ ~ cf env lab-rat
Getting env variables for app lab-rat in org development / space development as me...
OK

System-Provided:
{
  "VCAP_SERVICES": {
    "p-rabbitmq": [
      {
        "credentials": {
          "dashboard_url": "http://pivotal-rabbitmq.crystal.pepsi.cf-app.com/#/login/fd87c5d4-ca55-41f3-ba8a-cf0b0711f052/243bvlaa8c5m5dka1701b207ff",
```

## Logging

A TCP Syslog endpoint can be configured in Ops Manager. Logs are currently only forwarded for the RabbitMQ cluster.

## RabbitMQ CLI

If you want to run commands such as `rabbitmqctl` then you have two options:

SSH into one of the machines running the rabbitmq-server. IPs can be found from the Status tab and access credentials from the Credentials tab within the RabbitMQ component of the installer. From there you need to bring RabbitMQ and Erlang into your environment and from there you can use

```
rabbitmqctl :
```

```
bash-4.1# export PATH=$PATH:/var/vcap/packages/rabbitmq-server/bin
bash-4.1# export PATH=$PATH:/var/vcap/packages/erlang/bin
bash-4.1# rabbitmqctl cluster_status
Cluster status of node rabbit@node0 ...
[{nodes,[{disc,[rabbit@node0,rabbit@node1,rabbit@node2,rabbit@node3]}]},
 {running_nodes,[rabbit@node3,rabbit@node2,rabbit@node1,rabbit@node0]},
 {partitions,[]}]]
...done.
```

Alternatively, install RabbitMQ and Erlang on a machine of your choice. Be sure to match versions of both to the cluster: the Management UI shows both the version of RabbitMQ and Erlang.

Then set your `~/.erlang.cookie` to match the cookie used in the cluster (you may have supplied this as part of the installation; see above).

You will need to set up your `/etc/hosts` file to match the RabbitMQ nodes.