

VMware RabbitMQ for Tanzu Application Service v1.14

VMware RabbitMQ for Tanzu Application Service 1.14

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

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

Contents

RabbitMQ for PCF	14
RabbitMQ for PCF	14
About RabbitMQ for PCF	14
What are On-Demand Instances	14
About RabbitMQ	14
Product Snapshot	15
Features	15
On-Demand	15
Pre-Provisioned	16
Release Notes and Known Issues	16
RabbitMQ for PCF and Other PCF Services	16
Feedback	17
RabbitMQ® for PCF Release Notes	17
Upgrade to the Latest Version	17
Pivotal Is Continuing Support for the Pre-Provisioned Service	17
v1.14.14	17
Resolved Issue	17
Known Issues	18
Packages	18
v1.14.13	18
Resolved Issue	18
Known Issues	19
Packages	19
v1.14.12	20
Features	20
Security Fix	20
Resolved Issue	20
Known Issues	20
Packages	21
v1.14.10	21
Resolved Issues	22
Known Issues	22
Packages	23

v1.14.9	23
Features	23
Known Issues	23
Packages	24
v1.14.8	24
Features	25
Known Issues	25
Packages	26
v1.14.7	26
Features	27
Known Issues	27
Packages	27
v1.14.6	27
Features	28
Fixed Issue	28
Known Issues	28
Packages	29
v1.14.5	29
Features	29
Known Issues	29
Packages	30
v1.14.4	30
Security Fixes	31
Features	31
Known Issues	31
Packages	31
v1.14.3	31
Features	32
Known Issues	32
Packages	32
v1.14.1 - Withdrawn	33
Features	33
Known Issues	34
Packages	34
View Release Notes for Another Version	34
 Unlocking the Power of On-Demand RabbitMQ for PCF	 34
Introduction	35
Deciding Which Service Plan to Use	35

On-Demand Single Node Plan Using RabbitMQ 3.7	37
On-Demand Cluster Plan Using RabbitMQ 3.7	37
General Principles of the Cluster Plan	37
Designed for Consistency	37
Number of Nodes	38
Network Latency	38
Consistency or Availability Tradeoff	38
RabbitMQ Queue Availability	39
Managing On-Demand Resources Through Plans	39
Customizing Plan Options	40
Configuration Options	40
Single Node and Cluster Plans	40
Cluster Plan Only	40
Things That Are Preconfigured	40
Monitoring On-Demand RabbitMQ Clusters	42
About Migrating a Pre-Provisioned Instance to an On-Demand Instance	42
Differences between Pre-Provisioned and On-Demand Services Instances	42
Architecture	44
On-Demand Service Architecture	44
Service Network Requirement	44
Default Network and Service Network	44
Required Networking Rules for On-Demand Services	45
Deploying the RabbitMQ Pre-Provisioned Service	47
Default Deployment	47
Considerations for this deployment	48
Recommended Deployment	48
Upgrading to this deployment from a single AZ deployment	49
Upgrading to this deployment from a multi AZ deployment	49
Considerations for this deployment	49
Advanced Deployment	50
Upgrading to this deployment from the recommended deployment	50
Downgrading from this deployment to the recommended deployment	51
Resource requirements	51
Notes:	52
Operator Guide: On-Demand	53
Installing and Configuring	53

Preparing for TLS	53
Overview	53
Workflow for Enabling and Using TLS	54
Provide or Generate a CA Certificate	54
Create a UAA Client	54
Add the CA Certificate	56
Installing and Configuring the On-Demand Service	58
Role-Based Access in Ops Manager	59
Prerequisites for Deploying the On-Demand Service	59
Download and Install RabbitMQ for PCF	59
Configure On-Demand RabbitMQ for PCF	59
Configure AZs and Networks	60
Configure Logging and Monitoring	61
Configure Global Settings	61
Configure Security	63
Configure the Service Plan	64
Determine which AZs a Service Instance Uses	68
Understanding RabbitMQ VM Types and Persistent Disk Size	68
Verify the Stemcell	69
Apply Changes from Your Configuration	69
Errands	69
Post-Deploy Errands	71
Pre-Delete Errands	72
Create an Admin User for a Service Instance	72
RabbitMQ Server Settings That Cannot Be Disabled	73
Smoke Tests	73
Smoke Test Steps	74
Troubleshooting	74
Monitoring and KPIs for On-Demand RabbitMQ for PCF	74
Set up Syslog Forwarding and Metrics Polling Interval	75
Logging Format	76
RabbitMQ Program Names	77
What Are Metrics	78
Key Performance Indicators	78
Component Heartbeats	79
Service Broker Heartbeat	79

Server Heartbeat	79
RabbitMQ Server KPIs	80
File Descriptors	80
Erlang Processes	80
BOSH System Health Metrics	81
RAM	81
CPU	81
Ephemeral Disk	82
Persistent Disk	82
Determine If There Is a Network Partition	83
Recover from a Network Partition	83
Component Metric Reference	83
RabbitMQ Server Metrics	83
Managing the On-Demand Service	85
Rotating CA Certificates	85
Setting Limits for On-Demand Service Instances	85
Create Global-level Quotas	86
Create Plan-level Quotas	86
Create and Set Org-level Quotas	86
Create and Set Space-level Quotas	87
View Current Org and Space-level Quotas	88
Monitor Quota Use and Service Instance Count	88
Calculate Resource Costs for On-Demand Plans	88
Calculate Maximum Resource Cost Per On-Demand Plan	90
Calculate Maximum Resource Cost for All On-Demand Plans	91
Calculate Actual Resource Cost of all On-Demand Plans	91
Controlling Access to Service Plans by Org	91
Change Access to Service Plans	91
Troubleshooting and FAQs for On-Demand RabbitMQ for PCF	92
How to Retrieve a Service Instance GUID	92
Troubleshooting Errors	92
Failed Installation	92
Cannot Create or Delete Service Instances	93
Broker Request Timeouts	93
Cannot Bind to or Unbind from Service Instances	94
Instance Does Not Exist	94

Other Errors	94
Cannot Connect to a Service Instance	94
Upgrade All Service Instances Failures	95
Missing Logs and Metrics	95
Failed Deployment on Upgrade or after Apply Changes	95
Troubleshooting Components	96
BOSH Problems	96
Large BOSH Queue	96
Configuration	96
Service Instances in Failing State	96
Authentication	96
UAA Changes	96
Networking	96
Validate Service Broker Connectivity to Service Instances	97
Validate App Access to Service Instance	97
Quotas	97
Plan Quota Issues	97
Global Quota Issues	97
Failing Jobs and Unhealthy Instances	98
Techniques for Troubleshooting	98
Parse a Cloud Foundry (CF) Error Message	98
Access Broker and Instance Logs and VMs	99
Access Broker Logs and VMs	99
Access Service Instance Logs and VMs	100
Run Service Broker Errands to Manage Brokers and Instances	100
Register Broker	101
Deregister Broker	101
Upgrade All Service Instances	101
Delete All Service Instances	102
Detect Orphaned Service Instances	103
Get Admin Credentials for a Service Instance	104
Reinstall a Tile	104
View Resource Saturation and Scaling	105
Identify a Service Instance Owner	105
Monitor the Quota Saturation and Service Instance Count	106
Drop and Restore AMQP(S) Traffic to a RabbitMQ Instance	106
Frequently Asked Questions	106
What should I check before deploying a new version of the tile?	106
What is the correct way to stop and start RabbitMQ in PCF?	107

What happens when I run bosh stop rabbitmq-server?	107
What happens when bosh stop rabbitmq-server fails?	107
What do I do when bosh stop rabbitmq-server fails?	107
How can I manually back up the state of the RabbitMQ cluster?	108
Back up Manually	108
Back up and Restore with a Script	108
What pre-upgrade checks should I do?	108
Knowledge Base (Community)	108
File a Support Ticket	108
 Operator Guide: Pre-Provisioned	 110
 Turning Off the Pre-Provisioned Service	 110
Disable the Pre-Provisioned Service Manually	110
Disable the Pre-Provisioned Service Using Automation	111
 Installing and Configuring	 111
 Installing and Configuring the Pre-Provisioned Service	 111
Role-Based Access in Ops Manager	112
Download and Install the Tile	112
Configure Pre-Provisioned RabbitMQ for PCF	112
Assign AZs and Networks	113
Pre-Provisioned RabbitMQ	114
RabbitMQ Admin User Credentials	114
Plugins	115
Erlang Cookie	115
Known Issues with Erlang Cookie	116
Security Issue with the Tile Generated Erlang Cookie	116
Cluster Scaling Known Issue	116
Changing the Erlang Cookie Value Known Issue	117
Enable Custom Policy on New Instances	117
External Load Balancer	117
HAProxy Ports	117
SSL	118
TLS Support	120
RabbitMQ Configuration	120
Select the Network Partition Behavior of the RabbitMQ Cluster	121
Disk Free Alarm Limit	121
Dangers of Setting This Value Too Low	122

Disadvantages of Setting This Value Too High	122
When to Use the 50MB Value	122
Specify Static IP Addresses	122
Configure Syslog Forwarding and Metrics Polling Interval	123
Global Settings	125
Dedicated Instance: Single Node Plan	125
Errands	125
Post-Deploy Errands	127
Pre-Delete Errands	128
Stemcell	128
Apply Configuration and Complete the Installation	128
Other Configuration Topics	128
Connecting to a Highly Available RabbitMQ Cluster	128
Port to protocol mappings	129
Security Groups	129
Application Security Groups	130
Application Container Network Connections	130
Assigned IPs	130
Preserving Dynamically Assigned IPs	130
RabbitMQ Server Settings that Cannot be Overwritten	131
Smoke Tests	132
Smoke Test Steps	132
Troubleshooting	132
Monitoring and KPIs for Pre- Provisioned RabbitMQ for PCF	132
Setting up Syslog Forwarding	133
Logging Formats	133
RabbitMQ Program Names	134
Metrics	135
Partition Indicator	135
Recovering from a network partition	135
Key Performance Indicators	135
Component Heartbeats	136
Service Broker Heartbeat	136
HAProxy Heartbeat	136
Server Heartbeat	137
RabbitMQ Server KPIs	137
File Descriptors	137
Erlang Processes	138

BOSH System Health Metrics	138
RAM	139
CPU	139
Ephemeral Disk	139
Persistent Disk	140
Component Metric Reference	140
RabbitMQ Server Metrics	140
HAProxy Metrics	142
Managing the Pre-Provisioned Service	143
Isolating Clusters with the RabbitMQ for PCF Replicator	143
Overview	143
Common Use Cases	143
Running SCS on a Dedicated RabbitMQ Cluster	144
Providing a Pre-Provisioned Dedicated Cluster	144
Using Replicas While Offering the On-Demand RabbitMQ Service	145
Blue-Green Upgrades (Advanced)	145
Generating Replica Tiles	145
Prerequisites	145
Download the Replicator	145
Generate Replica Tiles	145
Naming Conventions in Original and Replica Tiles	145
Installing Replica Tiles	146
Limiting Access to Replica Tiles to Specific Orgs	146
Upgrading Replica Tiles	147
Example of an In-Place Upgrade of a Replica	147
Limitations	149
Setting Default Policies for the RabbitMQ Service	149
About RabbitMQ Policies	149
Rules for Policies Set in the Tile	149
An Example Policy: Mirror on Two Nodes	149
Best Practice for Syncing Queues	150
Setting or Changing the Policy	150
Viewing Policies in the RabbitMQ Management UI	151
Frequently Asked Questions for Pre-Provisioned RabbitMQ for PCF	151
Frequently Asked Questions	152
What should I check before deploying a new version of the tile?	152

What is the correct way to stop and start RabbitMQ in PCF?	152
What happens when I run bosh stop rabbitmq-server?	152
What happens when bosh stop rabbitmq-server fails?	152
What do I do when bosh stop rabbitmq-server fails?	152
How can I manually back up the state of the RabbitMQ cluster?	153
Back up Manually	153
Back up and Restore with a Script	153
What pre-upgrade checks should I do?	153
Knowledge Base (Community)	153
File a Support Ticket	153
Using the RabbitMQ Management Dashboard	155
Access the RabbitMQ Management UI	155
Pre-Provisioned Service Instances	156
On-Demand Service Instances	156
Clustering and Network Partitions	158
Clustering in RabbitMQ for PCF	158
Automatic Network Partition Behaviors in RabbitMQ Clusters	158
Detecting a Network Partition	158
Recovering	159
Manually Synchronizing after a Partition	159
Upgrading RabbitMQ for PCF	160
Update Add-Ons to Run with Xenial Stemcell	160
About the Upgrade	160
General Notes About the Upgrade Process	160
Release Policy	161
Downtime When Upgrading	161
Upgrade RabbitMQ for PCF	162
Developer Guide	163
Using On-Demand RabbitMQ for PCF	163
Prerequisites	163
Developer Guide	163
Entries in the VCAP_SERVICES Environment Variable	163
The Create-Bind Process	164
Confirm Service Availability	164
Create a Service Instance	165
Using Transport Layer Security (TLS)	165

Configure TLS for Your Service Instance	166
Activate TLS for Java and Spring Apps	167
Bind a Service Instance to Your App	167
Use the RabbitMQ Service in Your App	168
Updating a Service Instance	168
Unbind a Service Instance to Your App	168
Delete a Service Instance	169
Create an Admin User for a Service Instance	169
Sharing Service Instances	169
Federate Exchanges and Queues	169
Shovel Exchanges and Queues	170
Modifying Apps for TLS	171
Prerequisites	171
Modify Your App for TLS	171
Repush or Rebind Your App	172
Migrating a RabbitMQ Service Instance to Another Service Instance	173
Migrate a RabbitMQ Service Instance to Another Service Instance	173
Migrate Policies and Parameters	173
Migrate Messages in Queues	175
Migrate Messages Using the RabbitMQ Shovel Plugin	175
Migrate Messages by Using Consumers to Drain the Old Service Instance	175
RabbitMQ Environment Variables	176
VCAP_SERVICES	176
Changing Enabled Plugins and Protocols	178
Troubleshooting On-Demand Instances	178
Troubleshoot Errors	178
Troubleshoot General Errors	178
Techniques for Troubleshooting	179
Parse a Cloud Foundry (CF) Error Message	179
Retrieve Service Instance Information	180
Retrieve RabbitMQ Instance Credentials	180
Error: Failed to Set Credentials in Credential Store	180
Knowledge Base (Community)	181
File a Support Ticket	181
Delete RabbitMQ Instances	181

RabbitMQ for PCF

RabbitMQ for PCF



Warning: RabbitMQ for PCF v1.14 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.

About RabbitMQ for PCF

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

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

This table summarizes the main differences between the two:

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



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

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

Element	Details
Version	1.14.14
Release date	June 17, 2019
Software component version	OSS RabbitMQ 3.7.14
Compatible Ops Manager version(s)	2.2.2 or later, 2.3, and 2.4
Compatible Pivotal Application Service version(s)	2.2, 2.3, and 2.4
IaaS support	AWS, Azure, GCP, OpenStack, and vSphere
IPsec support	No

Features

On-Demand

- Create up to five different on-demand RabbitMQ plans which can be provisioned through the Marketplace
- Choose whether a plan has one, three, five, or seven 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 dashboard access to app developers
- Deployment into an availability zone 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 colocated VMs to decrease deployment times

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

Pre-Provisioned

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

RabbitMQ for PCF and Other PCF Services

As well as RabbitMQ for PCF, other PCF 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 Pivotal Cloud Foundry services offer on-demand and pre-provisioned service plans:

Pivotal Cloud Foundry service tile	Standalone product related to the service	Supports on-demand	Supports pre-provisioned
RabbitMQ for PCF	Pivotal RabbitMQ	Yes	Yes. Only recommended for test environments.

Redis for PCF	Redis	Yes	Yes (shared-VM plan). Only recommended for test environments.
MySQL for Pivotal Platform	MySQL	Yes	No
Pivotal Cloud Cache (PCC)	Pivotal 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 provide any bugs, feature requests, or questions to the [PCF Feedback list](#).

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

RabbitMQ® for PCF Release Notes



Warning: RabbitMQ for PCF v1.14 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.

Upgrade to the Latest Version

Pivotal recommends that you upgrade to the latest version of your current minor line, then upgrade to the latest available version of the new minor line. For example, if you use an older v1.13.x version, upgrade to the latest v1.13.x version before upgrading to the latest v1.14.x version.

For product versions and upgrade paths, see the [Product Compatibility Matrix](#).

Pivotal Is Continuing Support for the Pre-Provisioned Service

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

For information about migrating from the pre-provisioned service to the on-demand service, see [About Migrating a Pre-Provisioned Instance to an On-Demand Instance](#). For instructions on how to install, configure, and deploy RabbitMQ for PCF as an on-demand service, see [Installing and Configuring RabbitMQ for PCF the On-Demand Service](#).

v1.14.14

Release Date: June 17, 2019

Resolved Issue

This release has the following fix:

- The pre-provisioned broker now terminates failed TCP connections when unbinding services. This prevents the file descriptor limit on the HAProxy and the pre-provisioned broker VMs from reaching capacity over time.

Known Issues

This release has the following known issues:

- **[Security Issue]** The method for generating the Erlang cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - ✦ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - ✦ If you have set the Erlang cookie explicitly in the tile configuration, you are not affected by this issue.
- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager might not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.14 - [Release notes](#)
- Erlang v20.3.8.21
- HAProxy v1.8.20

v1.14.13

Release Date: May 21, 2019

Resolved Issue

This release has the following fix:

- The pre-provisioned broker now terminates failed TCP connections for the bind, provision, and deprovision broker endpoints. This prevents the file descriptor limit on the HAProxy and the pre-provisioned broker VMs from reaching capacity over time.



Note: TCP connections still fail to terminate when unbinding services.

Known Issues

This release has the following known issues:

- The pre-provisioned broker does not terminate failed TCP connections when unbinding services. Over time, this can cause the file descriptor limit on the VMs that host HAProxy and the pre-provisioned broker to reach capacity:
 - ✦ When capacity is reached on the HAProxy VM, RabbitMQ is unreachable.
 - ✦ When capacity is reached on the pre-provisioned broker, app developers cannot create, bind, unbind, or delete service instances.

To resolve the issue, do one of the following:

- ✦ Restart the `haproxy` or the `rabbitmq-service-broker` monit job on the VM.
 - ✦ Increase the file descriptor limit on the VM. For how to increase the limit, see the example in [Unable to create RabbitMQ service-instance in PCF “too many open files”](#) in the Pivotal Support knowledge base.
 - ✦ If the VM is unresponsive, restart it.
- **[Security Issue]** The method for generating the Erlang cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - ✦ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - ✦ If you have set the Erlang cookie explicitly in the tile configuration, you are not affected by this issue.
 - Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
 - Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
 - When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
 - Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.14 - [Release notes](#)
- Erlang v20.3.8.20
- HAProxy v1.8.19

v1.14.12

Release Date: May 17, 2019

Features

New features and changes in this release:

- Errands and the pre-provisioned service broker can be deployed over HTTPS.
- Updates OSS RabbitMQ to 3.7.14

Security Fix

This release includes the following security fix:

- High [CVE-2019-3789: Gorouter allows space developer to hijack route services hosted outside the platform](#)

Resolved Issue

This release fixes the following issue:

- An issue with service metrics caused zombie processes to accumulate on `rabbitmq-server` VMs.

Known Issues

This release has the following known issues:

- The pre-provisioned broker does not terminate failed TCP connections when the request fails with a 4xx status code and when unbinding services. Over time, this can cause the file descriptor limit on the VMs that host HAProxy and the pre-provisioned broker to reach capacity:
 - ◊ When capacity is reached on the HAProxy VM, RabbitMQ is unreachable.
 - ◊ When capacity is reached on the pre-provisioned broker, app developers cannot create, bind, unbind, or delete service instances.

To resolve the issue, do one of the following:

- ◊ Restart the `haproxy` or the `rabbitmq-service-broker` monit job on the VM.
- ◊ Increase the file descriptor limit on the VM. For how to increase the limit, see the example in [Unable to create RabbitMQ service-instance in Pivotal Cloud Foundry “too many open files”](#) in the Pivotal Support knowledge base.
- ◊ If the VM is unresponsive, restart it.
- The pre-provisioned broker does not terminate failed TCP connections when the request fails with a 4xx status code and when unbinding services. Over time, this can cause the file descriptor limit on the VMs that host HAProxy and the pre-provisioned broker to reach capacity:
 - ◊ When capacity is reached on the HAProxy VM, RabbitMQ is unreachable.
 - ◊ When capacity is reached on the pre-provisioned broker, app developers cannot

create, bind, unbind, or delete service instances.

To resolve the issue, do one of the following:

- ✦ Restart the `haproxy` or the `rabbitmq-service-broker` monit job on the VM.
- ✦ Increase the file descriptor limit on the VM. For how to increase the limit, see the example in [Unable to create RabbitMQ service-instance in Pivotal Cloud Foundry “too many open files”](#) in the Pivotal Support knowledge base.
- ✦ If the VM is unresponsive, restart it.
- **[Security Issue]** The method for generating the Erlang cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - ✦ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - ✦ If you have set the Erlang cookie explicitly in the tile configuration, you are not affected by this issue.
- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.14
- Erlang v20.3.8.20
- HAProxy v1.8.19

v1.14.10

Release Date: March 28, 2019



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing](#)

Stemcells.



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Resolved Issues

This release fixes the following issue:

- Issues with the configuration of pre-provisioned RabbitMQ with external load balancers

Known Issues

This release has the following known issues:

- The pre-provisioned broker does not terminate failed TCP connections when the request fails with a 4xx status code and when unbinding services. Over time, this can cause the file descriptor limit on the VMs that host HAProxy and the pre-provisioned broker to reach capacity:
 - ✦ When capacity is reached on the HAProxy VM, RabbitMQ is unreachable.
 - ✦ When capacity is reached on the pre-provisioned broker, app developers cannot create, bind, unbind, or delete service instances.

To resolve the issue, do one of the following:

- ✦ Restart the `haproxy` or the `rabbitmq-service-broker` monit job on the VM.
- ✦ Increase the file descriptor limit on the VM. For how to increase the limit, see the example in [Unable to create RabbitMQ service-instance in Pivotal Cloud Foundry “too many open files”](#) in the Pivotal Support knowledge base.
- ✦ If the VM is unresponsive, restart it.
- **[Security Issue]** The method for generating the Erlang cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - ✦ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - ✦ If you have set the Erlang cookie explicitly in the tile configuration, you are not affected by this issue.
- An issue with service metrics causes zombie processes to accumulate on `rabbitmq-server` VMs. This can cause the VM to run out of resources and BOSH to restart it. If this occurs, Pivotal recommends that you stop the `service-metrics` job. However, if you stop the job, metrics will not be available.
- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).

- Changing networks or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.13
- Erlang v20.3.8.20
- HAProxy v1.8.19

v1.14.9

Release Date: March 14, 2019



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing Stemcells](#).



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Features

New features and changes in this release:

- Adds RabbitMQ 3.7.13

Known Issues

This release has the following known issues:

- The pre-provisioned broker does not terminate failed TCP connections when the request fails with a 4xx status code and when unbinding services. Over time, this can cause the file descriptor limit on the VMs that host HAProxy and the pre-provisioned broker to reach capacity:

- ✦ When capacity is reached on the HAProxy VM, RabbitMQ is unreachable.
- ✦ When capacity is reached on the pre-provisioned broker, app developers cannot create, bind, unbind, or delete service instances.

To resolve the issue, do one of the following:

- ✦ Restart the `haproxy` or the `rabbitmq-service-broker` monit job on the VM.
 - ✦ Increase the file descriptor limit on the VM. For how to increase the limit, see the example in [Unable to create RabbitMQ service-instance in Pivotal Cloud Foundry “too many open files”](#) in the Pivotal Support knowledge base.
 - ✦ If the VM is unresponsive, restart it.
- **[Security Issue]** The method for generating the Erlang cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - ✦ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - ✦ If you have set the Erlang cookie explicitly in the tile configuration, you are not affected by this issue.
 - An issue with service metrics causes zombie processes to accumulate on `rabbitmq-server` VMs. This can cause the VM to run out of resources and BOSH to restart it. If this occurs, Pivotal recommends that you stop the `service-metrics` job. However, if you stop the job, metrics will not be available.
 - Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
 - Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
 - When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
 - There are issues with the configuration of pre-provisioned RabbitMQ with external load balancers.
 - Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.13
- Erlang v20.3.8.20
- HAProxy v1.8.19

v1.14.8

Release Date: March 11, 2019



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing Stemcells](#).



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Features

New features and changes in this release:

- Adds the `recreate-all-service-instances` errand. For more information, see [Post-Deploy Errands](#).

Known Issues

This release has the following known issues:

- The pre-provisioned broker does not terminate failed TCP connections when the request fails with a 4xx status code and when unbinding services. Over time, this can cause the file descriptor limit on the VMs that host HAProxy and the pre-provisioned broker to reach capacity:
 - ✦ When capacity is reached on the HAProxy VM, RabbitMQ is unreachable.
 - ✦ When capacity is reached on the pre-provisioned broker, app developers cannot create, bind, unbind, or delete service instances.

To resolve the issue, do one of the following:

- ✦ Restart the `haproxy` or the `rabbitmq-service-broker` monit job on the VM.
- ✦ Increase the file descriptor limit on the VM. For how to increase the limit, see the example in [Unable to create RabbitMQ service-instance in Pivotal Cloud Foundry “too many open files”](#) in the Pivotal Support knowledge base.
- ✦ If the VM is unresponsive, restart it.
- **[Security Issue]** The method for generating the Erlang cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - ✦ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).

- If you have set the Erlang cookie explicitly in the tile configuration, you are not affected by this issue.
- An issue with service metrics causes zombie processes to accumulate on `rabbitmq-server` VMs. This can cause the VM to run out of resources and BOSH to restart it. If this occurs, Pivotal recommends that you stop the `service-metrics` job. However, if you stop the job, metrics will not be available.
- Cluster scaling or changing the `Erlang Cookie` value requires cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- There are issues with the configuration of pre-provisioned RabbitMQ with external load balancers.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.11
- Erlang v20.3.8.18
- HAProxy v1.8.17

v1.14.7

Release Date: February 13, 2019



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing Stemcells](#).



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Features

New features and changes in this release:

- Previously, when multiple Availability Zones (AZ) were selected for a single-node service plan, all instances of that plan would be deployed to the first AZ. Now the AZ is selected randomly from the configured list. This change does not affect existing instances and only applies to newly created instances.

Known Issues

This release has the following known issues:

- **[Security Issue]** The method for generating the Erlang cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - If you have set the Erlang cookie explicitly in the tile configuration, you are not affected by this issue.
- An issue with service metrics causes zombie processes to accumulate on `rabbitmq-server` VMs. This can cause the VM to run out of resources and BOSH to restart it. If this occurs, Pivotal recommends that you stop the `service-metrics` job. However, if you stop the job, metrics will not be available.
- Cluster scaling or changing the [Erlang Cookie](#) value requires cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks or IP addresses for the `RabbitMQ Server` job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.11
- Erlang v20.3.8.18
- HAProxy v1.8.17

v1.14.6

Release Date: December 28, 2018



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing Stemcells](#).



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Features

New features and changes in this release:

- Scripts to drop and restore AMQP(S) traffic. For more information, see [Drop and Restore AMQP\(S\) Traffic to a RabbitMQ Instance](#).

Fixed Issue

This release fixes the following issue:

- In RabbitMQ for PCF v1.14.5 and earlier, the maximum number of file descriptors was not set and this limited the maximum number of file descriptors used by RabbitMQ.

Known Issues

This release has the following known issues:

- **[Security Issue]** The method for generating the Erlang Cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - If you have set the Erlang Cookie explicitly in the tile configuration, you are not affected by this issue.
- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands

when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.

- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.9
- Erlang v20.3.8.15
- HAProxy v1.6.13

v1.14.5

Release Date: December 20, 2018



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing Stemcells](#).



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Features

New features and changes in this release:

- Fixed issue where RabbitMQ metrics were not visible on Loggregator Firehose after an upgrade
- Set swap to 1 GB for RabbitMQ nodes
- Supports the Log Cache cf CLI plugin that enables developers to access logs for an on-demand service instance using the command `cf tail`.

For more information about this feature, see [Access RabbitMQ Metrics for On-Demand Service Instances](#).

Known Issues

This release has the following known issues:

- **[Security Issue]** The method for generating the Erlang Cookie is not secure. For more

information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).

- ✦ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
- ✦ If you have set the Erlang Cookie explicitly in the tile configuration, you are not affected by this issue.
- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- The maximum number of file descriptors is not set. Therefore, RabbitMQ instances use the default value from the operating system.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.9
- Erlang v20.3.8.15
- HAProxy v1.6.13

v1.14.4

Release Date: November 22, 2018



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing Stemcells](#).



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Security Fixes

This release includes the following security fix:

- Critical [CVE-2018-15759: On Demand Services SDK Timing Attack Vulnerability](#)

Features

New features and changes in this release:

- Adds mutual TLS between service metrics and the Loggregator system
- Removes credentials from the dashboard URL in the pre-provisioned service

Known Issues

This release has the following known issues:

- **[Security Issue]** The method for generating the Erlang Cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - ◊ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - ◊ If you have set the Erlang Cookie explicitly in the tile configuration, you are not affected by this issue.
- If you upgrade to RabbitMQ for PCF v1.14, you cannot see RabbitMQ metrics on the Loggregator Firehose. This issue does not affect new installations.
- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.8
- Erlang v20.3.8.10
- HAProxy v1.6.13

v1.14.3

Release Date: November 1, 2018



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing Stemcells](#).



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Features

New features and changes in this release:

- Smoke tests are now more resilient when using external load balancers.

Known Issues

This release has the following known issues:

- **[Security Issue]** The method for generating the Erlang Cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - ✦ If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - ✦ If you have set the Erlang Cookie explicitly in the tile configuration, you are not affected by this issue.
- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.8
- Erlang v20.3.8.9
- HAProxy v1.6.13

v1.14.1 - Withdrawn

Release Date: October 2, 2018

This release has been removed from Pivotal Network because of an issue with open source RabbitMQ v3.7.7.



Breaking Change: RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. You must manually download and import a new stemcell into the Ops Manager Stemcell Library before deploying RabbitMQ for PCF v1.14. This might break automation you have set up to update RabbitMQ for PCF deployments that used Trusty stemcells. To download the Xenial stemcell from Pivotal Network, go to [Stemcells for PCF \(Ubuntu Xenial\)](#). For instructions on importing the stemcell, see [Importing and Managing Stemcells](#).



Breaking Change: Syslog forwarding uses RFC5424 format, and the Legacy Format is no longer available. If you are upgrading and have Legacy Format selected in your existing syslog configuration, that setting is automatically reconfigured to use the RFC5424 format.

Features

New features and changes in this release:

- The pre-provisioned RabbitMQ service is now optional. It is enabled by default for users who are upgrading from previous versions. It is disabled by default for all other users, but can be enabled in the configuration. For more information, see [Turning Off the Pre-Provisioned Service](#). If you use RabbitMQ for PCF for Spring Cloud Services, you must turn on the pre-provisioned RabbitMQ Service.
- The configuration tabs have been simplified to make it clearer which properties relate to the pre-provisioned service and which relate to the on-demand service.
- On-Demand instances with TLS enabled use HTTPS instead of HTTP inside the PCF network.
- RabbitMQ for PCF uses compiled BOSH releases, resulting in improved deployment times. However, you might still see some compilation when upgrading to v1.14. This is because if both compiled and source versions of the same BOSH release are available and the stemcell version used for deployment does not match the stemcell used for compilation, BOSH prefers to compile the source releases instead of using the compiled release.
- RabbitMQ for PCF now uses [floating stemcells](#). That means that RabbitMQ instances are recreated based on the new stemcell whenever a newer patch release of the stemcell

becomes available. You can opt out of using a given stemcell version for any specific tile or deployment using OpsManager [Stemcell Library](#).

Known Issues

This release has the following known issues:

- **[Security Issue]** The method for generating the Erlang Cookie is not secure. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).
 - If you rely on the tile to generate the cookie by leaving the Erlang cookie field blank, see [Security Issue with the Tile Generated Erlang Cookie](#).
 - If you have set the Erlang Cookie explicitly in the tile configuration, you are not affected by this issue.
- Cluster scaling or changing the [Erlang Cookie](#) value require cluster downtime, and might result in failed deployments. For more information, see [Cluster Scaling Known Issue](#) and [Changing the Erlang Cookie Value Known Issue](#).
- Changing networks and/or IP addresses for the [RabbitMQ Server](#) job results in a failed deployment. For more information, see [Changing Network or IP Addresses Results in a Failed Deployment](#).
- When errand run rules are set to **When Changed**, Ops Manager may not run the errands when the tile has relevant changes. For more information, see [Managing Errands in Ops Manager](#). Pivotal recommends leaving the default run rule set to **On**.
- When on-demand service instances are enabled for TLS, the `port` property in `VCAP_SERVICES` for `amqp+ssl` incorrectly reports that the port is `5672`. The actual port used for `amqp+ssl` is `5671`.
- Shareable instances are enabled in the **Global Settings for On-Demand Plans** tab for the pre-provisioned service as well as the on-demand service. For how to enable shareable instances, see [Configure Global Settings](#).

Packages

- OSS RabbitMQ v3.7.7
- Erlang v20.3.8.1
- HAProxy v1.6.13

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 PCF



Warning: RabbitMQ for PCF v1.14 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 two on-demand service plans.

Introduction

RabbitMQ for Pivotal Cloud Foundry (PCF) responds to the demands of PCF 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 for PCF cluster to meet their business requirements and empower app development teams to self-serve their own RabbitMQ cluster.

RabbitMQ for PCF 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 now offer their app developers three types of RabbitMQ for PCF 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 on 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.



Note: The RabbitMQ for PCF tile will only provide the on-demand service in the future. For more information, see [Deciding Which Service Plan to Use](#) below.

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

Deciding Which Service Plan to Use

In the future, Rabbit for PCF will only provide the on-demand service because it is designed for independent, isolated RabbitMQ instances. The existing pre-provisioned offering has many RabbitMQ instances on a single VM, in a multi-tenancy model. In this 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 Pivotal's plans for addressing these discrepancies. Pivotal encourages you to give feedback on how to meet your use case requirements with the on-demand service.

Feature	Pre-Provisioned Service	On-Demand Service
Configuration	Enabled via 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. See RabbitMQ Server Settings That Cannot Be Disabled .
RabbitMQ admin credentials to access RabbitMQ Management UI	Can set password using tile UI	Can access by creating a service key, see Create an Admin User for a Service Instance .
Erlang cookie	Operator can change, which has caused problems	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	Plan to address. IaaS-specific load balancers can still be used.
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	Uses client-side load-balancing.
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	No plans to address. Instant provisioning for on-demand is limited by the IaaS. This limitation will potentially be addressed by containers.
Individual instance upgrade	Possible when using tile replicator	Plan to address
Network partition behavior	Defaults to <code>pause_minority</code> . Configurable.	Defaults to <code>pause_minority</code> . Configurable for each plan.
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 Using RabbitMQ 3.7

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 plug-in.

This plan offers:

- Fast access to an isolated instance of RabbitMQ scoped for the app developer teams
- Org and Space Administrator 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 bug fixes
- Message resilience provided through RabbitMQ exchange, queue Federation, and Shovel plugins.

On-Demand Cluster Plan Using RabbitMQ 3.7

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 for PCF to multiple nodes.

This plan offers:

- Fast access to an isolated, clustered instance of RabbitMQ scoped to the app developer team Orgs and Spaces
- Administrator 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 bug 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 Ops 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 for PCF, 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 since 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 may decide to enable the single node on-demand plan across all app developer teams to meet their demand to isolate their workload. You may 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 for PCF 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 instance size (CPU and Memory)
- Set persistent disk size (Persisted Message Store) for the RabbitMQ 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 on PCF v2.0 or later, 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 on PCF v2.0 or later, 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 the Ops Manager, in the **Metrics polling interval** field, in the **Pre-Provisioned RabbitMQ** tab of the RabbitMQ for PCF 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 PCF](#). See also [Monitoring On-Demand RabbitMQ Clusters](#) below.
- **Logs**—RabbitMQ on-demand instance logs are forwarded using the same configuration as contained in the **Syslog and Metrics** tab of the RabbitMQ for PCF 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 via 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 PCF](#).

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 application 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 PCF 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 instance uses TLS, ensure that you enable TLS for on-demand instances. See [Configure 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 PCF v1.14 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 architecture for on-demand RabbitMQ® for Pivotal Cloud Foundry (PCF).

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

Service Network Requirement

When you deploy Pivotal Cloud Foundry, you must create a statically defined network to host the component virtual machines that constitute the Pivotal Cloud Foundry infrastructure.

Pivotal Cloud Foundry components, like the Cloud Controller and UAA, run on this infrastructure network. On-demand Pivotal Cloud Foundry services may require that you host them on a network that runs separately from the Pivotal Cloud Foundry default network. You can also deploy tiles on separate service networks to meet your own security requirement.

Pivotal Cloud Foundry v2.1 and later include dynamic networking. Operators can use this dynamic networking with asynchronous service provisioning to define dynamically-provisioned service networks. For more information, see [Default Network and Service Network](#).

In Pivotal Cloud Foundry v2.1 and later, on-demand services are enabled by default on all networks. Operators can create separate networks to host services in BOSH Director, but doing so is optional. Operators select which network hosts on-demand service instances when they configure the tile for that service.

Default Network and Service Network

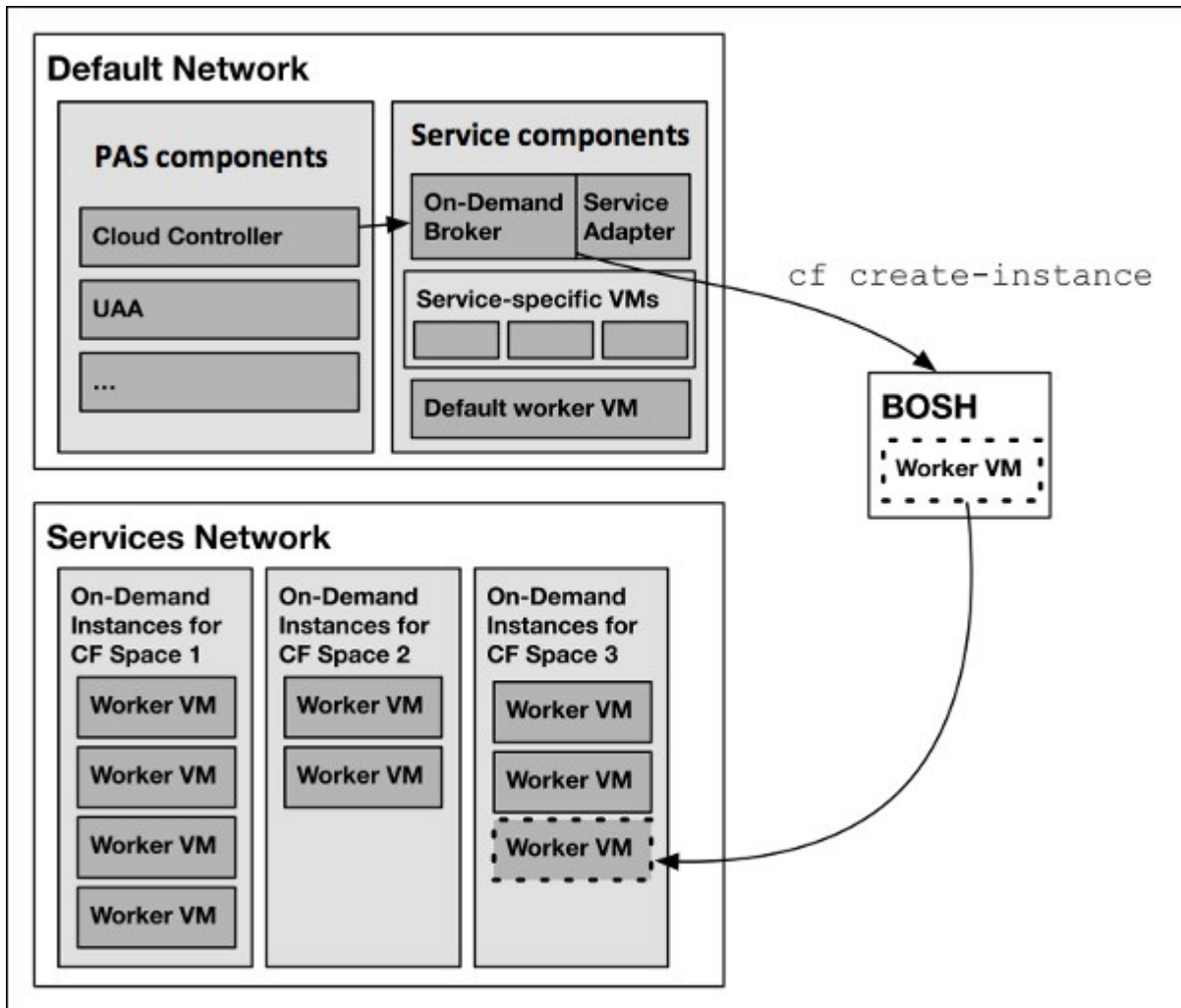
On-demand Rabbit for PCF services rely on the BOSH 2.0 ability to dynamically deploy VMs in a dedicated network. The on-demand service broker uses this capability to create single-tenant service instances in a dedicated service network.

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

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

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

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



Required Networking Rules for On-Demand Services

Before deploying a service tile that uses the on-demand service broker (ODB), request the needed network connections to allow components of Pivotal Cloud Foundry to communicate with ODB.

The specifics of how to open those connections varies for each IaaS.

See the following table for key components and their responsibilities in an on-demand architecture.

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

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

Source Component	Destination Component	Default TCP Port	Notes
ODB	BOSH Director	25555 (BOSH Director)	The default ports are not configurable.
	BOSH UAA	8443 (UAA)	
		8844 (CredHub)	
ODB	Deployed service instances	15672 (RabbitMQ Management UI)	This connection is for administrative tasks. Avoid opening general use, app-specific ports for this connection.
ODB	PAS	8443 (UAA)	The default port is not configurable.
Errand VMs	PAS	8443 (UAA)	The default port is not configurable.
	ODB	8080	
	Deployed Service Instances	15672 (RabbitMQ Management UI)	
		5672 (AMQP)	
		5671 (AMQPS)	

BOSH Agent	BOSH Director	4222	<p>The BOSH Agent runs on every VM in the system, including the BOSH Director VM.</p> <p>The BOSH Agent initiates the connection with the BOSH Director.</p> <p>The default port is not configurable.</p> <p>The communication between these components is two-way.</p>
Deployed apps on PAS	Deployed service instances	15672 (RabbitMQ Management UI) 5672 (AMQP) 5671 (AMQPS) 61613 (STOMP) 61614 (STOMPS) 1883 (MQTT) 8883 (MQTTS)	<p>This connection is for general use, app-specific tasks.</p> <p>Avoid opening administrative ports for this connection.</p>
PAS	ODB	8080	<p>This port may be different for individual services.</p> <p>This port may also be configurable by the operator if allowed by the tile developer.</p>
Deployed apps on PAS	Runtime CredHub	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 PCF v1.14 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

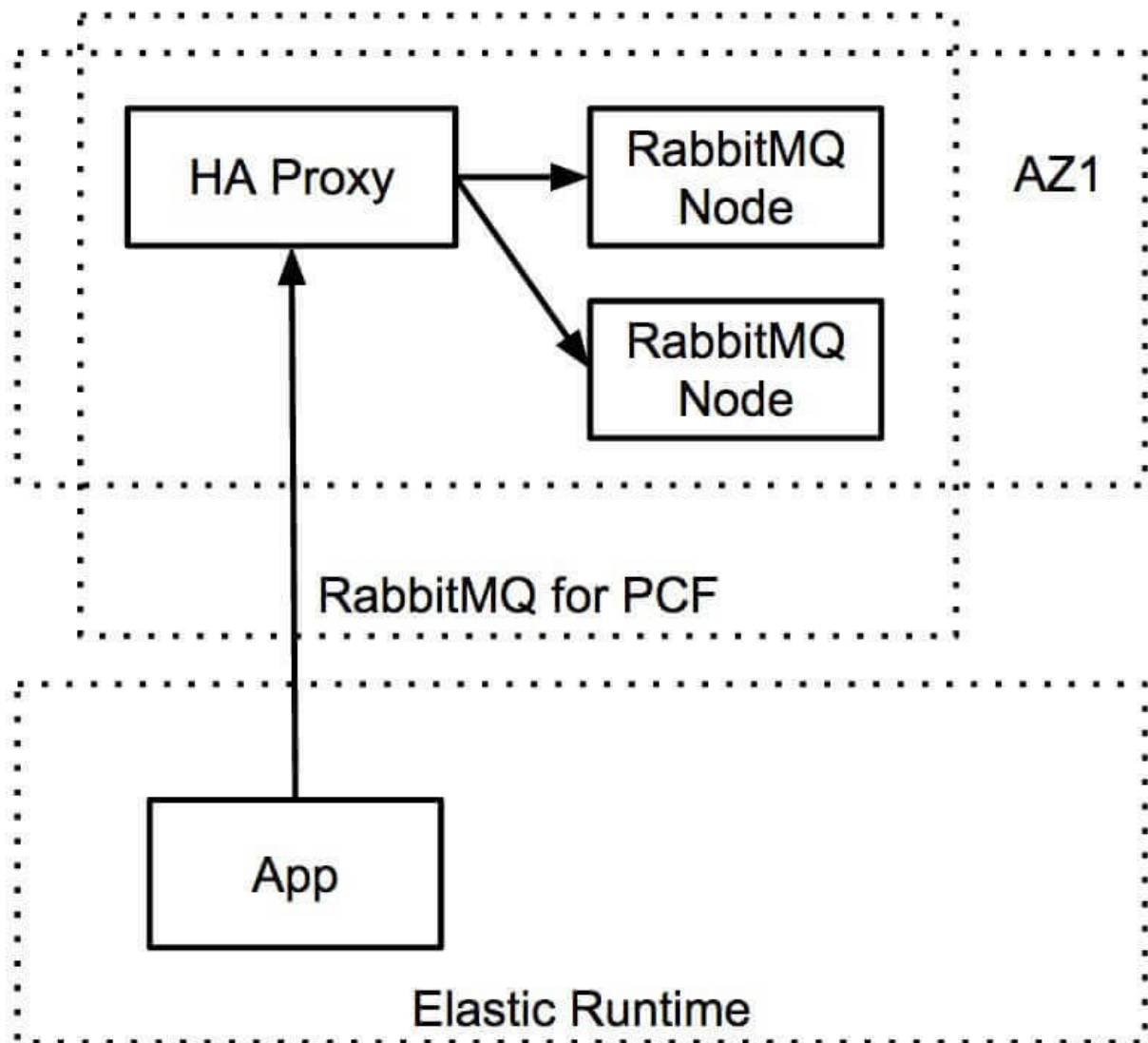
Deploying RabbitMQ for [Pivotal Cloud Foundry](#) (PCF) through Ops Manager deploys a RabbitMQ cluster of **3 nodes** by default.

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**



Considerations for this deployment

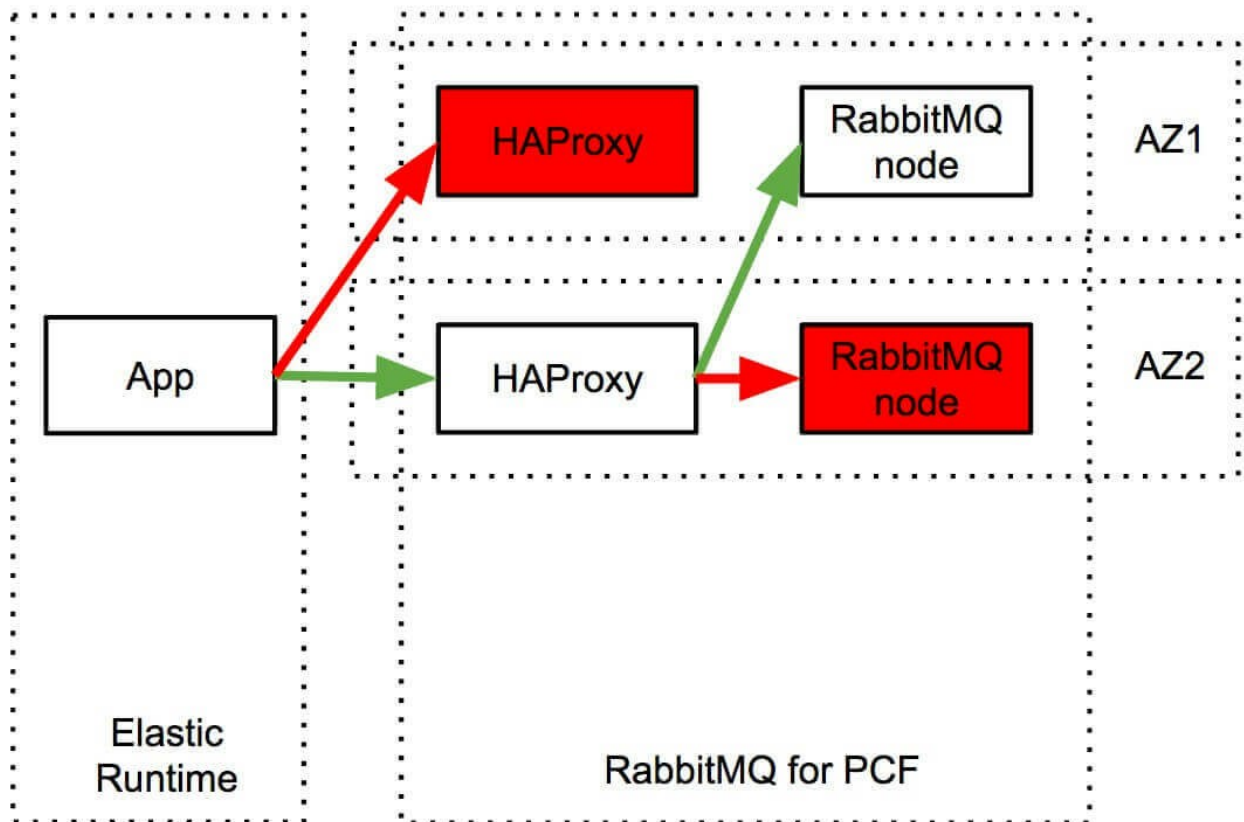
- Provides high availability for the RabbitMQ cluster
- Queues must be configured to be high availability as they are placed on one node by default
- Customers should decide which partition behavior is best suited to their use case. For two nodes 'automatic' is preferred.
- HAProxy is a single point 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 for PCF 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 above shows that you can now suffer the failure of a single HAProxy and single RabbitMQ node and still keep your cluster online and applications connected.

Upgrading to this deployment from a single AZ deployment

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

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

Upgrading to this deployment from a multi AZ deployment

If you have deployed the tile across two AZs, but with a single HAProxy instance, you can migrate to this setup by deploying an additional HAProxy instance through Ops Manager. New or re-bound applications to the RabbitMQ service see the IPs of both HAProxies immediately. Existing bound applications will continue to work, but only using the previously deployed HAProxy IP Address. They can be re-bound as required at your discretion.

Considerations for this deployment

- Requires IaaS configuration for AZs ahead of deploying the RabbitMQ tile
- Application developers are handed the IPs of each deployed HAProxy in their environment variables

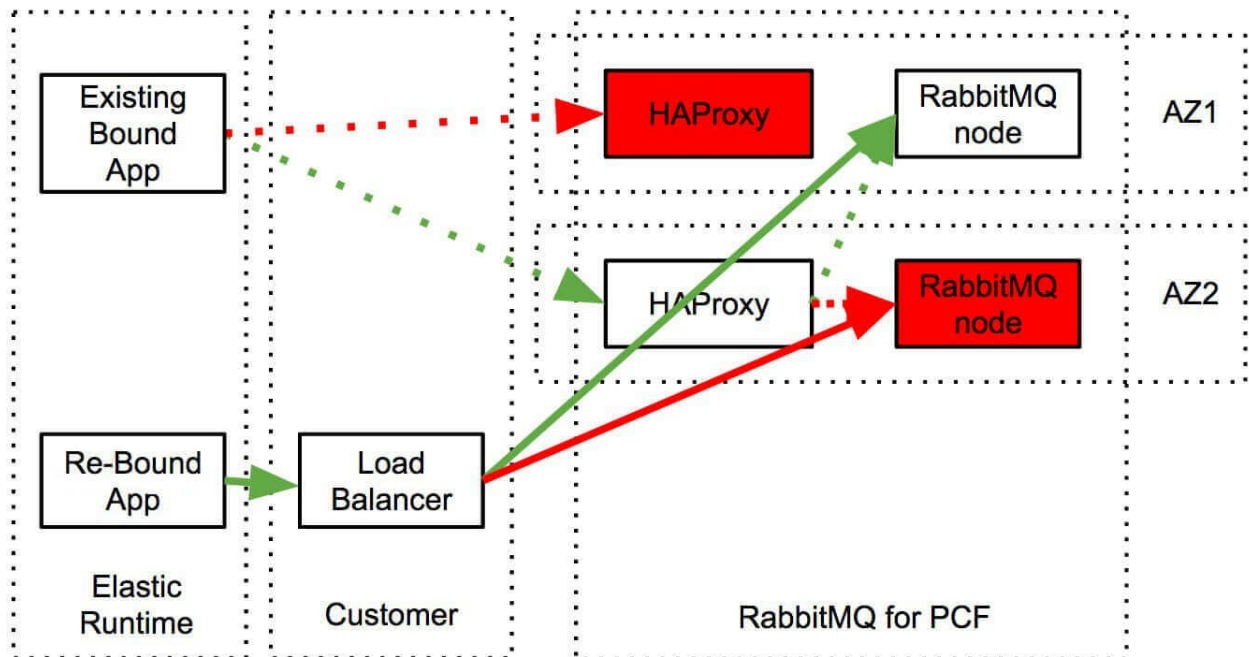
- Queues must be configured to be high availability as they are placed on one node by default
- Customers should decide on which partition behavior is best suited to their use case. For three or more nodes 'pause_minority' is preferred.

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 application developers.



Advantages

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

Disadvantages

- The load balancer needs to be configured with the IPs of the RabbitMQ Nodes. These 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 this deployment 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 applications. 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 instances of the RabbitMQ service **or** any re-bound connections will use the DNS name or IP address of the ELB in their `VCAP_SERVICES`. Any existing instances will continue to use the HAProxy IP addresses in their `VCAP_SERVICES`.
4. Phase the re-binding of existing applications to update their environment variables.
5. After all applications are updated, reduce the instance count of the `HAProxy` job in Ops Manager to 1.
6. Deploy the changes.

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

Downgrading from this deployment to the recommended deployment

If you are currently using an external load balancer, then you can move back to using 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.
- Changing the number of RabbitMQ nodes when the erlang cookie is not defined will restart the cluster. Check [here](#) for more information.

[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 PCF v1.14 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 an overview of how to prepare for using Transport Layer Security (TLS) with the RabbitMQ on-demand service to secure communication between apps and service instances. The topic also includes the procedure for providing an existing CA certificate to BOSH CredHub, or generating a new CA certificate with BOSH CredHub.



Note: To enable TLS for RabbitMQ for PCF, you must perform the procedures in [Provide or Generate a CA Certificate](#) below **before** installing and configuring the tile. This certificate is shared by multiple tiles. If you have already performed this procedure you do not need to repeat it. If you want to change this certificate, see [Rotating CA Certificates](#).

Overview

Enabling TLS provisions a RabbitMQ server with a certificate so that apps and clients can establish an encrypted connection with the data service. This certificate is a **server certificate** generated by **CredHub**, a component designed for centralized credential management in PCF, colocated on the BOSH Director.

CredHub generates the server certificate with a **Certificate Authority (CA)**. The operator can provide a certificate to CredHub or have CredHub generate a new certificate.

Apps and clients that communicate with the RabbitMQ server must have access to the public component of the CA certificate in order to use the certificate and to validate that the server certificate can be trusted.

PCF distributes the public component of the CA certificate to apps in two ways:

- PCF 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.

- PCF 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.

Workflow for Enabling and Using TLS

The following workflow is required to enable TLS and to allow developers to use it with their apps:

1. An operator provides a CA certificate to CredHub by completing the procedures in [Provide or Generate a CA Certificate](#) below.
2. An operator enables TLS in the tile configuration while installing RabbitMQ for PCF. See [Configure Security](#).
3. A developer enables TLS for an existing service instance. See [Configure TLS for Your Service Instance](#).
4. A developer modifies their app to communicate securely with the RabbitMQ server. See one of the following links, depending on the type of app:
 - ✦ For Java or Spring apps: [Activate TLS for Java and Spring Apps](#)
 - ✦ For other apps: [Modifying Apps for TLS](#)



Note: An operator must also rotate the CA certificate if it expires or if it becomes compromised. For information on how to rotate your CA certificate, see [Rotating CA Certificates](#).

Provide or Generate a CA Certificate

To enable and use TLS, you must complete the procedures below before installing and configuring the tile.



WARNING: This procedure requires restarting all of the VMs in your PCF deployment in order to apply a CA certificate. This operation can take a long time to complete.

Create a UAA Client

Follow these steps to create a UAA client for CredHub on your UAA server:

1. Retrieve the IP address of the BOSH Director VM and the Director credentials by completing the steps in [Gather Credential and IP Address Information](#).

Both the UAA and CredHub servers are colocated on the BOSH Director VM.

2. SSH into the Ops Manager VM by completing the steps in [SSH into Ops Manager VM](#).
3. From the Ops Manager VM, use the UAA Command Line Interface (UAAC) to target the

UAA server on the BOSH Director VM. In the UAAC command, specify the IP address for the BOSH Director VM and port 8443.

Run the following command:

```
uaac target BOSH-DIRECTOR:8443
```

Where **BOSH-DIRECTOR** is the IP address of the BOSH Director VM. You retrieved this address as part of the procedures in Step 1.

For example:

```
$ uaac target 10.0.0.5:8443
```

4. In the **Credentials** tab of the BOSH Director tile, retrieve and record the **identity** and **password** values for the following:

- ✦ **Uaa Login Client Credentials**
- ✦ **Uaa Admin User Credentials**



Note: These are the credentials for the UAA server colocated on the BOSH Director, not the UAA server colocated on Pivotal Application Service.

5. From the Ops Manager VM, use the UAAC to get a token.

Run the following command:

```
uaac token owner get login --secret=UAA-LOGIN-CLIENT-CRED
```

Where **UAA-LOGIN-CLIENT-CRED** is the **password** value of the **Uaa Login Client Credentials** that you retrieved in Step 4 above.

For example:

```
$ uaac token owner get \  
login --secret=abcdefghijklm123456789
```

6. When prompted for a user name and password, enter the values for **identity** and **password** of the **Uaa Admin User Credentials** you retrieved in step 5 above.

For example:

```
User name:  admin  
Password:  *****
```

7. Add a UAA client for CredHub with the correct grants.

Run the following command:

```
uaac client add \
  --authorized_grant_types client_credentials \
  --authorities credhub.read,credhub.write
```

- When prompted for Client ID, enter `credhub`. When prompted for `New client secret`, enter a secure password of your choice.

For example:

```
Client ID: credhub
New client secret: *****
Verify new client secret: *****
scope: uaa.none
client_id: credhub
resource_ids: none
authorized_grant_types: client_credentials
autoapprove:
authorities: credhub.write credhub.read
name: credhub
required_user_groups:
lastmodified: 1518198701452
id: credhub
created_by: f609e861-39ec-4a16-8aee-cba9e9b079e3
```

Add the CA Certificate

Follow these steps to log in to CredHub, provide or generate a CA certificate, and add the certificate to Ops Manager:

- From the Ops Manager VM, set the API target of the CredHub CLI to your CredHub server.

Run the following command:

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

Where `BOSH-DIRECTOR` 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
```

- Log in to CredHub.

Run the following command:

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

Where `CLIENT-SECRET` is the `New client secret` you set in step 8 in the procedure above.

For example:

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

3. Use the CredHub CLI to check whether a services CA certificate already is present.

Run the following command:

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

If you already have a certificate at the `services/tls_ca` path, skip step 4.

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



Note: Your PCF deployment may have multiple CA certificates. Pivotal recommends a dedicated CA certificate for services.

- ✦ If you do not have a CA certificate, use the CredHub CLI to generate one.

Run the following command:

```
credhub generate \
  --name="/services/tls_ca" \
  --type="certificate" \
  --no-overwrite \
  --is-ca \
  --common-name="rootCA"
```

- ✦ 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 enter the following command, specifying the path to `root.pem` and the private key for the certificate.

For example:

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

5. Use the BOSH CLI v2 to extract the `certificate` portion from the CA certificate and print it.

Run the following command:

```
bosh int <(credhub get \
  --name=/services/tls_ca) \
  --path /value/certificate
```

6. Copy the output.
7. Navigate to the Ops Manager **Installation Dashboard** and select the BOSH Director tile. Click **Security**.
8. Paste the contents of the CA certificate into **Trusted Certificates**, and then click **Save**.
9. After preparing your environment for TLS, you must enable TLS in the tile configuration, following the appropriate step below.



WARNING: Restarting all of the VMs in your PCF deployment in order to apply a CA certificate takes a long time to complete.

- ✦ **For an Existing Installation:** Complete the steps in [Configure Security](#) to enable TLS in the tile, click **Review Pending Changes**, and then click **Apply Changes** in the Installation Dashboard. This restarts all the VMs in your PCF deployment and applies your CA certificate.



Note: Make sure the `upgrade-all-service-instances` errand is set to **On**. This errand must be run to ensure all service instances have the needed changes.

- ✦ **For a New Installation:** Complete the procedures in [Installing and Configuring RabbitMQ for PCF](#), including enabling TLS in the **Security for On-Demand Plans** tab, click **Review Pending Changes**, and then click **Apply Changes** in the Installation Dashboard. This deploys RabbitMQ for PCF, restarts all the VMs in your PCF 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 PCF v1.14 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 Pivotal Cloud Foundry (PCF) operators about how to install, configure, and deploy the RabbitMQ for PCF 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 for PCF 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 Ops 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 allow 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 for PCF 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 for PCF



Note: Before using RabbitMQ for PCF v1.14 and later, you must update any BOSH add-ons to support Xenial stemcells. See [Update Add-ons to Run with Xenial Stemcell](#).

1. Download the product file from [Pivotal Network](#).
2. Navigate to the Ops Manager Installation Dashboard and click **Import a Product** to upload the product file.
3. Under the **Import a Product** button, click + next to the version number of RabbitMQ for PCF. This adds the tile to your staging area.
4. Click the newly added **RabbitMQ for PCF** tile. This lets you begin configuring the tile. The installation is complete when you apply the changes from the configuration.

Configure On-Demand RabbitMQ for PCF

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

tile.

RabbitMQ

Settings

Status

Credentials

Logs

✓ Assign AZs and Networks

✓ Pre-Provisioned RabbitMQ

✓ Global Settings for On-Demand Plans

✓ Security for On-Demand Plans

✓ On Demand Instance (RMQ 3.7): Plan 1

✓ On Demand Instance (RMQ 3.7): Plan 2

✓ On Demand Instance (RMQ 3.7): Plan 3

✓ On Demand Instance (RMQ 3.7): Plan 4

✓ On Demand Instance (RMQ 3.7): Plan 5

✓ Syslog and Metrics

✓ Errands

✓ Resource Config

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
Place singleton on jobs in	Select the region that you want for singleton VMs. PCF creates the RabbitMQ broker in this AZ.
Balance other jobs in	Select additional region. This selection does not affect the on-demand RabbitMQ for PCF service.
Network	<p>Select a network for the RabbitMQ On-Demand Broker.</p> <p>This should be a separate network from the one you select for Service Network. For more information about the Default Network, see Default Network and Service Network.</p> <p>Typically, you select the network used for Pivotal Application Service (PAS) components.</p>
Service Network	<p>Select a separate network that the on-demand service instances run on.</p> <p>A typical practice is to put all on-demand services on a single network, separate from the network that Pivotal Application Service (PAS) and the On-Demand Broker run on. For information about the Service Network, see Default Network and Service Network.</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>
<div>  <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.</p> </div>	

3. Click **Save**.

Configure Logging and Monitoring

Pivotal recommends that you configure logging to monitor the health of RabbitMQ for PCF. 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.

2. Configure the following:

- ✦ **Service instance quota** min: 0, max: 50 set the total number of on-demand service instances which can be deployed. For more information, see [Setting Limits for On-Demand Service Instances](#).

- ✦ **VM options:**

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



Note: Outbound network traffic rules also depend on your IaaS settings. Consult your network or IaaS administrator to ensure that your IaaS allows outbound traffic to the external networks you need.

- ✦ **(Beta) Shareable Instances:** Click **Yes** to enable the beta feature for sharing instances.



Note: This is a beta feature based on an experimental feature in

Cloud Foundry. Use the feature at your own risk in non-production environments. If you try this feature, please send your comments and feedback to [PCF Feedback List](#).

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 \(Beta\)](#).

- **(Beta) On Demand - Secure Service Instance Credentials with Runtime CredHub:** For on-demand services instances, click **Yes** to secure credentials with CredHub.



Note 1: For this feature to work you must also enable it in PAS. For instructions, see [Step 1: Configure the PAS Tile](#). After deploying the tile, notify developers that they must unbind and rebind existing service instances to secure their credentials with CredHub.

Note 2: This is a beta feature. Use it at your own risk in non-production environments. If you try this feature, please send your comments and feedback to [PCF Feedback List](#).

- **On Demand Service Broker Static IP:** Enter an IP address to assign to your on-demand service broker node. BOSH allocates an IP address if the field is left blank.

3. Click **Save**.

Configure Security

RabbitMQ for PCF 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.

Use this page to configure security settings for the RabbitMQ for PCF service.

TLS Options*



Not Configured



Optional - Developers may configure their service VMs to use TLS

Save

3. Under **TLS Options**, select **Optional**. This enables developers to configure their RabbitMQ

service instances to use TLS.

4. Click **Save**.
5. If you are using Ops Manager v2.3 or later, click **Review Pending Changes**. For more information about this Ops Manager page, see [Reviewing Pending Product Changes](#).
6. In the Installation Dashboard, make sure 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 instructions to developers, see [Configure 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:

On Demand Instance: Plan 1 (complete required fields even if you disable this plan):

Plan 1 Configuration

Please read the documentation before changing any of these settings, as improper use can lead to data loss.

CF Service Access*

Enable Service Access

Plan Name *

single-node

Plan Description *

Single node RabbitMQ dedicated instance

The plan description that will appear in the CF marketplace

Plan Features *

RabbitMQ

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

10

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

1

Network Partition Behaviour*

pause_minority

AZ Placement *

☒ us-central1-a

☒ us-central1-b

☐ us-central1-c

RabbitMQ VM Type*

micro (cpu: 1, ram: 1 GB, disk: 8 GB)

Persistent Disk Type*

2 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

On Demand Instance: Plan 2, 3, 4, and 5:

Plan 2 Configuration

Please read the documentation before changing any of these settings, as improper use can lead to data loss.

Enable This Plan *

☐ Plan Disable

☒ Plan Enable

CF Service Access *

Enable Service Access

Plan Name *

cluster

Plan Description *

RabbitMQ dedicated cluster

The plan description that will appear in the CF marketplace

Plan Features *

RabbitMQ

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

10

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

3

Network Partition Behaviour *

pause_minority

AZ Placement *

☒ us-central1-a

☒ us-central1-b

☐ us-central1-c

RabbitMQ VM Type *

micro (cpu: 1, ram: 1 GB, disk: 8 GB)

Persistent Disk Type *

2 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	(Plans 2 - 5 only) To enable, select Plan Enable .
CF Service Access	<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 Enable Service Access. 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 On-Demand Instance Smoke Tests errand to Off. For more information, see Errands.</p> <ul style="list-style-type: none"> ♦ Enable Service Access—Gives access to all orgs, and displays the service plan to all developers in the Marketplace. ♦ Disable Service Access—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. ♦ Leave Service Access Unchanged—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 Controlling Access to Service Plans by Org.
Plan Name	Accept the default or enter a name. This is the name that appears in the Marketplace.
Plan Description	Accept the default or enter a description. This description appears in the Marketplace.
Plan Features	Accept the default or enter a description. This description appears in Apps Manager.
Service Instance Quota	<p>Enter the maximum number of on-demand service instances that can be available at one time. For more information, see Setting Limits for On-Demand Service Instances.</p>
Number of Nodes	Enter 1, 3, 5 or 7. This setting only affects new service instances. Previously deployed service instances are not updated.
Network Partition Behaviour	Select <code>pause_minority</code> or <code>autoheal</code> . Pivotal recommends using pause minority. For more information, see Consistency or Availability Tradeoff .
RabbitMQ VM Type	Select a large VM type. The plan creates a service instance of this size. For more information, see Understanding RabbitMQ VM Types and Persistent Disk Size below.
Persistent Disk Type	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 Understanding RabbitMQ VM Types and Persistent Disk Size below.

AZ Placement	<p>This field is available after you complete the Assign AZs and Networks page.</p> <ul style="list-style-type: none"> For a single-node plan, select one or more AZs. 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 Network Latency and Consistency or Availability Tradeoff for details. <p>If you change this selection after deployment, existing instances are not affected by the change. See Determine which AZs a Service Instance Uses.</p>
---------------------	--

- 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.

Understanding RabbitMQ VM Types and Persistent Disk Size

The **RabbitMQ VM Type** and **Persistent disk type** are required fields on the service plan configuration pages. If you are installing on PCF v2.0 or later, 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 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 memory (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 (1.5xRAM)	Suggested disk size (2xRAM)
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. If you are using Ops Manager v2.3 or later, 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 for PCF.

Errands

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

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

Pending Changes

Revert

▼ INSTALL RabbitMQ

☒

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

Apply changes

Review Pending Changes

BETA

If you are using Ops Manager v2.3 or later, 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 for PCF 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 for PCF 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 for PCF **Settings** tab as well as the defaults for [pre-delete](#) errands.

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

RabbitMQ for PCF errands are colocated 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
Pre-Provisioned Broker Registrar	Makes the pre-provisioned RabbitMQ service plans available in the Marketplace
Pre-Provisioned Smoke Tests	Checks that a pre-provisioned RabbitMQ service instance can be bound to a Cloud Foundry app, and that the app can publish and subscribe to a RabbitMQ cluster. See Smoke Tests .

Register On Demand Service Broker	Makes the on-demand RabbitMQ service plans available in the Marketplace. If you change the Service Plan Configuration, you must run this errand in order for the changes to be reflected in the Marketplace.
On Demand Instance Smoke Tests	Checks that on-demand RabbitMQ service instances can be bound to a Cloud Foundry app, and that the app can publish and subscribe to a RabbitMQ cluster. Smoke tests only run against Plan 1. See Smoke Tests .
Upgrade All Service Instances	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 Off or When Changed , updates to on-demand plan settings will not be applied to existing service instances. Pivotal strongly recommends that this errand is configured to always run.
Recreate All Service Instances	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. This errand is off by default and should be enabled only when you want to recreate a VM.

Pre-Delete Errands

Errand	Description
Deregister and Purge Instances	Removes the pre-provisioned RabbitMQ service from the Marketplace and deletes all associated service instances and bindings. For more information, see Turning Off the Pre-Provisioned Service .
Delete All Service Instances	Unbinds and deletes existing on-demand service instances. The duration of this errand depends on the number of deployed on-demand instances.
Deregister On-Demand Service Broker	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 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/#/login/admin-username/admin-password",
  "username": "admin-username",
  "password": "admin-password",
  ...
}
```

RabbitMQ Server Settings That Cannot Be Disabled

The following plugins are enabled by default and cannot be disabled:

- `rabbitmq_management`
- `rabbitmq_federation`
- `rabbitmq_federation_management`
- `rabbitmq_shovel`
- `rabbitmq_shovel_management`

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

Smoke Tests



Warning: RabbitMQ for PCF v1.14 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 PCF 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 PCF



Warning: RabbitMQ for PCF v1.14 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 Cloud Foundry (PCF) service using the logs, metrics, and Key Performance Indicators (KPIs) generated by RabbitMQ for PCF component VMs.

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



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 PCF.

Set up Syslog Forwarding and Metrics Polling Interval

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

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

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

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

RabbitMQ

Settings Status Credentials Logs

Assign AZs and Networks

Pre-Provisioned RabbitMQ

Global Settings for On-Demand Plans

Security for On-Demand Plans

On Demand Instance (RMQ 3.7): Plan 1

On Demand Instance (RMQ 3.7): Plan 2

On Demand Instance (RMQ 3.7): Plan 3

On Demand Instance (RMQ 3.7): Plan 4

On Demand Instance (RMQ 3.7): Plan 5

Syslog and Metrics

Errands

Resource Config

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

Metrics polling interval (min: 10) *

10

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

☐ No

☒ Yes

Syslog address *

Syslog port *

22282

Transport protocol *

TCP

☐ Enable TLS

Permitted Peer

Custom CA Certificate

Save

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

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

Metrics polling interval	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. Changing this setting affects all deployed instances.
Syslog address	IP or DNS address of the syslog server
Syslog port	Port of the syslog server
Transport protocol	Transport protocol of the syslog server. One of <code>udp</code> , <code>tcp</code> , <code>relp</code> .
Enable TLS	Enable TLS to the syslog server.
Permitted Peer	If there are several peer servers that can respond to remote syslog connections, then you may provide a wildcard in the domain, such as <code>*.example.com</code> .
Custom CA Certificate	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.
- If you are using Ops Manager v2.3 or later, 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 for PCF logging configured, two types of components generate logs: the server nodes and the service broker:

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

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

For users familiar with documentation for previous versions of the tile, the tag we used to call the `app_name` is now called the `program_name`.

The generic log format is as follows:

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

The raw logs look similar to the following:

```
<7>2017-06-28T16:06:10.733560+00:00 10.244.16.133 vcap.agent [job=rmq index=0 id=e37ec
```



```
dca-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 may be presented in a different format.

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

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

RabbitMQ Program Names

Program Name	Description
<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 PAS deployment.
<code>rabbitmq_server</code>	The Erlang VM and RabbitMQ apps. <i>Logs may span multiple lines.</i>
<code>rabbitmq_server_drain</code>	Shuts down the Erlang VM and RabbitMQ apps. Runs as part of the BOSH lifecycle.

Program Name	Description
<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.

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**). 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 for PCF on-demand service.

Key Performance Indicators

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

Pivotal provides the following KPIs as general alerting and response guidance for typical RabbitMQ for PCF installations. Pivotal recommends that operators continue to fine-tune the alert measures to

their installation by observing historical trends. Pivotal also recommends that operators expand beyond this guidance and create new, installation-specific monitoring metrics, thresholds, and alerts based on learning from their own installations.

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

Component Heartbeats

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

Service Broker Heartbeat

p.rabbitmq/service_broker/heartbeat

Description	<p>RabbitMQ Service Broker <code>is alive</code> poll, which indicates if the component is available and able to respond to requests.</p> <p>Use: 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>Origin: Doppler/Firehose</p> <p>Type: boolean</p> <p>Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	<p>Yellow warning: N/A</p> <p>Red critical: < 1</p>
Recommended response	<p>Check the RabbitMQ Service Broker logs for errors. You can find this VM by targeting your RabbitMQ deployment with BOSH and running the command. <code>bosh -d service-instance_GUID vms</code></p>

Server Heartbeat

p.rabbitmq/rabbitmq/heartbeat

Description	<p>RabbitMQ Server <code>is alive</code> poll, which indicates if the component is available and able to respond to requests.</p> <p>Use: If the server does not emit heartbeats, this indicates that it is offline. To be functional, service instances require RabbitMQ Server.</p> <p>Origin: Doppler/Firehose</p> <p>Type: boolean</p> <p>Frequency: 30 s (default), 10 s (configurable minimum)</p>
--------------------	--


Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	Yellow warning: N/A Red critical: < 1
Recommended response	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

Description	<p>File descriptors consumed.</p> <p>Use: If the number of file descriptors consumed becomes too large, the VM might lose the ability to perform disk IO, which can cause data loss.</p> <div>  <p>Note: This assumes non-persistent messages are handled by retries or some other logic by the producers.</p> </div> <p>Origin: Doppler/Firehose Type: count Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	Yellow warning: > 250000 Red critical: > 280000
Recommended response	The default <code>ulimit</code> for RabbitMQ for PCF 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

Description	<p>Erlang processes consumed by RabbitMQ, which runs on an Erlang VM.</p> <p>Use: This is the key indicator of the processing capability of a node.</p> <p>Origin: Doppler/Firehose Type: count Frequency: 30 s (default), 10 s (configurable minimum)</p>
--------------------	--

Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	Yellow warning: > 900000 Red critical: > 950000
Recommended response	The default Erlang process limit in RabbitMQ for PCF v1.6 and later is 1,048,816. If this metric meets or exceeds the recommended thresholds for extended periods of time, consider scaling the RabbitMQ nodes in the tile Resource Config pane.

BOSH System Health Metrics

The BOSH layer that underlies Pivotal Cloud Foundry generates `healthmonitor` metrics for all VMs in the deployment. As of Pivotal Cloud Foundry 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 for PCF components, and serve as KPIs for the RabbitMQ for PCF service.

RAM

`system.mem.percent`

Description	RAM being consumed by the <code>p.rabbitmq</code> VM. Use: 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. Origin: BOSH HM Type: percent Frequency: 30 s (default), 10 s (configurable minimum)
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	Yellow warning: > 40 Red critical: > 50
Recommended response	Add more consumers to drain the queue as fast as possible.

CPU

`system.cpu.percent`

Description	<p>CPU being consumed by the <code>p.rabbitmq</code> VM.</p> <p>Use: 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>Origin: BOSH HM Type: percent Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p>Yellow warning: > 60 Red critical: > 75</p>
Recommended response	Remember that “an empty rabbit is a happy rabbit” . Add more consumers to drain the queue as fast as possible.

Ephemeral Disk

`system.disk.percent`

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

Persistent Disk

`persistent.disk.percent`

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

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

RabbitMQ Server Metrics

RabbitMQ for PCF message server components emit the following metrics.

Full Name	Unit	Description
/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 CA Certificates



Warning: RabbitMQ for PCF v1.14 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 are using TLS for RabbitMQ for PCF, you provided a CA certificate to BOSH CredHub when performing the procedures in [Provide or Generate a CA Certificate](#).

To rotate this CA certificate, see [Rotating CA Certificates for Pivotal Cloud Foundry Services](#) in the Pivotal Support knowledge base.



Warning: Do not rotate a CA certificate on your own. Contact [Pivotal Support](#) and follow the procedure in the Knowledge Base article with their help.

[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 PCF v1.14 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.

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 Pivotal Cloud Foundry (PCF) 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 PCF service independently. For example, if you have Redis for PCF and RabbitMQ for PCF, 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 PCF services and sets the maximum number of service instances an organization can create within PCF. 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 PCF 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 PCF services and sets the maximum number of service instances that can be created within a given space in PCF. 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 PCF 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-s
ervice-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 for PCF on-demand plan, do the following:

1. Navigate to **Ops Manager Installation Dashboard > RabbitMQ for PCF > Settings**.
2. Click the section for the plan you want to view. For example, **On Demand Instance (RMQ 3.7): 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 PCF v1.14 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 PCF



Warning: RabbitMQ for PCF v1.14 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 Cloud Foundry (PCF).

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.

Troubleshooting Errors

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

Failed Installation

1. **Certificate issues:** The on-demand broker (ODB) requires valid certificates. Ensure that your certificates are valid and generate new ones if necessary. To generate new certificates, contact [Pivotal Support](#).
2. **Deploy fails:** Deploys can fail for a variety of reasons. View the logs using Ops Manager to determine why the deploy is failing.
3. **[Networking problems:](#)**
 - ◆ Cloud Foundry cannot reach the RabbitMQ for PCF broker
 - ◆ Cloud Foundry cannot reach the service instances

- The service network cannot access the BOSH director
- 4. [Register broker errand](#) fails.
- 5. The smoke test errand fails.
- 6. Resource sizing issues: These occur when the resource sizes selected for a given plan are less than RabbitMQ for PCF requires to function. Check your resource configuration in Ops Manager and ensure that the configuration matches that recommended by the service.
- 7. Other service-specific issues.

Cannot Create or Delete Service Instances

If developers report errors such as:

```
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
```

Follow these steps:

1. If the BOSH error shows a problem with the deployment manifest, open the manifest in a text editor to inspect it.
2. To continue troubleshooting, [Log in to BOSH](#) and target the RabbitMQ for PCF service instance using the instructions on [parsing a Cloud Foundry error message](#).
3. Retrieve the BOSH task ID from the error message and run the following command:

```
bosh task TASK-ID
```

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

Broker Request Timeouts

If developers report errors such as:

```
Server error, status code: 504, error code: 10001, message: The request to the service broker timed out: https://BROKER-URL/v2/service_instances/e34046d3-2379-40d0-a318-d54fc7a5b13f/service_bindings/aa635a3b-ef6d-41c3-a23f-55752f3f651b
```

Follow these steps:

1. Confirm that Cloud Foundry (CF) is [connected to the service broker](#).
2. Check the BOSH queue size:
 1. Log into BOSH as an admin.
 2. Run `bosh tasks`.

3. If there are a large number of queued tasks, the system may be under too much load. BOSH is configured with two workers and one status worker, which may not be sufficient resources for the level of load. Advise app developers to try again once the system is under less load.

Cannot Bind to or Unbind from Service Instances

Instance Does Not Exist

If developers report errors such as:

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

Follow these steps:

1. Confirm that the RabbitMQ for PCF service instance exists in BOSH and obtain the GUID CF by running:

```
cf service MY-INSTANCE --guid
```

2. Using the GUID obtained above, the following BOSH CLI command:

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

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

Other Errors

If developers report errors such as:

```
Server error, status code: 502, error code: 10001, message: Service broker error: There was a problem completing your request. Please contact your operations team providing the following information: service: example-service, service-instance-guid: 8d69de6c-88c6-4283-b8bc-1c46103714e2, broker-request-id: 15f4f87e-200a-4b1a-b76c-1c4b6597c2e1, operation: bind
```

To find out the exact issue with the binding process:

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

Cannot Connect to a Service Instance

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

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

1. Check that [application security groups](#) are configured correctly. Access should be configured for the service network that the tile is deployed to.
2. Ensure that the network the Pivotal Application Service (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.
3. In Ops Manager go into the service tile and see the service network that is configured in the networks tab.
4. In Ops Manager go into the PAS tile and see the network it is assigned to. Make sure that these networks can access each other.

Upgrade All Service Instances Failures

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

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

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

Missing Logs and Metrics

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

1. Ensure you have configured syslog for the tile.
2. Ensure that you have network connectivity between the networks that the tile is using and the syslog destination. If the destination is external, you need to use the [public ip](#) VM extension feature available in your Ops Manager tile configuration settings.
3. Verify that the Firehose is emitting metrics:
 1. Install the `cf nozzle` plugin. For instructions, see the [firehose plugin](#) GitHub repository.
 2. To find logs from your service in the `cf nozzle` output, run the following:

```
cf nozzle -f ValueMetric | grep --line-buffered "on-demand-broker/MY-SERVICE"
```

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

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

Failed Deployment on Upgrade or after Apply Changes

If the deployment fails after editing the **Assign AZs and Networks** pane of the RabbitMQ for PCF tile,

it might be due to a change to the IP addresses assigned to the [RabbitMQ Server](#) job. RabbitMQ for PCF requires that these IP addresses do not change once assigned. If you change them, the deployment fails. This includes changes made to your current installation or during an upgrade.

To diagnose and solve this issue, see [Changing Network or IP Addresses Results in a Failed Deployment](#).

Troubleshooting 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 for PCF service instance sees `create in progress` in the Cloud Foundry Command Line Interface (cf CLI) until BOSH processes the queued request.

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

Configuration

Service Instances in Failing State

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

Authentication

UAA Changes

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

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



Note: You must ensure that any changes to UAA credentials are reflected in the Ops Manager `credentials` tab of the Pivotal Application Service (PAS) tile.

Networking

Common issues with networking include:

Issue

Solution

Latency when connecting to the RabbitMQ for PCF service instance to create or delete a binding.	Try again or improve network performance.
Firewall rules are blocking connections from the RabbitMQ for PCF service broker to the service instance.	Open the RabbitMQ for PCF tile in Ops Manager and check the two networks configured in the Networks pane. Ensure that these networks allow access to each other.
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. To SSH into the RabbitMQ for PCF service broker, run the following command:

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

2. If no BOSH `task-id` appears in the error message, look in the broker log using the `broker-request-id` from the task.

Validate App Access to Service Instance

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

Quotas

Plan Quota Issues

If developers report errors such as:

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

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

Global Quota Issues

If developers report errors such as:

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

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

Failing Jobs and Unhealthy Instances

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

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

For additional information, run the following command:

```
bosh 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 Pivotal 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 into 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 service-instance_GUID ssh
```

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

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

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

Log Name	Description
broker.log	Requests to the on-demand broker and the actions the broker performs while orchestrating the request (e.g. generating a manifest and calling BOSH). Start here when troubleshooting.

broker_ctl.log	Control script logs for starting and stopping the on-demand broker.
post-start.stderr.log	Errors that occur during post-start verification.
post-start.stdout.log	Post-start verification.
drain.stderr.log	Errors that occur while running the drain script.

Access Service Instance Logs and VMs

1. To target an individual service instance deployment, retrieve the GUID of your service instance with the following cf CLI command:

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

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

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

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

The errand does the following:

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

To run the errand, run the following command:

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

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

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

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

- Collects all of the service instances the on-demand broker has registered
- For each instance the errand does the following serially

- ✦ Issues an upgrade command to the on-demand broker
- ✦ Regenerates the service instance manifest based on its latest configuration from the tile
- ✦ Deploys the new manifest for the service instance
- ✦ Waits for this operation to complete, then proceeds to the next instance
- Adds to a retry list any instances that have ongoing BOSH tasks at the time of upgrade
- Retries any instances in the retry list until all are upgraded

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

To run the errand, do one of the following:

- Select the errand through the Ops Manager UI and have it run when you click **Apply Changes**.
- Run the following command.

```
bosh -d DEPLOYMENT-NAME run-errand upgrade-all-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, Pivotal 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

1. [Identify the service deployment by GUID.](#)
2. [Log in to BOSH.](#)
3. Open the manifest in a text editor.
4. Look in the manifest for the credentials.

Reinstall a Tile

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

1. Ensure that the previous tile was correctly uninstalled as follows:
 1. Log in as an admin by running:

```
cf login
```

2. Confirm that the Marketplace does not list RabbitMQ for PCF 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 for PCF 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 a Service Instance Owner

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

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(S) Traffic to a RabbitMQ Instance

While debugging a RabbitMQ 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 instance, do the following:

1. To stop all AMQP(S) traffic to a RabbitMQ 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(S) 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 in PCF?

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 [Pivotal Knowledge Base](#).

File a Support Ticket

You can file a ticket with [Pivotal 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 PCF v1.14 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 PCF 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
```

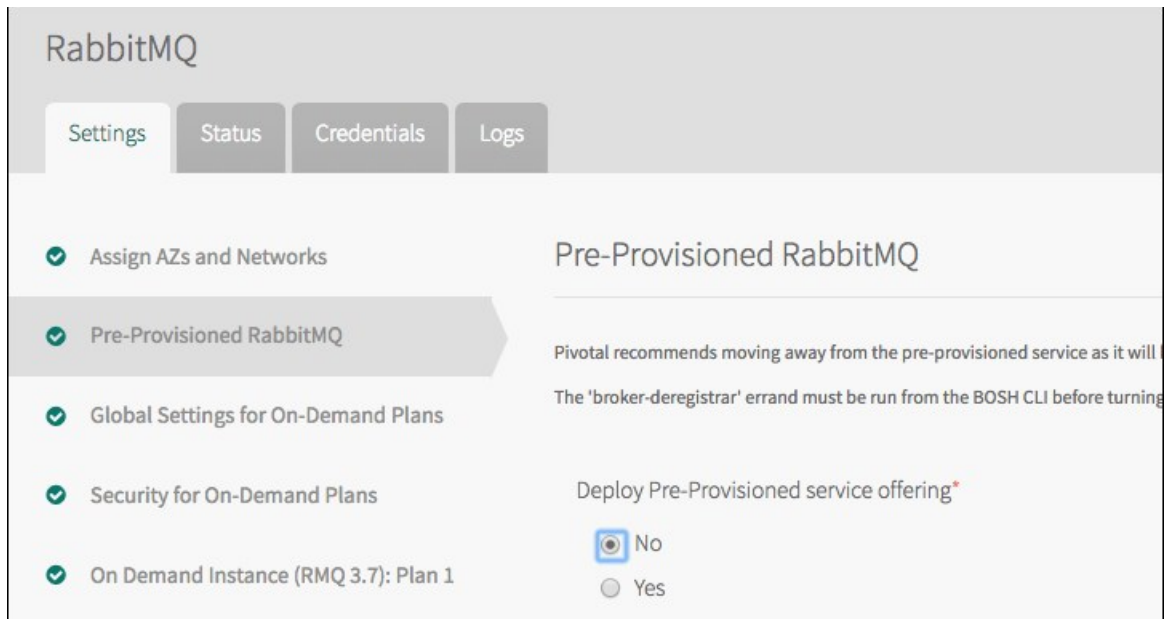
Where `GUID` is the RabbitMQ for PCF deployment GUID.

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 Ops 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. If you are using Ops Manager v2.3 or later, 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 for PCF 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 PCF v1.14 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 Pivotal Cloud Foundry (PCF) operators about how to install, configure, and deploy the RabbitMQ for PCF 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 Ops 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 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



Note: Before using RabbitMQ for PCF v1.14 and later, you must update any BOSH add-ons to support Xenial stemcells. See [Update Add-ons to Run with Xenial Stemcell](#).

1. Download the product file from [Pivotal Network](#).
2. Navigate to the Ops Manager Installation Dashboard and click **Import a Product** to upload the product file.
3. Under the **Import a Product** button, click **+** next to the version number of RabbitMQ for PCF. This adds the tile to your staging area.
4. Click the newly added **RabbitMQ for PCF** tile. This lets you begin configuring the tile. The installation is complete when you apply the changes from the configuration.

Configure Pre-Provisioned RabbitMQ for PCF

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

RabbitMQ

Settings
Status
Credentials
Logs

Assign AZs and Networks

Pre-Provisioned RabbitMQ
Global Settings for On-Demand Plans
Security for On-Demand Plans
On Demand Instance (RMQ 3.7): Plan 1
On Demand Instance (RMQ 3.7): Plan 2
On Demand Instance (RMQ 3.7): Plan 3
On Demand Instance (RMQ 3.7): Plan 4
On Demand Instance (RMQ 3.7): Plan 5
Syslog and Metrics
Errands
Resource Config

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](#) below.

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
Place singlet on jobs in	Select a region. This selection only affects the on-demand service.
Balanc e other jobs in	Select additional region(s). This selection only affects the pre-provisioned service.
Netwo rk	Select a network for the RabbitMQ Broker. This should be a separate network from the one you select for Service Network . This network is represented by the Default Network, described in Default Network and Service Network . Typically, you select the network used for Pivotal Application Service (PAS) components.
Servic e Netwo rk	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, in the [Settings](#) screen, click **Pre-Provisioned RabbitMQ**.

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

Plugins

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 if you want to connect directly to the RabbitMQ cluster, for example with `rabbitmqctl`, or to connect other VMs running Erlang.



Note: Leaving this field blank is a security risk. See [Security Issue with the Tile Generated Erlang Cookie](#) below.

Erlang cookie used by RabbitMQ nodes and rabbitmqctl

Known Issues with Erlang Cookie

There are three known issues associated with the Erlang cookie:

- [Security Issue with the Tile Generated Erlang Cookie](#)
- [Cluster Scaling Known Issue](#)
- [Changing the Erlang Cookie Value Known Issue](#)

Security Issue with the Tile Generated Erlang Cookie

If you leave the Erlang cookie field blank, the tile generates the cookie in a way that can be reverse-engineered. For more information about this security issue, see [CVE-2018-1279: RabbitMQ cluster compromise due to deterministically generated cookie](#).

To avoid this issue, set the Erlang cookie to a secure password value. This requires cluster downtime, see [Changing the Erlang Cookie Value Known Issue](#) below.

Cluster Scaling Known Issue

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

1. Follow the steps in the link below, up to and including the section *Log in to the BOSH Director*: [Advanced Troubleshooting with the BOSH CLI](#)

2. Run the following command:

```
bosh ssh rabbitmq-server/0
```

3. Run the following commands:

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

4. Paste the value returned from the last command into the Erlang cookie field in the **Pre-Provisioned RabbitMQ** tab. This field is shown [above](#).
5. To increase the size of your cluster, navigate to the **Resource Config** tab and, in the first row, raise the value for the number of **Instances** of the **RabbitMQ node**.
6. Return to the Ops Manager Installation Dashboard.
7. If you are using Ops Manager v2.3 or later, 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 the known issue below does **not** apply.

Changing the Erlang Cookie Value Known Issue

Changing the Erlang cookie value requires cluster downtime. Pivotal *strongly* 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.

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](#).

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**.

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 the following:
 - ✦ 15672 - Management
 - ✦ 5672 - AMQP
 - ✦ 5671 - AMQP+SSL
 - ✦ 1883 - MQTT

- ✦ 8883 - MQTT+SSL
 - ✦ 61613 - STOMP
 - ✦ 61614 - STOMP+SSL
 - ✦ 15674 - WebSTOMP
- You must leave the management plugin listening on port 15672 and load balance that port.
 - If you change the topology of your RabbitMQ cluster, the HAProxy is automatically reconfigured during the deployment.

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

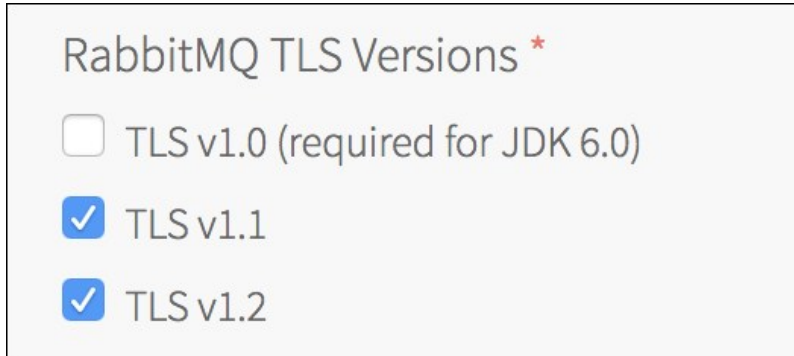
- SSL is simultaneously provided for AMQPS, STOMP and MQTT. No other plugins are automatically configured for use with SSL.
- If you provide SSL keys and certificates, non-SSL support is disabled. If you previously deployed this service without SSL support and have apps connected to the service, these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

RabbitMQ configuration

```
WwogIHtyYWJiaXRtcV9tYW5hZ2VtZW50LCBbCiAgICB7cmF0ZXNfbW9kZSwgZGV0YWlsZWR9CiAgXX0KXS4K
```

RabbitMQ config file contents, can be blank

You can see an example `rabbitmq.config` file [here](#). 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 HAProxies spread across low latency availability zones.

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:

Disk free alarm limit*

☐ 50MB (legacy)

☒ 100% Memory (default)

☐ 150% Memory (recommended)

☐ 200% Memory (conservative)

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

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 not recommended and 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 may:
 - ✦ Write persistent messages to disk (using up some disk space).
 - ✦ Accept more memory-based messages into various queues.
 - ✦ Page all memory-based messages to disk.

In 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 does not recommend 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 that RabbitMQ crashes and loses data.

Specify Static IP Addresses

To specify static IP addresses, do the following:

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** drop-down list 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.

Static IP Allocations

Please read the documentation before changing any of these settings, as improper use can lead to data loss.

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 for PCF, 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 for PCF log messages, do the following:

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

RabbitMQ

Settings Status Credentials Logs

✓ Assign AZs and Networks

✓ Pre-Provisioned RabbitMQ

✓ Global Settings for On-Demand Plans

✓ Security for On-Demand Plans

✓ On Demand Instance (RMQ 3.7): Plan 1

✓ On Demand Instance (RMQ 3.7): Plan 2

✓ On Demand Instance (RMQ 3.7): Plan 3

✓ On Demand Instance (RMQ 3.7): Plan 4

✓ On Demand Instance (RMQ 3.7): Plan 5

✓ Syslog and Metrics

✓ Errands

✓ Resource Config

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

Metrics polling interval (min: 10) *

10

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

☐ No

☒ Yes

Syslog address *

Syslog port *

22282

Transport protocol*

TCP

☐ Enable TLS

Permitted Peer

Custom CA Certificate

Save

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

Field	Description
Metrics polling interval	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. Changing this setting affects all deployed instances.
Syslog addresses	IP or DNS address of the syslog server
Syslog port	Port of the syslog server
Transport protocol	Transport protocol of the syslog server. One of <code>udp</code> , <code>tcp</code> , <code>relp</code> .

Enable TLS	Enable TLS to the syslog server.
Permitted Peer	If there are several peer servers that can respond to remote syslog connections, then you may provide a wildcard in the domain, such as <code>*.example.com</code> .
Custom CA Certificate	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.
- If you are using Ops Manager v2.3 or later, 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 beta 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 \(Experimental\)](#).



WARNING: This is a beta feature based on an experimental feature in Cloud Foundry. Use the feature at your own risk in non-production environments.

- Click **Global Settings**.
- Select **(Beta) Shareable Instances** to enable the beta feature for sharing instances.

☒ [Beta] Shareable Instances

App Developers can have the ability to share their Cloud Foundry Service Instances across Orgs and Spaces

- 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 on 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.

Errands can be thought of as tasks. For example, when deploying or updating RabbitMQ for PCF, Ops Manager can optionally run a series of [post-deploy errands](#). An example is the **Smoke Tests** errand, which checks the health of the RabbitMQ cluster after a deploy or upgrade.

You can decide whether to run post-deploy errands by toggling them on or off before you click **Apply Changes** to update a configuration in the Ops Manager Installation Dashboard:

Pending Changes

[↶ Revert](#)


▼ INSTALL RabbitMQ

- ☒ 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

[Apply changes](#)[Review Pending Changes](#) **BETA**

If you are using Ops Manager v2.3 or later, 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

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



Warning: In RabbitMQ for PCF 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 for PCF might fall behind if the Upgrade All Service Instances errand is not on by default.

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

Post-Deploy Errands

Errand	Description
Pre-Provisioned Broker Registrar	Makes the pre-provisioned RabbitMQ service plans available in the Marketplace
Pre-Provisioned Smoke Tests	Checks that a pre-provisioned RabbitMQ service instance can be bound to a Cloud Foundry app, and that the app can publish and subscribe to a RabbitMQ cluster. See Smoke Tests .
Register On Demand Service Broker	Makes the on-demand RabbitMQ service plans available in the Marketplace. If you change the Service Plan Configuration, you must run this errand in order for the changes to be reflected in the Marketplace.

On Demand Instance Smoke Tests	Checks that on-demand RabbitMQ service instances can be bound to a Cloud Foundry app, and that the app can publish and subscribe to a RabbitMQ cluster. See Smoke Tests .
Upgrade All Service Instances	On-Demand instances are updated and redeployed if there are changes to the Dedicated Instance settings or the tile is upgraded. If this errand is set to Off or When Changed , updates to Dedicated Instance settings will not be applied to existing service instances. Pivotal recommends that this errand is configured to always run.
Recreate All Service Instances	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. This errand is off by default and should be enabled only when you want to re-create a VM.

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
Deregister and Purge Instances	Removes the pre-provisioned RabbitMQ service from the Marketplace and deletes all associated service instances and bindings. For more information, see Turning Off the Pre-Provisioned Service .
Delete All Service Instances	Unbinds and deletes existing dedicated service instances. The duration of this errand depends on the number of deployed on-demand instances.
Deregister On-Demand Service Broker	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. If you are using Ops Manager v2.3 or later, 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 for PCF.

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 will be 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 for 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 Cloud Foundry security groups: [

```
{ "protocol": "tcp", "destination": "<haproxy-node-IP-
```

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

Application Security Groups

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



Note: The service is unusable without Application Security Groups.

Application Container Network Connections

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

Destination	Ports	Protocol	Reason
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

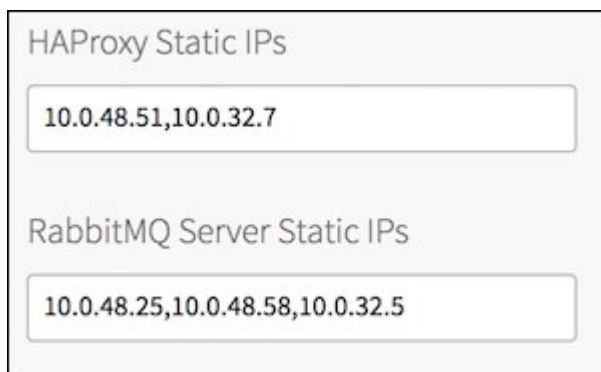
RabbitMQ for PCF does not support changing the IP addresses which have been assigned to the RabbitMQ deployments. For example, you cannot change the subnet into which the RabbitMQ cluster was originally provisioned. Doing so causes the deployment to fail. For more information, see [Changing Network or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

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, follow these steps:

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 PCF v1.14 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 PCF 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 PCF



Warning: RabbitMQ for PCF v1.14 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 Cloud Foundry (PCF) service using the logs, metrics, and Key Performance Indicators (KPIs)

generated by RabbitMQ for PCF component VMs.

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

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

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

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

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

For users familiar with documentation for previous versions of the tile, the tag we used to call the `app_name` is now called the `program_name`.

The generic log format is as follows:

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

The raw logs look similar to the following:

```
<7>2017-06-28T16:06:10.733560+00:00 10.244.16.133 vcap.agent [job=rmq index=0 id=e37ecdca-5b10-4141-abd8-e1d777dfd8b5] 2017/06/28 16:06:10 CEF:0|CloudFoundry|BOSH|1|agent_api|ssh|1|duser=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 may be presented in a different format.

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

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

RabbitMQ Program Names

Program Name	Description
<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 (PAS) deployment.
<code>rabbitmq_route_registrar_stdout</code>	Registers the route for the management API with the Gorouter in your PAS deployment.
<code>rabbitmq_server</code>	The Erlang VM and RabbitMQ apps. <i>Logs may 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.

Program Name	Description
<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**). 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) since each node is expected to emit 3 (2 reachable nodes and itself). Otherwise, an alert can be sent to the team.

Recovering from a network partition

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

Key Performance Indicators

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

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

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

Component Heartbeats

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

Service Broker Heartbeat

p-rabbitmq.service_broker.heartbeat

Description	<p>RabbitMQ Service Broker is alive poll, which indicates if the component is available and able to respond to requests.</p> <p>Use: 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>Origin: Doppler/Firehose Type: boolean Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	<p>Yellow warning: N/A Red critical: < 1</p>
Recommended response	<p>Check the RabbitMQ Service Broker logs for errors. You can find this VM by targeting your RabbitMQ deployment with BOSH and running the following command:</p> <pre>bosh -d service-instance_GUID vms</pre>

HAProxy Heartbeat

p-rabbitmq.haproxy.heartbeat

Description	<p>RabbitMQ HAProxy is alive poll, which indicates if the component is available and able to respond to requests.</p> <p>Use: If the HAProxy does not emit heartbeats, this indicates that it is offline. To be functional, service instances require HAProxy.</p> <p>Origin: Doppler/Firehose Type: boolean Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	<p>Yellow warning: N/A Red critical: < 1</p>
Recommended response	<p>Check the RabbitMQ HAProxy logs for errors. You can find the VM by targeting your RabbitMQ deployment with BOSH and running the following command, which lists HAProxy_GUID:</p> <pre>bosh -d service-instance_GUID vms</pre>

Server Heartbeat

p-rabbitmq.rabbitmq.heartbeat


Description	<p>RabbitMQ Server is alive poll, which indicates if the component is available and able to respond to requests.</p> <p>Use: If the server does not emit heartbeats, this indicates that it is offline. To be functional, service instances require RabbitMQ Server.</p> <p>Origin: Doppler/Firehose Type: boolean Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 5 minutes
Recommended alert thresholds	<p>Yellow warning: N/A Red critical: < 1</p>
Recommended response	<p>Check the RabbitMQ Server logs for errors. You can find the VM by targeting your RabbitMQ deployment with BOSH and running one of the following commands, which lists rabbitmq:</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

Description	<p>File descriptors consumed.</p> <p>Use: If the number of file descriptors consumed becomes too large, the VM may lose the ability to perform disk IO, which can cause data loss.</p> <div>  <p>Note: This assumes non-persistent messages are handled by retries or some other logic by the producers.</p> </div> <p>Origin: Doppler/Firehose Type: count Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p>Yellow warning: > 250000</p> <p>Red critical: > 280000</p>
Recommended response	<p>The default <code>ulimit</code> for RabbitMQ for PCF 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"> • Scaling the rabbit nodes in the tile Resource Config pane. • Reduce the load on the server

Erlang Processes

p-rabbitmq.rabbitmq.erlang.erlang_processes

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

BOSH System Health Metrics

The BOSH layer that underlies Pivotal Cloud Foundry generates `healthmonitor` metrics for all VMs in the deployment. As of Pivotal Cloud Foundry 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 for PCF components, and serve as KPIs for the RabbitMQ for PCF

service.

RAM

system.mem.percent

Description	<p>RAM being consumed by the <code>p-rabbitmq</code> VM.</p> <p>Use: 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>Origin: BOSH HM Type: percent Frequency: 30 s (default), 10 s (configurable minimum)</p>
Recommended measurement	Average over last 10 minutes
Recommended alert thresholds	<p>Yellow warning: > 40</p> <p>Red critical: > 50</p>
Recommended response	Add more consumers to drain the queue as fast as possible.

CPU

system.cpu.percent

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

Ephemeral Disk

system.disk.percent

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

Persistent Disk**persistent.disk.percent**

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

Component Metric Reference

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

RabbitMQ Server Metrics

RabbitMQ for PCF message server components emit the following metrics.

Full Name	Unit	Description
/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/system/memory	MB	The memory in MB used by the node
/p-rabbitmq/rabbitmq/system/mem_alarm	boolean	Indicates if the memory alarm went off
/p-rabbitmq/rabbitmq/system/disk_free_alarm	boolean	Indicates if the disk free alarm went off
/p-rabbitmq/rabbitmq/system/disk_free	MB	The disk space available on the node
/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
/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
/p-rabbitmq/rabbitmq/queues/VHOST-NAME/QUEUE-NAME/consumers	count	The number of consumers per virtual host per queue
/p-rabbitmq/rabbitmq/queues/VHOST-NAME/QUEUE-NAME/depth	count	The number of messages with the status <code>messages_unacknowledged</code> or <code>messages_ready</code> per virtual host per queue

HAProxy Metrics

RabbitMQ for PCF HAProxy components emit the following metrics.

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

[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 PCF Replicator



Warning: RabbitMQ for PCF v1.14 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

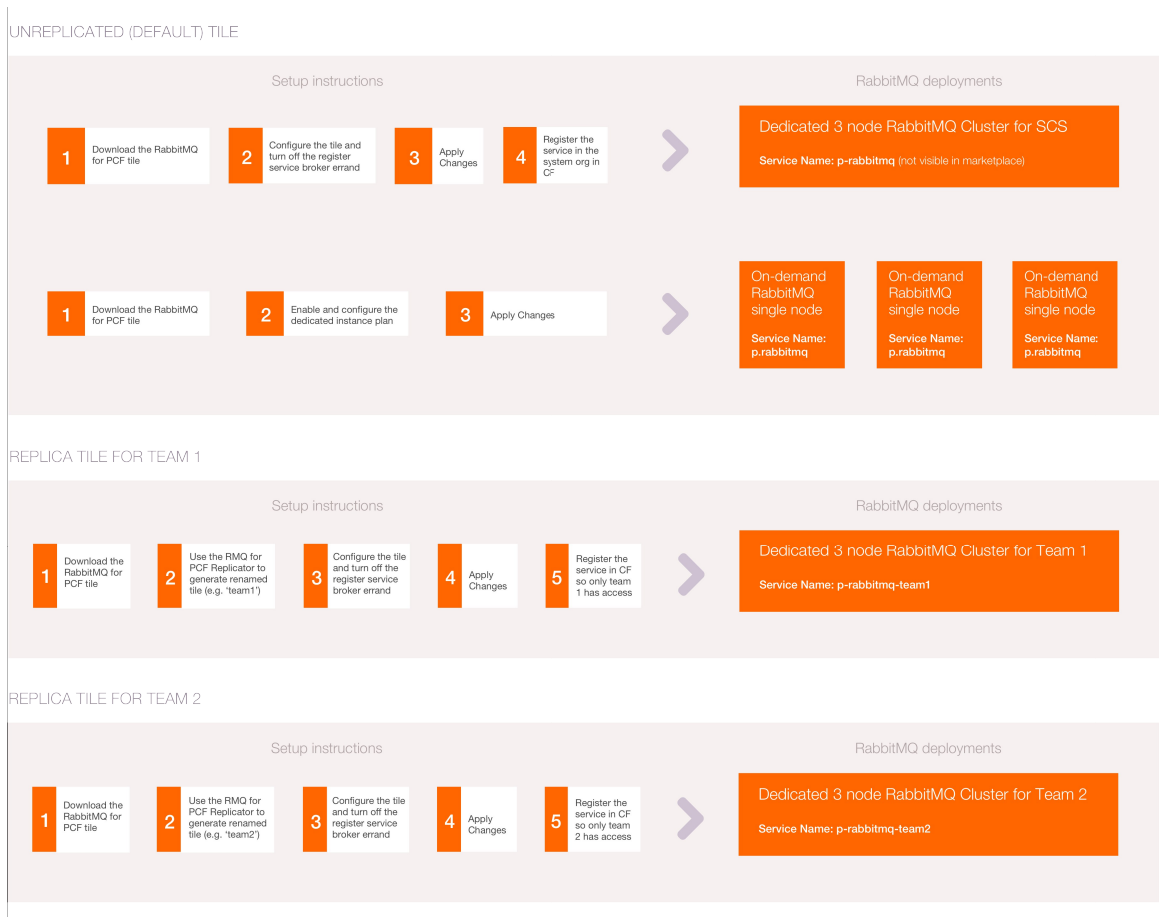
RabbitMQ for PCF Replicator is a tool that allows you to install multiple RabbitMQ for Pivotal Cloud Foundry (PCF) tiles in a single Ops Manager environment. This lets you run multiple pre-provisioned RabbitMQ clusters that are isolated from each other.

For example, you may want to isolate the cluster serving Spring Cloud Services (SCS) from the cluster serving apps in the Marketplace. Or you may want to give a certain team their own dedicated, pre-provisioned cluster that you manage for them. 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 for PCF from clustered and single node service instances dedicated to different teams. In this use case:

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



[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 for PCF tile for SCS, as shown in the diagram below. You can then add replica RabbitMQ clusters for use by apps in the Marketplace.

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

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

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

Providing a Pre-Provisioned Dedicated Cluster

To reserve a RabbitMQ cluster for use by a specific team, 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 for PCF tile as shown in the diagram above.

Blue-Green Upgrades (Advanced)

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

Generating Replica Tiles

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

Prerequisites

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

Download the Replicator

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

Generate Replica Tiles

The following is the syntax for generating a replica tile:

```
./rabbitmq-replicator-darwin\
--name YOUR-DESIRED-TILE-NAME\
--path PATH-TO-TILE\
--output DESIRED-FILE-NAME.pivotal
```

The following are the parameters expected in the above syntax:

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

Naming Conventions in Original and Replica Tiles

The table below shows the naming conventions for various components related to the original RabbitMQ for PCF 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 for PCF 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 for PCF, 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

Broker Registrar

Default (On) 

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 for PCF tile, using the replicator, and give the new replica the same `name` as the existing replica. This is shown in the example workflow below.

Example of an In-Place Upgrade of a Replica

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

```
./rabbitmq-replicator-darwin\  
--name trading-team\  

```

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

To upgrade to v2, follow these steps:

1. Download and unzip the new RabbitMQ for PCF Tile Replicator from [Pivotal Network](#). You must download the version of Tile Replicator that corresponds with the version of the RabbitMQ for PCF 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 for PCF 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 PCF v1.14 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 Cloud Foundry (PCF) 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 for PCF 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 Ops 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.

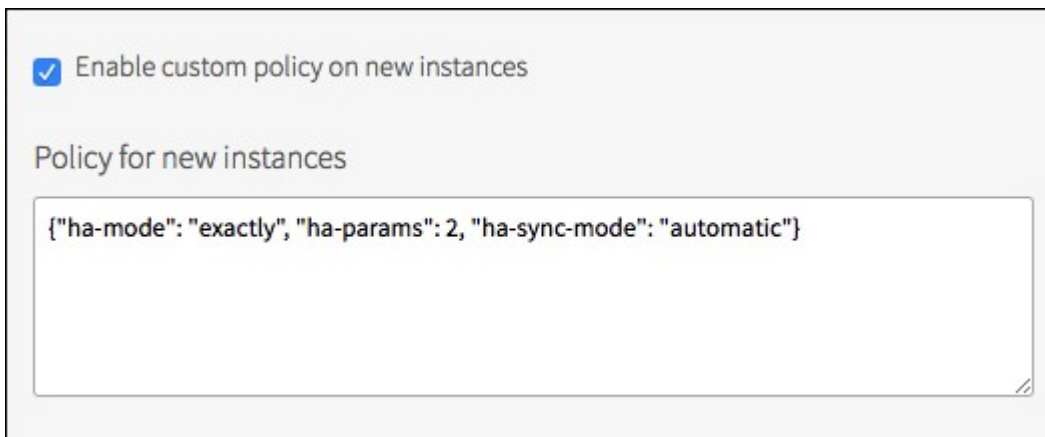
For example:

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 for PCF tile and then click the **Pre-Provisioned RabbitMQ** tab.



The screenshot shows a configuration window for RabbitMQ. At the top, there is a checkbox labeled "Enable custom policy on new instances" which is checked. Below this, there is a section titled "Policy for new instances" containing a text area with the following JSON policy: {"ha-mode": "exactly", "ha-params": 2, "ha-sync-mode": "automatic"}. A small edit icon is visible in the bottom right corner of the text area.

2. Select **Enable custom policy on new instances**.
3. 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 for PCF 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
operator_set_policy	.*	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	
test	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 PCF



Warning: RabbitMQ for PCF v1.14 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 Cloud Foundry (PCF) 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 in PCF?

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 [Pivotal Knowledge Base](#).

File a Support Ticket

You can file a ticket with [Pivotal 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 Dashboard



Warning: RabbitMQ for PCF v1.14 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 d	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 (PAS) 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 Ops 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**.

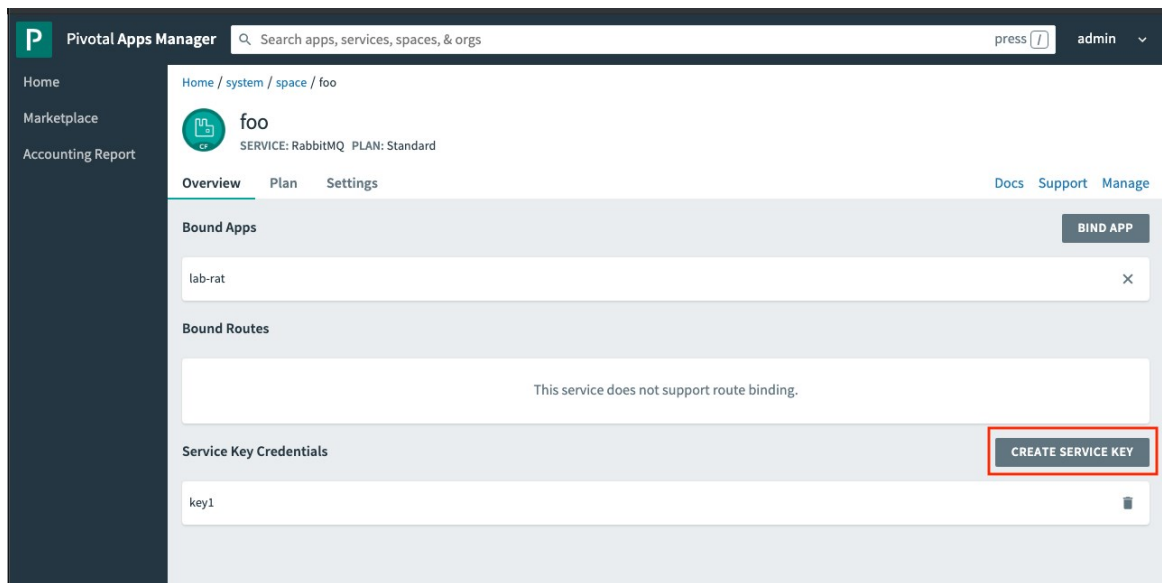
To access the RabbitMQ Management UI, do one of the following:

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 PCF v1.14 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 PCF

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

What is Replicated between nodes in a RabbitMQ cluster?

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

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

Automatic Network Partition Behaviors in RabbitMQ Clusters

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

You can also choose the `autoheal` option in the **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 ERL_DIR=$PWD/erlang/bin/`
4. `$ cd rabbitmq-server/bin/`
5. `$./rabbitmqctl cluster_status`

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

Recovering

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

Manually Synchronizing after a Partition

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

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

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

Upgrading RabbitMQ for PCF



Warning: RabbitMQ for PCF v1.14 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 PCF 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.

The upgrade paths for each version are detailed in the [Product Compatibility Matrix](#).

This topic applies to both the on-demand and pre-provisioned services.

Update Add-Ons to Run with Xenial Stemcell

RabbitMQ for PCF v1.14 and later requires a Xenial stemcell. If you are using any of the following BOSH add-ons with your PCF deployment, you must update the add-on definition to include the Xenial stemcell before you deploy RabbitMQ for PCF v1.14:

- File Integrity Monitoring for PCF Add-on. For update instructions, see [Updating FIM Add-on for PCF to Run with Xenial Stemcells](#).
- ClamAV for PCF Add-on. For update instructions, see [Updating ClamAV Add-on for PCF to Run with Xenial Stemcells](#).
- IPsec for PCF Add-on. For update instructions, see [Updating IPsec Add-on for PCF to Run with Xenial Stemcells](#).

About the Upgrade

This section provides information about the upgrade process, release cycle, and downtime during upgrades.

General Notes About the Upgrade Process

The following notes about the upgrade process apply to upgrading to any version of RabbitMQ for PCF.

- 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](#).
- Ops 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 for PCF is released soon after.

For more information about the Pivotal Cloud Foundry release policy, see [Release Policy](#).

Starting with RabbitMQ for PCF v1.14.0, the RabbitMQ tile uses [floating stemcells](#). 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 for PCF patch release from Pivnet.

Downtime When Upgrading

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 existing VM can just have the RabbitMQ software updated. 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 / Update
Major Tile Version	The RabbitMQ cluster is taken offline for the duration of the upgrade.
Minor Tile Version	The RabbitMQ cluster is taken offline for the duration of the upgrade.
Patch Tile Version	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 are specific migration paths that require downtime, which are identified in the release notes for that version.

Stemcell- Only Patch Tile Version	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.
--	---

Upgrade RabbitMQ for PCF

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 **Apply changes** in the Ops Manager Installation Dashboard. The rest of the process is automated.

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

Developer Guide

Using On-Demand RabbitMQ for PCF



Warning: RabbitMQ for PCF v1.14 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 service for their Pivotal Cloud Foundry (PCF) apps. RabbitMQ enables messaging between cloud-based servers, apps and devices.

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

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

Prerequisites

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

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

Developer Guide

Entries in the VCAP_SERVICES Environment Variable

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

```
{
  "p-rabbitmq": [{
```

```

    "credentials": {
      "dashboard_url": "http://pivotal-rabbitmq.your.pcf.example.com/#/login/b5d0ad14-4352-48e8-8982-d5b1d257029f/tavk86pnnns1ddiypsdtbchurn",
      "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
      "password": "tavk86pnnns1ddiypsdtbchurn",
      "protocols": {
        "amqp": {
          "password": "tavk86pnnns1ddiypsdtbchurn",
          "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
          "uris": [
            "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiypsdtbchurn@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
            "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiypsdtbchurn@10.0.0.51:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7"
          ]
        }
      }
    }
  }
}

```

For more information about the environment variable `VCAP_SERVICES`, see [RabbitMQ Environment Variables](#).

The Create-Bind Process

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

To use RabbitMQ in a PCF app:

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

Confirm Service Availability

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

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

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

```

$ cf marketplace
Getting services from marketplace in org my-org / space my-space as user@exampl

```



```
e.com...
OK
service           plans      description
[...]
ondemand-rabbitmq Solo       RabbitMQ Service
[...]
```

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

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

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

Create a Service Instance

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

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

1. Enter `cf create-service ondemand-rabbitmq Solo SERVICE-INSTANCE`

Where `SERVICE-INSTANCE` is a name you choose to identify the service instance. This name will appear under `service` [sic] in output from `cf services`.

2. Enter `watch cf services` and wait for the `last operation` for your instance to show as `create succeeded`.

```
$ cf create-service ondemand-rabbitmq Solo my-instance

Creating service my-instance in org my-org / space my-space as user@example.com
...
OK

$ watch cf services

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

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

Using Transport Layer Security (TLS)

To use TLS to secure communication between your apps and RabbitMQ service instances, you must complete these procedures:

1. [Configure TLS for Your Service Instance](#)
2. Update your apps to use encrypted communication using one of these procedures:
 - ✦ For Java or Spring apps: [Activate TLS for Java and Spring Apps](#)
 - ✦ For other apps: [Modify Apps for TLS](#)

Configure TLS for Your Service Instance

To secure the communication between your app and a RabbitMQ service instance, you must enable TLS on the service instance. You cannot disable TLS after it is enabled.

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](#).



Note: If you enable TLS for a service instance, you can no longer connect to it without using TLS.

To use TLS for a service instance, follow the steps below.

1. Follow these steps to find the IP addresses of the RabbitMQ nodes:

1. Create a service key by running the following command:

```
cf create-service-key MY-INSTANCE MY-KEY
```

Where:

- **MY-INSTANCE** is the name you chose when you created your service instance.
- **MY-KEY** is a name you choose for the service key.

2. Display the service key by running:

```
cf service-key MY-INSTANCE MY-KEY
```

3. In the output of the previous command, find the IP addresses in the **hostnames** field.

4. Delete the service key because it is no longer needed:

```
cf delete-service-key MY-INSTANCE MY-KEY
```

2. Update your service instance with a valid TLS certificate by running:

```
cf update-service MY-INSTANCE -c '{"tls": [HOSTNAMES]}'
```

Where:

- ✦ **HOSTNAMES** is a comma-separated list of the IP addresses you found above.

For example:

```
cf update-service MY-INSTANCE -c '{"tls": ["10.10.10.10", "10.10.10.11"]}'
```

```
$ cf update-service my-instance -c '{"tls": ["10.1.1.5", "10.1.1.6", "10.1.1.7"]}'
```



Note: After the service instance has been updated to enable TLS, you cannot disable TLS or change the listed IP addresses for that service.

3. Check that the process completed by running:

```
cf service MY-INSTANCE
```

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, repush the apps with `cf push`.
- For apps bound to an existing service instance, follow these steps:

1. Stop the app. For example:

```
$ cf stop my-app
```

2. Unbind the app from the service instance. For example:

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

3. Re-bind the app to the service instance. For example:

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

4. Restart the app. For example:

```
$ cf restart my-app
```



Note: If the operator used a self-signed certificate, configure your app to use the same CA certificate, and valid certificate and key.

Bind a Service Instance to Your App

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

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

1. Enter `cf bind-service APP SERVICE-INSTANCE`

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

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

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

Use the RabbitMQ Service in Your App

To access the RabbitMQ service from your app:

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

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

Updating 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.

1. Enter `cf restart-app APP`

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

```
$ cf restart my-app
```

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

Unbind a Service Instance to Your App

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

1. Enter `cf unbind-service APP SERVICE-INSTANCE`

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

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

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

OK

Delete a Service Instance

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

1. Enter `cf delete-service SERVICE-INSTANCE`

Where `SERVICE-INSTANCE` is the name of the service to delete.

```
$ cf delete-service my-instance

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

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

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

Create an Admin User for a Service Instance

If you want to get admin privileges to 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](#).

Sharing Service Instances

In order to [share service instances](#) the feature-flag `service_instance_sharing` must be enabled by your Operator. You can then follow the [documentation](#) to share your service instances across Cloud Foundry Organizations and Spaces.

Federate Exchanges and Queues

You can federate exchanges and queues in RabbitMQ for PCF, as you would in any RabbitMQ deployment.

To federate exchanges and queues, do the following:

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 will have this pattern:

```
{
  ...
  "uri": "amqp://USERNAME:PASSWORD@IP_ADDRESS/VHOST",
```

```
"uris": [
  "amqp://USERNAME:PASSWORD@IP_ADDRESS/VHOST"
],
...
}
```

For example:

```
{
  ...
  "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
  "uris": [
    "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
    "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.51:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7"
  ],
  ...
}
```

3. Set up federation as you normally would, 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 for PCF, as you would in any RabbitMQ deployment.

To shovel exchanges and queues, do the following:

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 will have this pattern:

```
{
  ...
  "uri": "amqp://USERNAME:PASSWORD@IP_ADDRESS/VHOST",
  "uris": [
    "amqp://USERNAME:PASSWORD@IP_ADDRESS/VHOST"
  ],
  ...
}
```

For example:

```
{
```

```
...
"uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
"uris": [
  "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
  "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.51:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7"]},
...
}
```

3. Set up shovel as you normally would, 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](#). At the moment, RabbitMQ for PCF 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 PCF v1.14 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 provides instructions to developers to modify apps that are not written in Java or Spring to use TLS to secure their connection with RabbitMQ on-demand service instances.

Prerequisites

The following are prerequisites to procedures in this topic:

- 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 [Configure 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.

To modify your app, do the following:

1. Use one of the code snippets below.

In these examples, `VCAP_SERVICES` is an environment variable available from the app.

- ❖ **Option 1:** 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']['uris'].sample
conn = Bunny.new(uri)
conn.start
```

- ❖ **Option 2:** If the operator used a self-signed certificate, configure the RabbitMQ client to use the same CA certificate, and valid certificate and key. Use the example code below, replacing the variables for `PATH_TO_CERTIFICATE`, `PATH_TO_KEY`, and `PATH_TO_CA_CERTIFICATE`.

```
require 'json'
require 'bunny'

vcap_services = JSON.parse(ENV['VCAP_SERVICES'])
uri = vcap_services['p.rabbitmq'][0]['credentials']['protocols']['amqp+ssl']['uris'].sample
conn = Bunny.new(uri, tls_cert: PATH_TO_CERTIFICATE, tls_key: PATH_TO_KEY,
, tls_ca_certificates: [PATH_TO_CA_CERTIFICATE])
conn.start
```

Repush or Rebind Your App

After modifying your app, repush it with `cf push`.



WARNING: Any apps using an existing service instance must be rebound after enabling TLS for the instance.

Follow these steps to rebound an app using an existing service instance:

1. Stop the app. For example:

```
$ cf stop my-app
```

2. Unbind the app from the service instance. For example:

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

3. Re-bind the app to the service instance. For example:

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

4. Restage the app. For example:


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

Your app should now communicate 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 Service Instance to Another Service Instance



Warning: RabbitMQ for PCF v1.14 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](#)

RabbitMQ Environment Variables



Warning: RabbitMQ for PCF v1.14 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 Cloud Foundry (PCF) stores for RabbitMQ for PCF service instances. These variables include the credentials that apps use to access the service instances.

VCAP_SERVICES

Apps running in PCF 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": "standard",
    "tags": ["rabbitmq", "messaging", "message-queue", "amqp", "pivotal"],
    "credentials": {
      "dashboard_url": "http://pivotal-rabbitmq.your.pcf.example.com/#/login/b5d0ad14-4352-48e8-8982-d5b1d257029f/tavk86pnns1ddiqpsdtbchurn",
      "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
```

```

    "vhost": "62e5ab21-7b38-44ac-b139-6aa97af01cd7",
    "password": "#passwordexample123456789",
    "ssl": false,
    "hostname": "10.0.0.41",
    "hostnames": [
        "10.0.0.41",
        "10.0.0.51"],
    "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@10.0.0.41/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
    "uris": [
        "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@10.0.0.41/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
        "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@10.0.0.51/62e5ab21-7b38-44ac-b139-6aa97af01cd7"],
    "http_api_uri": "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@10.0.0.41:15672/api",
    "http_api_uris": [
        "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@10.0.0.41:15672/api",
        "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:tavk86pnnns1ddiqpsdtbchurn@10.0.0.51:15672/api"],
    "protocols": {
        "amqp": {
            "password": "passwordexample123456789",
            "port": 5672,
            "ssl": false,
            "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
            "vhost": "62e5ab21-7b38-44ac-b139-6aa97af01cd7",
            "host": "10.0.0.41",
            "hosts": [
                "10.0.0.41",
                "10.0.0.51"],
            "uri": "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
            "uris": [
                "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.41:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7",
                "amqp://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.51:5672/62e5ab21-7b38-44ac-b139-6aa97af01cd7"]},
        "management": {
            "username": "b5d0ad14-4352-48e8-8982-d5b1d257029f",
            "password": "passwordexample123456789",
            "path": "/api",
            "port": 15672,
            "ssl": false,
            "host": "10.0.0.41",
            "hosts": [
                "10.0.0.41",
                "10.0.0.51"],
            "uri": "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.41:15672/api",
            "uris": [
                "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.41:15672/api",
                "http://b5d0ad14-4352-48e8-8982-d5b1d257029f:passwordexample123456789@10.0.0.51:15672/api"]}]]]]]]

```

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/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 PCF, 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 PCF v1.14 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 PCF instances.

Troubleshoot Errors

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

Troubleshoot General Errors

No Metrics from Log Cache

Symptom	You receive no metrics when running the <code>cf tail</code> command.
Cause	This might be because the Firehose is disabled in the PAS tile.
Solution	Ask your operator to ensure that the V2 Firehose checkbox is enabled, and the Enable Log Cache syslog ingestion checkbox is disabled in the PAS tile. For more information about configuring these checkboxes, see Enable Syslog Forwarding .

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 for PCF 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 Pivotal 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    ondemand-rabbitmq Solo      create succeeded
```

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

Retrieve RabbitMQ Instance Credentials

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

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

For example:

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

Error: Failed to Set Credentials in Credential Store

If you encounter:

```
error: failed to set credentials in credential store:
The request includes an unrecognized parameter 'mode'.
Please update or remove this parameter and retry your request.
```


after upgrading from PCF 2.3 to 2.4, it is likely because one or more installed on-demand brokers were not restarted during the upgrade. To clear the cached server version and enable your on-demand brokers to communicate with CredHub v2, do the procedure in [Run BOSH Restart on Your On-Demand Service Brokers](#).

Knowledge Base (Community)

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

File a Support Ticket

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

To 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.

Delete RabbitMQ Instances

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

1. Run the following command to delete all instances of the On-Demand Broker:



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

```
bosh run-errand delete-sub-deployments
```

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