

# VMware RabbitMQ for Tanzu Application Service v1.18

VMware RabbitMQ for Tanzu Application Service 1.18

You can find the most up-to-date technical documentation on the VMware website at:

<https://docs.vmware.com/>

**VMware, Inc.**  
3401 Hillview Ave.  
Palo Alto, CA 94304  
[www.vmware.com](http://www.vmware.com)

Copyright © 2023 VMware, Inc. All rights reserved. [Copyright and trademark information.](#)

# Contents

RabbitMQ for Pivotal Platform	15
RabbitMQ for Pivotal Platform	15
About RabbitMQ for Pivotal Platform	15
What are On-Demand Instances	15
About RabbitMQ	16
Product Snapshot	16
Features	16
On-Demand	16
Pre-Provisioned	17
Release Notes and Known Issues	17
RabbitMQ and Other On-Demand Services	17
Feedback	18
RabbitMQ for Pivotal Platform Release Notes	18
Upgrade to the Latest Version	18
v1.18.9	19
Features	19
Resolved Issues	19
Known Issues	19
Compatibility	19
v1.18.8	20
Resolved Issues	20
Known Issues	21
Compatibility	21
v1.18.6	21
Resolved Issues	22
Known Issues	22
Compatibility	22
v1.18.5	23
Resolved Issues	23
Known Issues	24
Compatibility	24
v1.18.4	25
Security Fix	25

Features	25
Resolved Issues	25
Known Issues	26
Compatibility	26
v1.18.3	27
Resolved Issues	27
Known Issues	27
Compatibility	28
v1.18.2	29
Features	29
Known Issues	29
Compatibility	30
View Release Notes for Another Version	30
<b>Unlocking the Power of On-Demand RabbitMQ for Pivotal Platform</b>	<b>31</b>
Introduction	31
Deciding Which Service Plan to Use	31
On-Demand Single Node Plan	33
On-Demand Cluster Plan	33
General Principles of the Cluster Plan	33
Designed for Consistency	34
Number of Nodes	34
Network Latency	34
Consistency or Availability Tradeoff	34
RabbitMQ Queue Availability	35
Managing On-Demand Resources Through Plans	35
Customizing Plan Options	36
Configuration Options	36
Single Node and Cluster Plans	36
Cluster Plan Only	36
Things That Are Preconfigured	37
Monitoring On-Demand RabbitMQ Clusters	38
About Migrating a Pre-Provisioned Instance to an On-Demand Instance	38
Differences between Pre-Provisioned and On-Demand Services Instances	39
<b>Architecture</b>	<b>40</b>
<b>On-Demand Service Architecture</b>	<b>40</b>
Service Network Requirement	40
Default Network and Service Network	40



Required Networking Rules for On-Demand Services	41
Deploying the RabbitMQ Pre-Provisioned Service	43
Default Deployment	44
Considerations for the Default Deployment	44
Recommended Deployment	45
Upgrading to the Recommended Deployment from a Single AZ Deployment	45
Upgrading to the Recommended Deployment from a Multi AZ Deployment	45
Considerations for the Recommended Deployment	46
Advanced Deployment	46
Upgrading to the Advanced from the Recommended Deployment	47
Downgrading from the Advanced Deployment to the Recommended Deployment	48
Resource Requirements	48
Notes:	48
Service-Gateway Access (Beta)	49
Overview	49
Architecture	49
Operator Guide: On- Demand	51
Installing and Configuring	51
Preparing for TLS	51
Overview	51
Prerequisite	52
Workflow	52
Find the CredHub Credentials in Ops Manager	52
Add the CA Certificate	53
Enable TLS in RabbitMQ	55
Installing and Configuring the On-Demand Service	55
Role-Based Access in Pivotal Operations Manager	56
Prerequisites for Deploying the On-Demand Service	56
Download and Install RabbitMQ	56
Configure On-Demand RabbitMQ	57
Configure AZs and Networks	57
Configure Logging and Monitoring	58
Configure Global Settings	58
Configure Security	61

Configure the Service Plan	62
Determine which AZs a Service Instance Uses	65
RabbitMQ VM Types and Persistent Disk Size	65
Verify the Stemcell	67
Apply Changes from Your Configuration	67
Errands	67
Post-Deploy Errands	68
Pre-Delete Errands	68
Create an Admin User for a Service Instance	68
RabbitMQ Server Plugins	69
Enabled RabbitMQ Server Plugins	70
Optional RabbitMQ Server Plugins	70
<b>Enabling Service-Gateway Access (Beta)</b>	<b>71</b>
Overview	71
Enable TCP Routing Using the PAS Tile	72
Configure the Firewall to Allow Incoming Traffic to the TCP Router	72
Configure the Load Balancer in the IaaS to Redirect Traffic to the TCP Router	72
Create a DNS Record That Maps to the Load Balancer	73
Configure a Service-Gateway-Enabled Plan	73
Disable Service-Gateway Access	75
Developer Workflow	76
<b>Smoke Tests</b>	<b>76</b>
Smoke Test Steps	76
Troubleshooting	76
<b>Monitoring and KPIs for On-Demand RabbitMQ for Pivotal Platform</b>	<b>76</b>
Set up Syslog Forwarding and Metrics Polling Interval	77
Logging Format	79
RabbitMQ Program Names	80
What Are Metrics	81
Key Performance Indicators	81
Component Heartbeats	82
Service Broker Heartbeat	82
Server Heartbeat	82
RabbitMQ Server KPIs	83
File Descriptors	83
Erlang Processes	83
BOSH System Health Metrics	84

RAM	84
CPU	84
Ephemeral Disk	85
Persistent Disk	85
Determine If There Is a Network Partition	86
Recover from a Network Partition	86
Component Metric Reference	86
RabbitMQ Server Metrics	86
Managing the On- Demand Service	88
Rotating Certificates	88
Setting Limits for On-Demand Service Instances	88
Create Global-level Quotas	89
Create Plan-level Quotas	90
Create and Set Org-level Quotas	90
Create and Set Space-level Quotas	90
View Current Org and Space-level Quotas	91
Monitor Quota Use and Service Instance Count	92
Calculate Resource Costs for On-Demand Plans	92
Calculate Maximum Resource Cost Per On-Demand Plan	93
Calculate Maximum Resource Cost for All On-Demand Plans	94
Calculate Actual Resource Cost of all On-Demand Plans	94
Controlling Access to Service Plans by Org	94
Change Access to Service Plans	94
Troubleshooting and FAQs for On-Demand RabbitMQ for Pivotal Platform	95
How to Retrieve a Service Instance GUID	95
Troubleshoot Errors	95
Common Services Errors	95
RabbitMQ-Specific Errors	100
Troubleshoot Components	101
BOSH Problems	101
Large BOSH Queue	101
Configuration	101
Service Instances in Failing State	101
Authentication	101
UAA Changes	101

Networking	101
Validate Service Broker Connectivity to Service Instances	102
Validate App Access to Service Instance	102
Quotas	102
Plan Quota Issues	102
Global Quota Issues	103
Failing Jobs and Unhealthy Instances	103
Techniques for Troubleshooting	103
Parse a Cloud Foundry (CF) Error Message	103
Access Broker and Instance Logs and VMs	104
Access Broker Logs and VMs	104
Access Service Instance Logs and VMs	105
Run Service Broker Errands to Manage Brokers and Instances	105
Register Broker	106
Deregister Broker	106
Upgrade All Service Instances	106
Delete All Service Instances	107
Detect Orphaned Service Instances	107
Get Admin Credentials for a Service Instance	109
Reinstall a Tile	110
View Resource Saturation and Scaling	111
Identify Apps using a Service Instance	111
Monitor the Quota Saturation and Service Instance Count	111
Drop and Restore AMQP or AMQPS Traffic to a RabbitMQ Service Instance	112
Frequently Asked Questions	112
What should I check before deploying a new version of the tile?	112
What is the correct way to stop and start RabbitMQ?	113
What happens when I run bosh stop rabbitmq-server?	113
What happens when bosh stop rabbitmq-server fails?	113
What do I do when bosh stop rabbitmq-server fails?	113
How can I manually back up the state of the RabbitMQ cluster?	114
Back up Manually	114
Back up and Restore with a Script	114
What pre-upgrade checks should I do?	114
Knowledge Base (Community)	114
File a Support Ticket	114
Operator Guide: Pre- Provisioned	116
Turning Off the Pre-Provisioned Service	116

Disable the Pre-Provisioned Service Manually	116
Disable the Pre-Provisioned Service Using Automation	117
Installing and Configuring	117
Installing and Configuring the Pre-Provisioned Service	117
Role-Based Access in Pivotal Operations Manager	118
Download and Install the Tile	118
Configure Pre-Provisioned RabbitMQ	118
Assign AZs and Networks	119
Pre-Provisioned RabbitMQ	120
RabbitMQ Admin User Credentials	120
Plugins	121
Erlang Cookie	121
Erlang Cookie Security Fix	122
Changing the Erlang Cookie Value Known Issue	122
Set the Erlang Cookie to Its Existing Value	122
External Load Balancer	123
Enable Custom Policy on New Instances	123
HAProxy Ports	123
SSL	124
TLS Support	126
RabbitMQ Configuration	126
Select the Network Partition Behavior of the RabbitMQ Cluster	127
Disk Free Alarm Limit	127
Dangers of Setting This Value Too Low	128
Disadvantages of Setting This Value Too High	128
When to Use the 50MB Value	128
Specify Static IP Addresses	128
Configure Syslog Forwarding and Metrics Polling Interval	129
Global Settings	131
Dedicated Instance: Single Node Plan	131
Errands	132
Post-Deploy Errands	132
Pre-Delete Errands	133
Stemcell	133
Apply Configuration and Complete the Installation	133
Other Configuration Topics	134
Connecting to a Highly Available RabbitMQ Cluster	134
Port to protocol mappings	134

Security Groups	134
Application Security Groups	135
Application Container Network Connections	135
Assigned IPs	136
Preserving Dynamically Assigned IPs	136
RabbitMQ Server Settings that Cannot be Overwritten	136
Smoke Tests	137
Smoke Test Steps	137
Troubleshooting	137
Monitoring and KPIs for Pre- Provisioned RabbitMQ for Pivotal Platform	138
Setting up Syslog Forwarding	138
Logging Formats	138
RabbitMQ Program Names	139
Metrics	140
Partition Indicator	140
Recovering from a network partition	141
Key Performance Indicators	141
Component Heartbeats	141
Service Broker Heartbeat	141
HAProxy Heartbeat	142
Server Heartbeat	142
RabbitMQ Server KPIs	143
File Descriptors	143
Erlang Processes	143
BOSH System Health Metrics	144
RAM	144
CPU	145
Ephemeral Disk	145
Persistent Disk	145
Component Metric Reference	146
RabbitMQ Server Metrics	146
HAProxy Metrics	148
Managing the Pre- Provisioned Service	148
Isolating Clusters with the RabbitMQ for Pivotal Platform Replicator	148
Overview	148
Common Use Cases	149

Running SCS on a Dedicated RabbitMQ Cluster	149
Providing a Pre-Provisioned Dedicated Cluster	150
Using Replicas While Offering the On-Demand RabbitMQ Service	150
Blue-Green Upgrades (Advanced)	150
Generating Replica Tiles	150
Prerequisites	150
Download the Replicator	150
Generate Replica Tiles	150
Naming Conventions in Original and Replica Tiles	151
Installing Replica Tiles	151
Limiting Access to Replica Tiles to Specific Orgs	151
Upgrading Replica Tiles	152
Example of an In-Place Upgrade of a Replica	152
Limitations	154
<b>Setting Default Policies for the RabbitMQ Service</b>	<b>154</b>
About RabbitMQ Policies	154
Rules for Policies Set in the Tile	154
An Example Policy: Mirror on Two Nodes	154
Best Practice for Syncing Queues	155
Setting or Changing the Policy	155
Viewing Policies in the RabbitMQ Management UI	156
<b>Frequently Asked Questions for Pre-Provisioned RabbitMQ for Pivotal Platform</b>	<b>156</b>
Frequently Asked Questions	157
What should I check before deploying a new version of the tile?	157
What is the correct way to stop and start RabbitMQ for Pivotal Platform?	157
What happens when I run <code>bosh stop rabbitmq-server</code> ?	157
What happens when <code>bosh stop rabbitmq-server</code> fails?	157
What do I do when <code>bosh stop rabbitmq-server</code> fails?	157
How can I manually back up the state of the RabbitMQ cluster?	158
Back up Manually	158
Back up and Restore with a Script	158
What pre-upgrade checks should I do?	158
Knowledge Base (Community)	158
File a Support Ticket	158
<b>Using the RabbitMQ Management UI</b>	<b>160</b>
Access the RabbitMQ Management UI	160

Pre-Provisioned Service Instances	161
On-Demand Service Instances	161
Clustering and Network Partitions	163
Clustering in RabbitMQ for Pivotal Platform	163
Automatic Network Partition Behaviors in RabbitMQ Clusters	163
Detecting a Network Partition	163
Recovering	164
Manually Synchronizing after a Partition	164
Upgrading RabbitMQ for Pivotal Platform	165
About the Upgrade	165
General Notes About Downtime	165
General Notes About the Upgrade Process	166
Release Policy	166
Upgrade RabbitMQ	166
Rolling Upgrades in RabbitMQ for Pivotal Platform	168
Overview	168
Running a Rolling Upgrade for RabbitMQ for Pivotal Platform	168
Example Rolling Upgrade Scenario	169
Configuration and Setup	169
Cluster Configuration	169
App Configuration	169
Observations	170
Cluster Configuration Considerations	171
Developer Guide	172
Using On-Demand RabbitMQ for Pivotal Platform	172
Prerequisites	172
Create Resilient Apps	172
Confirm Service Availability	172
Create a Service Instance	173
Use Transport Layer Security (TLS)	174
Enable TLS for Your Service Instance	174
Disable TLS for Your Service Instance	175
Activate TLS for Java and Spring Apps	176
Enable Optional Plugins	177
Disable Optional Plugins	178
Bind a Service Instance to Your App	178



Use the RabbitMQ Service in Your App	178
RabbitMQ Node Address Resolution	180
Update a Service Instance	180
Unbind a Service Instance From Your App	180
Delete a Service Instance	181
Create an Admin User for a Service Instance	181
Share Service Instances	181
Access RabbitMQ Metrics for On-Demand Service Instances	182
Federate Exchanges and Queues	182
Shovel Exchanges and Queues	183
<b>Modifying Apps for TLS</b>	<b>184</b>
Prerequisites	184
Modify Your App for TLS	184
Re-push or Rebind Your App	185
<b>Migrating a RabbitMQ for Pivotal Platform Service Instance to Another Service Instance</b>	<b>186</b>
Migrate a RabbitMQ Service Instance to Another Service Instance	186
Migrate Policies and Parameters	187
Migrate Messages in Queues	188
Migrate Messages Using the RabbitMQ Shovel Plugin	188
Migrate Messages by Using Consumers to Drain the Old Service Instance	189
<b>Creating a Service Instance with Service-Gateway Access</b>	<b>189</b>
Create a Service Instance with Service-Gateway Access	190
<b>RabbitMQ Environment Variables</b>	<b>190</b>
VCAP_SERVICES	191
Changing Enabled Plugins and Protocols	192
<b>Troubleshooting On-Demand Instances</b>	<b>192</b>
Troubleshoot Errors	193
Common Service Errors	193
Techniques for Troubleshooting	193
Parse a Cloud Foundry (CF) Error Message	193
Retrieve Service Instance Information	194
Retrieve RabbitMQ Service Instance Credentials	194
Delete RabbitMQ Service Instances	195
Knowledge Base (Community)	195
File a Support Ticket	195



# RabbitMQ for Pivotal Platform

## RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.



**Note:** Pivotal has renamed *RabbitMQ for Pivotal Cloud Foundry* to *RabbitMQ for Pivotal Platform*.



**Note:** Pivotal has renamed *Pivotal Cloud Foundry* to *Pivotal Platform*.

## About RabbitMQ for Pivotal Platform

RabbitMQ enables app developers to provision and use the RabbitMQ message broker with a single command.

As of v1.8, RabbitMQ supports two types of service, an *on-demand* service and a *pre-provisioned* service.

This table summarizes the main differences between the two:

	VMs it Runs On	How VMs are Created	Metrics Name Prefix
<b>On-Demand Service</b>	Dedicated VM that serves a single service instance. See <a href="#">this topic</a> for details.	Pivotal Platform creates each VM on-demand when app developer creates service instance	<code>p.rabbitmq</code> (with a dot)
<b>Pre-Provisioned Service</b>	Multi-tenant VMs shared by apps across your deployment	Pivotal Platform creates all VMs when operator deploys or updates service	<code>p-rabbitmq</code> (with a dash)



**Note:** For most new apps, Pivotal recommends that you use on-demand services instead of pre-provisioned services. On-Demand isolates workloads by creating a separate VM for each service instance.

## What are On-Demand Instances

In RabbitMQ versions before v1.8.0, the RabbitMQ service instances correspond to a unique RabbitMQ virtual host on the multi-tenant RabbitMQ cluster. RabbitMQ v1.8.0 introduced [On-Demand Broker \(ODB\)](#) support. That means you can create a new single-tenant cluster and dedicate it to a single app.

For more information, see [Unlocking the Power of On-Demand RabbitMQ for Pivotal Platform](#) and [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.

Element	Details
Version	1.18.9
Release date	September 23, 2020
Software component version	RabbitMQ OSS 3.8.8
Compatible Pivotal Operations Manager versions	2.6, 2.7, 2.8 and 2.9
Compatible Pivotal Application Service versions	2.6, 2.7, 2.8 and 2.9
IaaS support	AWS, Azure, GCP, OpenStack, and vSphere
IPsec support	No

## Features

### On-Demand

- Create up to 5 different on-demand RabbitMQ plans which can be provisioned through the Marketplace
- Choose whether a plan has 1, 3, 5, or 7 nodes
- Default resource sizes in plans to guide selection
- More control over which Orgs and Spaces have visibility of each configured plan
- Bind apps to an instance of the plan, providing unique credentials for each binding
- Management UI access to app developers
- Deployment into an availability zone (AZ) specified by the plan

- Enable Transport Layer Security (TLS) for the AMQP protocol
- 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
- Run smoke tests for on-demand plans on plan 1
- Errands are run on co-located VMs to decrease deployment times

For more information, see [Unlocking the Power of On-Demand RabbitMQ for Pivotal Platform](#).

## 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 UI access to operators and app developers
- Deployment across multiple AZs, with nodes striped across the AZs automatically
- Enable SSL (Secure Sockets Layer) for the AMQP, MQTT, STOMP protocols
- HA Proxy 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. For downtime requirements, see [Downtime When Upgrading](#).
- Configure the end point for the RabbitMQ syslog
- RabbitMQ and HA Proxy metrics are exposed on the Firehose
- Syslog forwarding on by default
- Errands are run on co-located VMs to decrease deployment times

## 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 at the top of the page.

## RabbitMQ and Other On-Demand Services

As well as RabbitMQ, other services offer *on-demand* service plans. These plans let developers provision service instances when they want.

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

The following table lists which service tiles offer on-demand and pre-provisioned service plans:

Service tile	Standalone product related to the service	Supports on-demand	Supports pre-provisioned
RabbitMQ for Pivotal Platform	Pivotal RabbitMQ	Yes	Yes. Only recommended for test environments.
Redis for Pivotal Platform	Redis	Yes	Yes (shared-VM plan). Only recommended for test environments.
MySQL for VMware Tanzu	MySQL	Yes	No
VMware Tanzu GemFire	VMware GemFire	Yes	No

For services that offer both on-demand and pre-provisioned plans, you can choose the plan you want to use when configuring the tile.

## Feedback

Please send any issue reports, feature requests, or questions to the [Feedback list](#).

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## RabbitMQ for Pivotal Platform Release Notes



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.



**Note:** Pivotal has renamed *RabbitMQ for Pivotal Cloud Foundry* to *RabbitMQ for Pivotal Platform*.



**Note:** Pivotal has renamed *Pivotal Cloud Foundry* to *Pivotal Platform*.



**Breaking Change:** In RabbitMQ v1.18.2 and later, users are no longer automatically logged in when navigating to the URL for the RabbitMQ Management UI. Instead, users must enter their username and password on the RabbitMQ Management UI login page. This is because usernames and passwords are removed from the RabbitMQ dashboard URL located in service bindings because of a potential XSS vector. For more information, see the pull request [Remove direct login route](#) in GitHub.

## Upgrade to the Latest Version

For RabbitMQ v1.15 and v1.16, you can skip a minor version when you upgrade.

See the following table to find your upgrade path:

If you are on...	Upgrade to...
v1.16	v1.18.4 or later patches
v1.15	v1.17.3 or later patches

If you are upgrading a version other than v1.16 or v1.15, do not skip minor versions.

For product versions and upgrade paths, see [Upgrade Planner](#).

## v1.18.9

**Release Date:** September 23, 2020

## Features

New features and changes in this release:

- **Erlang:** Upgraded to v23.0.3.

## Resolved Issues

This release has the following fix:

- **The on-demand adapter cleanly exits the drain script when the credentials are not present in UAA:** The drain script now has the correct output for success (0) and failure (non-zero).

## Known Issues

This release has the following issue:

- **Changing the Erlang cookie can cause failed deployments:** Changing the Erlang cookie value requires cluster downtime and can result in failed deployments. For more information, see [Changing the Erlang Cookie Value Known Issue](#).

## Compatibility

The following components are compatible with this release:

Component	Version
Erlang	23.0.3
HAProxy	1.8.26
OSS RabbitMQ*	3.8.8
Stemcell	621.x
Tanzu Application Service	2.7.x, 2.8.x, 2.9.x
bpm	1.1.9
cf-cli	1.29.0

cf-rabbitmq	318.0.0
cf-rabbitmq-multitenant-broker	78.0.0
cf-service-gateway	28.0.0
cf-rabbitmq-smoke-tests	64.0.0
loggregator-agent	3.21.5
on-demand-service-broker	0.40.0
rabbitmq-metrics	40.0.0
rabbitmq-on-demand-adapter	145.0.0
routing	0.207.0
service-metrics	1.12.5
syslog	11.6.1

\* For more information, see the [v3.8.8 GitHub documentation](#).

## v1.18.8

**Release Date:** August 5, 2020

## Resolved Issues

This release has the following fixes:

- The pre-stop script no longer fails with a timeout error.** Previously, the pre-stop script failed with the error: `Timed out waiting for mirror queue critical node to sync after 3600 seconds`. To prevent this, the following checks in the pre-stop script are disabled by default:
  - `rabbitmq-diagnostics check_if_node_is_mirror_sync_critical`
  - `rabbitmq-diagnostics check_if_node_is_quorum_critical`

To enable these checks, select the **Wait for Queue Synchronization** checkbox in the configuration for on-demands plans. For more information about this setting, see [Configure the Service Plan](#).
- Downstream federated queues consuming messages from an upstream queue are no longer duplicated after upgrading.** This is because the RabbitMQ cluster name defaults to `rabbit@localhost` instead of its service instance GUID. To set the cluster name to its service instance GUID, select the **Use Service Instance ID as cluster name** checkbox in **Global Settings for On-Demand Plans**. For more information about this setting, see [Configure Global Settings](#).
- Configure a custom apps domain to ensure smoke tests work:** Previously, the smoke test picked up the default Cloud Foundry domain, which could cause smoke tests to fail when the RabbitMQ service instance domain for on-demand services could not access the smoke tests domain. Operators can now configure the smoke tests domain to ensure that the service instance is



accessible during smoke tests.

For more information, see [Configure Global Settings](#).

## Known Issues

This release has the following issues:

- **Changing the Erlang cookie can cause failed deployments:** Changing the Erlang cookie value requires cluster downtime and can result in failed deployments. For more information, see [Changing the Erlang Cookie Value Known Issue](#).
- **The on-demand adapter does not cleanly exit the drain script when the credentials are not present in UAA.** The drain script is expected to output an integer to determine success (0) or failure (non-zero). In some cases, the drain script outputs errors even though the operation has succeeded. This causes a failure in the drain operation. This is fixed in v1.18.9.

## Compatibility

The following components are compatible with this release:

Component	Version
Erlang	22.3.4.3
HAProxy	1.8.25
OSS RabbitMQ*	3.8.5
Stemcell	621.x
Pivotal Application Service	2.7.x, 2.8.x, 2.9.x
bpm	1.1.8
cf-cli	1.27.0
cf-rabbitmq	312.0.0
cf-rabbitmq-multitenant-broker	73.0.0
cf-service-gateway	18.0.0
cf-rabbitmq-smoke-tests	55.0.0
loggregator-agent	3.21.5
on-demand-service-broker	0.40.0
rabbitmq-metrics	34.0.0
rabbitmq-on-demand-adapter	139.0.0
routing	0.203.0
service-metrics	1.12.5
syslog	11.6.1

\* For more information, see the [v3.8.5 GitHub documentation](#).

## v1.18.6

**Release Date:** June 10, 2020

### Resolved Issues

This release has the following fixes:

- Upgrading a replicated tile no longer fails with the error `this migration in a replicated tile caused the error "TypeError: Cannot read property 'value' of undefined"`.
- BOSH deployments no longer fail when the RabbitMQ admin password for the pre-provisioned cluster contains a special character.
- To prevent `rabbitmqctl node_health_check` timing out after 70 seconds, RabbitMQ has less intrusive health checks. For more information, see the [RabbitMQ documentation](#).

### Known Issues

This release has the following issues:

- **Changing the Erlang cookie can cause failed deployments:** Changing the Erlang cookie value requires cluster downtime and can result in failed deployments. For more information, see [Changing the Erlang Cookie Value Known Issue](#).
- **The default setting for mirrored queues can cause the pre-stop script to timeout:** In production, if you use mirrored queues with the default policy setting `ha-sync-mode: manual` and if you have not manually synced your queues, the pre-stop script can fail with the following error: `"Timed out waiting for mirror queue critical node to sync after 3600 seconds"`. To manually sync your queues, run `rabbitmqctl sync_queue`. To set `ha-sync-mode` to `automatic` instead, see [Setting or Changing the Policy](#).
- **After upgrading, the downstream federated queues that consume messages from an upstream queue are duplicated:** Federated queues are duplicated because RabbitMQ recreates them with a new cluster name. This issue occurs the first time you run the `upgrade-all-service-instances` errand after upgrading to RabbitMQ v1.18.5 or later, and does not reoccur if you upgrade to subsequent versions.
- **Smoke tests fail if the RabbitMQ service instance domain cannot access the test domain:** Smoke tests pick up the default Cloud Foundry domain, which can cause them to fail if the RabbitMQ service instance domain for on-demand services cannot access the smoke tests domain.
- **The on-demand adapter does not cleanly exit the drain script when the credentials are not present in UAA.** The drain script is expected to output an integer to determine success (0) or failure (non-zero). In some cases, the drain script outputs errors even though the operation has succeeded. This causes a failure in the drain operation. This is fixed in v1.18.9.

### Compatibility

The following components are compatible with this release:

Component	Version
-----------	---------

Erlang	22.3.4
HAProxy	1.8.25
OSS RabbitMQ*	3.8.3
Stemcell	621.x
Pivotal Platform	2.7.x, 2.8.x, 2.9.x
bpm	1.1.8
cf-cli	1.26.0
cf-rabbitmq	298.0.0
cf-rabbitmq-multitenant-broker	69.0.0
cf-service-gateway	13.0.0
cf-rabbitmq-smoke-tests	48.0.0
loggregator-agent	3.21.5
on-demand-service-broker	0.39.0
rabbitmq-metrics	28.0.0
rabbitmq-on-demand-adapter	121.0.0
routing	0.200.0
service-metrics	1.12.5
syslog	11.6.1

\* For more information, see the [v3.8.3 GitHub documentation](#).

## v1.18.5

**Release Date:** May 15, 2020

### Resolved Issues

This release has the following fixes:

- When used in production, quorum queues no longer experience downtime during a rolling upgrade. This is because RabbitMQ waits for quorum and mirror-sync critical nodes to sync before upgrades are applied.
- On demand instance plan features now display with bullet points in Apps Manager.
- BOSH `update_mode: converge` is now used instead of `feature.converge_variables`. This is because `feature.converge_variables` can result in unintended side effects. It regenerates all variables associated with a deployment, even those that are specified in a runtime config, causing downtime and other unexpected behavior.
- Operators can filter metrics by cluster name. This is because the RabbitMQ cluster name matches its service instance GUID rather than the default value `rabbit@localhost`.

## Known Issues

This release has the following issues:

- **Changing the Erlang cookie can cause failed deployments:** Changing the Erlang cookie value requires cluster downtime and can result in failed deployments. For more information, see [Changing the Erlang Cookie Value Known Issue](#).
- Upgrading a replicated tile fails with the error `this migration in a replicated tile caused the error "TypeError: Cannot read property 'value' of undefined"`
- In production use cases, `rabbitmqctl node_health_check` times out after 70 seconds.
- **The default setting for mirrored queues can cause the pre-stop script to timeout:** In production, if you use mirrored queues with the default policy setting `ha-sync-mode: manual` and if you have not manually synced your queues, the pre-stop script can fail with the following error: `"Timed out waiting for mirror queue critical node to sync after 3600 seconds"`. To manually sync your queues, run `rabbitmqctl sync_queue`. To set `ha-sync-mode` to `automatic` instead, see [Setting or Changing the Policy](#).
- **After upgrading, the downstream federated queues that consume messages from an upstream queue are duplicated:** Federated queues are duplicated because RabbitMQ recreates them with a new cluster name. This issue occurs the first time you run the `upgrade-all-service-instances` errand after upgrading to RabbitMQ v1.18.5 or later, and does not reoccur if you upgrade to subsequent versions.
- **Smoke tests fail if the RabbitMQ service instance domain cannot access the test domain:** Smoke tests pick up the default Cloud Foundry domain, which can cause them to fail if the RabbitMQ service instance domain for on-demand services cannot access the smoke tests domain.
- **The on-demand adapter does not cleanly exit the drain script when the credentials are not present in UAA.** The drain script is expected to output an integer to determine success (0) or failure (non-zero). In some cases, the drain script outputs errors even though the operation has succeeded. This causes a failure in the drain operation. This is fixed in v1.18.9.

## Compatibility

The following components are compatible with this release:

Component	Version
Erlang	22.3.2
HAProxy	1.8.25
OSS RabbitMQ*	3.8.3
Stemcell	621.x
Pivotal Platform	2.7, 2.8 and 2.9
Pivotal Cloud Foundry	2.5 and 2.6
bpm	1.1.8

cf-cli	1.26.0
cf-rabbitmq	291.0.0
cf-rabbitmq-multitenant-broker	65.0.0
cf-service-gateway	10.0.0
cf-rabbitmq-smoke-tests	45.0.0
loggregator-agent	3.21.5
on-demand-service-broker	0.39.0
rabbitmq-metrics	24.0.0
rabbitmq-on-demand-adapter	117.0.0
routing	0.200.0
service-metrics	1.12.5
syslog	11.6.1

\* For more information, see the [v3.8.3](#) GitHub documentation.

## v1.18.4

**Release Date:** March 19, 2020

## Security Fix

This release includes the following security fix:

- High [CVE-2020-9283](#)

## Features

New features and changes in this release:

- You can upgrade RabbitMQ from v1.16 to v1.18.4 and later. Pivotal recommends upgrading from the latest v1.16 release to the latest v1.18 release. You can skip v1.17 in this case to save time.



**Note:** Pivotal discourages other upgrade paths that skip a release, such as v1.14 to v1.16 or v1.15 to v1.18, because they are untested.

- This release updates the following dependencies:
  - ✦ OSS RabbitMQ to v3.8.3. For more information, see the [RabbitMQ v3.8.3](#) GitHub documentation.
  - ✦ Erlang to v22.2.8
  - ✦ HAProxy to v1.8.24

## Resolved Issues

This release has the following fix:

- Post-deploy scripts for the on-demand broker no longer fail because of a timing issue.

## Known Issues

This release has the following issues:

- When used in production, quorum queues might experience downtime during a rolling upgrade.
- **Changing the Erlang cookie can cause failed deployments:** Changing the Erlang cookie value requires cluster downtime and can result in failed deployments. For more information, see [Changing the Erlang Cookie Value Known Issue](#).
- On demand instance plan features do not display with bullet points in Apps Manager.
- BOSH `feature.converge_variables` can cause side effects. For example, BOSH runtime config variables might be regenerated, causing downtime and other unexpected behavior.
- Upgrading a replicated tile fails with the error `this migration in a replicated tile caused the error "TypeError: Cannot read property 'value' of undefined"`
- In production use cases, `rabbitmqctl node_health_check` times out after 70 seconds.
- Operators cannot filter metrics by cluster name because the RabbitMQ cluster name is always the default value `rabbit@localhost`.
- **Smoke tests fail if the RabbitMQ service instance domain cannot access the test domain:** Smoke tests pick up the default Cloud Foundry domain, which can cause them to fail if the RabbitMQ service instance domain for on-demand services cannot access the smoke tests domain.
- **The on-demand adapter does not cleanly exit the drain script when the credentials are not present in UAA.** The drain script is expected to output an integer to determine success (0) or failure (non-zero). In some cases, the drain script outputs errors even though the operation has succeeded. This causes a failure in the drain operation. This is fixed in v1.18.9.

## Compatibility

The following components are compatible with this release:

Component	Version
Stemcell	621.x
Pivotal Platform	2.7, 2.8 and 2.9
Pivotal Cloud Foundry	2.5 and 2.6
OSS RabbitMQ	3.8.3*
Erlang	22.2.8
HAProxy	1.8.24
bpm	1.1.7

service-metrics	1.12.5
on-demand-service-broker	0.38.0
rabbitmq-metrics	23.0.0
rabbitmq-on-demand-adapter	115.0.0
cf-service-gateway	9.0.0
cf-rabbitmq-multitenant-broker	64.0.0
loggregator-agent	3.21.5
routing	0.199.0
syslog	11.6.1
cf-rabbitmq-smoke-tests	44.0.0
cf-rabbitmq	288.0.0

\* For more information, see the [v3.8.3 GitHub documentation](#).

## v1.18.3

**Release Date:** January 10, 2020

## Resolved Issues

This release has the following fixes:

- These upgrade paths, each starting from v1.15.x with TLS set to optional, no longer fail:
  - ◊ v1.15.x to v1.17.0 to v1.18.3
  - ◊ v1.15.x to v1.17.1 to v1.18.3
  - ◊ v1.15.x to v1.17.3 to v1.18.3
- On platform on-demand service instances, created with a service-gateway-enabled plan, no longer fail to connect when using the MQTT and STOMP protocols. For more information, see [Service-Gateway Access \(Beta\)](#) and [RabbitMQ Server Plugins](#).

## Known Issues

This release has the following issues:

- When used in production, quorum queues might experience downtime during a rolling upgrade.
- **Changing the Erlang cookie can cause failed deployments:** Changing the Erlang cookie value requires cluster downtime and can result in failed deployments. For more information, see [Changing the Erlang Cookie Value Known Issue](#).
- In rare circumstances, post-deploy scripts for the on-demand broker fail with the error message `User admin does not have the correct permissions for vhost`. This is due to a timing issue.

- On demand instance plan features do not display with bullet points in Apps Manager.
- BOSH `feature.converge_variables` can cause side effects. For example, BOSH runtime config variables might be regenerated, causing downtime and other unexpected behavior.
- Upgrading a replicated tile fails with the error `this migration in a replicated tile caused the error "TypeError: Cannot read property 'value' of undefined"`
- In production use cases, `rabbitmqctl node_health_check` times out after 70 seconds.
- Operators cannot filter metrics by cluster name because the RabbitMQ cluster name is always the default value `rabbit@localhost`.
- **Smoke tests fail if the RabbitMQ service instance domain cannot access the test domain:** Smoke tests pick up the default Cloud Foundry domain, which can cause them to fail if the RabbitMQ service instance domain for on-demand services cannot access the smoke tests domain.
- **The on-demand adapter does not cleanly exit the drain script when the credentials are not present in UAA.** The drain script is expected to output an integer to determine success (0) or failure (non-zero). In some cases, the drain script outputs errors even though the operation has succeeded. This causes a failure in the drain operation. This is fixed in v1.18.9.

## Compatibility

The following components are compatible with this release:

Component	Version
Stemcell	621.x
Pivotal Platform	2.7 and 2.8
Pivotal Cloud Foundry	2.5 and 2.6
OSS RabbitMQ	3.8.2*
Erlang	22.1.8
HAProxy	1.8.23
bpm	1.1.6
service-metrics	1.12.5
on-demand-service-broker	0.36.0
cf-service-gateway	4.0.0
rabbitmq-metrics	22.0.0
rabbitmq-on-demand-adapter	111.0.0
cf-rabbitmq-multitenant-broker	61.0.0
loggregator-agent	3.21.5
routing	0.196.0
syslog	11.6.1



cf-rabbitmq-smoke-tests	40.0.0
cf-rabbitmq	283.0.0

\* For more information, see the [v3.8.2](#) GitHub documentation.

## v1.18.2

**Release Date:** December 17, 2019

## Features

New features and changes in this release:

- Operators can configure an on-demand plan that enables service instances to connect to components outside the Pivotal Platform foundation through AMQP. For more information, see [Enabling Service-Gateway Access \(Beta\)](#). This functionality is planned for other supported protocols in later releases.
- This release uses RabbitMQ v3.8.2. For more information, see [RabbitMQ 3.8.2](#) in GitHub.
- This version supports quorum queues. For more information about quorum queues, see the [RabbitMQ documentation](#).
- Previously, operators could not set an upper limit on the number of on-demand service instances higher than 200. This restriction no longer exists.

## Known Issues

This release has the following issues:

- When used in production, quorum queues might experience downtime during a rolling upgrade.
- **Changing the Erlang cookie can cause failed deployments:** Changing the Erlang cookie value requires cluster downtime and can result in failed deployments. For more information, see [Changing the Erlang Cookie Value Known Issue](#).
- In rare circumstances, post-deploy scripts for the on-demand broker fail with the error message `User admin does not have the correct permissions for vhost`. This is due to a timing issue.
- On demand instance plan features do not display with bullet points in Apps Manager.
- BOSH `feature.converge_variables` can cause side effects. For example, BOSH runtime config variables might be regenerated, causing downtime and other unexpected behavior.
- Upgrading a replicated tile fails with the error `this migration in a replicated tile caused the error "TypeError: Cannot read property 'value' of undefined"`
- In production use cases, `rabbitmqctl node_health_check` times out after 70 seconds.
- Operators cannot filter metrics by cluster name because the RabbitMQ cluster name is always the default value `rabbit@localhost`.
- **Smoke tests fail if the RabbitMQ service instance domain cannot access the test domain:**

Smoke tests pick up the default Cloud Foundry domain, which can cause them to fail if the RabbitMQ service instance domain for on-demand services cannot access the smoke tests domain.

- **The on-demand adapter does not cleanly exit the drain script when the credentials are not present in UAA.** The drain script is expected to output an integer to determine success (0) or failure (non-zero). In some cases, the drain script outputs errors even though the operation has succeeded. This causes a failure in the drain operation. This is fixed in v1.18.9.

## Compatibility

The following components are compatible with this release:

Component	Version
Stemcell	621.x
Pivotal Platform	2.7 and 2.8
Pivotal Cloud Foundry	2.5 and 2.6
OSS RabbitMQ	3.8.2*
Erlang	22.1.8
HAProxy	1.8.23
bpm	1.1.6
service-metrics	1.12.5
on-demand-service-broker	0.36.0
cf-service-gateway	4.0.0
rabbitmq-metrics	22.0.0
rabbitmq-on-demand-adapter	109.0.0
cf-rabbitmq-multitenant-broker	61.0.0
loggregator-agent	3.21.5
routing	0.196.0
syslog	11.6.1
cf-rabbitmq-smoke-tests	40.0.0
cf-rabbitmq	283.0.0
cf-service-gateway	4.0.0

\* For more information, see the [v3.8.2](#) GitHub documentation.

## View Release Notes for Another Version

To view the release notes for another product version, select the version from the dropdown at the top of this page.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

# Unlocking the Power of On-Demand RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic explains how to benefit from the on-demand service plans.

## Introduction

RabbitMQ for Pivotal Platform responds to the demands of operators by offering a RabbitMQ on-demand cluster for their app developer teams, in addition to the single-node on-demand plan.

The on-demand cluster plan is designed for workloads that require the same resilience requirements as the pre-provisioned offering, but also require their workloads be isolated. The platform operations team can configure a RabbitMQ cluster to meet their business requirements and empower app development teams to self-serve their own RabbitMQ cluster.

RabbitMQ also provides smoke tests for the on-demand plans so that operations teams can validate the app developer workflow for on-demand services. See [Smoke Tests](#).

With the on-demand cluster plan, platform operators can offer their app developers three types of RabbitMQ service plans:

- **On-Demand single node**—For app developer teams requiring greater isolation than provided by the pre-provisioned approach. App development teams can have full access to their own message broker to adapt the runtime parameters to their requirements. For more information about these parameters, see [Parameters and Policies](#) in the RabbitMQ documentation.
- **On-Demand cluster**—For an increased level of message resilience and cluster availability, as well as the benefits of workload isolation mentioned above.
- **Pre-provisioned**—For light to moderate messaging needs, this service is fully operated and managed by platform operators as a service.

For information about the pre-provisioned plan, see [Deploying the RabbitMQ Pre-Provisioned Service](#). For information about using pre-provisioned plans to isolate workloads, see [Creating Isolation with the Tile Replicator](#).

## Deciding Which Service Plan to Use

Pivotal recommends the on-demand service, which is designed for independent isolated RabbitMQ service instances. The existing pre-provisioned offering has many RabbitMQ service instances on a single VM, in a multi-tenancy model. In this multi-tenancy model, a single misbehaving app can take down the entire cluster for everyone.

From research and feedback on the issues customers had when using the pre-provisioned service,

Pivotal is making different design decisions for the on-demand service.

To provide you enough visibility to decide which service to use, the table below describes the current feature discrepancies between the pre-provisioned and on-demand services, and plans for addressing these discrepancies. Please give feedback on how to meet your use case requirements with the on-demand service.

Feature	Pre-Provisioned Service	On-Demand Service
Configuration	Enabled in base-64 encoded text box	Plan to address
Plugins	Tier-1 plugins enabled using checkboxes in UI.	A selection of tier-1 plugins are enabled by default on all instances. Other tier-1 plugins can be optionally enabled by an app developer. See <a href="#">RabbitMQ Server Plugins</a> .
RabbitMQ admin credentials to access RabbitMQ Management UI	Can set password using tile UI	Can access by creating a service key, see <a href="#">Create an Admin User for a Service Instance</a> .
Erlang cookie	Operator can change. This might cause problems. For more information, see <a href="#">Changing the Erlang Cookie Value Known Issue</a> .	This is managed by the service. No operator intervention needed.
RabbitMQ TLS versions	Available due to security concerns about the TLS packaged with the pre-provisioned service	All instances only have TLS v1.1 and TLS v1.2 available.
Resource Sharing	Yes. Service instances share resources on the same VM and can affect one another.	No. On-Demand ensures isolation between service instances by creating a separate VM per service instance.
External load balancer DNS name	Available	<ul style="list-style-type: none"> <li>Plan to address</li> <li>IaaS-specific load balancers can still be used.</li> </ul>
Disk free alarm limit	Configurable	No plans to address. The default persistent disk size is controlled at the plan level and is set relative to memory. This removes the ability to mis-configure the alarm limit.
Load-balancing	Available using HAProxy	Available using BOSH DNS
RabbitMQ servers static IP	Exists	Evaluating based on customer needs and feedback
Policy for new instances	Configurable	Plan to address
Almost-instant provisioning	Instantly provisioned	<ul style="list-style-type: none"> <li>No plans to address.</li> <li>Instant provisioning for on-demand is limited by the IaaS. This limitation might be addressed by containers.</li> </ul>
Individual instance upgrade	Possible when using tile replicator	Plan to address

Network partition behavior	<ul style="list-style-type: none"> <li>• Defaults to <code>pause_minority</code></li> <li>• Configurable</li> </ul>	<ul style="list-style-type: none"> <li>• Defaults to <code>pause_minority</code></li> <li>• Configurable for each plan</li> </ul>
TLS	TLS enabled between client and RabbitMQ broker, possible to configure peer validation	TLS between client and the RabbitMQ broker has been implemented.

## On-Demand Single Node Plan

This plan is designed to be simple to configure, deploy, and use. It gives app developer teams fast access to the power of the leading open source message broker backed by BOSH to meet all but the most demanding high availability app messaging requirements.

This plan can suit high-performance workloads requiring messaging resilience and asynchronous messaging replication. RabbitMQ copies messages to disk for resilience and allows asynchronous messaging replication through the RabbitMQ Federation plugin.

This plan offers:

- Fast access to an isolated instance of RabbitMQ scoped for the app developer teams
- Org and space admin access to the RabbitMQ Management UI so app developer teams can have full control over the node
- Updates and upgrades initiated and controlled by the operator to keep the instance up-to-date with the latest security patches and issue fixes
- Message resilience provided through RabbitMQ exchange, queue Federation, and Shovel plugins.

## On-Demand Cluster Plan

Like the single node plan, this plan is designed to be simple to configure, deploy and use. It gives app developer teams fast access to the power of the leading Open Source message broker backed by BOSH to meet all but the most demanding high availability app messaging requirements.

This plan can suit high performance workloads requiring messaging resilience (copied to disk) and asynchronous messaging replication through the RabbitMQ Federation plugin. With this plan, however, you also scale out RabbitMQ to multiple nodes.

This plan offers:

- Fast access to an isolated, clustered instance of RabbitMQ scoped to the app developer team orgs and spaces
- Admin access to the RabbitMQ Management UI to give app developer teams full control over the cluster
- Updates and upgrades initiated and controlled by the operator to keep the instance up-to-date with the latest security patches and issue fixes.
- Message resilience provided by mirroring queues across RabbitMQ nodes, and the option to use the Federation and Shovel plugins.

## General Principles of the Cluster Plan

The following are some general principles to be aware of when configuring the cluster plan:

### Designed for Consistency

RabbitMQ clustering is not primarily a solution for increased availability. Instead, it is designed for consistency and partition tolerance, as described in the [CAP theorem](#). RabbitMQ clustering provides increased message consistency through queue mirroring. This means that messages accessed in one queue are exactly the same as in another queue. For more information, see [Consistency or Availability Tradeoff](#).

Other options can be used for availability requirements, such as the use of federation between exchanges or queues.

For a detailed description of distributed RabbitMQ brokers, see the [RabbitMQ documentation](#).

### Number of Nodes

Every node in the on-demand cluster maintains a complete database of all metadata, and all changes to the metadata are confirmed by every node in the cluster. Therefore, going beyond seven nodes can have a significant negative impact on performance. For optimum resilience and performance, Pivotal recommends three nodes for most workloads.

### Network Latency

RabbitMQ clusters are only recommended for deployment in low latency networks, which normally means that it is not advisable to deploy these clusters across availability zones (AZs). The stability and performance of the RabbitMQ cluster is heavily influenced by the workload on the nodes, replication choices, and network latency.

For this reason, Pivotal recommends that you deploy RabbitMQ clusters into a single Pivotal Operations Manager AZ. However, where different AZs are in the same data center, with reliable low latency links, spanning AZs can be used.

For cloud IaaS deployments, Pivotal does not recommend that deployments span *regions*. For example, in Amazon Web Services (AWS) terms, deploying a RabbitMQ cluster across AZs within a region should provide high enough network performance to prevent impacting cluster stability. However, deploying across AWS regions is likely to lead to cluster instability. For more information, see the [AWS documentation](#).

## Consistency or Availability Tradeoff

In a distributed messaging system, a tradeoff must be made between availability or consistency when a network partition event occurs and one or more nodes are not able to communicate with each other. The cluster plan lets operators decide how they want the RabbitMQ cluster to react in the event of a network partition.

Pivotal recommends keeping the default cluster partition option of `pause_minority` because this satisfies most use cases. Choosing the `pause_minority` partition-handling strategy favors message consistency over availability. For more information about the options for handling partitions, see the [RabbitMQ documentation](#). For a detailed description of the options available in RabbitMQ, see

## Clustering and Network Partitions.

Here is an example of how `pause_minority` works. If you create a RabbitMQ cluster with three nodes and one node becomes unable to communicate with the other two, this node is in the minority. The node that is in the minority is paused, and the other two nodes continue serving traffic. If each of the nodes loses connectivity with the other two, then the entire cluster is paused to preserve data as no majority can be established. The cluster *heals* when two or more nodes are able to communicate with each other.

## RabbitMQ Queue Availability

It is important to be aware that message queue availability is different from cluster availability. So, having cluster availability does not mean that all of the messages within the queues are also available.

By default, queues within a RabbitMQ cluster are located on a single node—the node on which they were first declared. However, queues can be configured to mirror across multiple nodes, so that any message published to the queue is replicated to all mirrors. Enabling mirroring can have a negative impact on queue performance because messages must be copied to all mirrors before being acknowledged.

Each mirrored queue consists of one leader replica and one or more mirrors, with the oldest mirror being promoted to the new leader replica if the old leader replica disappears for any reason. Consumers are connected to the leader replica regardless of which node they connect to, and mirrors drop messages that have been acknowledged at the leader replica. Queue mirroring enhances queue availability, but does not distribute load across nodes because each of the participating nodes must still do all the work.

App developers must decide if they want to use queue mirroring and determine the policy they want to apply to their queues. These choices have significant impact on the availability of their queues. For more information, see the [RabbitMQ documentation](#).

Unlike the pre-provisioned plan, the cluster plan does not ship with a default load balancer. Therefore, developers must configure their app to use the array of hosts provided in `VCAP_SERVICES`. If developers enable queue mirroring, they must also ensure their apps have re-try logic and reconnection logic that iterates over the range of hosts provided. Most common RabbitMQ clients have this logic built into them. For more information, see the [Spring Advanced Message Queuing Protocol \(Spring AMQP\) documentation](#).

Because the cluster plan is designed to enable app developer teams to self-serve, not having a load balancer in front of the RabbitMQ cluster has these benefits:

- Manage resources better, as fewer VMs are needed.
- Help with troubleshooting. Client IP is now the IP of the source container and not the HAProxy.
- Reduce the number of hops between apps and broker. This helps with latency.
- Determine queue placement. This makes sense for larger scale deployments.
- Empower app developer teams to manage their cluster in the best way for their app.
- Require re-try logic in an app if it needs HA access to a queue. Thus, all nodes can route to a queue if it is available.

## Managing On-Demand Resources Through Plans

In configuring each plan, there are a number of operational controls that platform operations teams can use to manage the resources consumed by on-demand RabbitMQ:

- **Control Access**—Operators can choose the app development orgs and spaces for which the plans are available and visible. Each plan can be enabled or disabled, and service access and visibility can either be global, or enabled per org and space through the command line.

For example, you might decide to enable the single node on-demand plan across all app developer teams to meet their demand to isolate their workload. You might then choose to offer the on-demand cluster plan only to a subset of app developer teams who require the extra resources.

- **Set Quotas**—You can set a global quota for all on-demand instances that takes precedence over each plan quota. This lets you guard against the risk of over-committing resources, but allows the flexibility of over-committing each plan, so you can meet the fluctuating demands of your app developers.
- **Control Resource Consumption**—Each plan offers more fine-grained control over individual plan resource consumption. At the highest level, you can use the plan quota to control the number of instances that can be deployed within a foundation. For each plan, you can also configure the number of nodes that constitute a cluster (3, 5, or 7), the instance type, and persistent disk storage size to best suit your requirements.
- **Monitor**—You can monitor the number of instances that have been deployed against the quota you have set so that you can plan future resource requirements.

## Customizing Plan Options

The RabbitMQ on-demand plans expose a number of configuration options. In most cases, the default configurations meet most app demands. However, it is important for an operations team to consider the options to ensure that they provide the best service to their app developers. This section explains these options.

### Configuration Options

#### Single Node and Cluster Plans

- Enable/ Disable plan
- Determine which orgs and spaces can see and access the plan
- Set Service Instance Quota
- Select AZ placement (where applicable)
- Set RabbitMQ service instance size (CPU and Memory)
- Set persistent disk size (Persisted Message Store) for the RabbitMQ service instance. Ensure the size of the persistent disk is at least twice as large as the instance memory.

#### Cluster Plan Only



- Set number of nodes to 3, 5, or 7
- Determine network partition behavior. See [Consistency or Availability Tradeoff](#) above.



**Note:** A load balancer, such as HAProxy, is not deployed with on-demand cluster plans.

## Things That Are Preconfigured

The following are preconfigured for both the single node and the cluster plans:

- **RabbitMQ VM Type**—When installing using Ops Manager, each RabbitMQ node is configured to have the following properties:
  - ✦ CPUs: 2
  - ✦ RAM: 8 GB
  - ✦ Ephemeral disk: 16 GB

You can change these settings in the [Service Plan Configuration](#) page. Changing these settings affects all nodes.

- **Persistent Disk Type**—When installing using Ops Manager, each RabbitMQ node is configured to have 30 GB of persistent disk space.

You can change this setting in the [Service Plan Configuration](#) page. Pivotal recommends you set this value to be twice the amount of RAM of the selected **RabbitMQ VM Type**.

- **Metrics**—Emitted to the Loggregator Firehose for all on-demand instances. The polling interval is set in Ops Manager, in the **Metrics polling interval** field, in the **Pre-Provisioned RabbitMQ** tab of the RabbitMQ tile. Due to the impact of some of the cluster settings detailed below, Pivotal strongly recommends that you monitor the exposed metrics and configure alarms as recommended in [Monitoring and KPIs for On-Demand RabbitMQ for Pivotal Platform](#). See also [Monitoring On-Demand RabbitMQ Clusters](#) below.
- **Logs**—On-demand RabbitMQ service instance logs are forwarded using the same configuration as contained in the **Syslog and Metrics** tab of the RabbitMQ tile.
- **Disk free space limit**—The disk free space limit is set to 150% of RAM of the instance type you select. For example, if you select an instance type with 10 GB of RAM, the disk free space limit is set to 15 GB. A cluster-wide alarm is triggered if the amount of free disk space drops below this, and all publishers are blocked. Instances must be configured to have persistent disks that are at least twice the size of instance RAM. For more information, see the [RabbitMQ documentation](#).
- **Memory threshold for triggering flow control**—Threshold at which flow control is triggered is set to 40% of the instance RAM. This means that when the alarm is triggered, all connections publishing messages are blocked cluster-wide until the alarm is cleared.

For example, if you select an instance type with 10 GB of RAM, when more than 4 GB of memory is used, all publishing connections are blocked. For more information, see [Memory Alarms](#) in the RabbitMQ documentation.

- **Memory paging threshold**—This is the level at which RabbitMQ tries to free up memory by instructing queues to page their contents out to disk. This is done to try to avoid reaching the high watermark and blocking publishers. This threshold is set to 50% of the configured high watermark, which is 20% of configured memory.

For example, if you select an instance type with 10 GB of RAM, when more than 2 GB of memory is used, all queues start writing all queue contents to disk. For more information, see the [RabbitMQ documentation](#).

## Monitoring On-Demand RabbitMQ Clusters

- It is important to monitor and compare the number of instances that have been deployed against the quota you set using the metric exposed on the Firehose.
- Each instance is pre-configured to emit metrics to the Firehose and can be identified by the `deployment` tag, which has the service instance ID. It is important to monitor these metrics as recommended in [Monitoring and KPIs for On-Demand RabbitMQ for Pivotal Platform](#).

## About Migrating a Pre-Provisioned Instance to an On-Demand Instance

Pivotal recommends the on-demand service for production workloads due to its workload isolation.

For instructions for developers about migrating from a pre-provisioned to an on-demand instance, see [Migrating a RabbitMQ Service Instance to Another Service Instance](#). For how operators can turn off the pre-provisioned service, see [Turning Off the Pre-Provisioned Service](#).

When migrating from a pre-provisioned to an on-demand offering, be aware of the following:

- Pre-provisioned service instances have very low resource consumption, that is, a virtual host within an existing cluster. However, every on-demand service consists of dedicated VMs. Therefore, you must select an on-demand service plan that provides adequate resources, but avoids unnecessary resource consumption.
  - ✦ For example: you might select a single-node plan with a small VM for development purposes, but select a three-node cluster of large VMs for a mission critical system.
- Pivotal recommends that you define all required structures in your app to ensure they get defined if you connect to a new instance. These structures include:
  - ✦ Exchanges
  - ✦ Queues
  - ✦ Bindings
- If your pre-provisioned instance uses any of the following, ensure that you apply them to the on-demand instance:
  - ✦ Policies
  - ✦ virtual host-specific parameters, such as `max_connections`

You can apply them using the RabbitMQ Management UI or using APIs.

- You lose messages that have not been consumed when you delete your old service

instance. If you do not want to lose messages, do one of the following:

- ✦ Switch your producers to the new instance but keep the consumers bound to the old instance until the queues are empty.
- ✦ Use shovel or federation plugins to consume messages from the old instance.

## Differences between Pre-Provisioned and On-Demand Services Instances

There are some differences between pre-provisioned and on-demand services instances that you should be aware of:

- On-Demand instances are not fronted by a load balancer, therefore, ensure the following:
  - ✦ That you configure the RabbitMQ client in your app with all available nodes, not just one.
  - ✦ That your re-connection logic can handle a node failure. Pivotal recommends this behavior for Spring AMQP clients.
- The instance you are migrating to might use a different version of RabbitMQ than your old instance. For more information, see the [RabbitMQ for Pivotal Platform Release Notes](#) and the [RabbitMQ Changelog](#).
- Critical tier-1 plugins are enabled for on-demand. However, on-demand does not yet have the same plugins enabled as pre-provisioned. If you are missing a plugin, contact your Pivotal representative.
- You might have configured your on-demand instance differently. For example, you might have changed:
  - ✦ VM sizing—CPU, RAM, and disk
  - ✦ RabbitMQ network partition behavior, both offerings default to `pause_minority`
- If your RabbitMQ service instance uses TLS, ensure that you enable TLS for on-demand instances. See [Enable TLS for Your Service Instance](#).

[Create a pull request or raise an issue on the source for this page in GitHub](#)

# Architecture

## On-Demand Service Architecture



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

- [Service Network Requirement](#)
- [Default Network and Service Network](#)
- [Required Networking Rules for On-Demand Services](#)

This topic describes the architecture for on-demand RabbitMQ for Pivotal Platform.

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

## Service Network Requirement

When you deploy Pivotal Application Service (PAS), you must create a statically defined network to host the component VMs that make up the infrastructure. Components, such as Cloud Controller and UAA, run on this infrastructure network.

On-demand services might require you to host them on a separate network from the default network. You can also deploy on-demand services on a separate service networks to meet your own security requirements.

PAS supports dynamic networking. Operators can use dynamic networking with asynchronous service provisioning to define dynamically-provisioned service networks. For more information, see [Default Network and Service Network](#) below.

On-demand services are enabled by default on all networks. Operators can optionally create separate networks to host services in BOSH Director. Operators can select which network hosts on-demand service instances when they configure the tile for that service.

## Default Network and Service Network

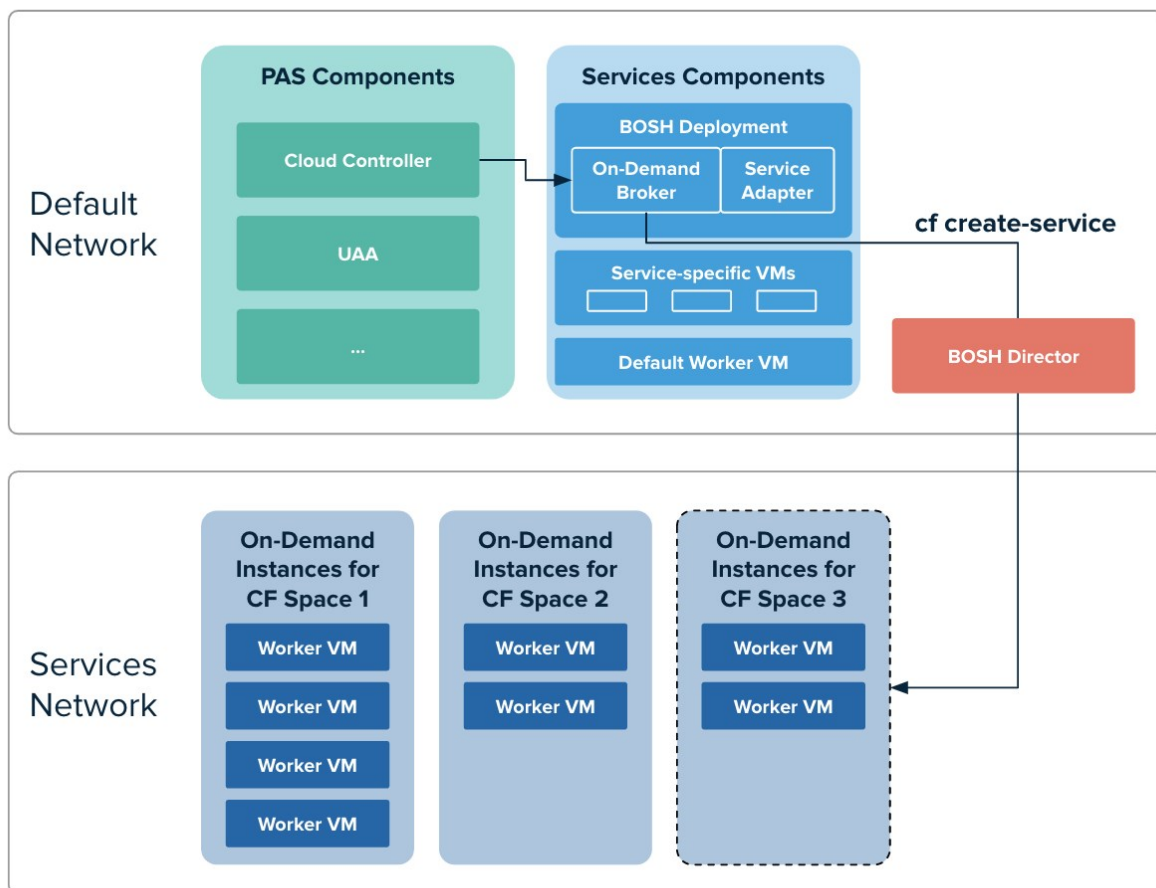
On-demand RabbitMQ services use BOSH to dynamically deploy VMs and create single-tenant service instances in a dedicated network. On-demand services use the dynamically-provisioned service network to host single-tenant worker VMs. These worker VMs run as service instances within development spaces.

This on-demand architecture has the following advantages:

- Developers can provision IaaS resources for their services instances when the instances are created. This removes the need for operators to pre-provision a fixed amount of IaaS resources when they deploy the service broker.
- Service instances run on a dedicated VM and do not share VMs with unrelated processes. This removes the “noisy neighbor” problem, where an app monopolizes resources on a shared cluster.
- Single-tenant services can support regulatory compliances where sensitive data must be separated across different machines.

An on-demand service separates operations between the default network and the service network. Shared service components, such as executive controllers and databases, Cloud Controller, UAA, and other on-demand components, run on the default network. Worker pools deployed to specific spaces run on the service network.

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



[View a larger version of this image](#)

## Required Networking Rules for On-Demand Services

Before deploying a service tile that uses the on-demand service broker (ODB), you must create

networking rules to enable components to communicate with ODB. For instructions for creating networking rules, see the documentation for your IaaS.

The following table lists key components and their responsibilities in the on-demand architecture.

Key Components	Component Responsibilities
<b>BOSH Director</b>	Creates and updates service instances as instructed by ODB.
<b>BOSH Agent</b>	Adds an agent on every VM that it deploys. The agent listens for instructions from the BOSH Director and executes those instructions. The agent receives job specifications from the BOSH Director and uses them to assign a role or job to the VM.
<b>BOSH UAA</b>	Issues OAuth2 tokens for clients to use when they act on behalf of BOSH users.
<b>Pivotal Application Service</b>	Contains the apps that consume services.
<b>ODB</b>	Instructs BOSH to create and update services. Connects to services to create bindings.
<b>Deployed service instance</b>	Runs the given service. For example, a deployed RabbitMQ service instance runs the RabbitMQ service.

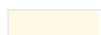
Regardless of the specific network layout, the operator must ensure network rules are set up so that connections are open as described in the table below.

Source Component	Destination Component	Default TCP Port	Notes
<b>ODB</b>	<b>BOSH Director</b>	25555 (BOSH Director)	The default ports are not configurable.
	<b>BOSH UAA</b>	8443 (UAA)	
		8844 (CredHub)	
<b>ODB</b>	<b>Deployed service instances</b>	15672 (RabbitMQ Management UI)	This connection is for administrative tasks.
		15671 (RabbitMQ Management UI over TLS)	Avoid opening general use, app-specific ports for this connection.
<b>ODB</b>	<b>Pivotal Application Service (PAS)</b>	8443 (UAA)	The default port is not configurable.

<b>Errand VMs</b>	<b>PAS</b>	8443 (UAA)	The default port is not configurable.
	<b>ODB</b>	8080	
	<b>Deployed Service Instances</b>	15672 (RabbitMQ Management UI)	
		15671 (RabbitMQ Management UI over TLS)	
		5672 (AMQP)	
		5671 (AMQPS)	
<b>BOSH Agent</b>	<b>BOSH Director</b>	4222	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.
			The communication between these components is two-way.
<b>Deployed apps on PAS</b>	<b>Deployed service instances</b>	15672 (RabbitMQ Management UI)	This connection is for general use, app-specific tasks.
		15671 (RabbitMQ Management UI over TLS)	Avoid opening administrative ports for this connection.
		5672 (AMQP)	
		5671 (AMQPS)	
		61613 (STOMP)	
		61614 (STOMPS)	
		1883 (MQTT)	
		8883 (MQTTS)	
<b>PAS</b>	<b>ODB</b>	8080	This port may be different for individual services.
			This port may also be configurable by the operator if allowed by the tile developer.
<b>Deployed apps on PAS</b>	<b>Runtime CredHub</b>	8844 (CredHub)	This port is needed if secure service instance credentials are enabled.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Deploying the RabbitMQ Pre-Provisioned Service





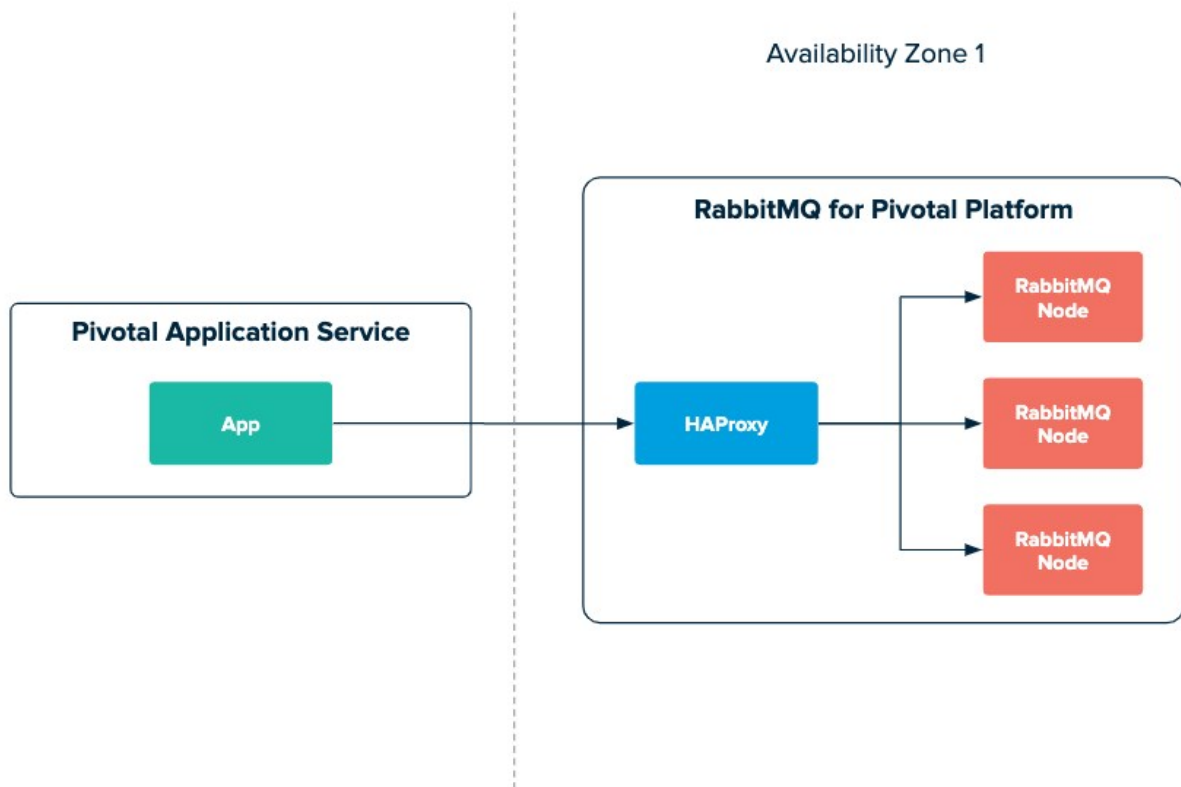
**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

## Default Deployment

Deploying RabbitMQ for Pivotal Platform through Pivotal Operations Manager deploys 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 occurs in a single availability zone (AZ).

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

The diagram below shows the default RabbitMQ pre-provisioned deployment.



## Considerations for the Default Deployment

- Provides high availability for the RabbitMQ cluster
- Queues must be configured to be high availability as they are placed on one node by default
- Customers should decide which partition behavior 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.

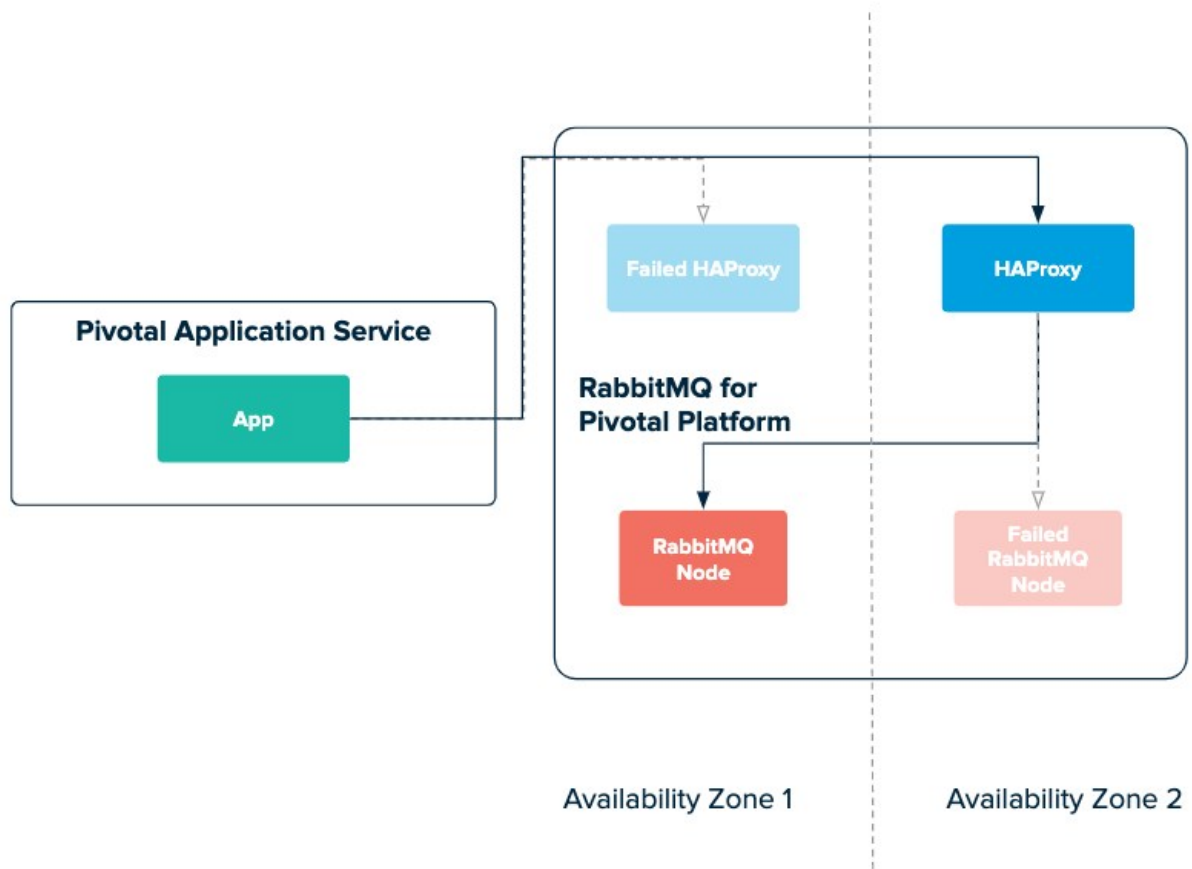
## Recommended Deployment

Pivotal recommends that RabbitMQ is deployed across at least two AZs. Scale RabbitMQ server nodes to an odd number that is greater than or equal to three.

Only use replication of queues 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.

The diagram below shows the recommended RabbitMQ pre-provisioned deployment. It shows that when you use this configuration, if a single HAProxy and single RabbitMQ node fail, your cluster can remain online and apps remain connected.



## Upgrading to the Recommended 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 after the tile has been deployed for the first time. This is to protect against data loss when moving jobs between AZs.

## Upgrading to the Recommended 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 by deploying an additional HAProxy instance through Ops Manager. New or re-bound apps to the RabbitMQ service instance see the IPs of both HAProxies immediately. Existing bound apps will continue to work, but only using the previously deployed HAProxy IP Address. They can be re-bound as required at your discretion.

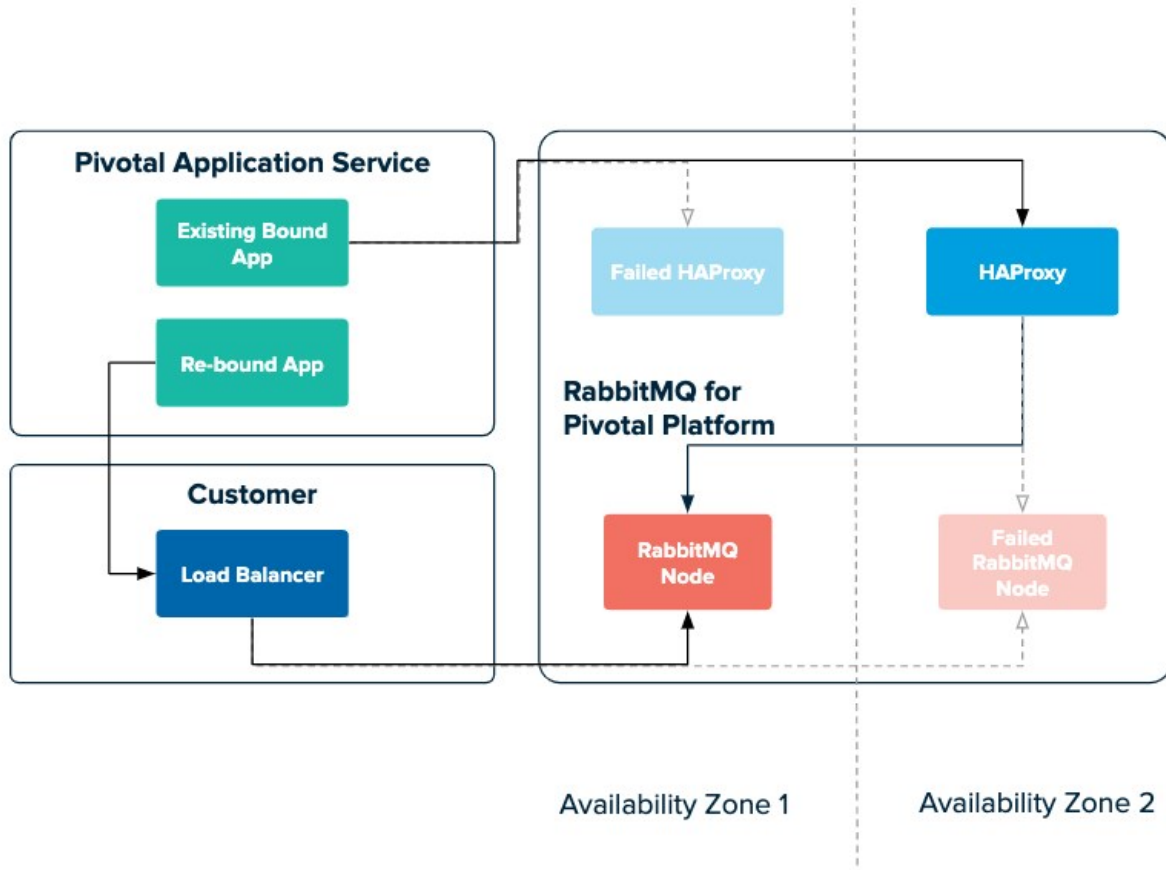
## Considerations for the Recommended Deployment

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

## Advanced Deployment

This deployment builds upon the above recommended deployment and so follows the same upgrade paths. This allows you to replace the use of HAProxy with your own external load balancer. You might choose to do this to remove any knowledge of the topology of the RabbitMQ setup from app developers.

The diagram below shows an advanced RabbitMQ pre-provisioned deployment.



#### Advantages:

- App developers do not need to handle multiple IPs for the HAProxy jobs in their apps

#### Disadvantages:

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

## Upgrading to the Advanced from the Recommended Deployment

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

This can be achieved without downtime to your apps. Follow these steps to do so:

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. Any new RabbitMQ service instances **or** any re-bound connections will use the DNS name or IP address of the ELB in their `VCAP_SERVICES`. Any existing service instances continue to use the HAProxy IP addresses in their `VCAP_SERVICES`.
4. Phase the re-binding of existing apps to update their environment variables.
5. After all apps are updated, reduce the instance count of the `HAProxy` job in Ops Manager to

0.

6. Deploy the changes.

This approach works as any existing bound apps have their `VCAP_SERVICES` information cached in Cloud Controller and are only updated by a re-bind request.

## Downgrading from the Advanced Deployment to the Recommended Deployment

If you are currently using an external load balancer, then you can move back to using HAProxies 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	CP U	Ra m	Ephemer al	Persiste nt	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
RabbitMQ	Recreate All Service Instances	1	1	1024	2048	0	0	1

**Notes:**

- The number of [RabbitMQ Node](#) can be increased if required.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Service-Gateway Access (Beta)



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

Service-gateway access enables a RabbitMQ for Pivotal Platform on-demand service instance to connect to external components that are not on the same foundation as the service instance. These components could be on another foundation or hosted outside of the foundation.

For related procedures, see:

- [Enabling Service-Gateway Access](#)
- [Create a Service Instance with Service-Gateway Access](#)



**Note:** Service-Gateway access is marked as beta because it is only compatible with Advanced Message Queuing Protocol (AMQP). Compatibility with other RabbitMQ protocols is planned. To give feedback on Service-Gateway access, email [pvtl-pcf-rabbitmq@vmware.com](mailto:pvtl-pcf-rabbitmq@vmware.com).

## Overview

There are multiple use cases for service-gateway access. For example:

- Accessing RabbitMQ from apps deployed to Pivotal Application Service (PAS) in a different foundation
- Replicating messages between RabbitMQ clusters in different foundations, for example, to set up federation for disaster recovery
- Enabling apps running outside the foundation to communicate, through RabbitMQ, with PAS-deployed apps
- Using RabbitMQ as a service for apps that are not deployed to PAS

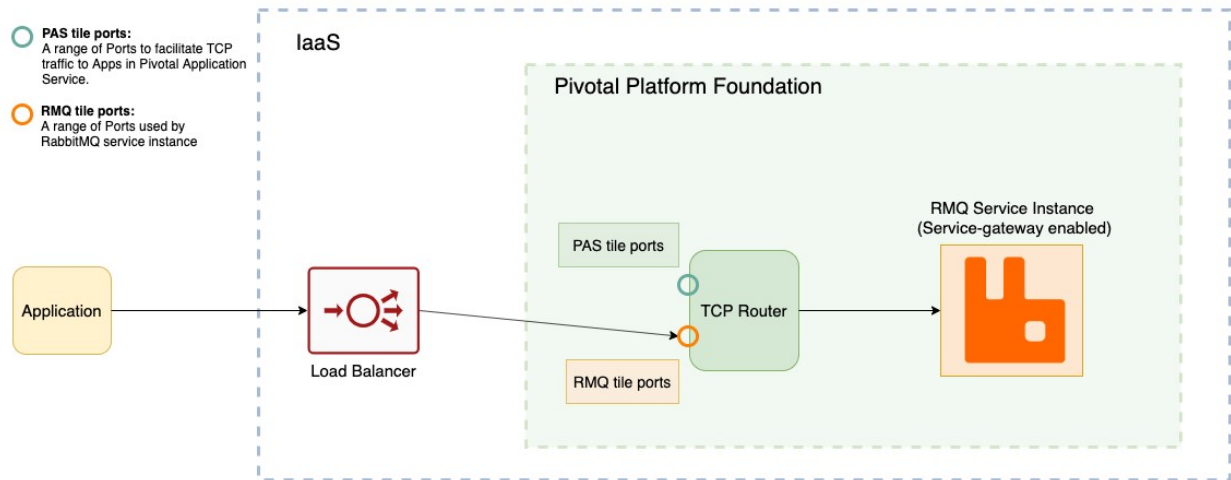
## Architecture

Service-gateway access to RabbitMQ instances leverages the TCP router in PAS.

Any RabbitMQ requests that an app makes are forwarded through DNS to a load balancer that can route traffic from outside to inside the foundation. This load balancer opens a range of ports that are reserved for RabbitMQ traffic. When an app developer creates a service instance on a plan with service-gateway access enabled, a port from the range is provisioned for that service instance. The load balancer then forwards the requests for this RabbitMQ service instance to the TCP router. The

TCP router internally load balances between the RabbitMQ service instance nodes.

The diagram below shows how the traffic is routed from apps to the RabbitMQ nodes in this case. All apps using this RabbitMQ service instance follow the same route, irrespective of whether they are hosted on the foundation or are hosted outside of the foundation.



[Click here to view a larger version of this image.](#)

[Create a pull request or raise an issue on the source for this page in GitHub](#)

# Operator Guide: On- Demand

## Installing and Configuring

### Preparing for TLS



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

- [Overview](#)
- [Workflow](#)
- [Find the CredHub Credentials in Ops Manager](#)
- [Add the CA Certificate](#)
- [Enable TLS in RabbitMQ](#)

#### Page last updated:

This topic provides an overview of how to prepare for using Transport Layer Security (TLS) with RabbitMQ for Pivotal Platform to secure communication between apps and service instances.

This topic explains how to provide an existing CA certificate to BOSH CredHub and generate a new CA certificate with BOSH CredHub.



**Warning:** This procedure involves restarting all of the VMs in your deployment to apply a CA certificate. The operation can take a long time to complete.



**Note:** This certificate is shared by multiple tiles. If you have already done this procedure, you do not need to repeat it.

However, an operator must rotate the this certificate if it expires or if it becomes compromised. For instructions about how to rotate your CA certificate follow the steps in [Rotating CA Certificates](#).

## Overview

When you use TLS, you are provisioned a RabbitMQ server with a certificate. With this certificate, apps and clients can establish an encrypted connection with the service.

CredHub generates a server certificate using a Certificate Authority (CA) certificate. You can either provide the CA or generate it using CredHub. For an overview of the purpose and functionality of the CredHub component, see [CredHub](#).

Apps and clients use this CA certificate to check that the server certificate is trustworthy. A trustworthy server certificate allows apps and clients to securely communicate with the RabbitMQ server.

Pivotal Application Service (PAS) shares the CA certificate public component in the following ways:

- PAS provisions a copy of the CA certificate in the trusted store of each container's operating system. Apps written in Java and Spring automatically discover the CA certificate in the trusted store.
- PAS supplies the public CA certificate in an environment variable called `VCAP_SERVICES` that exists in every container. Apps not written in Java and Spring can retrieve the public component of the CA certificate from `VCAP_SERVICES` and use it to establish an encrypted connection with the data service.

## Prerequisite

Before you do the procedures in this topic, you must install the Credhub CLI. For instructions, see [credhub-cli](#) on GitHub.

## Workflow

The following workflow describes enabling TLS for RabbitMQ for Pivotal Platform:

1. An operator provides a CA certificate to CredHub by performing the procedures in [Find the CredHub Credentials in Ops Manager](#) and in [Add the CA Certificate](#) below.
2. An operator enables TLS in the tile configuration while installing RabbitMQ. See [Enable TLS in RabbitMQ](#) below.
3. A developer enables TLS for an existing service instance. See [Enable TLS for Your Service Instance](#)
4. A developer modifies their app to communicate securely with the RabbitMQ server:
  - **For Java or Spring apps:** See [Activate TLS for Java and Spring Apps](#).
  - **For all other apps:** See [Modifying Apps for TLS](#)

## Find the CredHub Credentials in Ops Manager

To find the BOSH CredHub client name and client secret:

1. In the Ops Manager Installation Dashboard, click the BOSH Director tile.
2. Click the **Credentials** tab.
3. In the BOSH Director section, click the link to the **BOSH Commandline Credentials**.



JOB	NAME	CREDENTIALS
BOSH Director	Vm Credentials	<a href="#">Link to Credential</a>
	Agent Credentials	<a href="#">Link to Credential</a>
	Registry Credentials	<a href="#">Link to Credential</a>
	Director Credentials	<a href="#">Link to Credential</a>
	Nats Credentials	<a href="#">Link to Credential</a>
	Postgres Credentials	<a href="#">Link to Credential</a>
	Blobstore Credentials	<a href="#">Link to Credential</a>
	Health Monitor Credentials	<a href="#">Link to Credential</a>
	Uaa Admin User Credentials	<a href="#">Link to Credential</a>
	Uaa Login Client Credentials	<a href="#">Link to Credential</a>
	Uaa Jwt Key	<a href="#">Link to Credential</a>
	Bbr Ssh Credentials	<a href="#">Link to Credential</a>
	Uaa Bbr Client Credentials	<a href="#">Link to Credential</a>
	Bosh Commandline Credentials	<a href="#">Link to Credential</a>
	Nats Client Ca	<a href="#">Link to Credential</a>
	Nats Server Certificate	<a href="#">Link to Credential</a>
	Nats Director Client Certificate	<a href="#">Link to Credential</a>
	Nats Health Monitor Client Certificate	<a href="#">Link to Credential</a>
	Blobstore Certificate	<a href="#">Link to Credential</a>

- Record the values for `BOSH_CLIENT` and `BOSH_CLIENT_SECRET`.

Here is an example of the credentials page:

```
{ "credential": "BOSH_CLIENT=ops_manager
BOSH_CLIENT_SECRET=abCdE1FgHIjKl2m3n-3Pqrst4EUvwXy5
BOSH_CA_CERT=/var/tempest/workspaces/default/root_ca_certificate
BOSH_ENVIRONMENT=10.0.0.5 bosh " }
```

The `BOSH_CLIENT` is the BOSH CredHub client name and the `BOSH_CLIENT_SECRET` is the BOSH CredHub client secret.

## Add the CA Certificate

To generate and add the CA Certificate to Ops Manager:

- Record the information needed to log in to the BOSH Director VM by following the procedure in [Gather Credential and IP Address Information](#).
- Log in to the Ops Manager VM by following the procedure in [Log in to the Ops Manager VM with SSH](#).
- Set the API target of the CredHub CLI as your CredHub server by running:

```
credhub api \
```

```
https://BOSH-DIRECTOR-IP:8844 \
--ca-cert=/var/tempest/workspaces/default/root_ca_certificate
```

Where **BOSH-DIRECTOR-IP** is the IP address of the BOSH Director VM. For example:

```
$ credhub api \
https://10.0.0.5:8844 \
--ca-cert=/var/tempest/workspaces/default/root_ca_certificate
```

4. Log in to CredHub by running:

```
credhub login \
--client-name=CREDHUB-CLIENT-NAME \
--client-secret=CREDHUB-CLIENT-SECRET
```

Where

- ✦ **CREDHUB-CLIENT-NAME** is the value you recorded for **BOSH\_CLIENT** in [Find the CredHub Credentials in Ops Manager](#) above.
- ✦ **CREDHUB-CLIENT-SECRET** is the value you recorded for **BOSH\_CLIENT\_SECRET** in [Find the CredHub Credentials in Ops Manager](#) above.

For example:

```
$ credhub login \
--client-name=credhub \
--client-secret=abcdefghijklm123456789
```

5. If you are using Ops Manager v2.6 or later, skip this step and the next step. Otherwise, find out if a services CA certificate already is present by running the following command:

```
credhub get --name="/services/tls_ca"
```

If you already have a certificate at the **services/tls\_ca** path, skip the next step.

6. Use the CredHub CLI to generate a CA certificate or provide an existing one.



**Note:** Your deployment can have multiple CA certificates. VMware recommends a dedicated CA certificate for services.

- ✦ **If you do not have a CA certificate:** Use the CredHub CLI to generate one by running:

```
$ credhub generate \
--name="/services/tls_ca" \
--type="certificate" \
--is-ca \
--common-name="rootCA"
```



**Note:** When you run the above command, the generated certificate is valid for one year by default. You can use the **--duration** flag to set the validity period to a specific time. For example, if you add **--**

`duration=1825` to the above command, the generated certificate is valid for five years.

- ❖ **If you have an existing CA certificate that you want to use:** Create a new file called `root.pem` with the contents of the certificate. Then run the following command, specifying the path to `root.pem` and the private key for the certificate:

```
$ credhub set \
  --name="/services/tls_ca" \
  --type="certificate" \
  --certificate=./root.pem \
  --private=ERKSOSMFF...
```

- Record the `certificate` portion from the CA certificate by running:

```
credhub get \
  --name=/services/tls_ca \
  -k ca
```

- Navigate to the Ops Manager **Installation Dashboard** > **BOSH Director** > **Security**.
- Append the contents of the CA certificate you recorded in the previous step into **Trusted Certificates**.
- Click **Save**.

## Enable TLS in RabbitMQ

To enable TLS in the RabbitMQ tile:

- Enable TLS by doing one of the following:
  - ❖ **If you are configuring TLS for an existing installation:** Follow the procedure in [Configure Security](#).
  - ❖ **If you are configuring TLS for a new installation:** Follow the procedures in [Installing and Configuring RabbitMQ](#), including enabling TLS in the [Configure Security](#) section.
- Navigate to **Ops Manager Installation Dashboard** > **Review Pending Changes**.
- Ensure that the CA certificate is deployed to all VMs by selecting:
  - ❖ Pivotal Application Service
  - ❖ RabbitMQ for Pivotal Platform
  - ❖ The **Upgrade All On-Demand Service Instances** errand
- Click **Apply Changes**. This restarts all the VMs in your deployment and applies your CA certificate.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Installing and Configuring the On-Demand Service



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

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

The RabbitMQ open source product provides additional documentation. For more information about getting started with RabbitMQ and ensuring production readiness, see the Production Checklist in the [RabbitMQ Documentation](#).



#### Notes:

- For how to install and configure the RabbitMQ pre-provisioned service, see [Installing and Configuring the Pre-Provisioned Service](#).
- For how to turn off the pre-provisioned service, see [Turning Off the Pre-Provisioned Service](#).

## Role-Based Access in Pivotal Operations Manager

Ops Manager administrators can use Role-Based Access Control (RBAC) to manage which operators can make deployment changes, view credentials, and manage user roles in Ops Manager. Therefore, your role permissions might not permit you to perform every procedure in this operator guide.

For more information about roles in Ops Manager, see [Understand Roles in Ops Manager](#).

## Prerequisites for Deploying the On-Demand Service

Before deploying RabbitMQ as an on-demand service, ensure the following:

- **Networking:** 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.
- **Transport Layer Security (TLS):** If you want to use TLS to secure communication between apps and RabbitMQ service instances, you must complete the procedures in [Provide or Generate a CA Certificate](#) before installing and configuring the tile.

## Download and Install RabbitMQ

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. This adds the tile to your staging area.

4. Click the newly added **RabbitMQ** tile. This lets you begin configuring the tile. The installation is complete when you apply the changes from the configuration.

## Configure On-Demand RabbitMQ

The configuration screen below appears when you click the RabbitMQ tile in Ops Manager. An orange circle indicates that you are required to configure fields on the tab before you can deploy the tile. A green checkmark indicates that the tab is preconfigured and you may optionally change its settings.

### AZ and Network Assignments

Place singleton jobs in

☒ us-central1-f

☐ us-central1-a

☐ us-central1-c

Balance other jobs in

☒ us-central1-f

☒ us-central1-a

☒ us-central1-c

Network

steelteal-pas-subnet

Service Network


steelteal-services-subnet

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 [Apply Changes from Your Configuration](#) below.

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

Field	Instructions
<b>Place singleton jobs in</b>	Select the region that you want for singleton VMs. Ops Manager creates the RabbitMQ broker in this AZ.
<b>Balance other jobs in</b>	Select additional region. This selection does not affect the on-demand RabbitMQ service.
<b>Network</b>	<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>. For more information about the Default Network, see <a href="#">Default Network and Service Network</a>.</p> <p>Typically, you select the network used for Pivotal Application Service (PAS) components.</p>
<b>Service Network</b>	<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 PAS and the On-Demand Broker run on. For information about the Service Network, see <a href="#">Default Network and Service Network</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 after you have configured them 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**.



**Note:** BOSH randomly deploys single node on-demand service instances across configured AZs. This minimizes the impact of an AZ outage and removes single points of failure.

## Configure Logging and Monitoring

Pivotal recommends that you configure logging to monitor the health of RabbitMQ. Follow the instructions in [Set Up Syslog Forwarding and Metrics Polling Interval](#).


## Configure Global Settings

Follow the steps below to configure global settings.

1. Click the **Global Settings for On-Demand Plans** tab.

Global Settings for On-Demand Plans

Dedicated Instance Service Quota ( min: 0 ) \*

20 

Dedicated Instance VM Options

☐ Allow outbound internet access (IaaS-dependent)

RabbitMQ plugins that can be enabled by App Developers

☐ rabbitmq\_event\_exchange

☐ rabbitmq\_mqtt

☐ rabbitmq\_stomp

☐ rabbitmq\_amqp1\_0

☐ Use Service Instance ID as cluster name

External TCP domain

External URL of the TCP Router

Port Range

Shareable Instances\*

☒ No

☐ Yes

On Demand - Secure Service Instance Credentials with Runtime CredHub\*

☒ No

☐ Yes

On Demand Service Broker Static IP

10.0.4.105

Number of canary instances to upgrade or recreate ( min: 0 ) \*



1

Maximum number of instances upgraded or recreated in parallel (min: 1) \*


Custom Apps Domain for smoke tests

Save

2. Configure the following:

Field	Instructions
<b>Dedicated Instance Service Quota</b>	Set the total number of on-demand service instances that can be deployed. For more information, see <a href="#">Setting Limits for On-Demand Service Instances</a> .
<b>Allow outbound internet access (IaaS-dependent)</b>	<p>Select this checkbox to enable external log forwarding, send backup artifacts to external destinations, or communicate with an external BOSH blobstore.</p> <div>  <p><b>Note:</b> Outbound network traffic rules also depend on your IaaS settings. Consult your network or IaaS admin to ensure that your IaaS permits outbound traffic to the external networks you need.</p> </div>
<b>RabbitMQ plugins that can be enabled by App Developers</b>	For more information, see <a href="#">Optional RabbitMQ Server Plugins</a> below.
<b>Use Service Instance ID as cluster name</b>	<p><b>In RabbitMQ v1.18.7 and later patches:</b> Select this checkbox to set the cluster name to its service instance GUID instead of the default name <code>rabbit@localhost</code>. This enables you to filter metrics by cluster name.</p> <div>  <p><b>Warning:</b> If you have federated queues and you select this checkbox, RabbitMQ recreates the federated queues with a new cluster name. This duplicates the number of downstream federated queues that consume messages from an upstream queue.</p> </div>
<b>External TCP domain</b>	Enter the external URL assigned to the TCP router. This corresponds to your external load balancer.
<b>Port Range</b>	Enter a range of ports to expose on the TCP Router. You must have at least one port for each service instance. Ensure that the range has enough capacity to accommodate the maximum number of service instances.
<b>Shareable Instances</b>	<p>Click <b>Yes</b> to enable the feature for sharing instances.</p> <p>Sharing a service instance between spaces enables apps in different spaces to share databases and messaging queues. For more information, see <a href="#">Sharing Service Instances</a>.</p>



<b>On Demand - Secure Service Instance Credentials with Runtime CredHub</b>	<p>For on-demand services instances, click <b>Yes</b> to secure credentials with CredHub.</p> <div>  <p><b>Note:</b> For this feature to work you must also enable it in PAS. For instructions, see <a href="#">Step 1: Configure the PAS Tile</a>. After deploying the tile, notify developers that they must unbind and rebind existing service instances to secure their credentials with CredHub.</p> </div>
<b>On Demand Service Broker Static IP</b>	Enter an IP address to assign to your on-demand service broker node. BOSH allocates an IP address if the field is left blank.
<b>Number of upgrade canary instances</b>	Set the number of canary instances on which to run the <code>upgrade-all-service-instances</code> or the <code>recreate-all-service-instances</code> errands first. If the errand succeeds on all canary instances, it runs on the remaining instances.
<b>Maximum number of instances upgraded in parallel</b>	Set the limit for the number of instances on which to simultaneously run the <code>upgrade-all-service-instances</code> errand or the <code>recreate-all-service-instances</code> errand. The number of available BOSH workers limits the number of simultaneous runs. See <a href="#">workers</a> in the BOSH documentation. Set the value lower than this limit to avoid over-saturating BOSH.
<b>Custom apps domain for smoke tests</b>	<p>Configure to provide a custom apps domain for the smoke tests. If left blank, the default CF apps domain is used.</p> <p>Failure to configure a custom apps domain could cause smoke tests to fail if the smoke tests domain is not accessible from the RabbitMQ Service Instance domain for on-demand services.</p>

3. Click **Save**.

## Configure Security

RabbitMQ lets you use Transport Layer Security (TLS) to secure communication between apps and RabbitMQ service instances.

To configure the TLS settings, do the following:

1. Ensure that you have performed the procedures in [Provide or Generate a CA Certificate](#) before configuring the tile and applying changes.
2. Click the **Security for On-Demand Plans** tab.

### Security for On-Demand Plans

A CA certificate must be added to Credhub when enabling TLS. See the docs for more information.

TLS Options\*

☒ Not Configured - Developers will not be able to configure their On Demand service instances to use TLS  
☐ Optional - Developers may configure their On Demand service instances to use TLS  
☐ Enforced - All new and existing On Demand service instances will be configured to use TLS. On applying this setting, any applications that used non-TLS bindings will require re-binding and must support TLS connections

**Save**

3. Under **TLS Options**, select either **Optional** or **Enforced**. **Optional** enables developers to configure their RabbitMQ service instances to use TLS. **Enforced** requires TLS to be

enabled.



**Breaking Change:** If TLS is set to **Enforced** then all existing service instances use TLS after changes from the **Upgrade All Service Instances** errand are applied. Any apps not using TLS are no longer able to communicate with their service instances. Such apps require a new binding and must be configured to communicate with their RabbitMQ service instance through TLS.



**Note:** When TLS is subsequently set to **Not Configured**, existing service instances continue to use TLS. However, new instances are not configured with TLS.

4. Click **Save**.
5. Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
6. In the Installation Dashboard, ensure the **Upgrade All Service Instances** errand is set to **On**, and then click **Apply Changes**.
7. After deploying the tile, app developers can configure their service instances to use TLS. For developer instructions, see [Enable Transport Layer Security \(TLS\) for Your Service Instance](#).

## Configure the Service Plan

To enable the on-demand service, you must configure at least one on-demand plan.

- You can configure up to five on-demand plans: **On Demand Instance: Plan 1 – On Demand Instance: Plan 5**.
- All on-demand plans can be configured to have 1, 3, 5, or 7 RabbitMQ nodes.
- If the on-demand service is not enabled, the on-demand broker is deployed alongside the RabbitMQ installation, but it is not available in the Marketplace.



**Note:** You must fully configure **On Demand Instance: Plan 1** even if you disable access to this plan. See **CF Service Access** in the table below.

1. Choose the on-demand service instance you want to configure. You must complete the required fields for **On Demand Instance: Plan 1** even if you disable this plan.

### Plan 1 Configuration

CF Service Access\*

Enable Service Access

Plan Name \*

single-node

Plan Description \*

This plan provides a single dedicated RabbitMQ node

☐ Paid Plan

☐ Service-Gateway Access

☐ Wait for Queue Synchronization

Plan Features \*

RabbitMQ

Plan Quota (min: 0) \*

10

Number of Nodes (min: 1, max: 7) \*

1

Network Partition Behaviour \*

pause\_minority

AZ Placement \*

☒ us-central1-f

☒ us-central1-c

☒ us-central1-b

RabbitMQ VM Type \*

Automatic: large (cpu: 2, ram: 8 GB, disk: 16 GB)

Persistent Disk Type \*

Automatic: 30 GB


I acknowledge that I have configured the Persistent Disk Size to be at least 2x the amount of RAM of the selected VM type \*

☒ Acknowledge

Save

2. Configure the fields as follows:

Field	Instructions
Enable This Plan	For Plans 2 - 5, enable the plan by selecting <b>Plan Enable</b> .

<b>CF Service Access</b>	<p>Enable or disable access to this plan, or leave access unchanged.</p> <p>If you enable Plan 1, the default setting for Plans 2 - 5 is <b>Enable Service Access</b>. If you change this default setting, the smoke tests fail. Therefore, if you enable Plan 1 and want to change this default, before doing so, set the <b>On-Demand Instance Smoke Tests</b> errand to <b>Off</b>. For more information, see the <a href="#">Errands</a> section below.</p> <ul style="list-style-type: none"> <li>✦ <b>Enable Service Access</b>—Gives access to all orgs, and displays the service plan to all developers in the Marketplace.</li> <li>✦ <b>Disable Service Access</b>—Disables access to all orgs, and hides the service plan to all developers in the Marketplace. This setting cannot be selected at a later time in the UI.</li> <li>✦ <b>Leave Service Access Unchanged</b>—Keeps any existing access settings, and only displays the service plan in the Marketplace to members of orgs that have access to the plan. You can change the access settings later using the cf CLI. For instructions, see <a href="#">Controlling Access to Service Plans by Org</a>.</li> </ul>
<b>Plan Name</b>	Accept the default or enter a name. This is the name that appears in the Marketplace.
<b>Plan Description</b>	Accept the default or enter a description. This description appears in the Marketplace.
<b>Paid Plan</b>	Select this checkbox to indicate that this service plan is paid. The plan is marked with an asterisk in the <code>cf marketplace</code> list and labeled <code>paid</code> in the <b>free or paid</b> column when individual plans are listed.
<b>Service Gateway Access</b>	Select this checkbox to enable service gateway access. For information, see <a href="#">Enabling Service-Gateway Access</a> .
<b>Wait for Queue Synchronization</b>	<p><b>In RabbitMQ v1.18.7 and later patches:</b> Select this checkbox to enable the following pre-stop checks:</p> <ul style="list-style-type: none"> <li>✦ <code>rabbitmq-upgrade await_online_quorum_plus_one --timeout 3600</code></li> <li>✦ <code>rabbitmq-upgrade await_online_synchronized_mirror --timeout 3600</code></li> </ul> <p>Enabling these checks ensures that queues are synced before an upgrade.</p> <div data-bbox="389 1438 1414 1673">  <p><b>Warning:</b> If you enable these checks, the pre-stop script can fail with the error:</p> <pre>Timed out waiting for mirror queue critical node to sync after 3600 seconds</pre> <p>For more information, see <a href="#">Pre-Stop Script Times Out When Waiting for Queue Synchronization</a>.</p> </div>
<b>Plan Features</b>	Accept the default or enter a description. This description appears in Apps Manager.
<b>Plan Quota</b>	Enter the maximum number of on-demand service instances that can be available at one time. For more information, see <a href="#">Setting Limits for On-Demand Service Instances</a> .
<b>Number of Nodes</b>	Enter 1, 3, 5 or 7. This setting only affects new service instances. Previously deployed service instances are <b>not</b> updated.

<b>Network Partition Behavior</b>	Select <code>pause_minority</code> or <code>autoheal</code> . VMware recommends using <code>pause minority</code> . For more information, see <a href="#">Consistency or Availability Tradeoff</a> .
<b>AZ Placement</b>	<p>This field is available after you complete the Assign AZs and Networks page. See the <a href="#">Assign AZs and Networks</a> section above for more information.</p> <ul style="list-style-type: none"> <li>For a single-node plan, select one or more AZs.</li> <li>For a plan containing multiple nodes, select only one AZ. Pivotal recommends this for multi-node plans to minimize risks due to network latency and partitions. See <a href="#">Network Latency</a> and <a href="#">Consistency or Availability Tradeoff</a> for details.</li> </ul> <p>If you change this selection after deployment, existing instances are not affected by the change. See <a href="#">Determine which AZs a Service Instance Uses</a> below for more information.</p>
<b>RabbitMQ VM Type</b>	Select a large VM type. The plan creates a service instance of this size. For more information, see <a href="#">About RabbitMQ VM Types and Persistent Disk Size</a> below.
<b>Persistent Disk Type</b>	This is where RabbitMQ pages messages to disk. Service instance deployments fail if this value is less than twice the volume of RAM of the selected RabbitMQ VM Type. For more information, see <a href="#">About RabbitMQ VM Types and Persistent Disk Size</a> below.

- Click **Save**.

### Determine which AZs a Service Instance Uses



**Important:** If you change this configuration after you have selected AZs and deployed service instances, existing instances are not placed in the newly configured AZs when the Upgrade All Service Instances or Recreate All Service Instances errands are run. This prevents re-creation of the VMs in different AZs, which could lead to data loss.

All new service instances, however, are created in the newly configured AZs.

To determine which AZs a service instance is placed in, do one of the following:

- Retrieve the service GUID using the `cf service SERVICE_INSTANCE --guid` command and then run the BOSH `instances` command for the `service-instance_GUID` deployment.
- With syslog forwarding enabled, inspect the service broker logs when running the **Upgrade All Service Instances** errand. For each existing service instance, the log message includes the service instance GUID and the AZs the service instance is running in.

### RabbitMQ VM Types and Persistent Disk Size

The **RabbitMQ VM Type** and **Persistent disk type** are required fields on the service plan configuration pages. These properties are pre-configured by default.

RabbitMQ VM Type\*

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

Persistent disk type\*

10 GB

Pivotal recommends that the value of **Persistent disk type** be twice the amount of RAM of the selected **RabbitMQ VM Type**.

- You can change the RabbitMQ VM type and the size of the persistent disk that is attached to the RabbitMQ service instances. For example, if you are running out of disk space you might want to increase the persistent disk size by changing the **Persistent disk type** field. If you make changes, ensure that the persistent disk size is still twice the size of the RAM of the RabbitMQ VM type.
- RabbitMQ raises alarms when disk space drops below the configured limit. Incorrect disk sizes might cause the deployed instance not to start. RabbitMQ declines to start if there is not enough space available according to the threshold.
- On-Demand instances are configured with a threshold set to the 150% of the RAM of the VM. Use the following table as a guide 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 (1.5 x RAM)	Suggested disk size (2 x RAM)
10 GB	15 GB	20 GB
16 GB	24 GB	32 GB
32 GB	48 GB	64 GB

Minimum resources required for each RabbitMQ VM:

- CPU: 2
- RAM: 1 GB
- Ephemeral disk: 2 GB
- Persistent disk: 4 GB

For more information, see the following:

- [Memory and Disk Alarms](#)
- [Disk Alarms](#)

For information on all preconfigured settings, see [Things that are Preconfigured](#).

## Verify the Stemcell

To verify that you have the correct stemcell, follow the procedure in [Importing and Managing Stemcells](#).

## 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 **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes** to complete the installation of RabbitMQ.

## Errands

When deploying or updating RabbitMQ, Ops Manager can optionally run a series of [Post-Deploy Errands](#), detailed in the section below. An example is the [Smoke Tests](#) errand, which checks the health of the RabbitMQ cluster after a deploy or upgrade.

You can toggle errands on and off on the **Review Pending Changes** page.

☒



**RabbitMQ**  
Version

**Depends on**  
Small Footprint PAS (cf) >= 1.11.0

**ERRANDS**

**Select errands to run during the deploy**

☒ Pre-Provisioned Broker Registrar

☒ Pre-Provisioned Smoke Tests

☒ Register On Demand Service Broker

☒ On Demand Instance Smoke Tests

☒ Upgrade All Service Instances

☐ Recreate All Service Instances



**Warning:** In RabbitMQ v1.9.0 and later, post-deploy errands are on by default except for the Recreate All Service Instances errand. Pivotal recommends keeping these defaults, because the smoke tests can encounter unexpected issues, and on-demand instances of RabbitMQ might fall behind if the Upgrade All Service Instances errand is not on by default.

You can change these defaults by clicking **Errands** in the RabbitMQ **Settings** tab as well as the

defaults for [pre-delete](#) errands.

For more information on errand run rules, see [Errand Run Rules](#).

RabbitMQ errands are co-located with their brokers to decrease errand run time and VM footprint. In earlier releases, a new VM was deployed for each errand. For more information about errands, see [Errands](#).

## Post-Deploy Errands

Errand	Description
<b>Pre-Provisioned Broker Registrar</b>	Makes the pre-provisioned RabbitMQ service plans available in the Marketplace
<b>Pre-Provisioned Smoke Tests</b>	Checks that a pre-provisioned RabbitMQ service instance can be bound to a PAS app, and that the app can publish and subscribe to a RabbitMQ cluster. See <a href="#">Smoke Tests</a> .
<b>Register On Demand Service Broker</b>	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.
<b>On Demand Instance Smoke Tests</b>	Checks that on-demand RabbitMQ service instances can be bound to a PAS app, and that the app can publish and subscribe to a RabbitMQ cluster. Smoke tests only run against Plan 1. See <a href="#">Smoke Tests</a> .
<b>Upgrade All Service Instances</b>	On-Demand instances are updated and redeployed if there are changes to on-demand plan settings or the tile is upgraded. If this errand is set to <b>Off</b> , updates to on-demand plan settings are <b>not</b> applied to existing service instances. <b>Pivotal recommends that this errand is configured to always run.</b>
<b>Recreate All Service Instances</b>	This errand re-creates all on-demand instance VMs managed by the on-demand broker. It is useful for tasks that require re-creating a service instance VM, such as rotating the Ops Manager root certificate authority (CA) or fully restoring the platform during disaster recovery or migration. <b>This errand is off by default and should be enabled only when you want to re-create a VM.</b>

## Pre-Delete Errands

Errand	Description
<b>Deregister and Purge Instances</b>	Removes the pre-provisioned RabbitMQ service from the Marketplace and deletes all associated service instances and bindings. For more information, see <a href="#">Turning Off the Pre-Provisioned Service</a> .
<b>Delete All Service Instances</b>	Unbinds and deletes existing on-demand service instances. The duration of this errand depends on the number of deployed on-demand instances.
<b>Deregister On-Demand Service Broker</b>	Removes the on-demand RabbitMQ service from the Marketplace



## 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 service 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 RabbitMQ 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",
  "username": "admin-username",
  "password": "admin-password",
  ...
}
```

## RabbitMQ Server Plugins

RabbitMQ supports a subset of available RabbitMQ plugins. See the sections below for which plugins are supported, and whether they are enabled or disabled by default.

## Enabled RabbitMQ Server Plugins

The following plugins are enabled by default, you cannot disable them:

- `rabbitmq_management`: For more information, see [Management Plugin](#) in the RabbitMQ documentation.
- `rabbitmq_federation`: For more information, see [Federate Exchanges and Queues](#).
- `rabbitmq_federation_management`
- `rabbitmq_shovel`: For more information, see [Shovel Exchanges and Queues](#).
- `rabbitmq_shovel_management`
- `rabbitmq_sharding`: For more information, see [rabbitmq sharding](#) in GitHub.
- `rabbitmq_consistent_hash_exchange`: For more information, see [rabbitmq consistent hash exchange](#) in GitHub.

## Optional RabbitMQ Server Plugins

The following plugins are disabled by default:


- `rabbitmq_event_exchange`: For more information, see the [Event Exchange Plugin](#) in the RabbitMQ documentation.
- `rabbitmq_mqtt`: For more information, see the [MQTT Plugin](#) in the RabbitMQ documentation.
- `rabbitmq_stomp`: For more information, see [STOMP Plugin](#) in the RabbitMQ documentation.
- `rabbitmq_amqp1_0`: For more information, see [AMQP 1.0 Plugin](#) in GitHub.

You can enable disabled plugins by doing the following:

1. Navigate to **Ops Manager Installation Dashboard > RabbitMQ > Settings > Global Settings for On-Demand Plans**.
2. Select the plugins you want to enable under **RabbitMQ plugins that can be enabled by App Developers**.

## Global Settings for On-Demand Plans

Dedicated Instance Service Quota (min: 0) \*

20 

Dedicated Instance VM Options

☐ Allow outbound internet access (IaaS-dependent)

RabbitMQ plugins that can be enabled by App Developers

☒ rabbitmq\_event\_exchange If selected, these plugins can be enabled by App Developers.

☐ rabbitmq\_mqtt

After plugins are enabled for the platform, developers can enable them for each service instance. For more information, see [Enable Optional Plugins](#).

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Enabling Service-Gateway Access (Beta)



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic explains how to enable service-gateway access.

## Overview

Service-gateway access enables a RabbitMQ for Pivotal Platform on-demand service instance to connect to external components that are not on the same foundation as the service instance.

For a more detailed overview, see [Service-Gateway Access](#).

To enable service-gateway access for an on-demand offering:

1. [Enable TCP routing using the PAS tile](#)
2. [Configure the firewall to allow incoming traffic to the TCP router](#)
3. [Configure the load balancer in the IaaS to redirect traffic to the TCP router](#)
4. [Create a DNS record that maps to the load balancer](#)
5. [Configure a service-gateway-enabled plan](#)



**Warning:** Pivotal recommends that you configure Transport Layer Security (TLS)

alongside service-gateway access to prevent man-in-the-middle attacks. For instructions on how to configure TLS, see [Configure Security](#).

## Enable TCP Routing Using the PAS Tile

TCP routing is disabled by default. To enable TCP routing:

1. Go to the Networking pane of the PAS tile.
2. Under **Enable TCP requests to apps through specific ports on the TCP router**, select **Enable TCP routing**.
3. For TCP routing ports, enter one or more ports to which the load balancer forwards requests. For example, `1024` for a single port or `1024 – 1123` for a range of ports.
4. Apply changes in Ops Manager for the PAS tile to create the TCP router.
5. From the status tab of the PAS tile, make note of the cloud identity (CID) of the TCP router.

Small Footprint Pivotal Application Service

Settings Status Credentials Logs

JOB	INDEX	IPS	AZ	CID
Database	0	10.0.4.5	us-central1-f	vm-a048c4af-2aa3-4b76-75ea-6dc8551a8f63
File Storage	0	10.0.4.6	us-central1-f	vm-c9c22052-dd27-4eac-6ade-b2a22468afad
Control	0	10.0.4.7	us-central1-f	vm-a4d9640e-c178-4d8f-52db-c6fa68ab2d41
Compute	0	10.0.4.8	us-central1-f	vm-670f569a-1845-4d90-67ba-98c335868440
Router	0	10.0.4.9	us-central1-f	vm-bbeef2fc-6736-45e6-4128-466431fe0c96
MySQL Monitor	0	10.0.4.11	us-central1-f	vm-f93d65b2-63ce-4ee3-51e8-425311734109
TCP Router	0	10.0.4.12	us-central1-f	vm-21ca9051-9e43-4d46-4b84-8414878e0a56

[Click here to view a larger version of this image.](#)

## Configure the Firewall to Allow Incoming Traffic to the TCP Router

1. Allow incoming traffic to the TCP router VM created in [Enable TCP Routing Using the PAS Tile](#) above. For information about how to do so, see the documentation for your IaaS.

## Configure the Load Balancer in the IaaS to Redirect Traffic to the TCP Router

To configure the load balancer:

1. Use the IaaS console and the CID you noted down earlier to find the VM that runs the TCP router.
2. Create an external TCP load balancer that points to the VM running the TCP router.
3. Configure a new external port range that does not overlap with the TCP networking port or port range you configured in [Enable TCP Routing Using the PAS Tile](#) above. For example, if your TCP routing port range is `1024–1123` then your load balancer port range must not be

1000–1100.



**Note:** Each RabbitMQ service instance using service-gateway access requires a unique port. Ensure that the port range configured above has enough capacity to accommodate the maximum number of service instances desired. The start port and the end port are both inclusive.

4. Note down this port range.

## Create a DNS Record That Maps to the Load Balancer

To create a DNS record and prepare to map it:

1. Following the documentation for your IaaS, create a new DNS record of type A that maps to the external IP address of the load balancer created in [Configure the Load Balancer in the IaaS to Redirect Traffic to the TCP Router](#) above.
2. Note down the domain used for this DNS record.


## Configure a Service-Gateway-Enabled Plan

To configure a service-gateway-enabled plan:

1. In the **Global Settings for On-Demand Plans** pane in the RMQ tile, fill in the following fields:
  - ✦ **External TCP domain:** This must correspond to the DNS entry for the external load balancer, noted in [Create a DNS Record That Maps to the Load Balancer](#) above.
  - ✦ **Port Range:** This is the range of ports you configured for the external load balancer for RabbitMQ service instances in [Configure the Load Balancer in the IaaS to Redirect Traffic to the TCP Router](#) above.

## Global Settings for On-Demand Plans

Dedicated Instance Service Quota (min: 0) \*

20 

Dedicated Instance VM Options

☐ Allow outbound internet access (IaaS-dependent)

RabbitMQ plugins that can be enabled by App Developers

☐ rabbitmq\_event\_exchange

☐ rabbitmq\_mqtt

☐ rabbitmq\_stomp

☐ rabbitmq\_amqp1\_0

☐ Use Service Instance ID as cluster name

External TCP domain

tcp.elcajon.cf-app.com

Port Range

1024-2025



**Warning:** If you already have service instances using service-gateway, any modifications to this range must include ports that are already assigned to these service instances. If the port range does not contain the ports already assigned to service instances, the upgrades for the service instances fail. For example, if service-gateway access has the port range 1000-1005, and there are service instances that correspond to ports 1000, 1001, and 1002, then the new port range must have ports 1000, 1001, and 1002.

2. Navigate to the service plan that you want to use and select the **Service-Gateway Access** checkbox.



**Note:** Pivotal recommends that you change the name or the description of

the plan to indicate that service-gateway access is enabled for that plan.

CF Service Access\*

Enable Service Access

Plan Name \*

service-gateway

Plan Description \*

This plan provides a single service gateway enabled RabbitMQ 3.7 node

☐ Paid Plan

☒ Service Gateway Access



**Note:** If service-gateway access is disabled and then re-enabled, app developers must create new service keys to obtain a new set of credentials for service-gateway access.

3. Go back to **Ops Manager Installation Dashboard > Review Pending Changes**.
4. Click **Apply Changes** to apply the changes to the RabbitMQ tile.

## Disable Service-Gateway Access



**Note:** If service-gateway access is disabled and then re-enabled, app developers must create new service keys to obtain a new set of credentials for service-gateway access.

To disable service-gateway access:

1. Navigate to the service plan that you want to disable service-gateway access for and clear the **Service-Gateway Access** checkbox.



**Note:** Pivotal recommends that you change the name or the description of the plan to indicate that service-gateway access is disabled for that plan.

2. Go back to **Ops Manager Installation Dashboard > Review Pending Changes**.
3. Click **Apply Changes** to apply the changes to the RabbitMQ tile.

## Developer Workflow

For instructions for app developers, see [Create a Service Instance with Service-Gateway Access](#).

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Smoke Tests



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

RabbitMQ for Pivotal Platform runs a set of smoke tests during installation to confirm system health.

## Smoke Test Steps

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

1. Targets the org `system` and creates a space to run the tests.
2. Deploys an instance of the [CF RabbitMQ Example App](#) to this space
3. Creates a RabbitMQ service instance and binds it to the CF RabbitMQ Example App
4. Checks that the CF RabbitMQ Example App can write to and read from the RabbitMQ service instance
5. Cleans up all deployed application and all its service bindings. Finally, the cf space is deleted.



**Note:** Smoke tests fail unless you enable global default application security groups (ASGs). You can enable global default ASGs by binding the ASG to the `system` org without specifying a space. To enable global default ASGs, use `cf bind-running-security-group`.

## Troubleshooting

If errors occur while the smoke tests run, they are summarized at the end of the errand log output. Detailed logs can be found where the failure occurs.

When encountering an error when running smoke tests, it can be helpful to search the log for other instances of the error summary printed at the end of the tests, for example, `Failed to target Cloud Foundry`. Lookout for **TIP:** ... in the logs next to any error output for further troubleshooting hints.

[Create a pull request or raise an issue on the source for this page in GitHub](#)



# Monitoring and KPIs for On-Demand RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

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

On-Demand RabbitMQ components generate many of the same metrics as the pre-provisioned RabbitMQ service components. For information about metrics for the pre-provisioned service, see [Monitoring and KPIs for Pre- Provisioned RabbitMQ for Pivotal Platform](#).



**Note:** On-Demand service metrics are prefixed with `p.rabbitmq` (with a dot), to distinguish them from the pre-provisioned service metrics.

See [Overview of Logging and Metrics](#) for general information about logging and metrics in Pivotal Application Service.

## Set up Syslog Forwarding and Metrics Polling Interval

As of RabbitMQ 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 Pivotal Operations Manager **Settings** tab.

To enable monitoring for RabbitMQ, 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 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.

## Syslog and Metrics settings for both Pre-Provisioned and On-Demand service

Metrics polling interval \*

Do you want to configure log forwarding to a syslog server? \*

☐ No

☒ Yes

Syslog address \*

Syslog port \*

Transport protocol \*

☐ Enable TLS

Permitted Peer

Custom CA Certificate

**Save**

- Click **Syslog and Metrics**.
- Configure the fields on the Syslog pane as follows:

Field	Description
-------	-------------

<b>Metrics polling interval</b>	The default setting is 30 seconds for all deployed components. Pivotal recommends that you do not change this interval. In order to avoid overwhelming components, do not set this below 10 seconds. Set this to -1 to disable Metrics. Changing this setting affects all deployed instances.
---------------------------------	---

<b>Syslog address</b>	Enter the IP or DNS address of the syslog server
<b>Syslog port</b>	Enter the port of the syslog server
<b>Transport protocol</b>	Select the transport protocol of the syslog server. The options are <b>TLS</b> , <b>UDP</b> , or <b>REL</b> .
<b>Enable TLS</b>	Enable TLS to the syslog server.
<b>Permitted Peer</b>	If there are several peer servers that can respond to remote syslog connections, enter a wildcard in the domain, such as <code>*.example.com</code> .
<b>Custom CA Certificate</b>	If the server certificate is not signed by a known authority, for example, an internal syslog server, enter the CA certificate of the log management service endpoint.

- Click **Save**.
- Return to the Ops Manager Installation Dashboard.
- Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
- Click **Apply Changes** to redeploy with the changes.

## Logging Format

With on-demand RabbitMQ 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.

For users familiar with documentation for previous versions of the tile, the tag formerly called 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=director.be5a66bb-a9b4-459f-a0d3-1fc5c9c3ed79.be148cc6-91ef-4eed-a788-237b0b8c63b7 src=10.254.50.4 spt=4222 shost=5ae233e0-ecc5-4868-9ae0-f9767571251b
<86>2017-06-28T16:06:16.704572+00:00 10.244.16.133 useradd [job=rmq index=0 id=e37ecdc
```

```
a-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=e37ecdc
a-5b10-4141-abd8-e1d777dfd8b5] new user: name=bosh_ly0d2rbjr, UID=1001, GID=1003, hom
e=/var/vcap/bosh_ssh/bosh_ly0d2rbjr, shell=/bin/bash
<86>2017-06-28T16:06:16.736932+00:00 10.244.16.133 usermod [job=rmq index=0 id=e37ecdc
a-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=e37ecdc
a-5b10-4141-abd8-e1d777dfd8b5] add 'bosh_ly0d2rbjr' to group 'vcap'
```

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

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

Tag	Description
<b>PRI</b>	This is a value which in future will be used to describe the severity of the log entry and which facility it came from.
<b>TIMESTAMP</b>	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 entry was generated.
<b>IP_ADDRESS</b>	The internal IP address of server on which the log entry originated
<b>PROGRAM_NAME</b>	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.
<b>NAME</b>	The BOSH instance group name (for example, <code>rabbitmq_server</code> )
<b>JOB_INDEX</b>	BOSH job index. Used to distinguish between multiple instances of the same job.
<b>JOB_ID</b>	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.
<b>MESSAGE</b>	The log entry that appears

## RabbitMQ Program Names

The following table lists the program names used in the logs:

Program Name	Description
<code>rabbitmq_server_cluster_check</code>	Checks that the RabbitMQ cluster is healthy. Runs after every deploy.
<code>rabbitmq_server_node_check</code>	Checks that the RabbitMQ node is healthy. Runs after every deploy.
<code>rabbitmq_route_registrar_stderr</code>	Registers the route for the management API with the Gorouter in your Pivotal Application Service deployment.
<code>rabbitmq_route_registrar_stdout</code>	Registers the route for the management API with the Gorouter in your Pivotal Application Service deployment.
<code>rabbitmq_server</code>	The Erlang VM and RabbitMQ apps. <i>Logs can span multiple lines.</i>
<code>rabbitmq_server_drain</code>	Shuts down the Erlang VM and RabbitMQ apps. Runs as part of the BOSH lifecycle.
<code>rabbitmq_server_http_api_access</code>	Access to the RabbitMQ Management UI.

Program Name	Description
<code>rabbitmq_server_init</code>	Starts the Erlang VM and RabbitMQ.
<code>rabbitmq_server_post_deploy_stderr</code>	Runs the node check and cluster check. Runs after every deploy.
<code>rabbitmq_server_post_deploy_stdout</code>	Runs the node check and cluster check. Runs after every deploy.
<code>rabbitmq_server_pre_start</code>	Runs before the rabbitmq-server job is started.
<code>rabbitmq_server_sasl</code>	Supervisor, progress, and crash reporting for the Erlang VM and RabbitMQ apps.
<code>rabbitmq_server_shutdown_stderr</code>	Stops the RabbitMQ app and Erlang VM.
<code>rabbitmq_server_shutdown_stdout</code>	Stops the RabbitMQ app and Erlang VM.
<code>rabbitmq_server_startup_stderr</code>	Starts the RabbitMQ app and Erlang VM, then configures users and permissions.
<code>rabbitmq_server_startup_stdout</code>	Starts the RabbitMQ app and Erlang VM, then configures users and permissions.
<code>rabbitmq_server_upgrade</code>	Shuts down Erlang VM and RabbitMQ app if required during an upgrade.

## What Are Metrics

Metrics are regularly-generated log messages that report measured component states. The metrics polling interval defaults to 30 seconds. The **metrics polling interval** is a configuration option on the RabbitMQ tile (**Settings > RabbitMQ**). Setting this interval to -1 disables metrics. 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/memory" value:1024 unit:"MB">
```

The following sections describe the metrics used as Key Performance Indicators and other useful metrics for monitoring the RabbitMQ on-demand service.

## Key Performance Indicators

Key Performance Indicators (KPIs) for RabbitMQ 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 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 raw component metrics, see [Component Metrics Reference](#).

## Component Heartbeats

Key RabbitMQ 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 <b>is alive</b> 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>	Check the RabbitMQ Service Broker logs for errors. You can find this VM by targeting your RabbitMQ deployment with BOSH and running the command. <code>bosh -d service-instance_GUID vms</code>

### Server Heartbeat

#### p.rabbitmq/rabbitmq/heartbeat

<b>Description</b>	<p>RabbitMQ Server <b>is alive</b> 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>
<b>Recommended measurement</b>	Average over last 5 minutes


<b>Recommended alert thresholds</b>	<b>Yellow warning:</b> N/A <b>Red critical:</b> < 1
<b>Recommended response</b>	Check the RabbitMQ Server logs for errors. You can find the VM by targeting your RabbitMQ deployment with BOSH and running the following command, which lists <code>rabbitmq: bosh -d service-instance_GUID vms</code>

## 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 might lose the ability to perform disk I/O, which can cause data loss.</p> <div>  <p><b>Note:</b> This assumes non-persistent messages are handled by retries or some other logic by the producers.</p> </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>	<b>Yellow warning:</b> > 250000 <b>Red critical:</b> > 280000
<b>Recommended response</b>	The default <code>ulimit</code> for RabbitMQ is 300000. If this metric is met or exceeded for an extended period of time, consider reducing the load on the server.

### Erlang Processes

#### p.rabbitmq/rabbitmq/erlang/erlang\_processes

<b>Description</b>	<p>Erlang 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>	<b>Yellow warning:</b> > 900000 <b>Red critical:</b> > 950000
<b>Recommended response</b>	The default Erlang process limit in RabbitMQ 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.

## BOSH System Health Metrics

The BOSH layer that underlies Ops Manager generates `healthmonitor` metrics for all VMs in the deployment. As of Ops Manager v2.0, these metrics are included in the Loggregator Firehose by default. For more information, see [BOSH System Metrics Available in Loggregator Firehose](#) in *Pivotal Application Service Release Notes*.

All BOSH-deployed components generate the system health metrics below. These component metrics are from RabbitMQ components, and serve as KPIs for the RabbitMQ service.

### RAM

#### system.mem.percent

<b>Description</b>	RAM being consumed by the <code>p.rabbitmq</code> VM.  <b>Use:</b> RabbitMQ is considered to be in a good state when it has few 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.  Healthmonitor reports when RabbitMQ uses more than 40% of its RAM for the past ten minutes.  <b>Origin:</b> BOSH HM <b>Type:</b> percent <b>Frequency:</b> 30 s (default), 10 s (configurable minimum)
<b>Recommended measurement</b>	Average over last 10 minutes
<b>Recommended alert thresholds</b>	<b>Yellow warning:</b> > 40 <b>Red critical:</b> > 50
<b>Recommended response</b>	Add more consumers to drain the queue as fast as possible.

### CPU

#### system.cpu.user



<b>Description</b>	<p>CPU being consumed by user processes on the <code>p.rabbitmq</code> VM.</p> <p><b>Use:</b> A node that experiences context switching or high CPU usage becomes 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> BOSH HM  <b>Type:</b> percent  <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  <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.

## Ephemeral Disk

### `system.disk.ephemeral.percent`

<b>Description</b>	<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 50% of its Ephemeral Disk for the past ten minutes.</p> <p><b>Origin:</b> BOSH HM  <b>Type:</b> percent  <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; 50  <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. Insufficient disk space leads to node failures and might result in data loss due to all disk writes failing.

## Persistent Disk

### `system.disk.persistent.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 50% of its Persistent Disk.</p> <p><b>Origin:</b> BOSH HM  <b>Type:</b> percent  <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; 50  <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. Insufficient disk space leads to node failures and might result in data loss due to all disk writes failing.

## Determine If There Is a Network Partition

You can use the `reachable_nodes` metric to help to identify network partitions. This metric shows how many nodes in the cluster each individual node is aware of. A good indication that a node might be in a partition is when it is aware of only itself.

Here is an example of this metrics:

```
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/reachable_nodes" value:3 unit:"count">
```

You can create monitors to emit alerts in case a cluster seems to be in a partition. In a healthy cluster that is not undergoing upgrades, each node's `reachable_nodes` count is equal to the number of nodes in the cluster.

To monitor for network partition, Pivotal recommends alerting when one of the nodes starts reporting a `reachable_nodes` count that is less than the size of the cluster.

During rolling upgrades, nodes lose contact with other nodes. Therefore, only alert if a lowered `reachable_nodes` count persists longer than the expected upgrade time.

## Recover from a Network Partition

For information about how to recover from a network partition, see the [RabbitMQ documentation](#).

## Component Metric Reference

RabbitMQ 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 message server components emit the following metrics.

Full Name	Unit	Description
/p.rabbitmq/rabbitmq/heartbeat	boolean	Indicates whether the RabbitMQ server is available and able to respond to requests
/p.rabbitmq/rabbitmq/erlang/erlang_processes	count	The number of Erlang processes
/p.rabbitmq/rabbitmq/erlang/reachable_nodes	count	The number of nodes the current node can reach
/p.rabbitmq/rabbitmq/system/memory	MB	The memory in MB used by the node
/p.rabbitmq/rabbitmq/system/memory_alarm	boolean	Indicates if the memory alarm went off
/p.rabbitmq/rabbitmq/system/disk_free	MB	The disk space available on the node
/p.rabbitmq/rabbitmq/system/disk_free_alarm	boolean	Indicates if the disk free alarm went off
/p.rabbitmq/rabbitmq/connections/count	count	The total number of connections to the node
/p.rabbitmq/rabbitmq/consumers/count	count	The total number of consumers registered in the node
/p.rabbitmq/rabbitmq/messages/delivered	count	The total number of messages with the status <code>deliver_get</code> on the node
/p.rabbitmq/rabbitmq/messages/delivered_noack	count	The number of messages with the status <code>deliver_noack</code> on the node
/p.rabbitmq/rabbitmq/messages/delivered_rate	rate	The rate per second at which messages are being delivered to consumers or clients on the node
/p.rabbitmq/rabbitmq/messages/published	count	The total number of messages with the status <code>publish</code> on the node
/p.rabbitmq/rabbitmq/messages/published_rate	rate	The rate per second at which messages are being published by the node
/p.rabbitmq/rabbitmq/messages/redelivered	count	The total number of messages with the status <code>redeliver</code> on the node
/p.rabbitmq/rabbitmq/messages/redelivered_rate	rate	The rate per second at which messages are getting the status <code>redeliver</code> on the node
/p.rabbitmq/rabbitmq/messages/get_no_ack	count	The number of messages with the status <code>get_no_ack</code> on the node
/p.rabbitmq/rabbitmq/messages/get_no_ack_rate	rate	The rate per second at which messages get the status <code>get_no_ack</code> on the node
/p.rabbitmq/rabbitmq/messages/pending	count	The number of messages with the status <code>messages_unacknowledged</code> on the node
/p.rabbitmq/rabbitmq/messages/depth	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
<code>/p.rabbitmq/rabbitmq/exchanges/count</code>	count	The total number of exchanges on the node
<code>/p.rabbitmq/rabbitmq/messages/available</code>	count	The total number of messages with the status <code>messages_ready</code> on the node
<code>/p.rabbitmq/rabbitmq/queues/count</code>	count	The number of queues on the node
<code>/p.rabbitmq/rabbitmq/channels/count</code>	count	The number of channels on the node
<code>/p.rabbitmq/rabbitmq/vhosts/count</code>	count	The number of vhosts

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Managing the On- Demand Service

### Rotating Certificates



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

- [Access BOSH CredHub](#)
- [Check Expiration Dates](#)
- [Rotate CA Certificates Manually](#)

This topic describes how to access BOSH CredHub, check expiration dates, and rotate certificates using RabbitMQ for Pivotal Platform.

To rotate the Services TLS CA and its leaf certificates, use one of the following procedures:

- ◆ **Ops Manager v2.10:** See [Rotating the Services TLS CA and Its Leaf Certificates](#).
- ◆ **Ops Manager v2.9:** See [Rotating the Services TLS CA and Its Leaf Certificates](#).
- ◆ **Ops Manager v2.8:** See [Rotating the Services TLS CA and Its Leaf Certificates](#).
- ◆ **Ops Manager v2.7:** See [Rotating the Services TLS CA and Its Leaf Certificates](#).

Ops Manager v2.9 and later is compatible with CredHub Maestro. RabbitMQ v1.20 and later is compatible with CredHub Maestro.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

### Setting Limits for On-Demand Service Instances



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has

reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

- [Create Global-level Quotas](#)
- [Create Plan-level Quotas](#)
- [Create and Set Org-level Quotas](#)
- [Create and Set Space-level Quotas](#)
- [View Current Org and Space-level Quotas](#)
- [Monitor Quota Use and Service Instance Count](#)
- [Calculate Resource Costs for On-Demand Plans](#)
  - [Calculate Maximum Resource Cost Per On-Demand Plan](#)
  - [Calculate Maximum Resource Cost for All On-Demand Plans](#)
  - [Calculate Actual Resource Cost of all On-Demand Plans](#)

**Page last updated:**

On-demand provisioning is intended to accelerate app development by eliminating the need for development teams to request and wait for operators to create a service instance. However, to control costs, operations teams and administrators must ensure responsible use of resources.

There are several ways to control the provisioning of on-demand service instances by setting various **quotas** at these levels:

- [Global](#)
- [Plan](#)
- [Org](#)
- [Space](#)

After you set quotas, you can:

- [View Current Org and Space-level Quotas](#)
- [Monitor Quota Use and Service Instance Count](#)
- [Calculate Resource Costs for On-Demand Plans](#)

## Create Global-level Quotas

Each on-demand service has a separate service broker. A global quota at the service level sets the maximum number of service instances that can be created by a given service broker. If a service has more than one plan, then the number of service instances for all plans combined cannot exceed the global quota for the service.

The operator sets a global quota for each service tile independently. For example, if you have two service tiles, you must set a separate global service quota for each of them.

When the global quota is reached for a service, no more instances of that service can be created unless the quota is increased, or some instances of that service are deleted.

## Create Plan-level Quotas

A service may offer one or more plans. You can set a separate quota per plan so that instances of that plan cannot exceed the plan quota. For a service with multiple plans, the total number of instances created for all plans combined cannot exceed the [global quota](#) for the service.

When the plan quota is reached, no more instances of that plan can be created unless the plan quota is increased or some instances of that plan are deleted.

## Create and Set Org-level Quotas

An org-level quota applies to all on-demand services and sets the maximum number of service instances an organization can create within their foundation. For example, if you set your org-level quota to 100, developers can create up to 100 service instances in that org using any combination of on-demand services.

When this quota is met, no more service instances of any kind can be created in the org unless the quota is increased or some service instances are deleted.

To create and set an org-level quota, do the following:

1. Run this command to create a quota for service instances at the org level:

```
cf create-quota QUOTA-NAME -m TOTAL-MEMORY -i INSTANCE-MEMORY -r ROUTES -s SERVICE-INSTANCES --allow-paid-service-plans
```

Where:

- ✦ **QUOTA-NAME**—A name for this quota
- ✦ **TOTAL-MEMORY**—Maximum memory used by all service instances combined
- ✦ **INSTANCE-MEMORY**—Maximum memory used by any single service instance
- ✦ **ROUTES**—Maximum number of routes allowed for all service instances combined
- ✦ **SERVICE-INSTANCES**—Maximum number of service instances allowed for the org

For example:

```
cf create-quota myquota -m 1024mb -i 16gb -r 30 -s 50 --allow-paid-service-plans
```

2. Associate the quota you created above with a specific org by running the following command:

```
cf set-quota ORG-NAME QUOTA-NAME
```

For example:

```
cf set-quota dev_org myquota
```

For more information on managing org-level quotas, see [Creating and Modifying Quota Plans](#).

## Create and Set Space-level Quotas

A space-level service quota applies to all on-demand services and sets the maximum number of service instances that can be created within a given space in a foundation. For example, if you set your space-level quota to 100, developers can create up to 100 service instances in that space using any combination of on-demand services.

When this quota is met, no more service instances of any kind can be created in the space unless the quota is updated or some service instances are deleted.

To create and set a space-level quota, do the following:

1. Run the following command to create the quota:

```
cf create-space-quota QUOTA-NAME -m TOTAL-MEMORY -i INSTANCE-MEMORY -r ROUTES -s SERVICE-INSTANCES --allow-paid-service-plans
```

Where:

- ✦ **QUOTA-NAME**—A name for this quota
- ✦ **TOTAL-MEMORY**—Maximum memory used by all service instances combined
- ✦ **INSTANCE-MEMORY**—Maximum memory used by any single service instance
- ✦ **ROUTES**—Maximum number of routes allowed for all service instances combined
- ✦ **SERVICE-INSTANCES**—Maximum number of service instances allowed for the org

For example:

```
cf create-space-quota myspacequota -m 1024mb -i 16gb -r 30 -s 50 --allow-paid-service-plans
```

2. Associate the quota you created above with a specific space by running the following command:

```
cf set-space-quota SPACE-NAME QUOTA-NAME
```

For example:

```
cf set-space-quota myspace myspacequota
```

For more information on managing space-level quotas, see [Creating and Modifying Quota Plans](#).

## View Current Org and Space-level Quotas

To view **org** quotas, run the following command.

```
cf org ORG-NAME
```

To view **space** quotas, run the following command:

```
cf space SPACE-NAME
```

For more information on managing org and space-level quotas, see the [Creating and Modifying Quota Plans](#).

## Monitor Quota Use and Service Instance Count

Service-level and plan-level quota use, and total number of service instances, are available through the on-demand broker metrics emitted to Loggregator. These metrics are listed below:

Metric Name	Description
<code>on-demand-broker/SERVICE-NAME/quota_remaining</code>	Quota remaining for all instances across all plans
<code>on-demand-broker/SERVICE-NAME/PLAN-NAME/quota_remaining</code>	Quota remaining for a specific plan
<code>on-demand-broker/SERVICE-NAME/total_instances</code>	Total instances created across all plans
<code>on-demand-broker/SERVICE-NAME/PLAN-NAME/total_instances</code>	Total instances created for a specific plan



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

You can also view service instance usage information in Apps Manager. For more information, see [Reporting Instance Usage with Apps Manager](#).

## Calculate Resource Costs for On-Demand Plans

On-demand plans use dedicated VMs, disks, and various other resources from an IaaS, such as AWS. To calculate maximum resource cost for plans individually or combined, you multiply the quota by the cost of the resources selected in the plan configuration(s). The specific costs depend on your IaaS.

To view configurations for your RabbitMQ on-demand plan, do the following:

1. Navigate to **Ops Manager Installation Dashboard > RabbitMQ > Settings**.
2. Click the section for the plan you want to view. For example, **On Demand Instance: Plan 1**.

The image below shows an example that includes the VM type and persistent disk selected for the server VMs, as well as the quota for this plan.



Plan Quota ( min: 0, max: 50 ) \*

Number of Nodes ( min: 1, max: 7 ) \*

Network Partition Behaviour\*

AZ Placement \*

☒ us-central1-a

☒ us-central1-b

☐ us-central1-c

RabbitMQ VM Type\*

Persistent Disk Type\*



**Note:** Although operators can limit on-demand instances with plan quotas and a global quota, as described in the above topics, IaaS resource usage still varies based on the number of on-demand instances provisioned.

## Calculate Maximum Resource Cost Per On-Demand Plan

To calculate the maximum cost of VMs and persistent disk for each plan, do the following calculation:

**plan quota x cost of selected resources**

For example, if you selected the options in the above image, you have selected a VM type **micro**

and a persistent disk type **20 GB**, and the plan quota is **15**. The VM and persistent disk types have an associated cost for the IaaS you are using. Therefore, to calculate the maximum cost of resources for this plan, multiply the cost of the resources selected by the plan quota:

**(15 x cost of micro VM type) + (15 x cost of 20 GB persistent disk) = max cost per plan**

## Calculate Maximum Resource Cost for All On-Demand Plans

To calculate the maximum cost for all plans combined, add together the maximum costs for each plan. This assumes that the sum of your individual plan quotas is less than the global quota.

Here is an example:

**(plan1 quota x plan1 resource cost) + (plan2 quota x plan2 resource cost) = max cost for all plans**

## Calculate Actual Resource Cost of all On-Demand Plans

To calculate the current actual resource cost across all your on-demand plans:

1. Find the number of instances currently provisioned for each active plan by looking at the `total_instance` metric for that plan.
2. Multiply the `total_instance` count for each plan by that plan's resource costs. Record the costs for each plan.
3. Add up the costs noted in Step 2 to get your total current resource costs.

For example:

**(plan1 total\_instances x plan1 resource cost) + (plan2 total\_instances x plan2 resource cost) = current cost for all plans**

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Controlling Access to Service Plans by Org



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

If you want to restrict access to a service plan to a specific org, follow the instructions below.

You can also limit the number of service instances by setting quotas—for instructions, see [Setting Limits for On-Demand Instances](#).

## Change Access to Service Plans



**Note:** If the plan you are restricting is currently enabled for all orgs, you must first **Disable Service Access** for the plan, then grant access to the plan to specific orgs. Use the **CF Service Access** field to disable access in the [service plan configuration](#).

To restrict access to a plan for a specific org, run this command:

```
cf enable-service-access p.rabbitmq -p PLAN_NAME -o ORG_NAME
```

For example:

```
$ cf enable-service-access p.rabbitmq -p my-cluster-plan -o my-dev-org
```

For more information about the above command, see [Access Control](#).

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Troubleshooting and FAQs for On-Demand RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic provides operators with basic troubleshooting techniques and FAQs for on-demand RabbitMQ for Pivotal Platform.

### How to Retrieve a Service Instance GUID

You need the GUID of your service instance to run some BOSH commands. To retrieve the GUID, run the command:

```
cf service SERVICE-INSTANCE-NAME --guid
```

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.

### Troubleshoot Errors

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

### Common Services Errors

The following errors occur in multiple services:

- [Failed Installation](#)
- [Cannot Create or Delete Service Instances](#)
- [Broker Request Timeouts](#)
- [Instance Does Not Exist](#)
- [Cannot Bind to or Unbind from Service Instances](#)
- [Cannot Connect to a Service Instance](#)
- [Cannot Update a Service Instance](#)

- [Upgrade All Service Instances Errand Fails](#)
- [Missing Logs and Metrics](#)

## Failed Installation

<b>Symptom</b>	RabbitMQ fails to install.
<b>Cause</b>	<p>Reasons for a failed installation include:</p> <ul style="list-style-type: none"> <li>• Certificate issues: The on-demand broker (ODB) requires valid certificates.</li> <li>• Deploy fails. This could be due to a variety of reasons.</li> <li>• Networking problems: <ul style="list-style-type: none"> <li>✦ Cloud Foundry cannot reach the RabbitMQ broker</li> <li>✦ Cloud Foundry cannot reach the service instances</li> <li>✦ The service network cannot access the BOSH director</li> </ul> </li> <li>• The Register broker errand fails.</li> <li>• The smoke test errand fails.</li> <li>• Resource sizing issues: These occur when the resource sizes selected for a given plan are less than RabbitMQ requires to function.</li> <li>• Other service-specific issues.</li> </ul>
<b>Solution</b>	<p>To troubleshoot:</p> <ul style="list-style-type: none"> <li>• Certificate issues: Ensure that your certificates are valid and generate new ones if necessary. To generate new certificates, contact <a href="#">Support</a>.</li> <li>• Deploy fails: View the logs using Ops Manager to determine why the deploy is failing.</li> <li>• Networking problems: For how to troubleshoot, see <a href="#">Networking problems</a>.</li> <li>• Register broker errand fails: For how to troubleshoot, see <a href="#">Register broker errand</a>.</li> <li>• Resource sizing issues: Check your resource configuration in Ops Manager and ensure that the configuration matches that recommended by the service.</li> </ul>

## Cannot Create or Delete Service Instances

<b>Symptom</b>	<p>If developers report errors such as:</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"> <pre>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</pre> </div>
----------------	---

<b>Cause</b>	<p>Reasons include:</p> <ul style="list-style-type: none"> <li>• Problems with the deployment manifest</li> <li>• Authentication errors</li> <li>• Network errors</li> <li>• Quota errors</li> </ul>
<b>Solution</b>	<p>To troubleshoot:</p> <ol style="list-style-type: none"> <li>1. If the BOSH error shows a problem with the deployment manifest, open the manifest in a text editor to inspect it.</li> <li>2. To continue troubleshooting, <a href="#">Log in to BOSH</a> and target the RabbitMQ instance using the instructions on <a href="#">parsing a Cloud Foundry error message</a>.</li> <li>3. Retrieve the BOSH task ID from the error message and run the following command: <pre>bosh task TASK-ID</pre> </li> <li>4. If you need more information, <a href="#">access the broker logs</a> and use the <code>broker-request-id</code> from the error message above to search the logs for more information. Check for: <ul style="list-style-type: none"> <li>✦ <a href="#">Authentication errors</a></li> <li>✦ <a href="#">Network errors</a></li> <li>✦ <a href="#">Quota errors</a></li> </ul> </li> </ol>

## Broker Request Timeouts

<b>Symptom</b>	<p>If developers report errors such as:</p> <pre>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/service_bindings/aa635a3b-ef6d-41c3-a23f-55752f3f651b</pre>
<b>Cause</b>	<p>Cloud Foundry might not be connected to the service broker, or there might be a large number of queued tasks.</p>
<b>Solution</b>	<p>To troubleshoot:</p> <ol style="list-style-type: none"> <li>1. Confirm that Cloud Foundry (CF) is <a href="#">connected to the service broker</a>.</li> <li>2. Check the BOSH queue size: <ol style="list-style-type: none"> <li>1. Log in to BOSH as an admin.</li> <li>2. Run <pre>bosh tasks</pre> </li> </ol> <p>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 might not be sufficient resources for the level of load.</p> </li> <li>3. If the task queue is long, advise app developers to try again once the system is under less load.</li> </ol>

## Instance Does Not Exist

<b>Symptom</b>	<p>If developers report errors such as:</p> <pre>Server error, status code: 502, error code: 10001, message: Service broker error: instance does not exist`</pre>
<b>Cause</b>	The instance might have been deleted.
<b>Solution</b>	<p>To troubleshoot:</p> <ol style="list-style-type: none"> <li>1. Confirm that the RabbitMQ instance exists in BOSH and obtain the GUID CF by running: <pre>cf service MY-INSTANCE --guid</pre> </li> <li>2. Using the GUID obtained above, run: <pre>bosh -d service-instance_GUID vms</pre> </li> </ol> <p>If the BOSH deployment is not found, it has been deleted from BOSH. Contact Support for further assistance.</p>

## Cannot Bind to or Unbind from Service Instances

<b>Symptom</b>	<p>If developers report errors such as:</p> <pre>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 information: service: example-service, service-instance-guid: 8d69de6c-88c6-4283-b8bc-1c46103714e2, broker-request-id: 15f4f87e-200a-4b1a-b76c-1c4b6597c2e1, operation: bind</pre>
<b>Cause</b>	This might be due to authentication or network errors.
<b>Solution</b>	<p>To find out the exact issue with the binding process:</p> <ol style="list-style-type: none"> <li>1. <a href="#">Access the service broker logs.</a></li> <li>2. Search the logs for the <code>broker-request-id</code> string listed in the error message above.</li> <li>3. Check for: <ul style="list-style-type: none"> <li>◆ <a href="#">Authentication errors</a></li> <li>◆ <a href="#">Network errors</a></li> </ul> </li> <li>4. Contact Support for further assistance if you are unable to resolve the problem.</li> </ol>

## Cannot Connect to a Service Instance

<b>Symptom</b>	Developers report that their app cannot use service instances that they have successfully created and bound.
<b>Cause</b>	The error might originate from the service or be network related.
<b>Solution</b>	<p>To solve this issue, ask the user to send application logs that show the connection error. If the error originates from the service, then follow RabbitMQ-specific instructions. If the issue appears to be network-related, then:</p> <ol style="list-style-type: none"> <li>1. Check that <a href="#">application security groups</a> are configured correctly. Access should be configured for the service network that the tile is deployed to.</li> <li>2. Ensure that the network the PAS tile is deployed to has network access to the service network. You can find the network definition for this service network in the BOSH Director tile.</li> <li>3. In Ops Manager go into the service tile and see the service network that is configured in the networks tab.</li> <li>4. In Ops Manager go into the PAS tile and see the network it is assigned to. Make sure that these networks can access each other.</li> </ol>

### Cannot Update a Service Instance

<b>Symptom</b>	<p>If developers report errors such as the following when trying to run <code>cf-update-service</code>:</p> <pre>Server error, status code: 502, error code: 10001, message: Service broker error: Service cannot be updated at this time, please try a gain later or contact your operator for more information.</pre>
<b>Cause</b>	Their service instance might not be running the latest service offering.
<b>Solution</b>	<p>Operators must run the <code>upgrade-all-service-instances</code> errand after upgrading to ensure all existing service instances are upgraded to the latest service offering. See <a href="#">Upgrade All Service Instances</a>.</p> <p>App developers cannot upgrade individual service instances to the latest service offering. They cannot set parameters or change plan until you upgrade their service instances.</p>

### Upgrade All Service Instances Errand Fails

<b>Symptom</b>	The <code>upgrade-all-service-instances</code> errand fails.
<b>Cause</b>	There might be a problem with a particular instance.
<b>Solution</b>	<p>To troubleshoot:</p> <ol style="list-style-type: none"> <li>1. Look at the errand output in the Ops Manager log.</li> <li>2. If an instance has failed to upgrade, debug and fix it before running the errand again to prevent any failure issues from spreading to other on-demand instances.</li> <li>3. After the Ops Manager log no longer lists the deployment as <code>failing</code>, <a href="#">re-run the errand</a> to upgrade the rest of the instances.</li> </ol>

## Missing Logs and Metrics

<b>Symptom</b>	No logs are being emitted by the on-demand broker.
<b>Cause</b>	Syslog might not be configured correctly, or you might have network access issues.
<b>Solution</b>	<p>To troubleshoot:</p> <ol style="list-style-type: none"> <li>1. Ensure you have configured syslog for the tile.</li> <li>2. Check that your syslog forwarding address is correct in Ops Manager.</li> <li>3. 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 <a href="#">public ip</a> VM extension feature available in your Ops Manager tile configuration settings.</li> <li>4. Verify that Loggregator is emitting metrics: <ol style="list-style-type: none"> <li>1. Install the <code>cf log-stream</code> plugin. For instructions, see the <a href="#">Log Stream CLI Plugin</a> GitHub repository.</li> <li>2. Find the GUID for your service instance by running: <pre>cf service SERVICE-INSTANCE --guid</pre> </li> <li>3. Find logs from your service instance by running: <pre>cf log-stream   grep "SERVICE-GUID"</pre> </li> <li>4. If no metrics appear within five minutes, verify that the broker network has access to the Loggregator system on all required ports.</li> </ol> </li> <li>5. If you are unable to resolve the issue, contact <a href="#">Support</a>.</li> </ol>

## RabbitMQ-Specific Errors

The following troubleshooting errors are specific to RabbitMQ:

### Failed Deployment on Upgrade or after Apply Changes

<b>Symptom</b>	Your deployment fails after editing the <b>Assign AZs and Networks</b> pane in the RabbitMQ tile.
<b>Cause</b>	This error might occur if you change the IP addresses assigned to the <code>RabbitMQ Server</code> job. RabbitMQ requires that you do not change these IP addresses after they are assigned. This includes changes made to your current installation or during an upgrade.
<b>Solution</b>	To diagnose and solve this issue, see <a href="#">Changing Network or IP Addresses Results in a Failed Deployment</a> .

### Pre-Stop Script Times Out When Waiting for Queue Synchronization



<b>Symptom</b>	<p>A pre-stop script times out with the error message:</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <pre>Timed out waiting for mirror queue critical node to sync after 3600 seconds</pre> </div>
<b>Cause</b>	<p>You have not manually synced your queues, but you selected the <b>Wait for Queue Synchronization</b> checkbox and have mirrored queues with the default policy setting <code>ha-sync-mode: manual</code>.</p>
<b>Solution</b>	<p>To manually sync your queues, run <code>rabbitmqctl sync_queue</code>. To set <code>ha-sync-mode</code> to <code>automatic</code> instead, see <a href="#">Setting or Changing the Policy</a>.</p>

## Troubleshoot Components

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

### BOSH Problems

#### 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 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

The VM or Disk type that you configured in the plan page of the tile in Ops Manager might not be large enough for the RabbitMQ 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 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 Pivotal Application Service tile.

## Networking

Common issues with networking include:

Issue	Solution
Latency when connecting to the RabbitMQ service instance to create or delete a binding.	Try again or improve network performance.
Firewall rules are blocking connections from the RabbitMQ service broker to the service instance.	Open the RabbitMQ tile in Ops Manager and check the two networks configured in the <b>Networks</b> pane. Ensure that these networks allow access to each other.
Firewall rules are blocking connections from the service network to the BOSH director network.	Ensure that service instances can access the Director so that the BOSH agents can report in.
Apps cannot access the service network.	Configure Cloud Foundry application security groups to allow runtime access to the service network.
Problems accessing BOSH's UAA or the BOSH director.	Follow network troubleshooting and check that the BOSH director is online

### Validate Service Broker Connectivity to Service Instances

To validate connectivity, do the following:

1. View the BOSH deployment name for your service broker by running:

```
bosh deployments
```

2. SSH into the RabbitMQ service broker by running:

```
bosh -d DEPLOYMENT-NAME ssh
```

3. 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 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 in to 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 in to 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 deployment:

1. Inspect the VMs by running:

```
bosh -d service-instance_GUID vms --vitals
```

2. For additional information, run:

```
bosh -d service-instance_GUID instances --ps --vitals
```

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

This section contains 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 Support when filing a ticket.

The `task-id` field maps to the BOSH task ID. For more information on a failed BOSH task, use the `bosh 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 in to the [cf CLI](#) and the [BOSH CLI](#).

### Access Broker Logs and VMs

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 the following command:

```
bosh deployments
```

2. View VMs in the deployment by running the following command:

```
bosh -d DEPLOYMENT-NAME instances
```

3. SSH onto the VM by running the following command:

```
bosh -d DEPLOYMENT-NAME ssh
```

4. Download the broker logs by running the following command:

```
bosh -d DEPLOYMENT-NAME logs
```

The archive generated by BOSH includes the following logs:

Log Name	Description
broker.std out.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.
bpm.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.stder r.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 following cf CLI command:

```
cf service MY-SERVICE --guid
```

2. To view VMs in the deployment, run the following command:

```
bosh -d service-instance_GUID instances
```

3. To SSH into a VM, run the following command:

```
bosh -d service-instance_GUID ssh
```

4. To download the instance logs, run the following command:

```
bosh -d service-instance_GUID logs
```

## 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 an errand, run the following command:

```
bosh -d DEPLOYMENT-NAME run-errand ERRAND-NAME
```

For example:

```
bosh -d my-deployment run-errand deregister-broker
```

## Register Broker

The `register-broker` errand does the following:

- Registers the service broker with Cloud Controller.
- Enables service access for any plans that are enabled on the tile.
- Disables service access for any plans that are disabled on the tile.
- Does nothing for any plans that are set to manual on the tile.

You should run this errand whenever the broker is re-deployed with new catalog metadata to update the Marketplace.

Plans with disabled service access are only visible to admin Cloud Foundry users. Non-admin Cloud Foundry users, including Org Managers and Space Managers, cannot see these plans.

## 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, run the following command:

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

## Upgrade All Service Instances

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

- Collects all of the service instances that the on-demand broker has registered.
- Issues an upgrade command and deploys the a new manifest to the on-demand broker for each service 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 instances are upgraded.

When you make changes to the plan configuration, the errand upgrades all the RabbitMQ service instances to the latest version of the plan.

If any instance fails to upgrade, the errand fails immediately. This prevents systemic problems from spreading to the rest of your service instances.

## Delete All Service Instances

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



**Note:** Orphan BOSH deployments do not correspond to a known service instance. While rare, orphan deployments can occur. Use the `orphan-deployments` errand to identify them.

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

1. Unbinds all apps from the service instances.
2. Deletes all service instances sequentially. Each service instance deletion includes:
  1. Running any pre-delete errands
  2. Deleting the BOSH deployment of the service instance
  3. Removing any ODB-managed secrets from BOSH CredHub
  4. Checking for instance deletion failure, which results in the errand failing immediately
3. Determines whether any instances have been created while the errand was running. If new instances are detected, the errand returns an error. In this case, VMware recommends running the errand again.



**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, run the following command:

```
bosh -d service-instance_GUID delete-deployment
```

## Detect Orphaned 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, run the following command:

```
bosh -d DEPLOYMENT-NAME run-errand orphan-deployments
```

**If orphan deployments exist**—The errand script does the following:

- 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 and any data is safe to delete.

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 and any data is safe to delete.\n",
  "logs": {
    "blobstore_id": "d830c4bf-8086-4bc2-8c1d-54d3a3c6d88d"
  }
}
```

**If no orphan deployments exist**—The errand script does the following:

- 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**—The errand script does the following:

- 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.

```
bosh delete-deployment service-instance_SERVICE-INSTANCE-GUID
```

## Get Admin Credentials for a Service Instance

To retrieve the admin credentials for a service instance from BOSH CredHub:

- Use the cf CLI to determine the GUID associated with the service instance for which you want to retrieve credentials by running:

```
cf service SERVICE-INSTANCE-NAME --guid
```

For example:

```
$ cf service my-service-instance --guid
12345678-90ab-cdef-1234-567890abcdef
```

If you do not know the name of the service instance, you can list service instances in the space with `cf services`.

- Follow the steps in [Gather Credential and IP Address Information](#) and [Log In to the Ops Manager VM with SSH](#) of *Advanced Troubleshooting with the BOSH CLI* to SSH into the Ops Manager VM.
- From the Ops Manager VM, log in to your BOSH Director with the BOSH CLI. See [Authenticate with the BOSH Director VM](#) in *Advanced Troubleshooting with the BOSH CLI*.
- Find the values for `BOSH_CLIENT` and `BOSH_CLIENT_SECRET`:
  - In the Ops Manager Installation Dashboard, click the **BOSH Director** tile.
  - Click the **Credentials** tab.
  - In the **BOSH Director** section, click the link to the **BOSH Commandline Credentials**.
  - Record the values for `BOSH_CLIENT` and `BOSH_CLIENT_SECRET`.
- Set the API target of the CredHub CLI to your BOSH CredHub server by running:

```
credhub api https://BOSH-DIRECTOR-IP:8844 \
  --ca-cert=/var/tempest/workspaces/default/root_ca_certificate
```

Where `BOSH-DIRECTOR-IP` is the IP address of the BOSH Director VM.

For example:

```
$ credhub api https://10.0.0.5:8844 \
  --ca-cert=/var/tempest/workspaces/default/root_ca_certificate
```

## 6. Log in to CredHub by running:

```
credhub login \
  --client-name=BOSH-CLIENT \
  --client-secret=BOSH-CLIENT-SECRET
```

For example:

```
$ credhub login \
  --client-name=credhub \
  --client-secret=abcdefghijklm123456789
```

## 7. Use the CredHub CLI to retrieve the credentials :

- Retrieve the password for the admin user by running:

```
credhub get -n /p-bosh/service-instance_GUID/admin_password
```

In the output, the password appears under **value**. Record the password.

For example:

```
$ credhub get \
  -n /p-bosh/service-instance_70d30bb6-7f30-441a-a87c-05a5e4afff26/admin_
password

id: d6e5bd10-3b60-4a1a-9e01-c76da688b847
name: /p-bosh/service-instance_70d30bb6-7f30-441a-a87c-05a5e4afff26/adm
in_password
type: password
value: UMF2DXsqNPP1CNWMdVMcNv7RC3Wi10
version_created_at: 2018-04-02T23:16:09Z
```

## 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:

1. Log in as an admin by running:

```
cf login
```

2. Confirm that the Marketplace does not list RabbitMQ by running:

```
cf m
```

3. Log in to BOSH as an admin by running:

```
bosh log-in
```

4. Display your BOSH deployments to confirm that the output does not show the RabbitMQ deployment by running:

```
bosh deployments
```

5. Run the “`delete-all-service-instances`” errand to delete every instance of the service.
6. Run the “`deregister-broker`” errand to delete the service broker.
7. Delete the service broker BOSH deployment by running:

```
bosh delete-deployment BROKER-DEPLOYMENT-NAME
```

8. Reinstall the tile.

## View Resource Saturation and Scaling

To view usage statistics for any service, do the following:

1. Run the following command:

```
bosh -d DEPLOYMENT-NAME vms --vitals
```

2. To view process-level information, run:

```
bosh -d DEPLOYMENT-NAME instances --ps
```

## Identify Apps using a Service Instance

To identify which apps are using a specific service instance from the name of the BOSH deployment:

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:
  1. `cf curl APP-URL` and record the app name under `entity.name`.
  2. `cf curl SPACE-URL` to obtain the space, using the `entity.space_url` from the above curl. Record the space name under `entity.name`.
  3. `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 the 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.

## Drop and Restore AMQP or AMQPS Traffic to a RabbitMQ Service Instance

While debugging a RabbitMQ service instance, you can prevent apps from sending and receiving messages, for example, to decrease the server load. You can use `drop-amqp-traffic` and `restore-amqp-traffic` scripts, which run the necessary `iptables` commands to achieve that.

To stop and then restore traffic to a RabbitMQ service instance, do the following:

1. To stop all AMQP or AMQPS traffic to a RabbitMQ service instance, enter the following command:

```
bosh -d service-instance_GUID ssh rabbitmq-server
"echo y | sudo /var/vcap/packages/rabbitmq-admin/bin/drop-amqp-traffic"
```

2. After performing the troubleshooting steps, restore the traffic. To do this, enter the following command:

```
bosh -d service-instance_GUID ssh rabbitmq-server
"echo y | sudo /var/vcap/packages/rabbitmq-admin/bin/restore-amqp-traffic"
```

Alternatively, you can run these scripts on individual nodes:

1. `bosh ssh` to a `rabbitmq-server` instance.
2. `sudo -s` to gain root privileges.
3. Execute `drop-amqp-traffic` to drop all AMQP or AMQPS traffic to this instance, or `restore-amqp-traffic` to start accepting traffic again.

## Frequently Asked Questions

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

Ensure that all nodes in the cluster are healthy from the RabbitMQ Management UI, or health metrics

exposed through Firehose. You cannot rely solely on the BOSH `instances` output as that reflects the state of the Erlang VM used by RabbitMQ and not the RabbitMQ app.

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

Only BOSH commands should be used by the operator to interact with the RabbitMQ app.

For example:

```
bosh stop rabbitmq-server and bosh 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 the stop command and specifying `JOB [index]`.



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

## What happens when I run `bosh stop rabbitmq-server`?

BOSH starts the shutdown sequence from the bootstrap instance.

We start by telling the RabbitMQ app to shutdown and then shutdown the Erlang VM within which it is running. If this succeeds, we run the following checks to ensure that the RabbitMQ app and Erlang VM have stopped:

1. 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. Notice that we are tracking the Erlang PID and not the RabbitMQ PID.
2. Check that `rabbitmqctl` does not return an Erlang VM PID.

Once this completes on the bootstrap instance, BOSH continues the same sequence on the next instance. All remaining `rabbitmq-server` instances stop one by one.

## What happens when `bosh stop rabbitmq-server` fails?

If the 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 `bosh stop rabbitmq-server` fails?

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

First, BOSH `ssh` into the failing `rabbitmq-server` instance and start the `rabbitmq-server` job by running `monit start rabbitmq-server`. You will not be able to start the job with BOSH `start` as this always runs the drain script first and will fail as the drain script is failing.

Once `rabbitmq-server` job is running (confirm this with `monit status`), run `DEBUG=1`

`/var/vcap/jobs/rabbitmq-server/bin/drain`. This tells you exactly why it is failing.

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

It is possible to back up the state of a RabbitMQ cluster for both the on-demand and pre-provisioned services using the RabbitMQ Management API. Backups include virtual hosts, 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:

- **For the backup:**

```
curl -u "$USERNAME:$PASSWORD" "http://$RABBIT-ADDRESS:15672/api/definitions"
-o "$BACKUP-FOLDER/rabbit-backup.json"
```

- **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 Ops 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 display as green, showing they are healthy.
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.

## Knowledge Base (Community)

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

## File a Support Ticket

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

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

[Create a pull request or raise an issue on the source for this page in GitHub](#)

# Operator Guide: Pre- Provisioned

## Turning Off the Pre-Provisioned Service



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic provides instructions on how to turn off the pre-provisioned service. If you are not planning to use the pre-provisioned service, disable it. Disabling the pre-provisioned service broker means that no instances for the pre-provisioned service will be deployed. This, in turn, saves money.



**Note:** Previous deprecation warnings for the pre-provisioned service are no longer in effect. Pivotal continues to support the pre-provisioned service.

## Disable the Pre-Provisioned Service Manually

To turn off the pre-provisioned service manually, do the following:

1. If you are upgrading to RabbitMQ for Pivotal Platform v1.14, or have previously configured the pre-provisioned service, de-register the pre-provisioned service broker by running the following command:

```
bosh -d p-rabbitmq-GUID run-errand broker-deregistrar
```

Where `GUID` is the RabbitMQ deployment GUID.

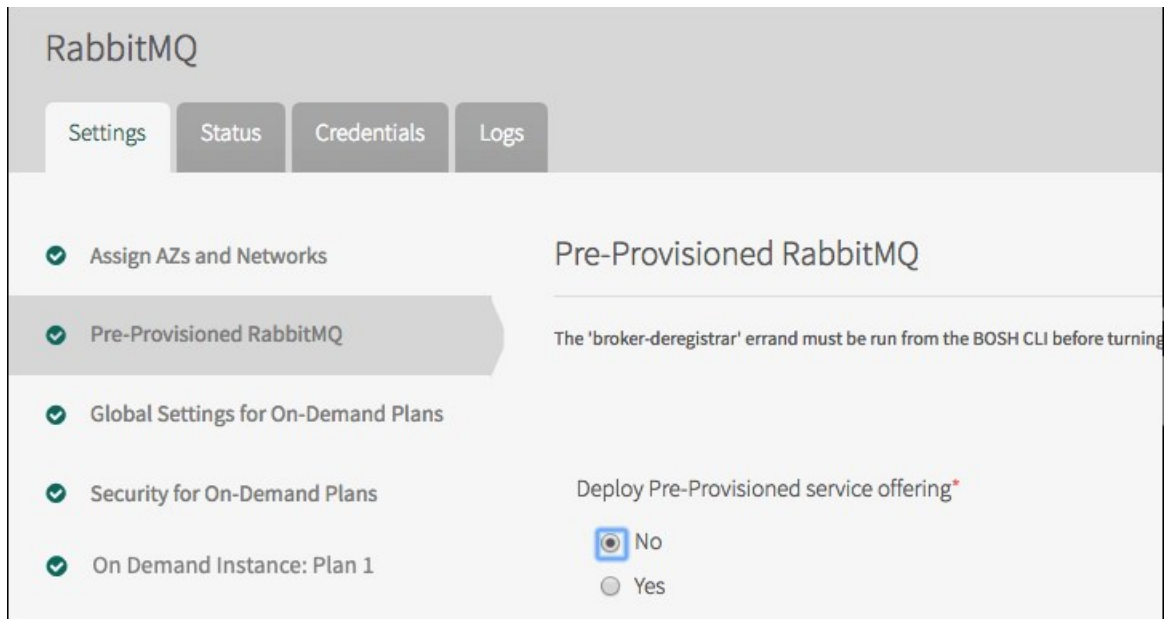
For example:

```
$ bosh -d p-rabbitmq-aeaea3ac-aba5-a6a4-afa7-aba8a0a7a0a9 run-errand broker-deregistrar
```

The broker-deregistrar errand removes the `p-rabbitmq` service offering from the Marketplace.

2. From the Pivotal Operations Manager Installation Dashboard, click the RabbitMQ tile.
3. In the RabbitMQ tile, click **Settings**.
4. Click **Pre-Provisioned RabbitMQ**, and click **No** under the **Deploy Pre-Provisioned service offering**.





5. Click **Save**, and then return to the Ops Manager Installation Dashboard.
6. Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
7. Click **Apply Changes** to redeploy and disable the pre-provisioned service.

## Disable the Pre-Provisioned Service Using Automation

You can turn off the pre-provisioned service if you are using automation such as scripts or pipelines to configure the RabbitMQ tile.

To do this, set the `.properties.multitenant_support` property to `disabled`. For more information about properties, see [Configuring products](#). When you make this change, Ops Manager automatically sets instance counts to `0`.



**WARNING:** For instance groups related to the pre-provisioned service, such as `rabbitmq-server`, `rabbitmq-broker`, `rabbitmq-haproxy`, if you modify instance counts to a non-zero value, BOSH fails to deploy the tile. Although the fields are inactive in the UI, this does not prevent you from overwriting instance counts through the API. Only set instance counts to `0` or do not set them at all.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Installing and Configuring

### Installing and Configuring the Pre-Provisioned Service



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support](#)

**Lifecycle Policy.** To stay up to date with the latest software and security updates, upgrade to a supported version.

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

The RabbitMQ open source product provides additional documentation. For more information about getting started with RabbitMQ and ensuring production readiness, see the Production Checklist in the [RabbitMQ Documentation](#).



**Note:** Previous deprecation warnings for the pre-provisioned service are no longer in effect. Pivotal continues to support the pre-provisioned service.

## Role-Based Access in Pivotal Operations Manager

Ops Manager admins can use Role-Based Access Control (RBAC) to manage which operators can make deployment changes, view credentials, and manage user roles in Ops Manager. Therefore, your role permissions might not allow you to perform every procedure in this operator guide.

For more information about roles in Ops Manager, see [Understand Roles in Ops Manager](#).

## 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. This adds the tile to your staging area.
4. Click the newly added **RabbitMQ** tile. This lets you begin configuring the tile. The installation is complete when you apply the changes from the configuration.

## Configure Pre-Provisioned RabbitMQ

The configuration screen below appears when you click the RabbitMQ tile in Ops Manager. An orange circle beside a tab indicates that you must complete a configuration in the tab. A green check mark indicates that the tab is preconfigured and you are able to change its settings. An orange circle indicates that you are required to configure fields on the tab before you can deploy the tile.

## AZ and Network Assignments

Place singleton jobs in

☒ us-central1-f

☐ us-central1-a

☐ us-central1-c

Balance other jobs in

☒ us-central1-f

☒ us-central1-a

☒ us-central1-c

Network

steelteal-pas-subnet

Service Network

steelteal-services-subnet

**Save**

## Assign AZs and Networks

Follow the steps below to configure the AZs and networks.

1. In the [Settings](#) screen, click **Assign AZs and Networks**.



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

2. Configure the fields on the **Assign AZs and Networks** as follows. **All fields are required**, though some do not apply to the pre-provisioned service.

Required Fields	Instructions

<b>Place singlet on jobs in</b>	Select a region. This selection only affects the on-demand service.
<b>Balance other jobs in</b>	Select additional region(s). This selection only affects the pre-provisioned service.
<b>Network</b>	<p>Select a network for the RabbitMQ 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, described in <a href="#">Default Network and Service Network</a>. Typically, you select the network used for Pivotal Application Service components.</p>
<b>Service Network</b>	Select a network. This selection only affects the on-demand service.



**Warning:** Changing the Network after you have configured it, or changing the IP configuration, results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).

3. Click **Save**.

## Pre-Provisioned RabbitMQ

To configure the following sections in the **Pre-Provisioned RabbitMQ** tab, on the **Settings** screen, click **Pre-Provisioned RabbitMQ** and under **Deploy Pre-Provisioned service offering**, click **Yes**.

## RabbitMQ Admin User Credentials

In the **RabbitMQ admin user credentials** field of the RabbitMQ pane, enter an admin username and password:

You can use a combination of upper or lowercase alphanumerics and supported special characters: – `=_+\";:/?.><,~.`



**Note:** You cannot use back quote ` or single quote '.

This grants you full admin access to the RabbitMQ Management UI.



**Note:** To rotate your admin 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

Choose which plugins you want to enable in this section of the **Pre-Provisioned RabbitMQ** tab.

You must leave the **rabbitmq\_management** plugin enabled for this product to 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

For more information about RabbitMQ plugins, see the [RabbitMQ documentation](#).

## Erlang Cookie

(Optional) Provide an Erlang cookie to be used by the cluster in this section of the **Pre-Provisioned RabbitMQ** tab. This is useful for connecting directly to the RabbitMQ cluster. For example, with `rabbitmqctl` or to connect other VMs running Erlang.

If you leave this field blank, BOSH CredHub generates a secure Erlang cookie.

Erlang cookie used by RabbitMQ nodes and rabbitmqctl

## Erlang Cookie Security Fix

In RabbitMQ v1.15 and earlier, if you left the Erlang cookie field blank, the tile generated the cookie in a way that could be reverse-engineered. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).

In RabbitMQ v1.16, BOSH CredHub securely generates the Erlang cookie when the field is left blank. If you rely on the tile to generate the Erlang cookie by leaving the field blank, do one of the following:

- (Recommended) Leave the Erlang cookie field blank so that BOSH CredHub generates a new cookie. This causes a few minutes of downtime.  
For more information, see [Changing the Erlang Cookie Value Known Issue](#) below.
- Set the cookie explicitly to the current value to avoid regeneration. This avoids downtime, but the Erlang cookie is not secure.  
For how to do this, see [Set the Erlang Cookie to Existing its Value](#) below.

## Changing the Erlang Cookie Value Known Issue

Changing the Erlang cookie value causes a few minutes of downtime because it requires a full cluster shutdown. Pivotal recommends that you do not change anything else during this time, because it is possible for the configuration to be inconsistently applied during this process.

The deployment might fail after this process. If so, redeploying fixes the issue.

## Set the Erlang Cookie to Its Existing Value



**WARNING:** Pivotal recommends that you change the Erlang cookie to prevent being affected by CVE-2018-1279. For more information, see [Erlang Cookie Security Fix](#) above.

If you rely on the tile to generate the Erlang cookie but need to upgrade to RabbitMQ v1.16 without a cluster shutdown, you can set the Erlang cookie to its existing value. To do so:

1. Follow the steps in the following link, up to and including the section *Log in to the BOSH Director*: [Advanced Troubleshooting with the BOSH CLI](#).
2. Run:  
`bosh ssh rabbitmq-server/0`
3. Run these 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 **Pre-Provisioned RabbitMQ** tab. This field is shown in the [Erlang Cookie](#) section 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.
7. Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
8. 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 there is no downtime.

## External Load Balancer

(Optional) Enter a DNS name or IP address of an external load balancer to be returned in the binding credentials (`VCAP_SERVICES`) to app developers. Enter this in this section of the **Pre-Provisioned RabbitMQ** tab:

External load balancer DNS name

If you configure an external load balancer, to avoid an unnecessary VM deployment, in the **Resource Config** tab set the **HAProxy for RabbitMQ** instance count to **0**.

## Enable Custom Policy on New Instances

You can specify a custom RabbitMQ policy in this part of the **Pre-Provisioned RabbitMQ** tab. For information and instructions, see [Setting Default Policies for the RabbitMQ Service](#).

## HAProxy Ports

Enter the ports HAProxy should load balance to the RabbitMQ nodes in this section of the **Pre-Provisioned RabbitMQ** tab:

RabbitMQ cluster HAProxy ports \*

- Enter a comma-separated list of ports that are proxied to RabbitMQ nodes. If you enable other protocol plugins or have a custom configuration that changes the ports that RabbitMQ listens on, you can extend this list. The list defaults to:
  - ✦ 15672 - Management
  - ✦ 5672 - AMQP
  - ✦ 5671 - AMQP+SSL
  - ✦ 1883 - MQTT
  - ✦ 8883 - MQTT+SSL

- ✦ 61613 - STOMP
- ✦ 61614 - STOMP+SSL
- ✦ 15674 - WebSTOMP



**Warning:** The Management UI must listen on port 15672. You must include this port on the list.



**Note:** You do not need to expose port 15671 because the pre-provisioned offering does not support access to the Management UI over TLS.

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

## SSL

(Optional) Provide SSL certificates and keys for use by the RabbitMQ cluster in this section of the **Pre-Provisioned RabbitMQ** tab:

---



## RabbitMQ server certificate

Certificate PEM

Private Key PEM

[Generate RSA Certificate](#)

RabbitMQ server CA certificate(s)

☐ Enable 'verify\_peer' SSL certificate verification (default is 'verify\_none')☐ Require peer certificate validation

SSL certificate verification depth ( min: 1, max: 32 ) \*

5

- SSL is simultaneously provided for AMQPS, STOMP and MQTT. No other plugins are automatically configured for use with SSL.
- If you provide SSL keys and certificates, non-SSL support for AMQP is disabled. If you previously deployed this service without SSL support and have apps connected to the service using AMQP, these apps lose their connections and must reconnect using AMQPS.

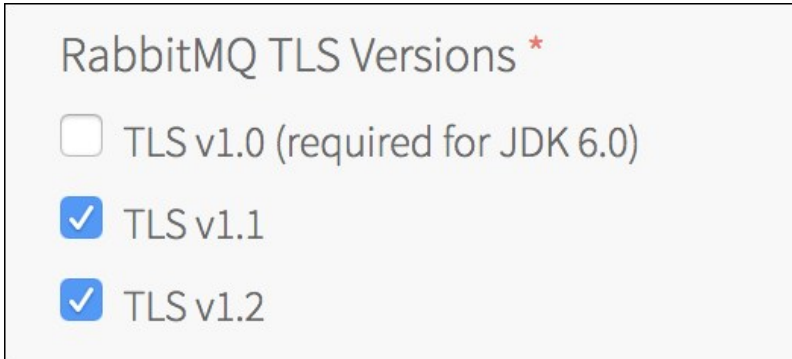
Apps connected to the cluster over MQTT and STOMP continue to function without SSL.

- SSL settings are applied equally across all VMs in the cluster.
- You can provide more than one CA certificate.

For more information about SSL support, see the [RabbitMQ documentation](#).

## TLS Support

Configure TLS in this section of the **Pre-Provisioned RabbitMQ** tab:



RabbitMQ TLS Versions \*

☐ TLS v1.0 (required for JDK 6.0)


☒ TLS v1.1

☒ TLS v1.2

- TLS v1.0 is disabled by default, due to security issues.
- TLS v1.1 and v1.2 are enabled by default and you can turn them on and off.

## RabbitMQ Configuration

(Optional) Provide a full `rabbitmq.config` file by pasting its contents in the **RabbitMQ configuration** field in the **Pre-Provisioned RabbitMQ** tab. This `rabbitmq.config` file is then provided to all the nodes in the cluster.



RabbitMQ configuration

RabbitMQ config file contents, can be blank

The input in this field must be Base64 encoded.

For example, suppose you want to configure the `rates_mode` of the `rabbitmq_management` stats below:

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

1. Encode the file into Base64:

```
WwogIHtyYWJiaXRtcV9tYW5hZ2VtZW50LCBbCiAgICB7cmF0ZXNfbW9kZSwgZGV0YWlsZWR9CiAgXX0KXS4K
```

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

For more information about the RabbitMQ configuration, see the [RabbitMQ documentation](#).

## Select the Network Partition Behavior of the RabbitMQ Cluster

In this part of the **Pre-Provisioned RabbitMQ** tab, you can select one of two behaviors to occur in the event of a network partition.

1. In the **Select the network partition behavior of the RabbitMQ cluster** field, choose the desired behavior: **pause\_minority** or **autoheal**.

For more information about these options and on RabbitMQ clusters and network partitions, see the [RabbitMQ documentation](#).

For production purposes, Pivotal recommends that customers have at least three RabbitMQ server nodes and two HA Proxies spread across low latency AZs.

## Disk Free Alarm Limit

Choose how much disk space RabbitMQ attempts to keep free at any given time in this section of the **Pre-Provisioned RabbitMQ** tab:

RabbitMQ periodically checks if there is sufficient free space on disk. If there is not, RabbitMQ temporarily stops accepting new messages. This gives your apps time to consume existing messages, and thus free up some disk space. The RabbitMQ tile provides four options for this value:

- **50MB** is the minimum value. (Not Recommended)  
Selecting **50MB** is discouraged because it can cause data loss. For more information, see [Dangers of Setting This Value Too Low](#) and [When to Use the 50MB Value](#) below.
- **100% Memory** ensures that at the time when RabbitMQ checks the available disk, there must be enough space for RabbitMQ 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, RabbitMQ might:
  - ✦ 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 the above situations, RabbitMQ might require more free disk than it has memory.

- **200% Memory** is a conservative value used when the operator wants higher confidence that RabbitMQ never runs out of disk space.

For more information about disk alarms, see the [RabbitMQ documentation](#).

### Dangers of Setting This Value Too Low

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

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

### 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 RabbitMQ is not able to free up enough disk space to accept new messages. 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 choose.

### When to Use the 50MB Value

Pivotal discourages using this value in production. However, if you are experimenting with a development environment you might want to use a small disk to keep down costs, though this increases the possibility of RabbitMQ crashing and losing data.

## Specify Static IP Addresses

To specify static IP addresses:

1. In the **Pre-Provisioned RabbitMQ** tab, configure the following fields:
  - ✦ **Pre-Provisioned HAProxy Static IPs:** Enter a comma-delimited list of IP addresses grouped in the order of AZs configured. These IP addresses are assigned to your Pre-Provisioned HAProxy nodes.
  - ✦ **Pre-Provisioned RabbitMQ Server Static IPs:** Enter a comma-delimited list of IP addresses grouped in the order of AZs configured. These IP addresses are assigned to your Pre-Provisioned RabbitMQ Server nodes.
  - ✦ **Pre-Provisioned Service Broker Static IP:** Enter an IP address. This address must be

in the network range of the network name specified in the **Network** dropdown in the **Assign AZs and Networks** tab and must not be in the **Service Network**. This IP address is assigned to your Pre-Provisioned Service Broker node.



**Note:** If any of the above fields are left blank, BOSH allocates an IP address.

Pre-Provisioned HAProxy Static IPs

Pre-Provisioned RabbitMQ Server Static IPs

Pre-Provisioned Service Broker Static IP

2. Click **Save**.

## Configure Syslog Forwarding and Metrics Polling Interval

To enable monitoring for RabbitMQ, operators forward the syslog by designating an external syslog endpoint for RabbitMQ component log messages. This endpoint serves as the input to a monitoring platform such as Datadog, Papertrail, or SumoLogic.

The metrics polling interval determines how often metrics are collected.

To specify the destination for RabbitMQ log messages:

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

## Syslog and Metrics settings for both Pre-Provisioned and On-Demand service

Metrics polling interval \*

Do you want to configure log forwarding to a syslog server? \*

☐ No

☒ Yes

Syslog address \*

Syslog port \*

Transport protocol \*

☐ Enable TLS

Permitted Peer

Custom CA Certificate

**Save**

- Click **Syslog and Metrics**.
- Configure the fields on the **Syslog** pane as follows:

Field	Description
-------	-------------

<b>Metrics polling interval</b>	The default setting is 30 seconds for all deployed components. Pivotal recommends that you do not change this interval. To avoid overwhelming components, do not set this below 10 seconds. Set this to -1 to disable Metrics. Changing this setting affects all deployed instances.
---------------------------------	--

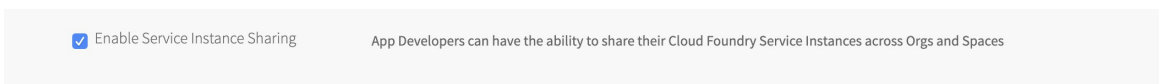
<b>Syslog address</b>	IP or DNS address of the syslog server
<b>Syslog port</b>	Port of the syslog server
<b>Transport protocol</b>	Transport protocol of the syslog server – one of <code>udp</code> , <code>tcp</code> , or <code>relp</code> .
<b>Enable TLS</b>	Enable TLS to the syslog server.
<b>Permitted Peer</b>	If there are several peer servers that can respond to remote syslog connections, then you can provide a wildcard in the domain, such as <code>*.example.com</code> .
<b>Custom CA Certificate</b>	If the server certificate is not signed by a known authority, for example, an internal syslog server, provide the CA certificate of the log management service endpoint.

- Click **Save**.
- Return to the Ops Manager Installation Dashboard.
- Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
- Click **Apply Changes** to redeploy with the changes.

## Global Settings

Follow the steps below to enable the shareable instances feature. Sharing a service instance between spaces, allows apps in different spaces to share databases, messaging queues, and many other types of services. For more information, see [Sharing Service Instances](#).

- Click **Pre-Provisioned RabbitMQ**.
- Select **Enable Service Instance Sharing**



- Click **Save**.

## Dedicated Instance: Single Node Plan

This tab only applies to the on-demand service. However, you must complete the fields on this tab even if you are not using the on-demand service. Therefore, if you are not using the on-demand service:

- Select or enter any values in the required fields, select the **Acknowledge** checkbox, and click **Save**.

For information about configuring the on-demand service, see [Installing and Configuring the On-](#)

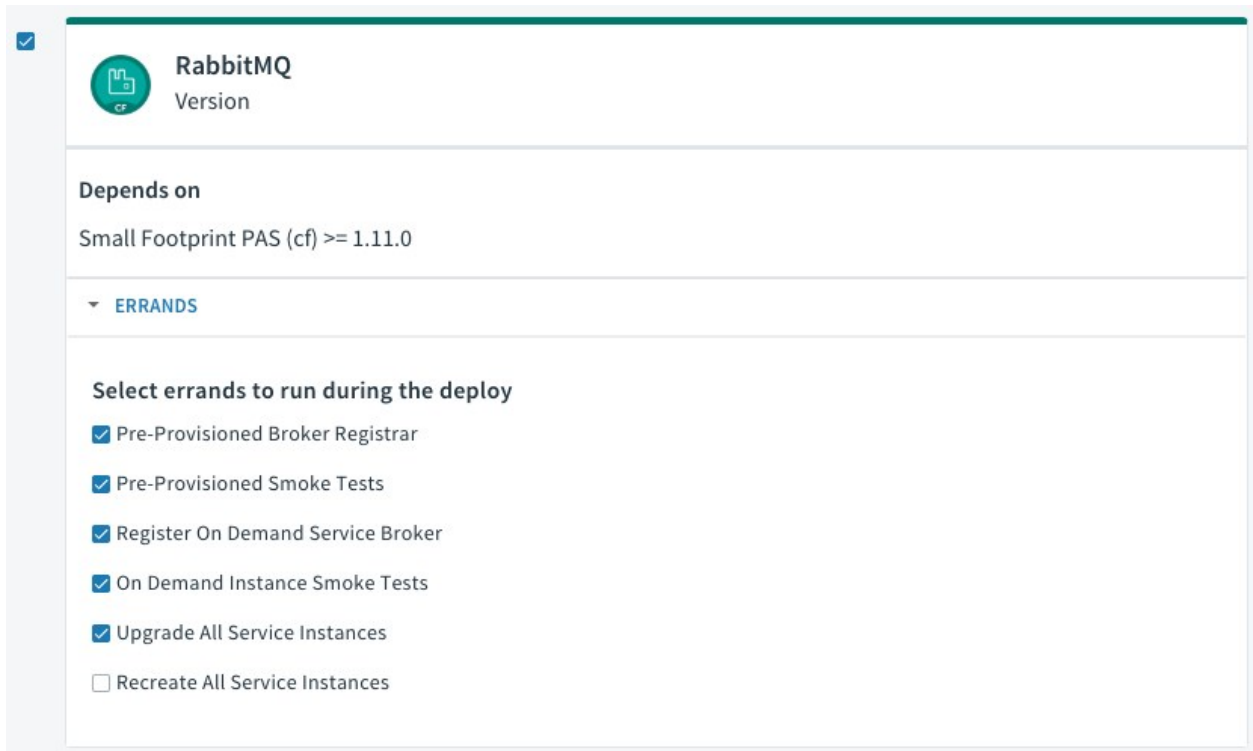
[Demand Service](#).

## Errands

(Optional) In the **Errands** tab, choose the defaults for when errands run.

You can think of errands as tasks. For example, when deploying or updating RabbitMQ, Ops Manager can optionally run a series of post-deploy errands. For more information, see the [post-deploy errands](#) section below. An example is the **Smoke Tests** errand, which checks the health of the RabbitMQ cluster after a deploy or upgrade.

You can toggle errands on and off on the Ops Manager **Review Pending Changes** page.



☒ **RabbitMQ**  
Version

**Depends on**  
Small Footprint PAS (cf) >= 1.11.0

▼ **ERRANDS**

**Select errands to run during the deploy**

- ☒ Pre-Provisioned Broker Registrar
- ☒ Pre-Provisioned Smoke Tests
- ☒ Register On Demand Service Broker
- ☒ On Demand Instance Smoke Tests
- ☒ Upgrade All Service Instances
- ☐ Recreate All Service Instances

This is a one-time action before an update. You can change these defaults by clicking **Errands** in the RabbitMQ **Settings** tab as well as the defaults for [pre-delete](#) errands.



**Warning:** In RabbitMQ v1.9.0 and later, post-deploy errands are on by default except the Recreate All Service Instances errand. Pivotal recommends keeping these defaults, because the smoke tests can encounter unexpected issues, and on-demand instances of RabbitMQ might fall behind if the Upgrade All Service Instances errand is not on by default.

For more information about errand run rules, see [Errand Run Rules](#).

## Post-Deploy Errands

Errand

Description



<b>Pre-Provisioned Broker Registrar</b>	Makes the pre-provisioned RabbitMQ service plans available in the Marketplace
<b>Pre-Provisioned Smoke Tests</b>	Checks that a pre-provisioned RabbitMQ service instance can be bound to a PAS app, and that the app can publish and subscribe to a RabbitMQ cluster. See <a href="#">Smoke Tests</a> .
<b>Register On Demand Service Broker</b>	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.
<b>On Demand Instance Smoke Tests</b>	Checks that on-demand RabbitMQ service instances can be bound to a PAS app, and that the app can publish and subscribe to a RabbitMQ cluster. See <a href="#">Smoke Tests</a> .
<b>Upgrade All Service Instances</b>	On-Demand instances are updated and redeployed if there are changes to the <b>Dedicated Instance</b> settings or the tile is upgraded. If this errand is set to <b>Off</b> , updates to <b>Dedicated Instance</b> settings are not applied to existing service instances. <b>Pivotal recommends that this errand is configured to always run.</b>
<b>Recreate All Service Instances</b>	This errand re-creates all On-Demand instance VMs managed by the On-Demand broker. It is useful for tasks that require re-creating the service instance VM, such as rotating the Ops Manager root certificate authority (CA) or fully restoring the platform during disaster recovery or migration. <b>This errand is off by default and should be enabled only when you want to re-create a VM.</b>

## Pre-Delete Errands

Pre-delete errands run after an operator chooses to delete a product in the Ops Manager Installation Dashboard, but before Ops Manager finishes deleting the product.

Errand	Description
<b>Deregister and Purge Instances</b>	Removes the pre-provisioned RabbitMQ service from the Marketplace and deletes all associated service instances and bindings. For more information, see <a href="#">Turning Off the Pre-Provisioned Service</a> .
<b>Delete All Service Instances</b>	Unbinds and deletes existing dedicated service instances. The duration of this errand depends on the number of deployed on-demand instances.
<b>Deregister On-Demand Service Broker</b>	Removes the on-demand RabbitMQ service from the Marketplace

## Stemcell

To verify that you have the correct stemcell, follow the procedure in [Importing and Managing Stemcells](#).

## Apply Configuration and Complete the Installation

1. Return to the Ops Manager Installation Dashboard.
2. Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
3. Click **Apply Changes** to complete the installation of RabbitMQ.

## Other Configuration Topics

### 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 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 are present:

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

### Port to protocol mappings

- 15672 = RabbitMQ Management UI
- 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

### 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 in the RabbitMQ tile. Ensure the following ports are enabled for those VMs:

- 15672
- 5672
- 5671

- 1883
- 8883
- 61613
- 61614
- 15674
- 4567
- 3457 - 3459

The following is a template for configuring your PAS 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
HAProxy IPs	5672	tcp	Application containers using AMQP
HAProxy IPs	5671	tcp	Application containers using AMQP over SSL
HAProxy IPs	1883	tcp	Application containers using MQTT
HAProxy IPs	8883	tcp	Application containers using MQTT over SSL
HAProxy IPs	61613	tcp	Application containers using STOMP
HAProxy IPs	61614	tcp	Application containers using STOMP over SSL
HAProxy IPs	61613	tcp	Application containers using Web STOMP

Create an ASG named `rabbitmq-app-containers` with the above configuration and bind it to either:

- The appropriate space
- The `default-running` ASG set if you want to provide access to all started apps Then 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. For example:

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

## Assigned IPs

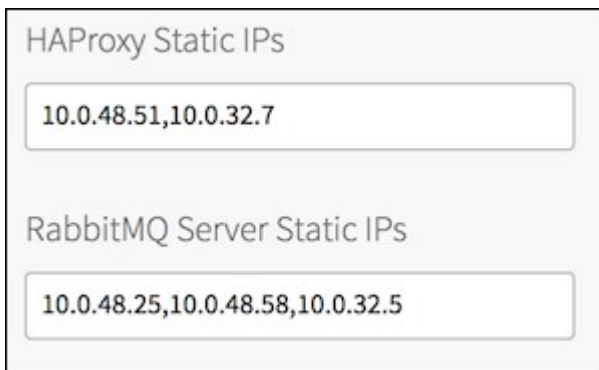
RabbitMQ does not support changing the IP addresses which have been assigned to the RabbitMQ deployments. For example, you cannot change the subnet into which the RabbitMQ cluster was originally provisioned. Doing so causes the deployment to fail. For more information, see [Changing Network or IP Addresses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

## Preserving Dynamically Assigned IPs

You cannot switch from dynamically assigned IP addresses to a different set of static IP addresses. However, you can configure Ops Manager so the current set of dynamically assigned IP addresses always continue to be used. This might be useful when upgrading.

To do this:

1. Go to the **Status** page in the RabbitMQ tile.
2. Take note of the IP addresses for the RabbitMQ Server and HAProxy for RabbitMQ jobs, in the order nodes appear in the UI.
3. Go to the **Settings** page, and click the **Pre-Provisioned RabbitMQ** tab.
4. Enter the IP addresses you got from the **Status** page as a comma-separated list.



The screenshot shows a configuration form with two sections. The first section is titled 'HAProxy Static IPs' and contains a text input field with the value '10.0.48.51,10.0.32.7'. The second section is titled 'RabbitMQ Server Static IPs' and contains a text input field with the value '10.0.48.25,10.0.48.58,10.0.32.5'.

5. Click **Save**.

## RabbitMQ Server Settings that Cannot be Overwritten

In all cases:

- `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]`

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Smoke Tests



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

RabbitMQ for Pivotal Platform runs a set of smoke tests during installation to confirm system health.

## Smoke Test Steps

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

1. Targets the org `system` and creates a space to run the tests.
2. Deploys an instance of the [CF RabbitMQ Example App](#) to this space
3. Creates a RabbitMQ service instance and binds it to the CF RabbitMQ Example App
4. Checks that the CF RabbitMQ Example App can write to and read from the RabbitMQ service instance
5. Cleans up all deployed application and all its service bindings. Finally, the cf space is deleted.



**Note:** Smoke tests fail unless you enable global default application security groups (ASGs). You can enable global default ASGs by binding the ASG to the `system` org without specifying a space. To enable global default ASGs, use `cf bind-running-security-group`.

## Troubleshooting

If errors occur while the smoke tests run, they are summarized at the end of the errand log output. Detailed logs can be found where the failure occurs.

When encountering an error when running smoke tests, it can be helpful to search the log for other

instances of the error summary printed at the end of the tests, for example, `Failed to target Cloud Foundry`. Lookout for `TIP: ...` in the logs next to any error output for further troubleshooting hints.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Monitoring and KPIs for Pre- Provisioned RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

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

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

See [Overview of Logging and Metrics](#) for general information about logging and metrics in Pivotal Application Service.

## Setting up Syslog Forwarding

Operators can enable log forwarding by configuring an external syslog endpoint for RabbitMQ component log messages. For instructions on setting up syslog forwarding, see [Configure Syslog Forwarding and Metrics Polling Interval](#). If syslog forwarding is enabled, log entries with timestamps can also be found locally in `/var/log/messages`. In any case, logs are available under `/var/vcap/sys/log/`.

## Logging Formats

With pre-provisioned RabbitMQ 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 Pivotal Operations 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.

For users familiar with documentation for previous versions of the tile, the tag formerly called 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=director.be5a66bb-a9b4-459f-a0d3-1fc5c9c3ed79.be148cc6-91ef-4eed-a788-237b0b8c63b7 src=10.254.50.4 spt=4222 shost=5ae233e0-ecc5-4868-9ae0-f9767571251b
<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_ssh/bosh_ly0d2rbjr, shell=/bin/bash
<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 might be presented in a different format.

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

Tag	Description
<b>PRI</b>	This is a value which in future will be used to describe the severity of the log entry and which facility it came from.
<b>TIMESTAMP</b>	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 entry was generated.
<b>IP_ADDRESS</b>	The internal IP address of server on which the log entry originated
<b>PROGRAM_NAME</b>	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.
<b>NAME</b>	The BOSH instance group name (for example, <code>rabbitmq_server</code> )
<b>JOB_INDEX</b>	BOSH job index. Used to distinguish between multiple instances of the same job.
<b>JOB_ID</b>	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.
<b>MESSAGE</b>	The log entry that appears

## RabbitMQ Program Names

The following table lists the program names used in the logs:

Program Name	Description
<code>rabbitmq_server_cluster_check</code>	Checks that the RabbitMQ cluster is healthy. Runs after every deploy.

Program Name	Description
<code>rabbitmq_server_node_check</code>	Checks that the RabbitMQ node is healthy. Runs after every deploy.
<code>rabbitmq_route_registrar_stderr</code>	Registers the route for the management API with the Gorouter in your Pivotal Application Service deployment.
<code>rabbitmq_route_registrar_stdout</code>	Registers the route for the management API with the Gorouter in your Pivotal Application Service deployment.
<code>rabbitmq_server</code>	The Erlang VM and RabbitMQ apps. <i>Logs can span multiple lines.</i>
<code>rabbitmq_server_drain</code>	Shuts down the Erlang VM and RabbitMQ apps. Runs as part of the BOSH lifecycle.
<code>rabbitmq_server_http_api_access</code>	Access to the RabbitMQ Management UI.
<code>rabbitmq_server_init</code>	Starts the Erlang VM and RabbitMQ.
<code>rabbitmq_server_post_deploy_stderr</code>	Runs the node check and cluster check. Runs after every deploy.
<code>rabbitmq_server_post_deploy_stdout</code>	Runs the node check and cluster check. Runs after every deploy.
<code>rabbitmq_server_pre_start</code>	Runs before the rabbitmq-server job is started.
<code>rabbitmq_server_sasl</code>	Supervisor, progress, and crash reporting for the Erlang VM and RabbitMQ apps.
<code>rabbitmq_server_shutdown_stderr</code>	Stops the RabbitMQ app and Erlang VM.
<code>rabbitmq_server_shutdown_stdout</code>	Stops the RabbitMQ app and Erlang VM.
<code>rabbitmq_server_startup_stderr</code>	Starts the RabbitMQ app and Erlang VM, then configures users and permissions.
<code>rabbitmq_server_startup_stdout</code>	Starts the RabbitMQ app and Erlang VM, then configures users and permissions.
<code>rabbitmq_server_upgrade</code>	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. The **metrics polling interval** is a configuration option on the RabbitMQ tile (**Settings** > **RabbitMQ**). Setting this interval to -1 disables metrics. 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/memory" value:1024 unit:"MB">
```

## 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/reachable_nodes" value:3 unit:"count">
```

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) because 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

See [Clustering and Network Partitions](#) in the RabbitMQ documentation to learn how to recover from a network partition.

## Key Performance Indicators

Key Performance Indicators (KPIs) for RabbitMQ 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 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 raw component metrics, see [Component Metrics Reference](#).

## Component Heartbeats

Key RabbitMQ 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 the following command:</p> <pre>bosh -d service-instance_GUID vms</pre>

## HAProxy Heartbeat

### p-rabbitmq.haproxy.heartbeat

<b>Description</b>	<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>
<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 HAProxy logs for errors. You can find the VM by targeting your RabbitMQ deployment with BOSH and running the following command, which lists <code>HAProxy_GUID</code>:</p> <pre>bosh -d service-instance_GUID vms</pre>

## Server Heartbeat

### p-rabbitmq.rabbitmq.heartbeat


<b>Description</b>	<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>
<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 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> <pre>bosh -d service-instance_GUID vms</pre>

## 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 might lose the ability to perform disk I/O, which can cause data loss.</p> <div>  <p><b>Note:</b> This assumes non-persistent messages are handled by retries or some other logic by the producers.</p> </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; 250000  <b>Red critical:</b> &gt; 280000</p>
<b>Recommended response</b>	<p>The default <code>ulimit</code> for RabbitMQ is 300000. 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>Reduce the load on the server</li> </ul>

### Erlang Processes

**p-rabbitmq.rabbitmq.erlang.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</p> <p><b>Type:</b> count</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; 900000</p> <p><b>Red critical:</b> &gt; 950000</p>
<b>Recommended response</b>	The default Erlang process limit in RabbitMQ 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.

## BOSH System Health Metrics

The BOSH layer that underlies Ops Manager generates [healthmonitor](#) metrics for all VMs in the deployment. As of Ops Manager v2.0, these metrics are included in the Loggregator Firehose by default. For more information, see [BOSH System Metrics Available in Loggregator Firehose](#) in *Pivotal Application Service Release Notes*.

All BOSH-deployed components generate the system health metrics below. These component metrics are from RabbitMQ components, and serve as KPIs for the RabbitMQ service.

### RAM

**system.mem.percent**

<b>Description</b>	<p>RAM being consumed by the <a href="#">p-rabbitmq</a> 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> 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; 40</p> <p><b>Red critical:</b> &gt; 50</p>
<b>Recommended response</b>	Add more consumers to drain the queue as fast as possible.

## CPU

### system.cpu.user

<b>Description</b>	<p>CPU being consumed by user processes on the <code>p-rabbitmq</code> VM.</p> <p><b>Use:</b> A node that experiences context switching or high CPU usage becomes 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> BOSH HM  <b>Type:</b> percent  <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  <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.

## Ephemeral Disk

### system.disk.ephemeral.percent

<b>Description</b>	<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> BOSH HM  <b>Type:</b> percent  <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  <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.

## Persistent Disk

**system.disk.persistent.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> BOSH HM  <b>Type:</b> percent  <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  <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 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 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 went off
<code>/p-rabbitmq/rabbitmq/system/disk_free_alarm</code>	boolean	Indicates if the disk free alarm went off
<code>/p-rabbitmq/rabbitmq/system/disk_free</code>	MB	The disk space available on the node
<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

/p-rabbitmq/rabbitmq/messages/delivered	count	The total number of messages with the status <code>deliver_get</code> on the node
/p-rabbitmq/rabbitmq/messages/delivered_noack	count	The number of messages with the status <code>deliver_noack</code> on the node
/p-rabbitmq/rabbitmq/messages/delivered_rate	rate	The rate per second at which messages are being delivered to consumers or clients on the node
/p-rabbitmq/rabbitmq/messages/published	count	The total number of messages with the status <code>publish</code> on the node
/p-rabbitmq/rabbitmq/messages/published_rate	rate	The rate per second at which messages are being published by the node
/p-rabbitmq/rabbitmq/messages/redelivered	count	The total number of messages with the status <code>redeliver</code> on the node
/p-rabbitmq/rabbitmq/messages/redelivered_rate	rate	The rate per second at which messages are getting the status <code>redeliver</code> on the node
/p-rabbitmq/rabbitmq/messages/get_no_ack	count	The number of messages with the status <code>get_no_ack</code> on the node
/p-rabbitmq/rabbitmq/messages/get_no_ack_rate	rate	The rate per second at which messages get the status <code>get_no_ack</code> on the node
/p-rabbitmq/rabbitmq/messages/pending	count	The number of messages with the status <code>messages_unacknowledged</code> on the node
/p-rabbitmq/rabbitmq/messages/depth	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_descriptors	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/vhosts/count	count	The number of vhosts

<code>/p-rabbitmq/rabbitmq/queues/VHOST-NAME/QUEUE-NAME/consumers</code>	count	The number of consumers per virtual host per queue
<code>/p-rabbitmq/rabbitmq/queues/VHOST-NAME/QUEUE-NAME/depth</code>	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 HAProxy components emit the following metrics.

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

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Managing the Pre- Provisioned Service

### Isolating Clusters with the RabbitMQ for Pivotal Platform Replicator



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.



**Note:** Previous deprecation warnings for the pre-provisioned service are no longer in effect. Pivotal continues to support the pre-provisioned service.

## Overview

RabbitMQ for Pivotal Platform Replicator is a tool that allows you to install multiple RabbitMQ tiles in a single Pivotal Operations Manager environment. This lets you run multiple pre-provisioned RabbitMQ clusters that are isolated from each other.

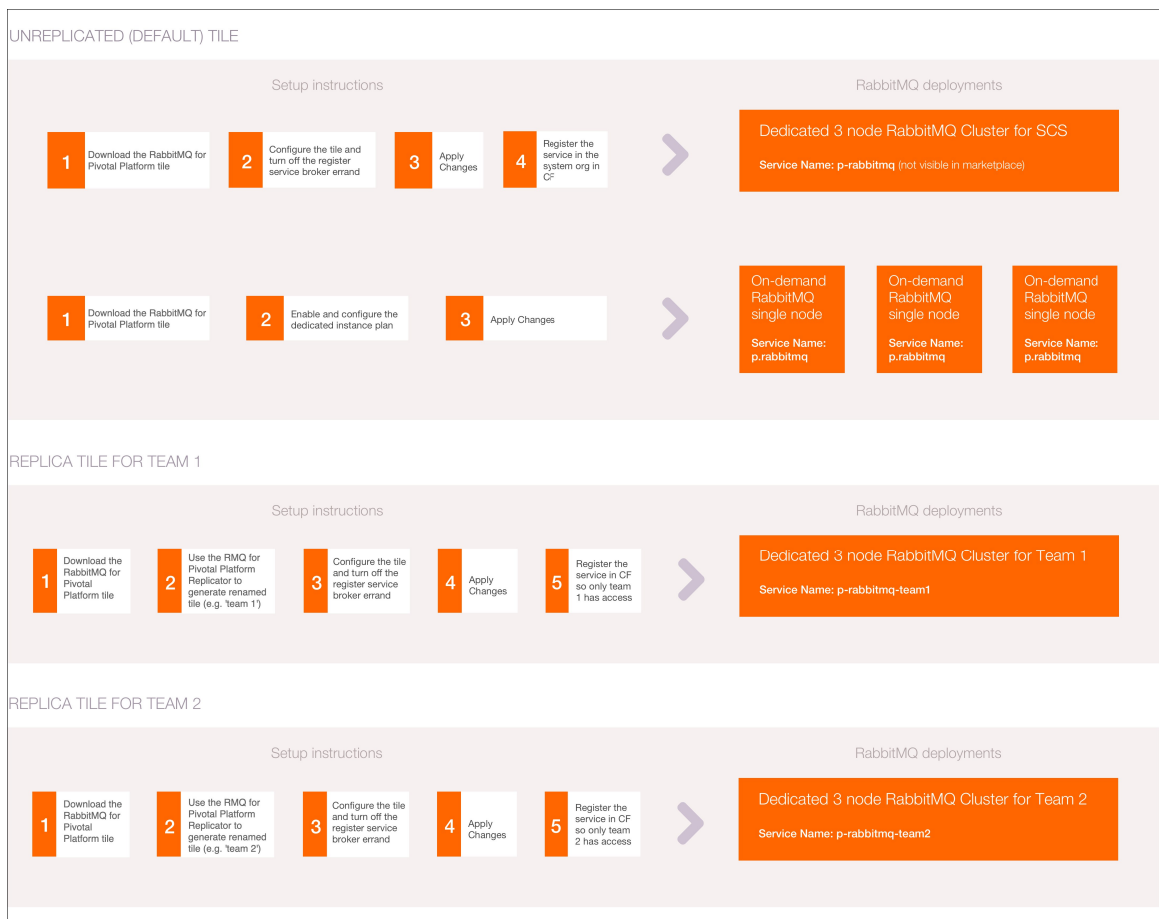


For example, you may want to isolate the cluster serving Spring Cloud Services (SCS) from the cluster serving apps in the Marketplace. Or you may want to give a certain team their own dedicated, pre-provisioned cluster that you manage for them. For information on how to accomplish these scenarios, see [Common Use Cases](#).

## Common Use Cases

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

- The unreplicated RabbitMQ 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.



[Click here to view a larger version of this image.](#)

These scenarios are explained below in [Running SCS on a Dedicated RabbitMQ Cluster](#) and [Providing a Pre-Provisioned Dedicated Cluster](#).

## 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 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 tile solely for SCS to avoid contention between apps using SCS, and apps using RabbitMQ 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, disable the **Broker Registrar** errand in Ops Manager. This prevents service registration in the Marketplace. For more information, see [Errands](#).

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 wish to offer on-demand service plans, use the unreplicated RabbitMQ 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.

### Prerequisites

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

### Download the Replicator

The RabbitMQ 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 source tile that you downloaded
<code>output</code>	The desired file name and path for the replica tile

## Naming Conventions in Original and Replica Tiles

The table below shows the naming conventions for various components related to the original RabbitMQ tile and to the replica tile.

For the purposes of this example, assume that when you generate a replica tile as shown above, in the `name` field you provide the string `finance\`. Then the attributes for the original and replica tiles are as follows:

Component	Name with Original Tile	Name with Replica Tile
Broker name	p-rabbitmq	p-rabbitmq-finance
Broker URL	pivotal-rabbitmq-broker.YOUR_CF_DOMAIN	pivotal-rabbitmq-broker-finance.YOUR_CF_DOMAIN
Service name	p-rabbitmq	p-rabbitmq-finance
URL for the RabbitMQ Management UI	pivotal-rabbitmq.YOUR_CF_DOMAIN	pivotal-rabbitmq-finance.YOUR_CF_DOMAIN
Tile display name in Ops Manager	RabbitMQ	RabbitMQ (finance)
Tile name used internally by Ops Manager	p-rabbitmq	p-rabbitmq-finance
Metrics/Logging Origin	p-rabbitmq	p-rabbitmq-finance

## 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 as a Pre-Provisioned Service](#). The On-Demand service is not offered on replica tiles.

## Limiting Access to Replica Tiles to Specific Orgs


When you replicate RabbitMQ, the replica tile has the **Broker Registrar** errand set to **On** by default. This field appears in the **Errands** tab in the tile:

## Errands


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

### Post-Deploy Errands

Pre-Provisioned Broker Registrar

Default (On) 

Pre-Provisioned Smoke Tests

Default (On) 

With any tile, if the **Broker Registrar** errand is set to **On**, it runs automatically when you finish installing the tile and causes the tile to be available to all CF orgs.

If you want to limit access to the tile to a specific org, follow these steps:

1. Set the broker registrar errand to **Off**, and apply your changes.
2. Manually register the tile with a specific CF org using the following command. See the above table for `BROKER_NAME`, `BROKER_URL`, and `SERVICE_NAME`:

```
cf create-service-broker BROKER_NAME BROKER_USERNAME BROKER_PASSWORD BROKER_URL
```

3. To give access to the org, use the following command and repeat for each additional org:

```
cf enable-service-access SERVICE_NAME -o ORG_NAME
```

## 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 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 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 and unzip the new RabbitMQ Tile Replicator from [Pivotal Network](#). You must download the version of Tile Replicator that corresponds with the version of the RabbitMQ Tile you want to replicate.
2. Find the plan UUID and Service UUID for the service plan you want to update.  
To do this, you can use the CF API. Run the following command, then search the result for the section about the service plan you want:

```
cf curl /v2/service_plans
```

The result is similar to the following, with some extra properties:

```
{
  "metadata": {
    "guid": "a0a2fae9-e5de-47f5-b2b3-23cf07ff6142",
  },
  "entity": {
    "name": "single-node",
    "service_guid": "3e3a9413-50bd-4c76-a226-8721b0d2b3d7",
  }
}
```

- ✦ The plan UUID is found under `metadata, guid`.
- ✦ The service UUID is found under `entity, service_guid`.

3. Run the replicator command below to create the replica:

```
./rabbitmq-replicator-darwin\
--name NAME-OF-EXISTING-REPLICA \
--path PATH-TO-NEW-TILE \
--plan-uuid PLAN-UUID \
--service-uuid SERVICE-UUID \
--output /output/p-rabbitmq-v2-trading-team.pivotal
```

Where:

- ✦ `NAME-OF-EXISTING-REPLICA` must be the same as the name used for the existing replica. This is **trading-team** in this example.
- ✦ `PATH-TO-NEW-TILE` is the path to the new RabbitMQ v2 tile.
- ✦ `PLAN-UUID` is the plan UUID found in Step 2.
- ✦ `SERVICE-UUID` is the service UUID found in Step 2.

Specifying the parameters `--plan-uuid` and `--service-uuid` ensures the new replica updates the existing service offering and plans.

4. 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.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Setting Default Policies for the RabbitMQ Service



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.



**Note:** Previous deprecation warnings for the pre-provisioned service are no longer in effect. Pivotal continues to support the pre-provisioned service.

## About RabbitMQ Policies

You can set a default queue and an exchange policy in the RabbitMQ for Pivotal Platform tile to be applied to the RabbitMQ cluster. After you deploy the tile, Pivotal recommends that you use the RabbitMQ Management Interface to make configuration changes.

For more information about RabbitMQ policies, see the [RabbitMQ documentation](#).

## Rules for Policies Set in the Tile

The following rules apply to policies set through the RabbitMQ tile:

- A new policy, or an update to a policy, only applies to new instances (vhosts). Existing instances are not affected by the policy.
- The policy can only be deleted manually from the RabbitMQ nodes.
- Policies can be added dynamically using the RabbitMQ Management Interface.
- It is not possible to use pattern matching with policies. Policies will be applied to all queues and exchanges.

For granular policy settings, Pivotal recommends using the RabbitMQ Management UI. Set a `priority number` lower than 50, the default `priority number` applied through the Pivotal Operations Manager configuration.

## An Example Policy: Mirror on Two Nodes

Here is an example policy that ensures messages are mirrored on two nodes:

```
{
```

```

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

```

Operators should consider some of the performance implications of making queues and exchanges highly available. For more information about highly available queues, see the [RabbitMQ documentation](#).

## Best Practice for Syncing Queues

When a queue syncs all its messages, they are loaded into memory. When queues are syncing, they can use as much memory as the total size of all messages. This applies to both nodes—the node where the queue leader runs (from node) and the node where the queue follower runs (to node), but only applies to newly created queue followers.

This behavior is especially relevant when any change affects the deployment, for example: stemcell updates, deployment configuration changes, and network changes. Verify that you have enough memory and disk available to support all messages.

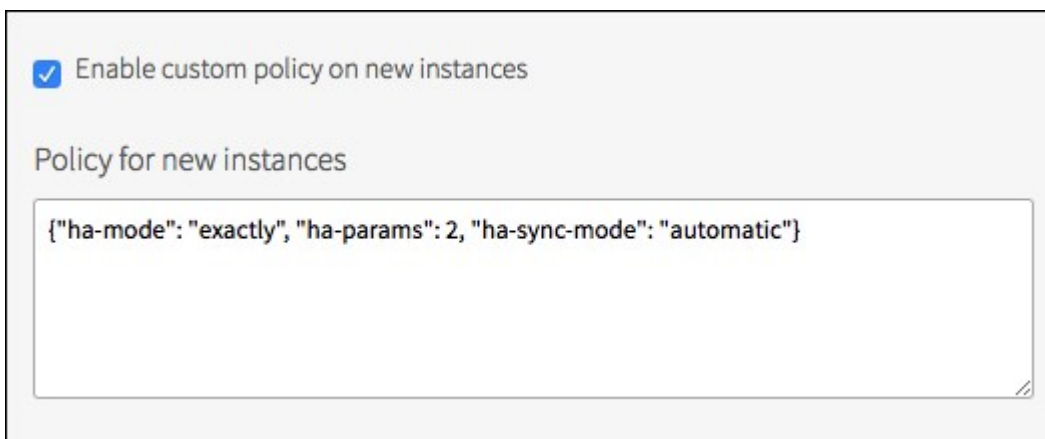
For example:

There are 5 GB of messages in a mirrored queue that is set to automatic sync. When this queue needs to sync, the node where the queue leader runs can use up to 5 GB of extra memory. The same applies to the node where the new queue follower is created.

## Setting or Changing the Policy

To set the RabbitMQ policy, do the following:

1. From the Ops Manager Installation Dashboard, click the RabbitMQ tile and then click the **Pre-Provisioned RabbitMQ** tab.
2. Under **Deploy Pre-Provisioned service offering**, click **Yes**.
3. Select **Enable custom policy on new instances**.



☒ Enable custom policy on new instances

Policy for new instances

```

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

```

4. In the **Policy for new instances** field, paste the policy.

The policy must be valid JSON and should meet valid RabbitMQ policy criteria.



**Note:** No policy validation occurs during the deployment, and errors can cause the

deployment to fail or policies to be applied incorrectly.

## Viewing Policies in the RabbitMQ Management UI

You can view RabbitMQ policies in the RabbitMQ Management UI, shown below. The example policy entered in the RabbitMQ tile above is applied to all queues and given a **Priority** of **50**. This allows you to override it by defining another policy with a higher priority.

### 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

In the **Queues** section shown below, you can see that 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>	D operator_set_policy	running	0	0	0				



**Note:** Developers can obtain the URL of the policy from `VCAP_SERVICES` for app developers.

Create a pull request or raise an issue on the source for this page in [GitHub](#)

## Frequently Asked Questions for Pre-Provisioned RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic lists frequently asked questions that apply to the RabbitMQ for Pivotal Platform pre-



provisioned service.

## Frequently Asked Questions

### 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. You cannot rely solely on the BOSH `instances` output as that reflects the state of the Erlang VM used by RabbitMQ and not the RabbitMQ application.

### What is the correct way to stop and start RabbitMQ for Pivotal Platform?

Only BOSH commands should be used by the operator to interact with the RabbitMQ app.

For example:

```
bosh stop rabbitmq-server and bosh 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 the stop command and specifying `JOB [index]`



**Note:** Do not use `monit stop rabbitmq-server` as this does not call the drain scripts

### What happens when I run `bosh stop rabbitmq-server`?

BOSH starts the shutdown sequence from the bootstrap instance.

We start by telling the RabbitMQ application to shutdown and then shutdown the Erlang VM within which it is running. If this succeeds, we run the following checks to ensure that the RabbitMQ application and Erlang VM have stopped:

1. 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. Notice that we are tracking the Erlang PID and not the RabbitMQ PID.
2. 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-server instances will be stopped one by one.

### What happens when `bosh stop rabbitmq-server` fails?

If the 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 `bosh stop rabbitmq-server` fails?

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.

First, BOSH `ssh` into the failing `rabbitmq-server` instance and start the `rabbitmq-server` job by running `monit start rabbitmq-server`. You will not be able to start the job via BOSH `start` as this always runs the drain script first and will fail since the drain script is failing.

Once `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.

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

It is possible to 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.

## Knowledge Base (Community)

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

## File a Support Ticket

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

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

[Create a pull request or raise an issue on the source for this page in GitHub](#)

# Using the RabbitMQ Management UI



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic describes how to access the RabbitMQ Management UI.

## Access the RabbitMQ Management UI

You can access the management UI for both on-demand and for pre-provisioned service instances. The image below shows an example RabbitMQ Management UI.



Overview Connections Channels Exchanges Queues Admin

## Overview

### Totals

Queued messages (chart: last minute) (?)



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@6031f2ce968ce856e9f684c504f6aec	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	6031f2ce968ce856e9f684c504f6aec	::	5672
amqp	623a32cce88bc994de244cec4e6c6085	::	5672
clustering	6031f2ce968ce856e9f684c504f6aec	::	25672
clustering	623a32cce88bc994de244cec4e6c6085	::	25672

Web contexts

Context	Node	Bound to	Port	SSL	Path
RabbitMQ Management	6031f2ce968ce856e9f684c504f6aec	0.0.0.0	15672	o	/
RabbitMQ Management	623a32cce88bc994de244cec4e6c6085	0.0.0.0	15672	o	/

## Pre-Provisioned Service Instances

To access the RabbitMQ Management UI as the `admin` user, do the following:

- Retrieve your system domain, by doing the following:
  - Navigate to your Pivotal Application Service tile.
  - In the **Domains** tab, record the value in the **System Domain** field.
- Visit <http://pivotal-rabbitmq.SYS-DOMAIN>, where `SYS-DOMAIN` is the system domain you found in step 1.
- The username and password for the Management UI is the username and password you provided in the RabbitMQ configuration in Pivotal Operations Manager.

## On-Demand Service Instances

You can access the RabbitMQ Management UI for an on-demand service instance using Apps Manager or the `cf` CLI. This is done using credentials that only provide access to a specific **vhost**.

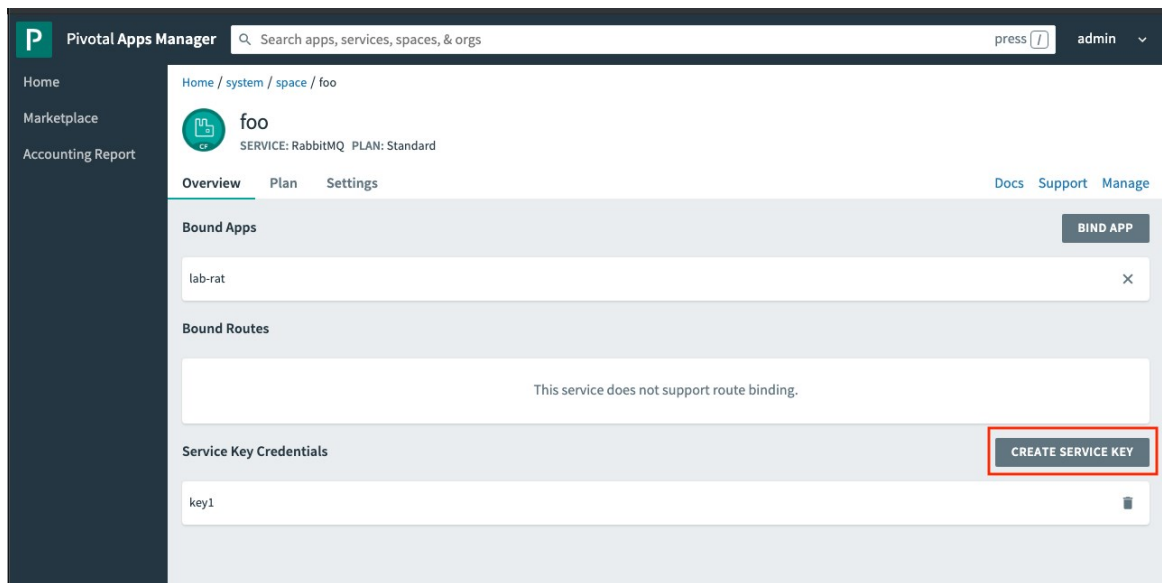
To access the RabbitMQ Management UI, do one of the following:

**Use the `cf` CLI:**

1. Create credentials using the cf CLI. For instructions, see [Create an Admin User for a Service Instance](#).
2. Using the output from step 2 of *Create an Admin User for a Service Instance*, navigate to the URL in `dashboard_url` and use the `username` and `password` to log in to the RabbitMQ Management UI.

### Use Apps Manager:

1. In Apps Manager, navigate to the service instance you want to access the Management UI for.
2. Click **CREATE SERVICE KEY** and give it a name of your choice.



The service key you create should appear in the **Service Key Credentials** section.

3. Select the service key you created in the previous step.
4. From the information displayed, navigate to the URL in `dashboard_url`. Use the `username` and `password` to log in to the RabbitMQ Management UI.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

# Clustering and Network Partitions



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic contains information about clustering and network partitions, and applies to both the on-demand and pre-provisioned services.

## Clustering in RabbitMQ for Pivotal Platform

In RabbitMQ, 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 **Pre-Provisioned RabbitMQ** 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 PATH=$PATH:erlang/bin/`
4. `$ 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 PATH=$PATH:erlang/bin/`
4. `$ cd rabbitmq-server/bin/`
5. `$ ./rabbitmqctl list_queues`
6. `$ ./rabbitmqctl sync_queue name`

[Create a pull request or raise an issue on the source for this page in GitHub](#)



# Upgrading RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

RabbitMQ for Pivotal Platform enables automated upgrades between versions of the product. In some versions, you might be required to take the RabbitMQ cluster offline. Whenever this is necessary, it is noted in the release notes for those versions.

For product versions and upgrade paths, see [Upgrade Planner](#).

This topic applies to both the on-demand and pre-provisioned services.

## About the Upgrade

This section provides information about the upgrade process, release cycle, and downtime during upgrades.

### General Notes About Downtime

Resilient apps are less likely to crash during downtime. For how developers can create resilient apps, see the [workloads](#) repository in GitHub.

A guide for downtime during upgrade deployments is shown in the table below. In some cases, the RabbitMQ cluster remains available during a tile upgrade, but individual queues on cluster nodes might be taken offline.

The length of the downtime depends on whether there is a stemcell update to replace the operating system image, or whether the update is for the RabbitMQ software only. Stemcell updates incur additional downtime while the IaaS creates the new VM.

The RabbitMQ cluster becomes unavailable only when upgrading between specific versions of Erlang or RabbitMQ. This is stated in the release notes for those versions.



**IMPORTANT:** The following table is only a guide. Always check the release notes for the version you are upgrading to.

Upgrade Type	Will Downtime Be Required For This Upgrade or Update?
--------------	---

<b>Major Tile Version</b>	Refer to the release notes for that version.
<b>Minor Tile Version</b>	<b>For 1.17.2 and earlier:</b> The RabbitMQ cluster is taken offline for the duration of the upgrade. <b>For 1.17.3 and later:</b> Normally these are rolling deployments with each node being updated in turn. In these cases, the cluster remains available, but individual queues might be taken offline as each node is restarted. There might be specific migration paths that require downtime. These are identified in the release notes for that version.
<b>Patch Tile Version</b>	Normally these are rolling deployments with each node being updated in turn. In these cases the cluster remains available, but individual queues might be taken offline as each node is restarted. There might be specific migration paths that require downtime. These are identified in the release notes for that version.
<b>Stemcell -Only Patch Tile Version</b>	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 remains available, but individual queues might be taken offline as each node is restarted.

## General Notes About the Upgrade Process

The following notes about the upgrade process apply to upgrading to any version of RabbitMQ.

- Upgrading to a newer version of the product does not cause any loss of data or configuration.
- It might take busy RabbitMQ nodes a long time to shut down during the upgrade and you must not interrupt this process.
- To benefit from rolling upgrades, configure your apps to reconnect after a node restarts. For more information, see [Handling Node Restarts in Applications](#) in the RabbitMQ documentation.
- The benefit you get from stemcell rolling upgrades depends on how you have configured network partition handling and the **Resource Config** tab. An HAProxy instance count of 2 and a RabbitMQ node count of 3 are required for rolling stemcell upgrades. These counts are the default. For more information, see [Clustering and Network Partitions](#).
- Pivotal Operations Manager ensures the instances are updated with the new packages and any configuration changes are applied automatically.

## Release Policy

When a new version of RabbitMQ is released, a new version of RabbitMQ is released soon after. For more information, see the [Release Policy](#).

The RabbitMQ tile uses floating stemcells in v1.14.0 and later. This means that the tile can be updated to use the latest minor version of a stemcell, without the need to download a new RabbitMQ patch release from Pivnet. For more information see [Floating Stemcells](#).

## Upgrade RabbitMQ

To upgrade the product, follow these steps:

1. Download the latest version of the product from [Pivotal Network](#).

2. Upload the new .pivotal file to Ops Manager.
3. Upload the stemcell associated with the update (*if required*).
4. Update any new mandatory configuration parameters (*if required*).
5. Click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
6. Click **Apply Changes**. The rest of the process is automated.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

# Rolling Upgrades in RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic describes the rolling upgrade strategy and how it can incur less downtime than other upgrade methods. It includes steps for running a rolling upgrade and a description of an experiment that illustrates the benefits of rolling upgrades in detail.

## Overview

A rolling upgrade is a strategy for updating a distributed system.

In a rolling upgrade, each VM is updated in turn. After the update completes, the VM is started, and, after the specified processes are running, the update procedure begins for the next VM in the sequence.

In a RabbitMQ cluster, each node runs on a separate VM. Rolling upgrades help to ensure availability by keeping at least one node up throughout the upgrade process.

Before v1.17.3, some upgrades required the whole cluster to be shut down. For example, when a major or minor version of RabbitMQ was updated, or when a major version of the Erlang distribution was updated.

As of v1.17.3, upgrades are performed using a rolling upgrade strategy. The only case where a cluster is required to fully shut down as part of an upgrade is where the Erlang cookie for the cluster is changed. Due to ongoing development, Pivotal cannot guarantee that rolling upgrades will always be possible in the future. Pivotal always recommends checking the release notes for each version before upgrading.

## Running a Rolling Upgrade for RabbitMQ for Pivotal Platform

On a single canary node in the cluster, the following steps are carried out:

- `rabbitmqctl stop` runs, stopping the RabbitMQ server process and the Erlang VM.
- The persistent disk is detached.
- The VM is torn down for the node.
- After 0 – 5 seconds the BOSH DNS service detects a failing health check for the node that has just gone down. It stops routing service traffic to the node.

- BOSH requests a new VM from the underlying IaaS. It attaches the persistent disk from the old VM.
- rabbitmq-server starts on the new VM. The node rejoins the cluster.
- The new node registers with the BOSH DNS service and begins receiving traffic.

The above steps are then carried out on the remaining nodes in the cluster, one by one.

## Example Rolling Upgrade Scenario

The experiment described below is an example of a rolling upgrade scenario.

In the experiment, an operator upgrades their platform to use a new version of RabbitMQ for Pivotal Platform. They upgrade from v1.17.4 to v1.18.1.

This experiment is designed to show a system performing a rolling upgrade under a heavy load: there is substantial disk I/O with both the underlying RabbitMQ and Erlang software upgrading to a newer version.

Without a rolling upgrade, the whole cluster must shut down, resulting in a service outage given publishers and consumers are unable to connect to the cluster. This experiment shows the extent of downtime associated with a rolling upgrade.



**Note:** The following is provided for example purposes only and is not intended to represent all upgrade situations. Your platform setup might have different results.

## Configuration and Setup

Details of the configuration and setup of the experiment are detailed in the sections below.

### Cluster Configuration

The IaaS used for this experiment is Google Cloud Platform (GCP).

The RabbitMQ node VMs are configured with:

- CPUs: 2
- RAM: 2 GB
- Disk: 8 GB
- Persistent disk: 5 GB

Initially, RabbitMQ for Pivotal Cloud Foundry v1.17.4 was installed with a plan configured to build a three-node cluster with queues being mirrored. This environment was then upgraded to use RabbitMQ v1.18.1.

### App Configuration

The RabbitMQ Performance Tool for Cloud Foundry simulates the workload on the cluster. This is a Java app that tests throughput of RabbitMQ.

This tool uses a resilient client with reconnection and retry logic. When the performance test is run, it

creates a direct exchange and a queue. In addition, it creates the necessary consumers and producers and binds them to the newly created queue.

In this experiment, the performance test is configured to use durable and mirrored queues and persistent messages, which ensure that messages are persisted to disk.

A protocol extension called Publisher Confirms is enabled to ensure that there is no data loss.

This setup ensures that there is a backlog of messages to be read from disk and consumed at any point during the upgrade.

The publishers are configured to constantly produce messages in three different bursts:

- 500 messages per second for 30 seconds
- 750 messages per second for 15 seconds
- 250 messages per second for 15 seconds

The publishers are expected to consume a total of 500 messages per second. Each message is a 50,000-byte JSON blob.

The equivalent app manifest for this test is as follows:

```
---
applications:
- name: rabbitmq-perf-test
  path: ./target/pcf-perf-test-1.0-SNAPSHOT.jar
  buildpacks:
  - https://github.com/cloudfoundry/java-buildpack.git
  memory: 2G
  health-check-type: process
  services: [rmq]
  env:
    VARIABLE_RATE: "500:30,750:15,250:15"
    CONSUMER_RATE: 500
    JSON_BODY: true
    SIZE: 50000
    SLOW_START: true
    METRICS_PROMETHEUS: true
    FLAG: persistent
    CONFIRM: 30000
```

For more information about concepts mentioned above, see:

Concept	More information in...
The RabbitMQ Performance Tool for Cloud Foundry	the <a href="#">RabbitMQ PerfTest for Cloud Foundry</a> repository in GitHub
Direct exchange	the <a href="#">RabbitMQ documentation</a>
Durable queues	the <a href="#">RabbitMQ documentation</a>
Mirrored queues	the <a href="#">RabbitMQ documentation</a>
Publisher Confirms protocol extension	the <a href="#">RabbitMQ documentation</a>

## Observations

Tests show that downtime experienced during this rolling upgrade is significantly reduced compared to a similar upgrade where the cluster is fully shut down.

The metrics indicate that the downtime, in this case a publisher being unable to publish a message to a queue, is 5 seconds at maximum.

This is because the internal BOSH DNS record used to round-robin messages to the nodes in the cluster has a 5-second time to live (TTL). The messages are routed to nodes that have just been replaced. Because the tested app has retry logic, no service outage is observed.

For more information about creating resilient apps, see the [workloads repository](#) in GitHub.

In most cases, downtime is longer for a cluster under a greater load. When a node comes back up and rejoins the cluster, messages from the other nodes sync with the newly joined node. Queues on the newly joined node reject publishers and consumers until the messages are synced.

## Cluster Configuration Considerations

There is downtime for a cluster without mirrored queues. This is because when the hosting node is down, the queue does not exist and any published messages are dropped unless the publisher uses the `mandatory` flag or the exchange is configured with an alternate exchange.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

# Developer Guide

## Using On-Demand RabbitMQ for Pivotal Platform



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic provides instructions for developers on how to use the on-demand RabbitMQ for Pivotal Platform service for their apps that run on Pivotal Application Service. 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 RabbitMQ with your PAS apps, you must:

- Have an Ops Manager installation with RabbitMQ installed and listed in the Marketplace. For how to verify availability in the Marketplace, see [Confirm Service Availability](#) below.
- Have a Space Developer or Admin account on the PAS installation. For more information, see [User Roles](#).
- Have a local machine with the following installed:
  - ✦ A browser
  - ✦ A shell
  - ✦ The Cloud Foundry Command-Line Interface (cf CLI). See [Installing the cf CLI](#).
  - ✦ The Linux watch command. See the [Linux Information Project website](#).
- Log in to the org and space containing your app. For instructions, see [Log In With the CLI](#).

## Create Resilient Apps

Resilient apps are less likely to crash during downtime, such as when upgrading RabbitMQ. For how to create resilient apps, see the [workloads](#) repository in GitHub.

## Confirm Service Availability



For an app to use a service, the service must be available in the Marketplace for its space.

To confirm this availability:

1. List the services that are available in the Marketplace:

```
cf marketplace
```

2. Verify that `p.rabbitmq` is in the `service` column. If it is, on-demand RabbitMQ is available. If it is not, ask your operator to install it.

For example:

```
$ cf marketplace
Getting services from marketplace in org my-org / space my-space as user@example.com...
OK
service      plans      description
[...]
p.rabbitmq    single-node  RabbitMQ service to provide dedicated instances of this high-performance multi-protocol messaging broker
[...]
```

## Create a Service Instance

For an app to use a service, an instance of the service must exist in its space. On-Demand services are created asynchronously, not immediately. The `watch` command shows you when your service instance is ready to bind and use.

You can enable TLS and additional plugins using the `cf create-service` command. For instructions on how to modify the `cf create-service` command, see [Enable TLS for Your Service Instance](#) or [Enable Optional Plugins](#) below.

To create an instance of the on-demand RabbitMQ service:

1. See if there is an existing service by running:

```
cf services
```

Look for instances that have `p.rabbitmq` in the `service` column. If there is an existing instance that you want to use, skip to the [Bind a Service Instance to Your App](#) section below.

For example:

```
$ 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  p.rabbitmq    single-node, plan-2      create succeeded
```

2. Create the service instance by running:

```
cf create-service p.rabbitmq SINGLE-NODE SERVICE-INSTANCE-NAME
```

Where:

- ✦ `SERVICE-INSTANCE-NAME` is a name you choose to identify the service instance. This name appears under `name` in the output from `cf services`.
- ✦ `SINGLE-NODE` is the plan. You can replace it with any of the other plans listed in the Marketplace for the `p.rabbitmq` service.

3. Verify that the process was successful by running:

```
watch cf services
```

If successful, `last operation` for your instance is reported as `create succeeded`.

For example:

```
$ cf create-service p.rabbitmq single-node 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	p.rabbitmq	single-node		create succeeded

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

## Use Transport Layer Security (TLS)

The operator must enable TLS in the **Security for On-Demand Plans** tab before you can use this feature. For more information, see [Configure Security](#).

To use TLS to secure communication between your app and a RabbitMQ service instance:

1. [Enable TLS for Your Service Instance](#)
2. If your app is written in Java or Spring, follow the steps in the [Activate TLS for Java and Spring Apps](#) section below. If it is not, follow the steps in [Modifying Apps for TLS](#).

## Enable TLS for Your Service Instance

If the operator has configured TLS to be optional, then follow the steps below to enable TLS on a service instance. If the operator has configured TLS to be enforced, then all service instances have TLS enabled and it is not necessary to follow the steps below. For more information, see [Configure Security](#).

To configure TLS for your service instance:



**Note:** If you enable TLS for a service instance, you can no longer connect to it without using TLS.

1. Enable TLS by doing one of the following steps:

- If you have already created a RabbitMQ service instance, update the service instance to use TLS by running:

```
cf update-service MY-INSTANCE -c '{"tls": true}'
```

Where **MY-INSTANCE** is the name you chose when you created your service instance.

- If you have not created a RabbitMQ service instance, create one with TLS enabled by running:

```
cf create-service p.rabbitmq PLAN MY-INSTANCE -c '{"tls": true}'
```

Where:

- **PLAN** is the name of the deployment plan to use for this service instance
- **MY-INSTANCE** is the name to give to the new service instance

2. Verify that the process was successful by running:

```
watch cf services
```

If successful, the **last operation** for your instance is reported as **create succeeded** or **update succeeded**.

For example:

```
$ 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    p.rabbitmq   single-node                                create succeeded
```

## Disable TLS for Your Service Instance

If the operator has configured TLS to be enforced, then all service instances have TLS enabled and it is not possible to disable TLS. If the operator has configured TLS to be optional or not configured, then follow the steps below to disable TLS on a service instance. For more information, see [Configure Security](#).

1. Disable TLS by doing one of the following:

- If you have already created a RabbitMQ service instance, update the service instance by running:

```
cf update-service MY-INSTANCE -c '{"tls": false}'
```

Where **MY-INSTANCE** is the name you chose when you created your service instance.

- If you have not created a RabbitMQ service instance, request a service instance with TLS disabled by running:

```
cf create-service p.rabbitmq PLAN MY-INSTANCE -c '{"tls": false}'
```

Where:

- **PLAN** is the name of the deployment plan to use for this service instance
- **MY-INSTANCE** is the name to give to the new service instance

2. Verify that the process was successful by running:

```
watch cf services
```

If successful, the last operation for your instance is reported as **update succeeded**.

For example:

```
$ 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    p.rabbitmq   single-node                                update succeeded
```

## Activate TLS for Java and Spring Apps



**Note:** If your app is not written in Java or Spring, see [Modifying Apps for TLS](#).

For an app to use TLS, you must update it to request encrypted communications when connecting to a RabbitMQ service instance. The procedure for updating your app depends on the app's language and framework. Java and Spring apps automatically detect TLS.

To activate TLS for Java and Spring apps, do one of the following:

- For apps that are not bound to an existing service instance, re-push the apps by running the **cf push** command.
- For apps bound to an existing service instance, re-bind the apps as follows:
  1. Stop the app:

```
cf stop APP-NAME
```

2. Unbind the app from the service instance:

```
cf unbind-service APP-NAME SERVICE-INSTANCE-NAME
```

3. Re-bind the app to the service instance:

```
cf bind-service APP-NAME SERVICE-INSTANCE-NAME
```

4. Restart the app.

```
cf restart APP-NAME
```



**Note:** If the operator used a self-signed certificate, configure your app to use the same CA certificate and valid certificate and key.

## Enable Optional Plugins

For a list of RabbitMQ plugins that are enabled or disabled by default in on-demand RabbitMQ instances, see [RabbitMQ Server Plugins](#).



**Note:** If a plugin is disabled by default, the operator must enable the plugin for the platform before you can enable it on your service instances. For more information, see [Optional RabbitMQ Server Plugins](#).

You can enable plugins when you create a service instance or by updating an existing service instance. When enabling more than one plugin, specify all the plugin names in the same command.

To enable plugins, run one of these commands, specifying the plugins as a configuration parameter in JSON:

- When creating a service instance, run:

```
cf create-service SERVICE PLAN SERVICE-INSTANCE-NAME -c PARAMETER-AS-JSON
```

Where the `PARAMETER-AS-JSON` key is the plugin and the value is `true`.

For example:

```
$ cf create-service p.rabbitmq single-node myservice -c '{ "plugins": { "rabbitmq_event_exchange": true } }'
```

For example, when enabling more than one plugin:

```
$ cf create-service p.rabbitmq single-node myservice -c '{ "plugins": { "rabbitmq_event_exchange": true, "rabbitmq_mqtt": true } }'
```

- When updating an existing service instance, run:

```
cf update-service SERVICE-INSTANCE-NAME -c PARAMETER-AS-JSON
```

Where the `PARAMETER-AS-JSON` key is the plugin and the value is `true`.

For example:

```
$ cf update-service myservice -c '{ "plugins": { "rabbitmq_event_exchange": true } }'
```

For example, when enabling more than one plugin:

```
$ cf update-service myservice -c '{ "plugins": { "rabbitmq_amqp1_0": true, "rab
```

```
bitmq_event_exchange": true } }'
```

## Disable Optional Plugins



**Note:** When creating a service instance, optional plugins are disabled by default.

To disable plugins for an existing service instance:

1. Run:

```
cf update-service SERVICE-INSTANCE-NAME -c PARAMETER-AS-JSON
```

Where the `PARAMETER-AS-JSON` key is the plugin and the value is `false`.

For example:

```
$ cf update-service myservice -c '{ "plugins": { "rabbitmq_event_exchange": false } }'
```

For example, when disabling more than one plugin:

```
$ cf update-service myservice -c '{ "plugins": { "rabbitmq_amqp1_0": false, "rabbitmq_event_exchange": false } }'
```

## 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 service instance:

1. Run:

```
cf bind-service APP-NAME SERVICE-INSTANCE-NAME
```

Where:

- ♦ `APP-NAME` is the app you want to use the RabbitMQ service instance.
- ♦ `SERVICE-INSTANCE-NAME` is the name you supplied when you ran `cf create-service`.

For example:

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

Binding service mydb to my-app in org my-org / space test as user@example.com..
.
OK
```

## Use the RabbitMQ Service in Your App

Apps running in PAS gain access to the bound service instances through an environment variable credentials hash called `VCAP_SERVICES`. An example hash is shown below:

```
{
  "p.rabbitmq": [{
    "label": "p.rabbitmq",
    "provider": null,
    "plan": "single-node",
    "name": "myservice",
    "tags": [
      "rabbitmq"
    ],
    "instance_name": "myservice",
    "binding_name": null,
    "credentials": {
      "dashboard_url": "https://rmq-62e5ab21-7b38-44ac-b139-6aa97af01cd7.your.pcf.example.com",
      "hostname": "q-s0.rabbitmq-server.default.service-instance-62e5ab21-7b38-44ac-b139-6aa97af01cd7.bosh",
      "hostnames": [
        "q-s0.rabbitmq-server.default.service-instance-62e5ab21-7b38-44ac-b139-6aa97af01cd7.bosh"
      ],
      "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
      "password": "tavk86pnns1ddiqpsdtbchurn",
      "vhost": "62e5ab21-7b38-44ac-b139-6aa97af01cd7",
      "http_api_uri": "https://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiqpsdtbchurn@rmq-62e5ab21-7b38-44ac-b139-6aa97af01cd7.your.pcf.example.com/api/",
      "http_api_uris": [
        "https://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiqpsdtbchurn@rmq-62e5ab21-7b38-44ac-b139-6aa97af01cd7.your.pcf.example.com/api/"
      ],
      "protocols": {
        "amqp": {
          "host": "q-s0.rabbitmq-server.default.service-instance-62e5ab21-7b38-44ac-b139-6aa97af01cd7.bosh",
          "hosts": [
            "q-s0.rabbitmq-server.default.service-instance-62e5ab21-7b38-44ac-b139-6aa97af01cd7.bosh"
          ],
          "password": "tavk86pnns1ddiqpsdtbchurn",
          "port": 5672,
          "ssl": false,
          "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiqpsdtbchurn@q-s0.rabbitmq-server.default.service-instance-62e5ab21-7b38-44ac-b139-6aa97af01cd7.bosh/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
          "uris": [
            "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiqpsdtbchurn@q-s0.rabbitmq-server.default.service-instance-62e5ab21-7b38-44ac-b139-6aa97af01cd7.bosh/62e5ab21-7b38-44ac-b139-6aa97af01cd7"
          ],
          "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
          "vhost": "62e5ab21-7b38-44ac-b139-6aa97af01cd7"
        }
      }
    },
    "ssl": false,
    "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnns1ddiqpsdtbchurn@q-s0.rabbitmq-server.default.service-instance-62e5ab21-7b38-44ac-b139-6aa97af01cd7.bosh/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
  ]
}
```

```

    "uris": [
      "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@q-s0.rabbitmq-server.default.service-instance-62e5ab21-7b38-44ac-b139-6aa97af01cd7.bosh/62e5ab21-7b38-44ac-b139-6aa97af01cd7"
    ]
  }
}]
}

```

Apps that have been bound to a service instance can read `VCAP_SERVICES` from their environment. To see the `VCAP_SERVICES` for an app, run `cf env APP-NAME` with the name of an app bound to the RabbitMQ instance.

For more information about the environment variable `VCAP_SERVICES`, see [RabbitMQ Environment Variables](#).

## RabbitMQ Node Address Resolution

The `hostnames` field in `VCAP_SERVICES` contains a single hostname, even if there is more than one RabbitMQ node. BOSH DNS resolves the hostname to a list of the IP addresses of all the available RabbitMQ nodes. The order of the IP addresses rotates so that there is load balancing.

A RabbitMQ client can use the first IP address returned. If a node becomes unavailable, BOSH DNS removes it from the list of resolved IP addresses. Because of this, apps should not cache the IP addresses of the nodes, as the cache might become outdated. The Java buildpack automatically disables DNS caching.

## Update a Service Instance

If you bind a new service or change the service bindings, you need to run `cf restart` to update the `VCAP_SERVICES` environment variable in the application container.

To restart your app:

1. Run:

```
cf restart-app APP-NAME
```

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

For example:

```
$ cf restart my-app
```

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

## Unbind a Service Instance From Your App

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

To unbind a service instance from your app:

1. Run:



```
cf unbind-service APP-NAME SERVICE-INSTANCE-NAME
```

Where:

- **APP-NAME** is the app you want to stop using the RabbitMQ service instance.
- **SERVICE-INSTANCE-NAME** is the name you supplied when you ran `cf create-service`.

For example:

```
$ 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



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

To delete a service instance:

1. Run:

```
cf delete-service SERVICE-INSTANCE-NAME
```

For example:

```
$ 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. Verify that the service instance was deleted by running:

```
watch cf service SERVICE-INSTANCE-NAME
```

and then wait for a `Service instance not found` error indicating that the instance no longer exists.

## Create an Admin User for a Service Instance

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

Both operators and app developers can use this procedure. For instructions, see [Create an Admin User for a Service Instance](#).

## Share Service Instances

To share service instances, your operator must enable the feature flag `service_instance_sharing`. You can then follow the [Cloud Foundry documentation](#) to share your service instances across Cloud Foundry orgs and spaces.

## Access RabbitMQ Metrics for On-Demand Service Instances

To access metrics for RabbitMQ service instances, you can use Loggregator's Log Cache feature with the Log Cache CLI plugin. Log Cache is enabled by default.



**Note:** To use this feature the **V2 Firehose** must be enabled and **Enable Log Cache syslog ingestion** must be disabled in the PAS tile. For more information about configuring these checkboxes, see [Enable Syslog Forwarding](#).

To access metrics for on-demand service instances:

1. Install the cf CLI plugin by running:

```
cf install-plugin -r CF-Community "log-cache"
```

2. To access metrics for a service instance, run:

```
cf tail SERVICE-INSTANCE-NAME
```

Where `SERVICE-INSTANCE-NAME` is the name of your service instance.

For example:

```
$ cf tail my-instance
Retrieving logs for service my-instance in org system / space pivotal-services
as admin...
2018-12-06T11:01:06.03+0000 [my-instance] GAUGE /p.rabbitmq/rabbitmq/channels/c
ount:0.000000 count /p.rabbitmq/rabbitmq/connections/count:0.000000...
```

For more information about the metrics output, see [Monitoring and KPIs for On-Demand RabbitMQ](#)

For more information about how to enable Log Cache and about the `cf tail` command, see [Enable Log Cache](#).

## Federate Exchanges and Queues

You can federate exchanges and queues in RabbitMQ, just like in any other RabbitMQ deployment.

To federate exchanges and queues:

1. Create a service key by following the instructions in [Create an Admin User for a Service Instance](#). The output of the above procedure returns admin user credentials, along with other data.
2. In the output from the above step, look for the `uris` array. It has the following pattern

```
{
```

```
...
"uri": "amqp://USERNAME:PASSWORD@DNS_ENTRY/VHOST",
"uris": [
  "amqp://USERNAME:PASSWORD@DNS_ENTRY/VHOST"
],
...
}
```

as seen in the below example.

```
{
  ...
  "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
  "uris": [
    "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
    ...
  ]
}
```

3. Set up federation as usual, using the RabbitMQ Management UI or API, with the URIs found in the `uris` array you got from the step above.

For instructions on federation, see the [RabbitMQ documentation](#).

## Shovel Exchanges and Queues

You can shovel exchanges and queues in RabbitMQ, just like in any other RabbitMQ deployment.

To shovel exchanges and queues:

1. Create a service key by following the instructions in [Create an Admin User for a Service Instance](#). The output of the above procedure returns admin user credentials, along with other data.
2. In the output from the above step, look for the `uris` array. It has the following pattern

```
{
  ...
  "uri": "amqp://USERNAME:PASSWORD@DNS_ENTRY/VHOST",
  "uris": [
    "amqp://USERNAME:PASSWORD@DNS_ENTRY/VHOST"
  ],
  ...
}
```

as seen in the example below.

```
{
  ...
  "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
  "uris": [
```

```
"amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@q-s0.rabb
itm-q-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba8
51.bosh:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
...
}
```

3. Set up shovel as usual, using the RabbitMQ Management UI or API, with the URIs found in the `uris` array you got from the step above.

For shovel instructions, see the [RabbitMQ documentation](#). RabbitMQ currently only supports Dynamic Shovels.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Modifying Apps for TLS



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.



**Note:** If your app is written in Java or Spring, see [Activate TLS for Java and Spring Apps](#). For other types of apps, use the procedures in this topic.

This topic explains how to modify apps not written in Java or Spring to use TLS to secure their connection with RabbitMQ for Pivotal Platform on-demand service instances.

## Prerequisites

The prerequisites for the procedures in this topic are:

- The operator must complete these procedures, in this order:
  1. [Provide or Generate a CA Certificate](#)
  2. [Configure Security](#)
- The developer must complete the procedures in [Enable TLS for Your Service Instance](#).

## Modify Your App for TLS

To start using TLS for apps that are not written in Java or Spring, you must modify your app to use the correct protocol. The exact steps vary between client libraries. Consult the documentation for your library for the necessary configuration.

The following examples use the [ruby-amqp/bunny](#) library in GitHub. In these examples, `VCAP_SERVICES` is an environment variable available from the app.

To modify your app:

- **If the operator enabled TLS using a certificate from a trusted authority:**  
Use the code below.

```
require 'json'
require 'bunny'

vcap_services = JSON.parse(ENV['VCAP_SERVICES'])
uri = vcap_services['p.rabbitmq'][0]['credentials']['protocols']['amqp+ssl']['uri']
conn = Bunny.new(uri)
conn.start
```

- **If the operator used a self-signed CA certificate:** Use the code below to configure the RabbitMQ client to use the same CA certificate, as well as a valid TLS certificate and key.

```
require 'json'
require 'bunny'

vcap_services = JSON.parse(ENV['VCAP_SERVICES'])
uri = vcap_services['p.rabbitmq'][0]['credentials']['protocols']['amqp+ssl']['uri']
conn = Bunny.new(uri, tls_cert: PATH-TO-CERTIFICATE, tls_key: PATH-TO-KEY, tls_ca_certificates: [PATH-TO-CA-CERTIFICATE])
conn.start
```

Where:

- ✦ `PATH-TO-CERTIFICATE` is the path to your TLS certificate
- ✦ `PATH-TO-KEY` is the path to your TLS key
- ✦ `PATH-TO-CA-CERTIFICATE` is the path to the self-signed CA certificate the operator used

- **If connecting to a service-gateway instance:** Use the code below to configure the RabbitMQ client to verify the identity of the service instance.

```
require 'json'
require 'bunny'

service_key = JSON.load(File.open(PATH-TO-SERVICE-KEY-JSON))
uri = service_key['protocols']['amqp+ssl']['uri']
conn = Bunny.new(uri, tls: true, verify_peer: true, tls_ca_certificates: [PATH-TO-CA-CERTIFICATE])
conn.start
```

Where:

- ✦ `PATH-TO-SERVICE-KEY-JSON` is the path to the service key in JSON
- ✦ `PATH-TO-CA-CERTIFICATE` is the path to the CA certificate which signed the RabbitMQ server certificate

## Re-push or Rebind Your App

After modifying your app, re-push it with `cf push`.



**Warning:** Any apps using an existing service instance must be rebound after enabling TLS for the instance.

To rebound an app using an existing service instance:

1. Stop the app by running:

```
cf stop APP-NAME
```

2. Unbind the app from the service instance by running:

```
cf unbind-service APP-NAME SERVICE-INSTANCE-NAME
```

3. Re-bind the app to the service instance by running:

```
cf bind-service APP-NAME SERVICE-INSTANCE-NAME
```

4. Restage the app by running:

```
cf restage APP-NAME
```

Your app now communicates securely with the RabbitMQ service instance.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Migrating a RabbitMQ for Pivotal Platform Service Instance to Another Service Instance



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic describes how to safely and fully migrate a RabbitMQ service instance to another service instance. This migration can include migrating policies, parameters, and messages.

## Migrate a RabbitMQ Service Instance to Another Service Instance

To migrate from one service instance to another, do the following:

1. Create a new service instance. Pivotal recommends using the on-demand service for new deployments. For instructions, see [Create a Service Instance](#).
2. If the RabbitMQ client in your apps has not defined the model for you, create queues, exchanges, and bindings on the new service instance that match your old service instance.
3. Migrate policies and parameters. For instructions, see [Migrate Policies and Parameters](#) below.
4. If you need all the messages in the old service instance, migrate the messages in the

queues. For instructions, see [Migrate Messages in Queues](#) below.

5. When ready, delete your old instance. For instructions, see [Delete a Service Instance](#).
6. If you have service-keys for the old instance, create service-keys for the new instance and replace the old credentials.

## Migrate Policies and Parameters

If your old instance uses any of the following, ensure that you apply them to the new instance:

- Policies
- Virtual host-specific parameters, such as `max_connections`

You can apply them using the API by following the steps below.



**Note:** You can also apply them in the RabbitMQ Management UI. The API is the faster option if you have multiple instances to migrate.

1. Create a service key for the new instance with the admin username and password by running the following command:

```
cf create-service-key SERVICE-INSTANCE SERVICE-KEY -c '{"tags": "administrator"}'
```

Where:

- ✦ `SERVICE-INSTANCE` is your service instance
- ✦ `SERVICE-KEY` is a name of your choice for your service key

2. Retrieve the admin username and password with the following command:

```
cf service-key SERVICE-INSTANCE SERVICE-KEY
```

Where:

- ✦ `SERVICE-INSTANCE` is your service instance
- ✦ `SERVICE-KEY` is the service key name you chose in the step above

3. Retrieve the URL of the new service instance with the following command:

```
cf service SERVICE-INSTANCE
```

Where `SERVICE-INSTANCE` is your service instance.

4. Export the virtual host (vhost) limit parameters JSON to a local file with the following command:

```
curl -u USERNAME:PASSWORD OLD-MANAGEMENT-URL/api/definitions/VHOST | jq '{"parameters": [.parameters[]]}' > PATH-TO-FILE
```

Where:

- ✦ `USERNAME` is the admin username for the cluster
- ✦ `PASSWORD` is the admin password for the cluster
- ✦ `OLD-MANAGEMENT-URL` is the RabbitMQ Management UI URL for your old service instance
- ✦ `VHOST` is your virtual host ID

5. Upload the virtual host limit parameters JSON to the new instance with the following command:

```
curl -H 'Content-type: application/json' -i -XPOST -u USERNAME:PASSWORD NEW-MANAGEMENT-URL/api/definitions/VHOST -d "$(cat PATH-TO-FILE) "
```

Where:

- ✦ `USERNAME` is the admin username for the cluster
- ✦ `PASSWORD` is the admin password for the cluster
- ✦ `NEW-MANAGEMENT-URL` is the RabbitMQ Management UI URL for your new service instance
- ✦ `VHOST` is your virtual host ID

6. Export the policies JSON to a local file with the following command:

```
curl -u USERNAME:PASSWORD OLD-MANAGEMENT-URL/api/definitions/VHOST | jq '{"policies": [.policies[]]}' > PATH-TO-FILE
```

7. Upload the policies JSON to the new instance with the following command:

```
curl -H 'Content-type: application/json' -i -XPOST -u USERNAME:PASSWORD NEW-MANAGEMENT-URL/api/definitions/VHOST -d "$(cat PATH-TO-FILE) "
```

## Migrate Messages in Queues

To safely migrate messages from one service instance to another, you must do one of the following:

- [Migrate Messages Using the RabbitMQ Shovel Plugin](#)
- [Migrate Messages By Using Consumers to Drain the Old Service Instance](#)

Use of the shovel plugin requires less waiting time and manual steps, however, it does not preserve the ordering of messages.

## Migrate Messages Using the RabbitMQ Shovel Plugin

You can safely migrate messages to the new instance by using a shovel to move messages between queues on the two instances. For more information, see [the RabbitMQ documentation](#).

1. Log in to the RabbitMQ Management Web UI for both the old and new service instances.
2. For each queue, create a shovel from the queue on the old instance to the queue on the new instance.
3. Unbind the consumer apps from the old service instance. For instructions, see [Unbind a](#)



### Service Instance From Your App.

4. Bind the consumer apps to the new service instance. For instructions, see [Bind a Service Instance to Your App](#).
5. Restart your consumer apps. For more information, see [Restart Your App](#).
6. Using the Management Web UI, confirm that your consumer apps are successfully connected to the new service instance.
7. Carry out steps 3 – 5 for your producer apps.
8. Using the Management Web UI, confirm that your producer apps are successfully connected to the new service instance.
9. Ensure all messages are moved from the old instance to the new instance by using its Management Web UI.
10. Verify no messages are in the old instance by checking its Management Web UI.

## Migrate Messages by Using Consumers to Drain the Old Service Instance

You can safely migrate messages to the new instance without the use of the shovel plugin. This method is better for ensuring that messages are ordered consistently. However, it requires more manual steps and waiting for queues to be drained.

1. Log in to the RabbitMQ Management Web UI for both the old and new service instances.
2. Unbind the producer apps from the old service instance. For more information, see [Unbind a Service Instance From Your App](#).
3. Bind the producer apps to the new service instance. For more information, see [Bind a Service Instance to Your App](#).
4. Restart your producer apps. For more information, see [Restart Your App](#).
5. Using the Management Web UI, confirm that your producer apps are successfully connected to the new service instance.
6. Wait for the consumer apps to fully drain the queues of the old service instance.
7. Repeat steps 2 – 4 for your consumer apps.
8. Using the Management Web UI, confirm that your consumer apps are successfully connected to the new service instance.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Creating a Service Instance with Service-Gateway Access



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic explains how to create a service instance with service-gateway access.

Service-gateway access enables a RabbitMQ for Pivotal Platform on-demand service instance to connect to external components that are not on the same foundation as the service instance.

## Create a Service Instance with Service-Gateway Access

This section assumes that you meet the prerequisites for using on-demand RabbitMQ for Pivotal Platform. For more information, see [Prerequisites](#).

If operators have enabled a service-gateway plan, you can create a service instance that can connect to components outside the your foundation. Contact your operator if you are unsure which plans have been enabled for service-gateway access. For information about the architecture and use cases, see [Service-Gateway Access](#).

To create a service instance that enables service-gateway access:

1. Create a service instance with the service-gateway plan by running:

```
cf create-service p.rabbitmq SERVICE-GATEWAY-PLAN SERVICE-INSTANCE-NAME -c '{"tls": true}'
```



**Warning:** Pivotal recommends that you configure service-gateway access with TLS to prevent man-in-the-middle attacks. For instructions on using TLS on your platform, see [Use Transport Layer Security](#).

2. Obtain credentials by creating a service key:

```
cf create-service-key SERVICE-INSTANCE-NAME SERVICE-KEY-NAME
```

The `uri` section of the service key contains the address to the TCP router and a port from the port range. You can use this URI to connect to the service instance from outside the foundation.



**Note:** If the operator disables and then re-enables service-gateway access on a plan, you must create new service keys to obtain a new set of credentials for service-gateway access.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## RabbitMQ Environment Variables



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic provides a reference for the environment variables that Pivotal Application Service stores for RabbitMQ for Pivotal Platform service instances. These variables include the credentials that apps use to access the service instances.

## VCAP\_SERVICES

Apps running in PAS gain access to the bound service instances through 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": "single-node",
    "tags": ["rabbitmq"],
    "credentials": {
      "dashboard_url": "http://pivotal-rabbitmq.your.pcf.example.com",
      "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
      "vhost": "62e5ab21-7b38-44ac-b139-6aa97af01cd7",
      "password": "#passwordexample123456789",
      "ssl": false,
      "hostname": "q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh",
      "hostnames": [
        "q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh",
      ],
      "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
      "uris": [
        "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
      ],
      "http_api_uri": "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh:15672/api",
      "http_api_uris": [
        "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh:15672/api",
      ],
      "protocols": {
        "amqp": {
          "password": "passwordexample123456789",
          "port": 5672,
          "ssl": false,
          "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
          "vhost": "62e5ab21-7b38-44ac-b139-6aa97af01cd7",
          "host": "q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh",
          "hosts": [
            "q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh",
          ],
          "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@q-s0.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851.bosh:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
          "uris": [
            "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@q-s0"
```

```
.rabbitmq-server.services-subnet.service-instance-e0e7fc5f-c1c7-4211-81b9-284fb29ba851
.bosh:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
    ]
  }
}
}
}]
}
```

You can search for your service by its `name`, given when creating the service instance, or dynamically through 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 are `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 or protocols might cause apps bound with RabbitMQ to break.

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

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

In common with all services in PAS, the `VCAP_SERVICES` environment variable for an app is only modified when the app is bound to a service instance. Users need to `cf unbind-service`, `cf bind-service`, and `cf restage` their app in this scenario.

[Create a pull request or raise an issue on the source for this page in GitHub](#)

## Troubleshooting On-Demand Instances



**Warning:** RabbitMQ for Pivotal Platform v1.18 is no longer supported because it has reached the End of General Support (EOGS) phase as defined by the [Support Lifecycle Policy](#). To stay up to date with the latest software and security updates, upgrade to a supported version.

This topic provides basic instructions for app developers troubleshooting on-demand RabbitMQ for

Pivotal Platform instances.

## Troubleshoot Errors

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

### Common Service Errors

The following errors occur in multiple services:

#### No Metrics from Log Cache

<b>Symptom</b>	You receive no metrics when running the <code>cf tail</code> command.
<b>Cause</b>	This might be because the Firehose is disabled in the PAS tile.
<b>Solution</b>	Ask your operator to ensure that the <b>V2 Firehose</b> checkbox is enabled, and the <b>Enable Log Cache syslog ingestion</b> checkbox is disabled in the PAS tile. For more information about configuring these checkboxes, see <a href="#">Enable Syslog Forwarding</a> .

## Techniques for Troubleshooting

See the following sections for troubleshooting techniques when using the Cloud Foundry Command-Line Interface (cf CLI) to perform basic operations on a RabbitMQ service instance.

Basic cf CLI operations include `create`, `update`, `bind`, `unbind`, and `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-af27-1b08b0c70089,
        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 Support when filing a ticket.

The `task-id` field maps to the BOSH task ID. For more information on a failed BOSH task, use the `bosh 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    p.rabbitmq   single-node                               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 Service Instance Credentials

To access the RabbitMQ Management UI for troubleshooting, create a new service-key to retrieve RabbitMQ service 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 service 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`.
3. Navigate to the RabbitMQ Management UI using the URL from the service key.

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...
{
```

```
"dashboard_url": "https://rmq-62e5ab21-7b38-44ac-b139-6aa97af01cd7.your.pcf.example.com",
...
}
```

To gain admin privileges for the RabbitMQ Management UI, create an admin user for a service instance and obtain user credentials that you can share with other app developers.

Both operators and app developers can use this procedure. See [Create an Admin User for a Service Instance](#) for more information.

## Delete RabbitMQ Service Instances

The on-demand broker provides a BOSH command to delete all on-demand broker deployed service instances.



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

To delete all on-demand service instances:

1. Run:

```
bosh run-errand delete-sub-deployments
```

## Knowledge Base (Community)

Find the answer to your question and browse product discussions and solutions by searching the [VMware Tanzu 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 expedite troubleshooting, if possible, 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.

[Create a pull request or raise an issue on the source for this page in GitHub](#)