Pivotal

PRODUCT DOCUMENTATION

Redis for PCF®

Version 1.6

User's Guide

Table of Contents

Redis for PCF

Note: Redis for PCF v1.6 is no longer supported. The support period for v1.6 has expired. To stay up-to-date with the latest software and security updates, upgrade to Redis for PCF v1.10 or later.

This is documentation for the Redis for PCF service tile. This tile can be downloaded from Pivotal Network.

This documentation:

- Describes the features and architecture of Redis for PCF
- Instructs the PCF operator on how to install, configure, maintain and backup Redis for PCF
- Instructs the App developer on how to choose a service plan, create and delete Redis service instances, and bind an app

About Redis

Redis is an easy to use, high speed key-value store that can be used as a database, cache, and message broker. It supports a range of data structures including strings, lists, hashes, sets, bitmaps, hyperloglogs, and geospatial indexes. It is easy to install and configure and is popular with engineers as a straightforward NoSQL data store. It is used for everything from a quick way to store data for development and testing through to enterprise-scale apps like Twitter.

About Redis for PCF

Redis for PCF packages Redis for easy deployment and operability on Pivotal Cloud Foundry (PCF). Redis for PCF can be used as a datastore or cache. Metrics and logging enable operators to monitor Redis health with backups every 24 hours. New features are regularly introduced. Upgrading Redis for PCF is straightforward and preserves configuration and data.

Product Snapshot

Current Redis for PCF details

Version: v1.6.9
Release date: November 15, 2017
Software component version: Redis OSS v3.2.11
Compatible Ops Manager version: v1.8.0
Compatible Elastic Runtime version: v1.8.0
GCP support? Yes
vSphere support? Yes
OpenStack support? Yes
AWS support? Yes
Azure support? Yes
IPSec support? Yes

About Upgrading to the Latest Version

Consider the following compatibility information before upgrading Redis for PCF.

For more information, see the Product Compatibility Matrix.

Ops Manager Version	Supported Upgrades from Imported Redis Installation	
	From	To
v1.5.x, v1.6.x	v1.40 – v1.4.3	v1.4.4 – latest v1.4.x
		v1.5.0 – v1.5.7
	v1.4.4 – latest v1.4.x	Next v1.4.x – latest v1.4.x
		v1.5.0 – latest v1.5.x

	v1.5.0 – latest v1.5.x	Next v1.5.x – latest v1.5.x
v1.7.x	v1.5.0 – latest version	v1.5.1 – latest version
v1.8.0 – latest version	v1.5.17 – latest version	v1.6.0 – latest version
v1.9.0 – latest version	v1.5.17 – latest version	v1.6.0 – latest version

More Information

The following table lists where you can find topics related to the information on this page:

For more information about…	See…
product compatibility	Product Version Matrix
a particular version of Redis for PCF	Release Notes
how to upgrade Redis for PCF	Upgrading Redis for PCF
how to use Redis	Redis Documentation

Feedback

Please provide any bugs, feature requests, or questions to the Pivotal Cloud Foundry Feedback list.

Overview of Redis for PCF v1.6

This topic describes the significant new features in Redis for PCF v1.6. This topic also presents a checklist that you can use to decide if Redis for PCF is ready to meet your business requirements.

Introduction

Redis for PCF v1.6 focuses on increased operability and compatibility with Ops Manager v1.8.
As of v1.6, metrics are emitted via the Loggregator Firehose. Metrics help operators monitor service health and usage. New, enhanced logging provides a fine-grained history of service usage.

New in This Release

The following features are new in Redis for PCF v1.6

- **Metrics** — Redis emits a number of metrics you can use to monitor the health and performance of your Redis deployment. Redis for PCF sends these metrics to the Firehose. For more information, see Monitoring Redis for PCF.
- **Syslog forwarding protocols** — Redis for PCF v1.6 supports new protocols for syslog forwarding: TCP and RELP. These new transport protocols widen the range of log management tools that can be used. For more information, see Configuring Syslog Output.
- **Extended logging** — New logging includes service instance creation and deletion. Logs are emitted when a service instance is created, via `cf create-service p-redis <plan-name> <service-instance-name>` and also when a service instance is deleted, via `cf delete-service <service-instance-name>`.
- **Redis 3.2.1** — Redis for PCF v1.6 uses Redis 3.2.1. App developers can now use the new GEO API and BITFIELD commands. For more information, see the Redis Commands documentation.
- **Enhanced smoke tests** — In Redis for PCF v1.6 smoke tests run in a more restrictive application security group (ASG) in the `system` org and `redis-smoke-test-space`. Smoke tests run as an application instance; the restrictive ASG reduces security vulnerability. Running the tests in `system` ensures the smoke tests don't count as a billable application instance. For more information, see Redis for PCF Smoke Tests.

Enterprise-Ready Checklist

Review the following table to determine if Redis for PCF v1.6 has the features needed to support your enterprise.

Plans and Instances		**More Information**
Dedicated and shared plans	Redis for PCF v1.6 provides both dedicated VM and shared VM plans.	Plans
Customizable plans	For the dedicated VM plan, the operator can customize the VM and disk size.	Configuring
Installation and Upgrades		**More Information**
Product upgrades	Redis for PCF v1.6 can be upgraded from v1.5 tiles	Upgrading Redis for PCF
Deployment Smoke Tests	Redis for PCF v1.6 installation and upgrade runs a post deployment BOSH errand that validates basic Redis operations	Smoke Tests
Maintenance and Backups		**More Information**
Operational Monitoring and Logging	Redis for PCF v1.6 provides metrics for health monitoring and syslog redirection to external log ingestors.	Monitoring Redis for PCF
Backup and Restore	Redis for PCF v1.6 provides automatic backups every 24 hours and manual restore	Manual Backup and Restore of Redis for PCF
Scale and Availability		**More Information**
Scale	Redis for PCF has been tested with 60GB	
Ability to Scale Up / Down	Operators can scale VMs up, but not down	Configuring
Rolling Deployments	Redis for PCF does not support rolling deployments because it is single node; the service is unavailable during upgrades.	Upgrades
AZ Support	Assigning multiple AZs to Redis jobs does not guarantee high availability.	About Multiple AZs in Redis for PCF v1.6
Encryption		**More Information**
Encrypted		

Encrypted Communication in Transit	Redis for PCF v1.6 has been tested successfully with the BOSH IPsec Add-on	Securing Data in Transit with the IPsec Add-on

About Multiple AZs in Redis for PCF v1.6

Redis for PCF v1.6 supports configuring multiple AZs. However, assigning multiple AZs to Redis jobs does not guarantee high availability.

- Shared-VM instances run on a single node in just one of the configured availability zones and are therefore not highly available.
- Dedicated-VM instances can be assigned to any of the configured availability zones. However each instance still operates as a single node with no clustering. This separation over availability zones provides no high availability.

More Information

The following table lists where you can find topics related to the information on this page:

For more information about…	See…
the v1.6 releases	Release Notes

Release Notes

Redis for PCF 1.6.9

November 15, 2017

Compatibility

Stemcell line is 3445.

Redis for PCF 1.6.8

October 27, 2017

Compatibility

Stemcell line is now 3468
Uses Redis v3.2.11

Features

No change

Bug Fixes

No change

Known Issues

Stemcell line 3468 is not meant for adoption. Users should not install this version of Redis for PCF.

Redis for PCF 1.6.7

May 25, 2017

Compatibility

Stemcell line is now 3363

Uses Redis v3.2.8

Features

No change

Bug Fixes

No change

Resolved Security Issues

No change

Redis for PCF 1.6.6

April 4, 2017

Compatibility

No change

Features

No change

Bug Fixes

Fix bug: Tile now accepts floating stemcells

Resolved Security Issues

No change

Redis for PCF 1.6.5

January 19, 2017

Compatibility

No change

Features

No change

Bug Fixes

No change

Resolved Security Issues

Update service metrics and cf-routing-release to use golang to 1.7.4

Redis for PCF 1.6.4

December 22, 2016

Compatibility

No change

Features

No change

Bug Fixes

Fix bug: pivotal-cf/cf-redis-release #39: New Tile/BOSH Deployment installs fail

Resolved Security Issues

Update golang to 1.7.4 Update stemcell to 3263.13

Redis for PCF 1.6.3

December 7, 2016

Compatibility

No change

Features

No change

Bug Fixes

Fix a bug where stopping all the VMs via the IaaS and then starting them up again fails to bring everything back to a running state.

Resolved Security Issues

Update to Golang 1.7.3.

Known Issues

None

Redis for PCF v1.6.2

October 11, 2016

Compatibility

No change

Features

No change

Bug Fixes

No change

Resolved Security Issues

Updates loggregator release to eliminate spawning subshell for timestamps.

Known Issues

None

Redis for PCF v1.6.1

September 23, 2016

Compatibility

No change

Features

No change

Bug Fixes

No change

Resolved Security Issues

Updates for stemcell 3263.2

Known Issues

None

Redis for PCF v1.6.0

September 14, 2016

Compatibility

See About Upgrading to the Latest Version.

Features

See the Overview of Redis for PCF v1.6 topic.

Bug Fixes

Fixes an issue with long server names that caused Nginx to error. For more information, see related Tracker story.

Resolved Security Issues

Uses Golang v1.6.3 because it fixes an HTTPoxy security vulnerability; that particular vulnerability was not present in the product.

Known Issues

None

Redis for PCF 1.6 Architecture and Lifecycle

How Redis for PCF Configures Redis

You should be aware that Redis for PCF configures Redis in the following ways. These configurations cannot be changed.

- Redis is configured with a maxmemory-policy of no-eviction. This policy means that when the memory is full, the service does not evict any keys or perform any write operations until memory becomes available.
- Persistence is configured for both `RDB` and `AOF`. The default maximum number of connections, maxclients, is set at 10000 but this number is adjusted by Redis according to the number of file handles available.
- Replication and event notification are not configured.

A sample `redis.conf` from a Dedicated-VM plan instance can be viewed on Sample Redis Configuration

Service Plans

Redis for PCF offers Dedicated-VM and Shared-VM plans.

Shared-VM Plan

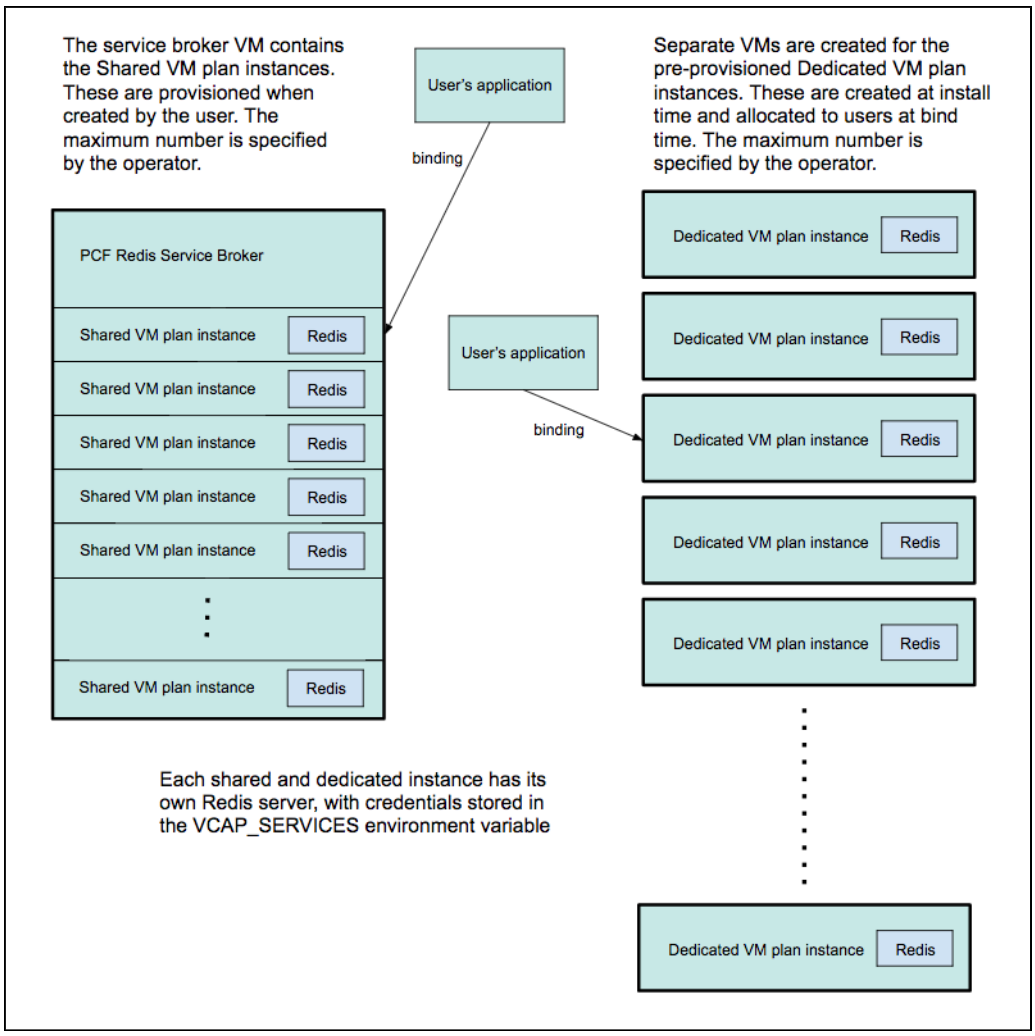
- This plan deploys a Redis instance inside the service broker VM.
- This plan can be disabled by setting the `Max instances limit` on the `Shared-VM plan` tab in OpsManager to be `0`.
- The maximum number of instances can be increased from the default 5 to a value of your choosing. If you increase the number of instances that can be run on this single VM, you should consider increasing the resources allocated to the VM. In particular RAM and CPU. You can overcommit to a certain extent, but may start to see performance degradations.
- You can also increase the maximum amount of RAM allocated to each Redis process (service instance) that is running on this VM
- If you decrease the service instance limit, any instances that are running where the count is now greater than the limit are not terminated. They are left to be removed naturally, until the total count drops below the new limit you cannot create any new instances. For example if you had a limit of 10 and all were used and reduced this to 8, the two instances will be left running until you terminate them yourself.

Dedicated-VM Plan

- This plan deploys the operator-configured number of dedicated Redis VMs alongside the service broker VM.
- These instances are pre-provisioned during the deployment of the tile from OpsManager into a **pool**. The VMs are provisioned and configured with a Redis process ready to be used when an instance of the Dedicated-VM plan is requested.
- A default deployment will provision `5 instances` of the Dedicated-VM plan into the **pool**. This number can be increased on the `Resource Config` tab in Ops Manager, either in the initial deployment, or subsequently thereafter. The number of VMs **cannot** be decreased once deployed.
- When a user provisions an instance, it is marked as in use and taken out of the **pool**.
- When a user deprovisions an instance, the instance is cleansed of any data and configuration to restore it to a fresh state and placed back into the pool, ready to be used again.
- This plan can be disabled by setting the number of instances of the `Dedicated node` job in Ops Manager to `0`.

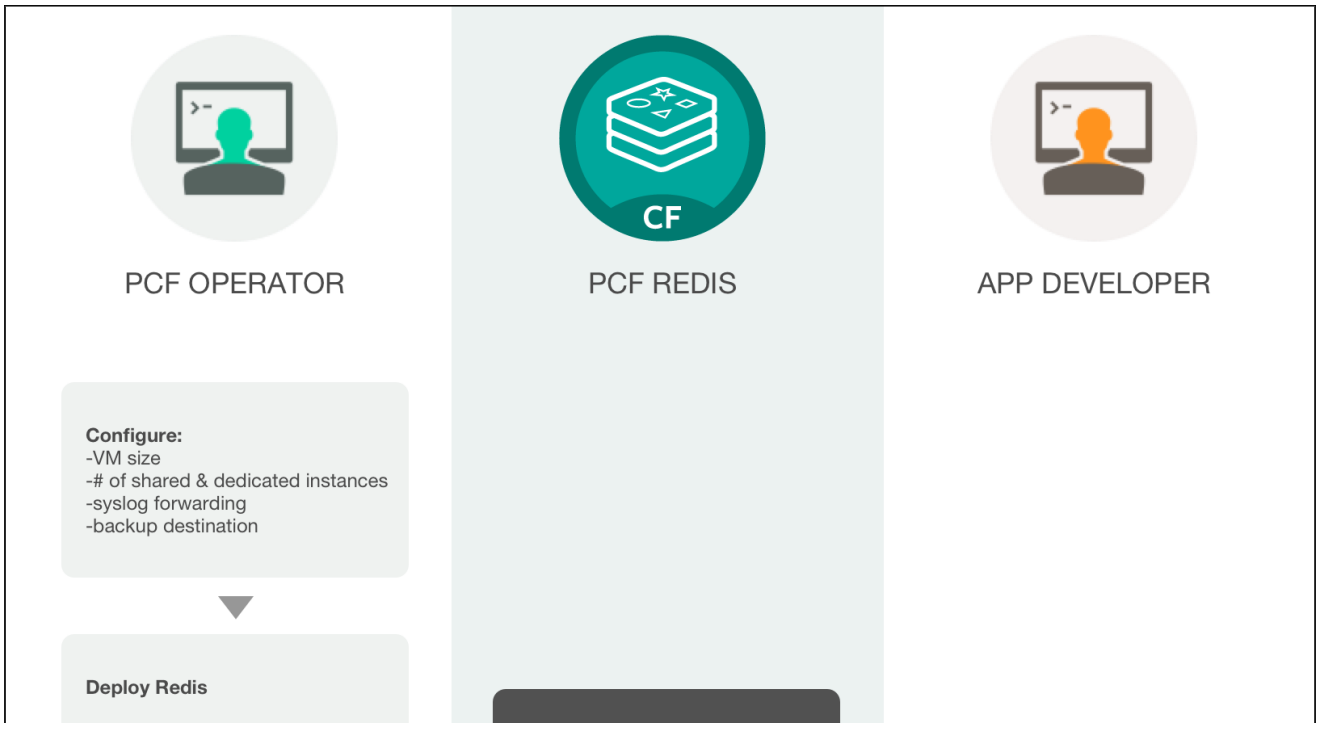
Redis for PCF Architecture

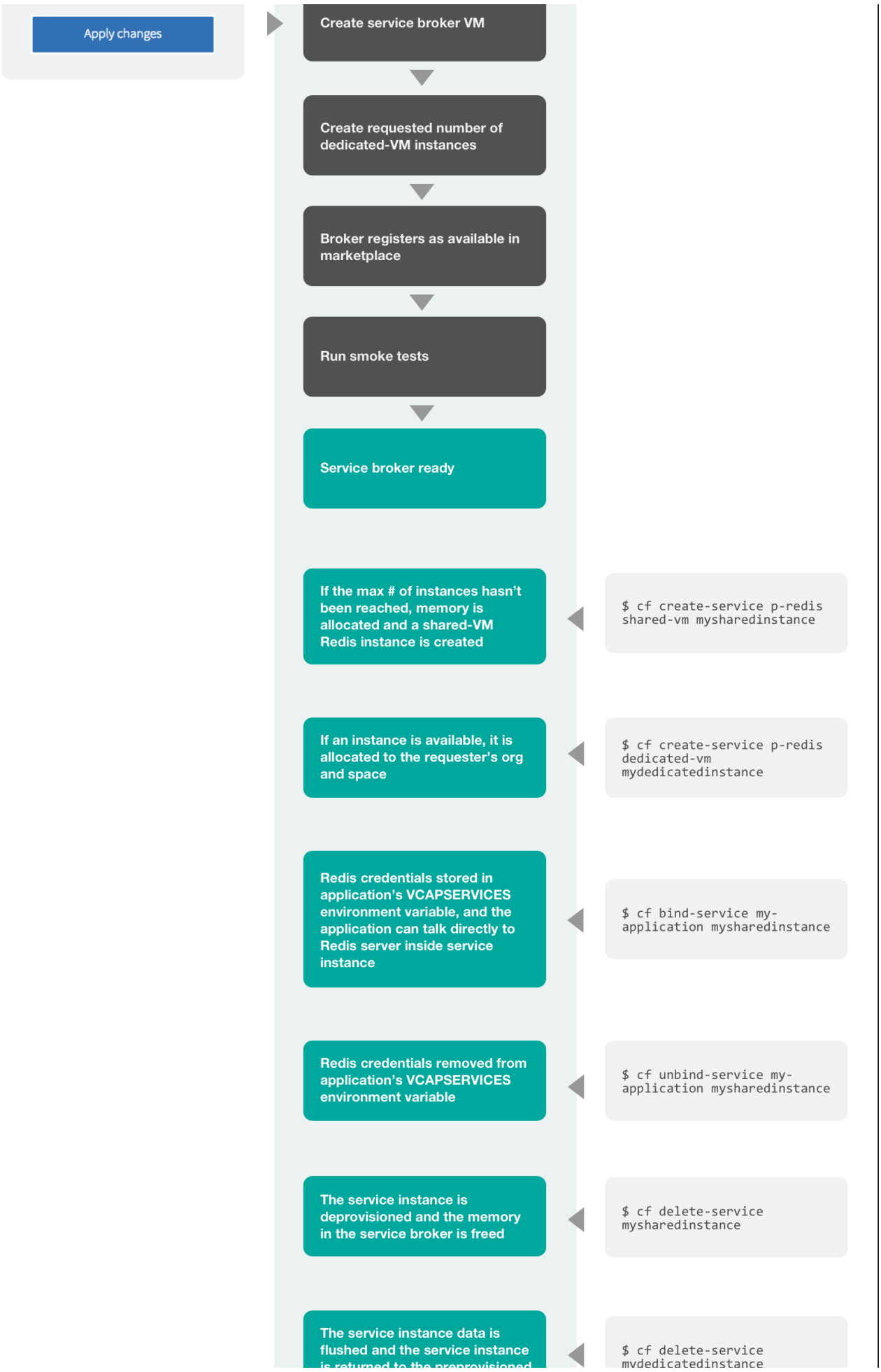
This diagram shows how the architecture of the service broker and Shared-VM and Dedicated-VM plans and how the user's app binds to a Redis instance.

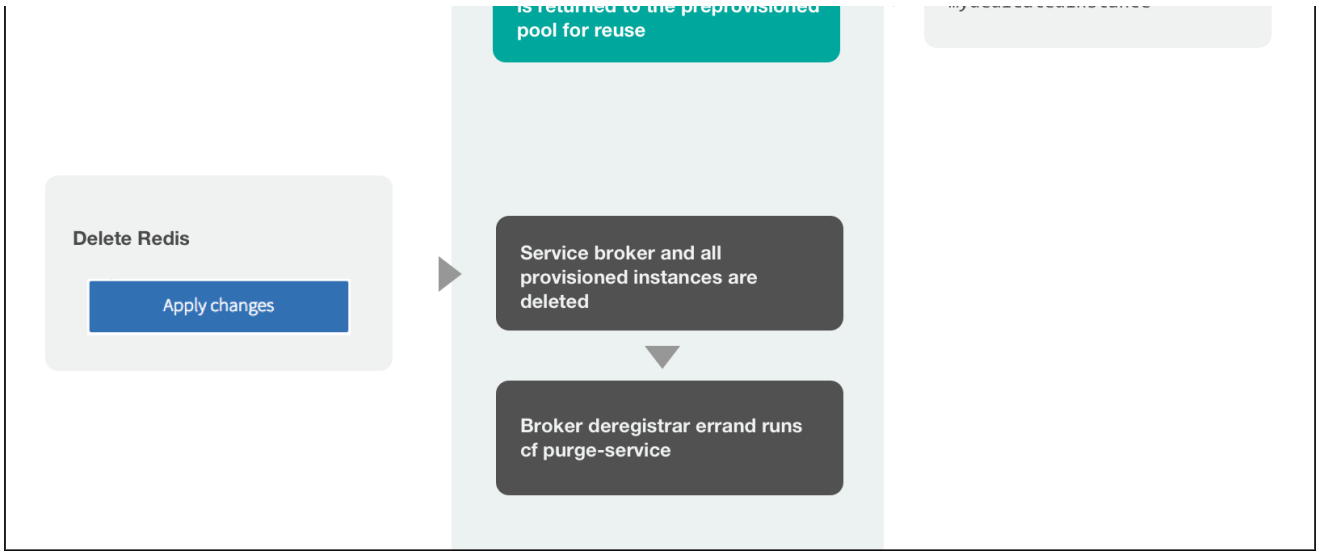


Redis for PCF Lifecycle

Here is the lifecycle of Redis for PCF, from an operator installing the tile through an app developer using the service then an operator deleting the tile.







Redis for PCF Recommended Usage and Limitations

Recommended Use Cases

Redis for PCF can be used as a datastore or cache.
It is configured with `maxmemory-policy = noeviction` and RDB and AOF persistence.

Redis can be used in many different ways, including: * Key/value store for strings and more complex data structures including Hashes, Lists, Sets, Sorted Sets * Session cache - persistence enables preservation of state * Full page cache - persistence enables preservation of state * Database cache - cache queries * Data ingestion - because Redis is in memory it can ingest data very quickly * Message Queues - list and set operations. `PUSH` , `POP` , and blocking queue commands. * Leaderboards/Counting - increments and decrements of sets and sorted sets using `ZRANGE` , `ZADD` , `ZREVRANGE` , `ZRANK` , `INCRBY` , `GETSET` * Pub/Sub - built in publish and subscribe operations - `PUBLISH` , `SUBSCRIBE` , `UNSUBSCRIBE`

Service Plan Recommended Usage and Limitations

Dedicated-VM Plan

- Each dedicated VM plan instances is deployed to its own VM and is suitable for production workloads.
- The number of Dedicated-VM plan instances available to developers is set by the operator. Configurations of up to 100 Dedicated-VM plan instances have been tested.
- No ability to change the Redis configuration. The `CONFIG` command is disabled.
- Cannot scale down the number of VMs on the plan once deployed.
- The default maximum number of connections, maxclients, is set at 10000 but this number is adjusted by Redis according to the number of file handles available.

Shared-VM Plan

- The Shared-VM plan does not manage ‘noisy neighbor’ problems so it is not recommended for production apps.
- The number of Shared VM instances available to developers is set by the operator. The maximum number of shared VM instances is relative to the memory allocated to each Shared VM instance and the total memory of the Redis service broker. Please see Configuring Service Plans for more detail.
- It cannot be scaled beyond a single virtual machine.
- The following commands are disabled: `CONFIG` , `MONITOR` , `SAVE` , `BGSAVE` , `SHUTDOWN` , `BGREWRITEAOF` , `SLAVEOF` , `DEBUG` , and `SYNC` .
- Constraining CPU and/or disk usage is not supported.
- The default maximum number of connections, maxclients, is set at 10000 but this number is adjusted by Redis according to the number of file handles available.

Availability Using Multiple AZs

Redis for PCF 1.6 supports configuring multiple availability zones but this configuration does not provide high availability.

Downtime During Redeploys

Redeploying PCF Redis for configuration changes or upgrades will result in Redis being inaccessible to apps for a brief period of time.

Redis Key Count and Memory Size

Redis can handle up to 2^{32} keys, and was tested in practice to handle at least 250 million keys per instance. Every hash, list, set, and sorted set, can hold 2^{32} elements. VM memory is more likely to be a limiting factor than number of keys that can be handled.

Redis for PCF Security

Security

Pivotal recommends that Redis for PCF is run in its own network.

Redis for PCF works with the IPsec Add-on for PCF. For information about the IPsec Add-on for PCF, see Securing Data in Transit with the IPsec Add-on.

To allow this service to have network access you must create Application Security Groups. For more information, see Networks, Security, and Assigning AZs.

Best Practices for Operating Redis for PCF

This topic is for PCF operators. It introduces some best practices, but does not provide details about operation.

Best Practices

Pivotal recommends that operators do the following:

- Resource Allocation — Work with app developers to anticipate memory requirements and to configure VM sizes. Redis for PCF is configured by default with small VMs. For information about configuring VM sizes, see Configure Redis Service Plans.
- Logs — Configure a syslog output. Storing logs in an external service helps operators debug issues both current and historical.
- Monitoring — Set up a monitoring dashboard for metrics to track the health of the installation.
- Backing Up Data — When using Redis for persistence, configure automatic backups so that data can be restored in an emergency. Validate the backed-up data with a test restore.

About Creating Backups of Redis Instances

You can back up Redis for PCF instances in two way:

- Configure automatic backups to be run for each instance, across both service plans. For information about setting up automatic backups, see Configure Backups.
- Create manual backups of individual instances. For information about how to make manual backups of instances, see Manual Backup and Restore of Redis for PCF.

About Monitoring Redis for PCF

Redis Metrics

Redis for PCF emits Redis metrics via the firehose. Details here

Logging

Syslog can be forwarded to an external log service.

The following example shows syslog message:

```
Nov 15 17:05:01 10.0.24.10 audispd: [job=dedicated-node index=4]  node=7bfe8b1b-6c
fd-4d33-b704-c9214ce6bb3e type=USER_ACCT msg=audit(1479229501.290:86): pid=6655 ui
d=0 auid=4294967295 ses=4294967295 msg='op=PAM:accounting acct="root" exe="/usr/sbi
n/cron" hostname=? addr=? terminal=cron res=success'
```

For information about how to set up syslog output, see Configure Syslog Output.

Smoke Tests

Redis for PCF has smoke tests that are run as a post-install errand by Ops Manager. Information on what they do is here. They can be run by the operator via `bosh run errand smoke-tests`.

Installing and Upgrading Redis for PCF

Installation Steps

To add Redis for PCF to Ops Manager, follow the procedure for adding Pivotal Ops Manager tiles:

1. Download the product file from Pivotal Network.
2. Upload the product file to your Ops Manager installation.
3. Click **Add** next to the uploaded product description in the Available Products view to add this product to your staging area.
4. (Optional) Click the newly added tile to configure your possible service plans, syslog draining, and backups.
5. Click **Apply Changes** to install the service.

Default Resources

Resource requirements for Redis for PCF

These are the default resource and IP requirements for installing the tile

Product	Resource	Instances	CPU	Ram	Ephemeral	Persistent	Static IP	Dynamic IP
Redis	Redis Broker	1	2	3072	4096	9216	1	0
Redis	Dedicated Node	5	2	1024	4096	4096	1	0
Redis	Broker Registrar	1	1	1024	2048	0	0	1
Redis	Broker De-Registrar	1	1	1024	2048	0	0	1
Redis	Compliation	2	2	1024	4096	0	0	1

Notes:

- The `shared-vm` plan is on the `Redis Broker` resource
- The `dedicated-vm` plan is on the `Dedicated Node` resource

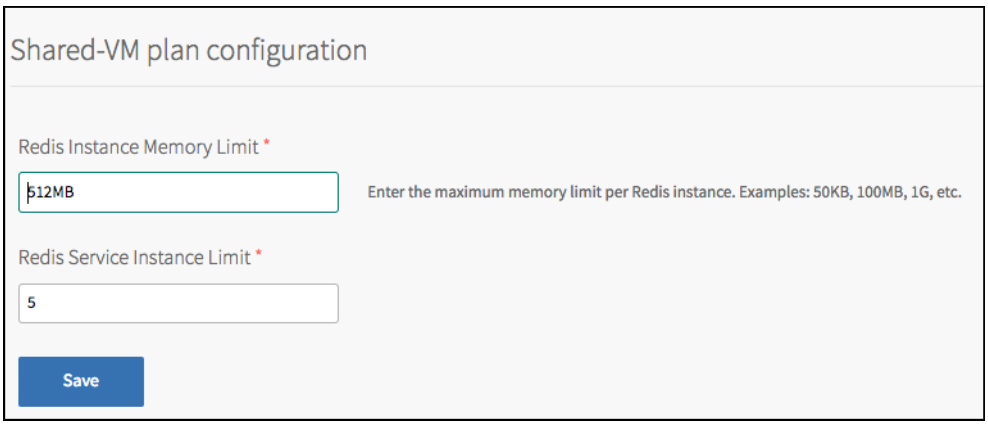
Configuring PCF Redis

Configure Redis Service Plans

Select the **Redis** tile from the Installation Dashboard to display the configuration page and allocate resources to Redis service plans.

Shared-VM Plan

1. Select the **Shared-VM Plan** tab to configure the memory limit for each Redis instance and the maximum number of instances that can be created.



2. Enter the maximum number of instances and the memory limit for each Redis instance.

3. Click the **Save** button.

   Shared-VM instances run on the Redis Broker.

   The memory and instance limits for your Shared-VM Redis instances should depend on the total memory of your Redis broker. When configuring the maximum number of Redis service instances that can be created you need to take into account the maximum memory each redis instance could use in correlation with how much total memory the Redis broker has. We recommend you only allow up to 45% of your Redis broker's total memory to be used by all Redis instances. This is due the amount of memory required to support Redis persistence, and run Redis broker & system tasks.

   See below for example cases:

Redis Broker Total Memory	Redis Instance Memory Limit	Redis Service Instance Limit
16GB	512MB	14
16GB	256MB	28
64GB	512MB	56

   It is possible to configure a larger Redis Service Instance Limit, if you are confident that the majority of the deployed instances will not be using a large amount of their allocated memory, for example in development or test environments.

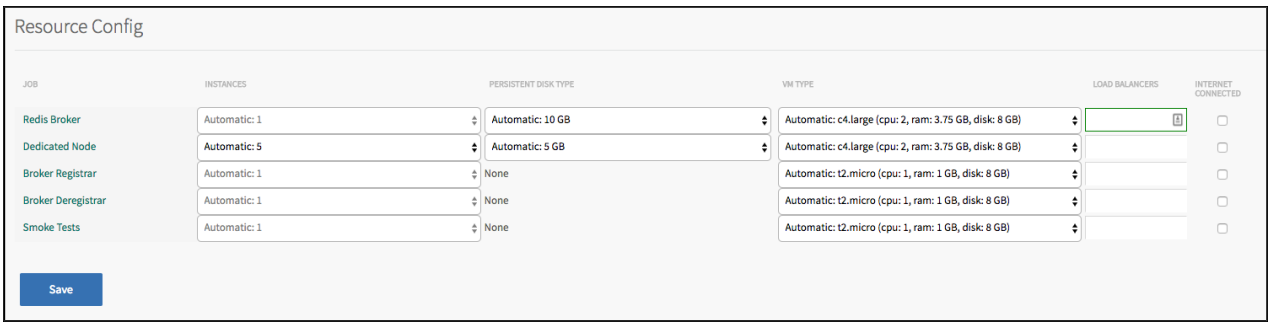
   Note: This is not supported, and could cause your server to run out of memory. If this happens your users may not be able to write any further data to any Redis instance.

4. Select the **Resource Config** tab to change the allocation of resources for the Redis Broker.
   The Redis Broker server will run all of the Redis instances for your Shared-VM plan. From this screen you may increase or decrease the CPU, RAM, Ephemeral Disk & Persistent Disk made available, as required.

5. Click the **Save** button.

Dedicated-VM Plan

1. Select the **Resource Config** tab to change the allocation of resources for the Dedicated Node.



   By default, 5 dedicated nodes will be created, each capable of running one Redis instance. You can increase or decrease the number of dedicated nodes, the size of the Persistent and Ephemeral Disks, and the CPU and RAM, as required. The default VM size is small; it is important that the

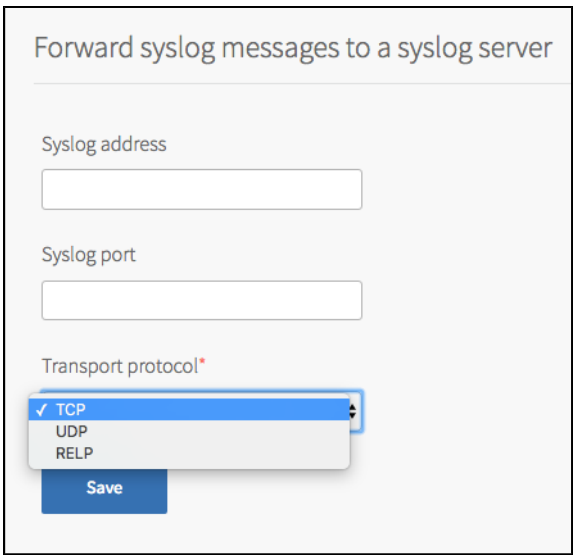
operator set the correct VM size to handle anticipated loads. Redis maxmemory is set to 45% of RAM. It is recommended the persistent disk be set to 3.5x the amount of RAM.

2. Click the **Save** button.

Configure Syslog Output

Pivotal recommends that operators configure a syslog output.

1. Add the Syslog address, Syslog port and transport protocol of your log management tool.
   The information required for these fields is provided by your log management tool.



2. Click the **Save** button.

Configure Backups

You can configure backups to be run for all instances, across both service plans.

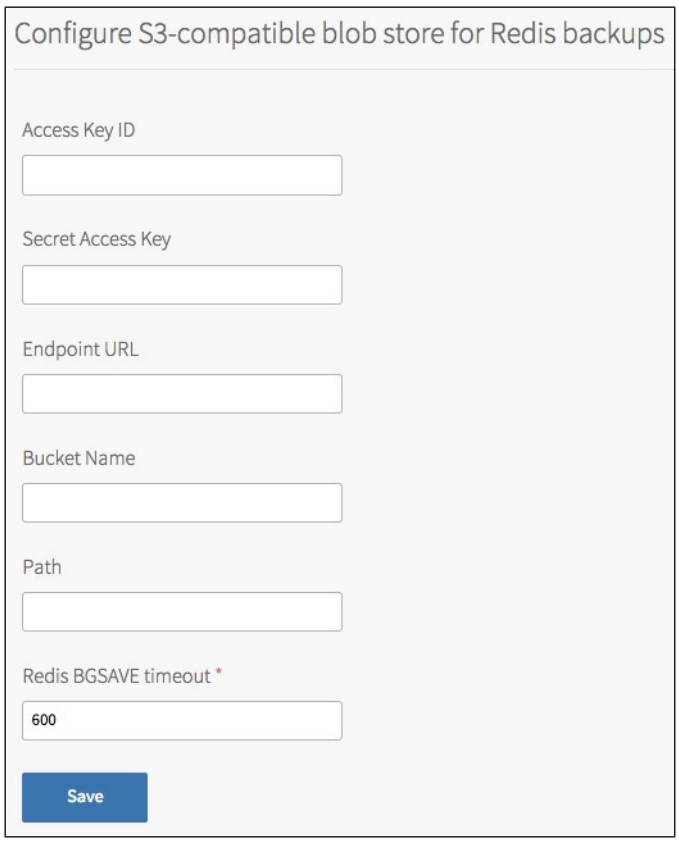
The key features are:

- Runs at midnight system time every day (not configurable)
- Every instance is backed up, across both service plans
- You can configure an S3 compatible blobstore as your destination
- Data from Redis is flushed to disk, before the backup is started by running a `BGSAVE` on each instance
- Currently certified and tested against AWS S3 only

Configuration

To enable backups to be taken, you need to configure the mandatory options in the `Redis for PCF` tile in OpsManager.

Click on the tile in OpsManager, followed by the `Backups` link on the left hand menu.



Access Key ID

This is your Access Key for your Blobstore

Required? No - this is optional, dependent upon whether it is required by your blobstore

Secret Access Key

This is your Secret associated with your access key id

Required? No - this is optional, dependent upon whether it is required by your blobstore

Endpoint URL

This is the endpoint for your blobstore e.g. `https://s3.amazonaws.com`

Required? Yes - If you want to enable backups to be run, you must populate this field.

Bucket Name

Name of the bucket inside your blobstore you wish the files to be stored in.

Required? Yes - If you want to enable backups to be run, you must populate this field.

Path

Path inside the bucket

Required? No - this is optional.

Redis BGSAVE Timeout

This is the amount of time that the backup process will wait for the BGSAVE command to complete on your instance, before transferring the RDB file to your configured blobstore.

You can increase this if required for your setup.

Required? - Yes, this defaults to 600 seconds.

AWS IAM Policy

The minimum set of policies required in order to upload the backup files are:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::<bucket-name>",
        "arn:aws:s3:::<bucket-name>/*"
      ]
    }
  ]
}
```

Make sure to replace `<bucket-name>` with your correct value.

Networks, Security, and Assigning AZs

Network Configuration

The following ports and ranges are used in this service:

Port	Protocol	Direction and Network	Reason
4001	tcp	Inbound to CloudFoundry network, outbound from service broker and service instance networks*	Used by the Redis metron_agent to forward metrics to the CloudFoundry etcd server
80	tcp	Outbound from CloudFoundry to the cf-redis-broker service broker network	(Only if using a cf-redis-broker) Access to the cf-redis-broker from the cloud controllers.
6379	tcp	Outbound from CloudFoundry to any service instance networks	Access to all nodes from the Diego Cell and Diego Brain network(s)
32768-61000	tcp	Outbound from CloudFoundry to the cf-redis-broker service broker network	From the Diego Cell and Diego Brain network(s) to the service broker VM. This is only required for the shared service plan.
80 or 443 (Typically)	http or https respectively	Outbound from any service instance networks	Access to the backup blobstore

* Typically the service broker network and service instance network(s) are the same.

Application Security Groups

To allow this service to have network access you must create Application Security Groups (ASGs). Ensure your security group allows access to the Redis Service Broker VM and Dedicated VMs configured in your deployment. You can obtain the IP addresses for these VMs in Ops Manager under the **Resource Config** section for the Redis tile.

Note: Without ASGs, this service is unusable.

Application Container Network Connections

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

Destination	Ports	Protocol	Reason
`ASSIGNED_NETWORK`	32768-61000	tcp	Enable application to access shared vm service instance
`ASSIGNED_NETWORK`	6379	tcp	Enable application to access dedicated vm service instance

Create an ASG called `redis-app-containers` with the above configuration and bind it to the appropriate space or, to give all started apps access, bind to the `default-running` ASG set and restart your apps. Example:

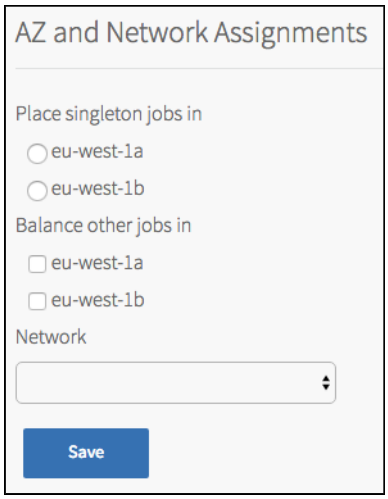
```
[
  {
    "protocol": "tcp",
    "destination": "<code>ASSIGNED_NETWORK</code>",
    "ports": "6379"
  }
]
```

Assigning AZs

Assigning multiple AZs to Redis jobs will not guarantee high availability.

All of your Shared-VM instances will run on a single node in just one of the configured availability zones and are therefore not highly availabile.

Each Dedicated-VM instance could be assigned to any of the configured availability zones, however each instance still operates as a single node with no clustering. This separation over availability zones provides no high availability.



Validating Installation

Smoke tests

Smoke tests are run as part of Redis for PCF installation to validate that the install succeeded. Smoke tests are described here.

Upgrading Redis for PCF

This product enables a reliable upgrade experience between versions of the product that is deployed through Ops Manager.

The upgrade paths are detailed here for each released version.

To upgrade the product:

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

During the upgrade deployment each Redis instance will experience a small period of downtime as each Redis instance is updated with the new software components. This downtime is because the Redis instances are single VMs operating in a non HA setup. The length of the downtime depends on whether there is a stemcell update to replace the operating system image or whether the existing VM can simply have the redis software updated. Stemcells updates incur additional downtime while the IaaS creates the new VM while updates without a stemcell update are faster.

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

Upgrading to a newer version of the product does not cause any loss of data or configuration. This is explicitly tested for during our build and test process for a new release of the product.

Release policy

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

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

Uninstalling Redis for PCF

To uninstall Redis for PCF, click on the trashcan icon in the lower right hand corner of the PCF Redis tile in the PCF Ops Manager Installation dashboard. Confirm deletion of the product and click apply changes.

Manual Backup and Restore of Redis for PCF

Manual Backups

It is possible to create a backup of an instance manually by following these steps:

- Follow these steps to log into your Ops Manager installation and target the Redis tile deployment.
- Identify the VM which holds your instance by running `bosh vms`.
  - For the `shared-vm` plan this will be the job name containing `cf-redis-broker`.
  - For the `dedicated-vm` plan this will be the job name containing `dedicated-node`.
  - You can identify the exact node for your `dedicated-vm` service instance by comparing the IP Address from your application bindings. Run `cf env <your-app-name>` to view your application bindings. The `host` field under `credentials` will contain the IP address of the dedicated VM instance. Use `bosh vms` to identify the VM with that IP address. In the `bosh ssh` step select the VM with that name.

An example output from `bosh vms`:

```
Deployment `p-redis-9dacffffa493b5e5a386'

Director task 129

Task 129 done

+-----------------------------------------+---------+-----------------------------------------+-----------+
| Job/index                               | State   | Resource Pool                           | IPs       |
+-----------------------------------------+---------+-----------------------------------------+-----------+
| cf-redis-broker-partition-default_az_guid/0 | running | cf-redis-broker-partition-default_az_guid | 10.0.0.58 |
| dedicated-node-partition-default_az_guid/0  | running | dedicated-node-partition-default_az_guid  | 10.0.0.59 |
+-----------------------------------------+---------+-----------------------------------------+-----------+
```

- Target your redis deployment with `bosh deployment`.
- `bosh ssh` into your desired node.

Persistence is enabled on these plans through the use of `RDB` files, using the following Redis config rules: `save 900 1 save 300 10 save 60 10000`

Shared-VM Plan

You can either take the latest RDB file held on disk, which is generated by the above the rules, or trigger a recent update by using the `redis-cli` to trigger a `BGSAVE`. Credentials to log into the `redis-cli` can be obtained from `VCAP_SERVICES` for your bound application.

The `redis-cli` is located in `/var/vcap/packages/redis/bin/redis-cli`.

On this plan, the `BGSAVE` command is aliased to a random string. This can be obtained from Ops Manager in the credentials tab.

Steps to Backup

- `bosh ssh` into your desired node. See the above section to identify the correct VM.
- Change to Root using `sudo -i`.
- Copy the contents of the `/var/vcap/store/cf-redis-broker` directory to a zip or tar file.
- Backup the file to your chosen location. You can use `bosh scp {job-name/index} --download /var/vcap/store/cf-redis-broker/redis-data/{instance id}/db/dump.rdb {location}` to copy the file from the service broker VM to another location. For example, `bosh scp cf-redis-broker/0 --download /var/vcap/store/cf-redis-broker/redis-data/{instance id}/db/dump.rdb /tmp` will copy the dump.rdb file to the `/tmp` directory on the local machine.

The `/var/vcap/store/cf-redis-broker` has sub-directories for each instance created of this plan. The backup file for each instance is called `dump.rdb`.

For example, here are two instances:

```
root@66358f3e-3428-46df-9bb3-9acc7770b188:/var/vcap/store/cf-redis-broker# find -type f | xargs ls -1 ./redis-data/3124f373-e9e2-44e1-ad12-a8865d8978b0/db/dump.rdb ./redis-data/3124f373-e9e2-44e1-ad12-a8865d8978b0/redis.conf ./redis-data/3124f373-e9e2-44e1-ad12-a8865d8978b0/redis-server.pid ./redis-data/62333bf9-f023-4566-b233-6686f26b8f4d/db/dump.rdb ./redis-data/62333bf9-f023-4566-b233-6686f26b8f4d/redis.conf ./redis-data/62333bf9-f023-4566-b233-6686f26b8f4d/redis-server.pid ./statefile.json
```

Dedicated-VM Plan

You can either take the latest RDB file on disk, as generated by the above rules, or trigger a more recent RDB file by executing the `BGSAVE` command using the `redis-cli`. Credentials can be obtained from the `VCAP_SERVICES` from your bound application. The `redis-cli` can be found in `/var/vcap/packages/redis/bin/redis-cli`.

Steps to Backup

- `bosh ssh` into your desired node. See the above section to identify the correct VM.
- Change to Root using `sudo -i`.
- Copy `/var/vcap/store/redis/dump.rdb` directory to a zip or tar file.
- Backup the file to your chosen location. You can use `bosh scp {job-name/index} --download /var/vcap/store/redis/dump.rdb {location}` to copy the file from the dedicated VM instance to another location. For example, `bosh scp dedicated-node/0 --download /var/vcap/store/redis/dump.rdb /tmp` will copy the dump.rdb file to the `/tmp` directory on the local machine.

Restore Redis Instance from a Backup

To a Local System

You can choose to restore the RDB file to a local Redis instance.

The steps to do this depend on your configuration and setup. Refer to the Redis documentation for more details.

To Pivotal Cloud Foundry

You can also restore your backup file to another instance of the `Redis for PCF` tile.

The below steps are manual.

Before restoring your RDB file you must have these prerequisites:

- Same resource configuration as the instance from which you backed up.
- The persistent disk should be increased to be `3.5 x size of the RDB file` if it is not already so. This allows space for the temporary files used during the restore process

1. Create a new instance of the plan that you wish to restore to.
2. Identify the VM which the instance of your plan is located on by following the steps from the `Manual Backups` section above.
3. `bosh ssh` into the identified VM.
4. Switch to root user `sudo su`
5. Follow the steps for Dedicated-VM Plan or Shared-VM Plan below.
6. You must to have access to the service instance password. This can be retrieved using the following command:
   - Password: `grep requirepass /var/vcap/store/redis/redis.conf`
7. Run `monit stop all`
8. Run `watch monit summary` to wait for monit services to enter the `not monitored` state.
9. Clean up existing Redis data files:
   - `rm -f /var/vcap/store/redis/appendonly.aof`
   - `rm -f /var/vcap/store/redis/dump.rdb`
10. Restore your Redis backup file to `/var/vcap/store/redis/dump.rdb` and correct the owner and permissions with `chown vcap:vcap /var/vcap/store/redis/dump.rdb && chmod 660 /var/vcap/store/redis/dump.rdb`

11. Edit the template Redis config file with `vim $(find /var/vcap/data/jobs/ -name redis.conf)` and make the following line changes:
    - `appendonly yes` -> `appendonly no`
12. Run `monit start all`
13. Run `watch monit summary` to wait for monit services to enter the `running` state.
14. Run `/var/vcap/packages/redis/bin/redis-cli -a {instance_password} BGREWRITEAOF`
15. Run `watch "/var/vcap/packages/redis/bin/redis-cli -a {instance_password} INFO | grep aof_rewrite_in_progress"` until `aof_rewrite_in_progress` is `0`
16. Run `monit stop all`
17. Run `watch monit summary` to wait for monit services to enter the `not monitored` state.
18. Edit the template Redis config file with `vim $(find /var/vcap/data/jobs/ -name redis.conf)` and make the following line changes:
    - `appendonly no` -> `appendonly yes`
19. Run `monit start all`

Shared-VM Plan

1. Make sure you have a service provisioned that you can restore to. The instance ID is required in future steps. You can retrieve this by running `cf service {instance name} --guid`
2. You must have access to the service instance password and port. This can be retrieved using the following commands:
    - Password: `grep requirepass /var/vcap/store/cf-redis-broker/redis-data/{instance_id}/redis.conf`
    - Port: `grep port /var/vcap/store/cf-redis-broker/redis-data/{instance_id}/redis.conf`
3. Run `monit stop all`
4. Run `watch monit summary` to wait for monit services to enter the `not monitored` state.
5. Run `pkill redis-server`
6. Clean up existing Redis data files:
    - `rm -f /var/vcap/store/cf-redis-broker/redis-data/{instance_id}/db/appendonly.aof`
    - `rm -f /var/vcap/store/cf-redis-broker/redis-data/{instance_id}/db/dump.rdb`
7. Confirm no running instances of `redis-server` with `ps aux | grep redis-server`
8. Restore your Redis backup file to `/var/vcap/store/cf-redis-broker/redis-data/{instance_id}/db/dump.rdb` and correct the owner and permissions with
   ```
   chown vcap:vcap /var/vcap/store/cf-redis-broker/redis-data/{instance_id}/db/dump.rdb && chmod 660 /var/vcap/store/cf-redis-broker/redis-data/{instance_id}/db/dump.rdb
   ```
9. Edit the template Redis config file with `vim $(find /var/vcap/data/jobs/ -name redis.conf)` and make the following line changes:
    - `appendonly yes` -> `appendonly no`
    - `rename-command BGREWRITEAOF ""` -> `rename-command BGREWRITEAOF "BGREWRITEAOFTEMP"`
10. Run `monit start all`
11. Run `watch monit summary` to wait for monit services to enter the `running` state.
12. Run `/var/vcap/packages/redis/bin/redis-cli -a {instance_password} -p {redis_port} BGREWRITEAOFTEMP`
13. Run
    `watch "/var/vcap/packages/redis/bin/redis-cli -a {instance_password} -p {instance_port} INFO | grep aof_rewrite_in_progress"`
    until `aof_rewrite_in_progress` is `0`
14. Run `monit stop all`
15. Run `watch monit summary` to wait for monit services to enter the `not monitored` state.
16. Run `pkill redis-server`
17. Confirm no running instances of `redis-server` with `ps aux | grep redis-server`

18. Edit the template Redis config file with `vim $(find /var/vcap/data/jobs/ -name redis.conf)` and make the following line changes:
    - `appendonly no` -> `appendonly yes`
    - `rename-command BGREWRITEAOF "BGREWRITEAOFTEMP"` -> `rename-command BGREWRITEAOF ""`
19. Run `monit start all`

Recovering Redis Instances

In the event of a recovery of Cloud Foundry, it is possible to recover bound Redis instances to healthy states that are in sync with Cloud Foundry. There are a few caveats to being able to recover previous instance state fully that depend on your plan.

Shared-VM Plan Caveats

- You need a backed up RDB Redis dump file - this would be stored in your S3 buckets if you have backups configured
- You need a backed up `/var/vcap/store/cf-redis-broker/redis-data` directory from the service broker node (you do not need to backup and `*.aof` or `*.rdb` files from subdirectories if you have backups configured)

Dedicated-VM Plan Caveats

- You need a backed up RDB Redis dump file - this would be stored in your S3 buckets if you have backups configured
- You need a backed up `/var/vcap/store/redis/statefile.json` from the service broker node

Note

This procedure assumes that a recovery of service information and service keys assigned to instances are restored with a restore of Cloud Foundry.

Recovery Procedure

After redeploying Redis, take the following steps.

Shared-VM Plan

1. `bosh ssh` into the service broker node of your Redis deployment
2. Run `monit stop all && pkill redis-server`
3. Wait for monit services to enter the `not monitored` state, you can watch this with `watch monit summary`
4. Confirm no running instances of `redis-server` with `ps aux | grep redis-server`
5. Copy the backed up `redis-data` directory into `/var/vcap/store/cf-redis-broker`
6. Follow the instructions here for your plan, skipping the first four steps described here, for restoring your backed up Redis data
7. Your Redis instance is now recovered

Dedicated-VM Plan

1. `bosh ssh` into the service broker node of your Redis deployment
2. Run `monit stop all`
3. Wait for monit services to enter the `not monitored` state, you can watch this with `watch monit summary`
4. Copy the backed up `/var/vcap/store/cf-redis-broker/statefile.json` and ensure ownership and permissions are correct with

```
chown vcap:vcap /var/vcap/store/redis/dump.rdb && chmod 660 /var/vcap/store/redis/dump.rdb
```

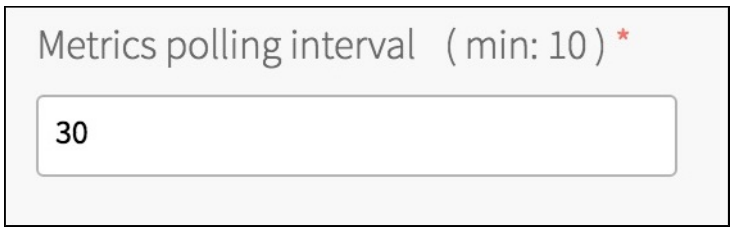
5. Follow the instructions here for your plan, skipping the first three steps described here, for restoring your backed up Redis data

6. Your Redis instance is now recovered

Monitoring Redis for PCF

The PCF firehose exposes Redis metrics.

The metrics polling interval defaults to 30 seconds. This can be changed by navigating to the Metrics configuration page and entering a new value in **Metrics polling interval (min: 10)**.



Third-party monitoring tools can consume Redis metrics to monitor Redis performance and health. For an example Datadog configuration that displays some of the significant metrics outlined below, see the CF Redis example dashboard. Pivotal does not endorse or provide support for any third party solution.

The following example shows the number of available instances for the Dedicated-VM plan metric:

```
origin:"p-redis" eventType:ValueMetric timestamp:1480084323333475533 deployment:"cf-redis" job:"cf-redis-broker" index:"3be5f4b9-cdf3-45c4-a3b2-19d923d63a01" ip:"10.0.1.49" valueMe
```

Redis Metrics

Redis emits a number of metrics that can be used to monitor the health and performance of your Redis deployment.

keyspace_hits

Description	Number of successful lookups of keys in the main dictionary. “/p-redis/info/stats/keyspace_hits”
Significance	In conjunction with `keyspace_misses`, it can be used to calculate the hit ratio.
Notes	A successful lookup is a lookup on a key that exists.

keyspace_misses

Description	Number of unsuccessful lookups of keys in the main dictionary. “/p-redis/info/stats/keyspace_misses”
Significance	In conjunction with `keyspace_hits`, it can be used to calculate the hit ratio.
Notes	An unsuccessful lookup is a lookup on a key that does not exist.

used_memory

Description	Number of bytes allocated by Redis. “/p-redis/info/memory/used_memory”
Significance	Grows as the number of unsaved keys increases.

maxmemory

Description	Maximum number of bytes available in Redis. “/p-redis/info/memory/maxmemory”
Significance	Grows as the number of unsaved keys increases.

blocked_clients

Description	Number of connected clients pending on a blocking call. “/p-redis/info/clients/blocked_clients”
Significance	Can be used as an indicator to detect deadlocks.

connected_clients

Description	Number of clients connected to the Redis instance. “/p-redis/info/clients/connected_clients”

rdb_changes_since_last_save

Description	Number of keys currently in memory. “/p-redis/info/persistence/rdb_changes_since_last_save”
Significance	Memory usage grows in proportion to the number of keys in memory. If the Redis instance is stopped ungracefully, these changes may be lost.
Notes	Performing a `BGSAVE` writes these keys to disk and frees up memory.

total_commands_processed

Description	Total number of commands processed by Redis. “/p-redis/info/stats/total_commands_processed”
Significance	A crude indicator of activity. Can be used in conjunction with `uptime_in_seconds`.

mem_fragmentation_ratio

Description	Ratio of memory allocated by the operating system to the memory requested by Redis. “/p-redis/info/memory/mem_fragmentation_ratio”
Significance	A ratio in excess of 1.5 indicates excessive fragmentation, with your Redis instance consuming 150% of the physical memory it requested..

total_instances

Description	Total number of `dedicated-vm` instances of Redis. “/p-redis/service-broker/dedicated_vm_plan/total_instances”
Significance	Used in conjunction with `available_instances`, provides information about used instances.

available_instances

Description	Number of available `dedicated-vm` instances of Redis. “/p-redis/service-broker/dedicated_vm_plan/total_instances”
Significance	If zero, no more instances are available.

Description	Total number of `shared-vm` instances of Redis. “/p-redis/service-broker/shared_vm_plan/total_instances”
Significance	Used in conjunction with `available_instances`, provides information about used instances.

available_instances

Description	Number of available `shared-vm` instances of Redis. “/p-redis/service-broker/shared_vm_plan/total_instances”
Significance	If zero, no more instances are available.

Other Metrics

Redis also exposes the following metrics. for more information, see the Redis documentation.

- `arch_bits`
- `uptime_in_seconds`
- `uptime_in_days`
- `hz`
- `lru_clock`
- `client_longest_output_list`
- `client_biggest_input_buf`
- `used_memory_rss`
- `used_memory_peak`
- `used_memory_lua`
- `mem_fragmentation_ratio`
- `loading`
- `rdb_bgsave_in_progress`
- `rdb_last_save_time`
- `rdb_last_bgsave_time_sec`
- `rdb_current_bgsave_time_sec`
- `aof_rewrite_in_progress`
- `aof_rewrite_scheduled`
- `aof_last_rewrite_time_sec`
- `aof_current_rewrite_time_sec`
- `total_connections_received`
- `total_commands_processed`
- `instantaneous_ops_per_sec`
- `total_net_input_bytes`
- `total_net_output_bytes`
- `instantaneous_input_kbps`
- `instantaneous_output_kbps`
- `rejected_connections`
- `sync_full`
- `sync_partial_ok`
- `sync_partial_err`
- `expired_keys`
- `evicted_keys`
- `keyspace_hits`
- `keyspace_misses`
- `pubsub_channels`
- `pubsub_patterns`
- `latest_fork_usec`
- `migrate_cached_sockets`
- `connected_slaves`
- `master_repl_offset`
- `repl_backlog_active`
- `repl_backlog_size`
- `repl_backlog_first_byte_offset`

- `repl_backlog_histlen`
- `used_cpu_sys`
- `used_cpu_user`
- `used_cpu_sys_children`
- `used_cpu_user_children`
- `cluster_enabled`
- `rdb_last_bgsave_status`
- `aof_last_bgrewrite_status`
- `aof_last_write_status`

Troubleshooting Redis for PCF

General troubleshooting guides for PCF:

- PCF 1.8
- PCF 1.7
- PCF 1.6

Knowledgebase containing PCF services articles Four Redis-specific articles:

- Can't redeploy PCF Redis if shared-vm persistent disk full
- Issue with upgrading tile
- Issue with deploy failing
- Redis Instance Alive after Successful De-provisioning

Other issues you might encounter:

Error	`Failed to target Cloud Foundry`
Cause	Your Pivotal Cloud Foundry is unresponsive
Solution	Examine the detailed error message in the logs and check the PCF Troubleshooting Guide for advice

Error	`Failed to bind Redis service instance to test app`
Cause	Your deployment's broker has not been registered with Pivotal Cloud Foundry
Solution	Examine the broker-registrar installation step output and troubleshoot any problems.

Useful Debugging Information

If you encounter an issue, here is a list of useful information to gather, especially before you perform any destructive operations such as `cf purge-service-offerings` or `bosh delete deployment`.

- PCF Redis version
- Previous PCF Redis version if upgrading
- Ops Manager version
- Ops Manager installation logs
- IaaS description
- For all VMs:
  - Copy of /var/vcap/sys/log (particularly the broker)
  - Logs from a forwarded endpoint
  - `monit summary`
  - `Full ps aux` - Has monit done its job?
  - `ps aux | grep redis-serve[r]` - Are Redis instances running?
  - `df -h` - disk usage
  - `free -m` - memory usage
  - `cf m`
  - `tree /var/vcap/store/cf-redis-broker/redis-data` (broker only)
  - Copy of /var/vcap/store/cf-redis-broker/statefile.json

For App Developers

Redis Configuration

Redis is configured with a maxmemory-policy of no-eviction. This policy means that the once memory is full, the service will not evict any keys and no write operations will be possible until memory becomes available. Persistence is configured for both RDB and AOF. The default maximum number of connections, maxclients, is set at 10000 but this number is adjusted by Redis according to the number of file handles available. Replication and event notification are not configured.

Service Plans

PCF Redis offers Dedicated VM and Shared VM plans. The memory allocated to the plans is determined by the operator at deploy time. For more information on the plans see the architecture and recommended usage pages.

Using Redis for PCF

Instructions for creating, binding to, and deleting an instance of the dedicated-VM or shared-VM plan are here.

Getting Started

Using PCF Redis with Spring

Spring Cloud Connectors can connect to PCF Redis. Spring Cloud Cloud Foundry connectors will automatically connect to PCF Redis.

PCF Dev

PCF Dev is a small footprint version of PCF that's small enough to run on a local developer machine. More info herehttps://pivotal.io/pcf-dev.

Redis Example App

Sample ruby code that uses PCF can be found here https://github.com/pivotal-cf/cf-redis-example-app.

Redis

To learn more about Redis itself, visit redis.io.

Using Redis for PCF

Redis for PCF can be used both via Pivotal Apps Manager and the CLI, both methods are outlined below. An example application has also been created to help application developers get started with Redis for PCF, and can be downloaded here.

See Redis for PCF Recommended Usage for recommendations regarding Redis for PCF service plans, and memory allocation.

Creating a Redis Service Instance

The following procedure describes how to create a Redis service instance in the Pivotal Cloud Foundry Elastic Runtime environment.

Available Plans

Before creating a Redis instance, it is worth being aware of the two available plans:

Plan Name	Suitable for	Tenancy Model per Instance	Highly Available	Backup Functionality
Shared-VM	Lighter workloads that do not require dedicated resources	Shared VM	No	Yes
Dedicated-VM	Increased workloads that require dedicated resources	Dedicated VM	No	Yes

Using Pivotal Apps Manager

1. From within Pivotal Apps Manager, select Marketplace from the left navigation menu under Spaces. The Services Marketplace displays.
2. Select **Redis** from the displayed tiles and click to view the available plans.
3. Click on the appropriate **Select this plan** button to select the required **Redis Service Plan**.
4. In the Instance Name field, enter a name that will identify this specific Redis service instance.
5. From the Add to Space drop-down list, select the space where you or other users will deploy the applications that will bind to the service.
6. Click the **Add** button.

Using the CLI

1. Run the following command to view the available service plans.

```
$ cf marketplace
```

This should produce the output:

```
Getting services from marketplace in org system / space apps-manager as admin...
OK

service      plans                  description
p-redis      shared-vm, dedicated-vm   Redis service to provide a key-value store

TIP:  Use 'cf marketplace -s SERVICE' to view descriptions of individual plans of a given service.
```

2. Type the following command to create the service plan:

```
$ cf create-service p-redis <service-plan-name> <service-instance-name>
```

The service-plan-name is as seen in the Services marketplace – in this example, “shared-vm” – and the service-instance-name is a descriptive name that you want to use for the service.

For example

```
$ cf create-service p-redis shared-vm redis
```

Binding an Application to the Redis Service

The following procedures describe how to bind a Redis service instance to your Pivotal Cloud Foundry application. This can be done via the Pivotal Apps Manager or Using the Pivotal Cloud Foundry CLI.

Using Pivotal Apps Manager

1. Select the application that you wish to bind to the service. A page displays showing the already bound services and instances for this application.
2. Click Bind. A list of available services displays.
3. Click the Bind button for the Redis service you want to bind to this application.
4. Using the Pivotal Cloud Foundry CLI, start or restage your application.

```
$ cf restage <application-name>
```

Using the CLI

1. Run the following command to view running service instances.

```
$ cf services
```

This should produce the output:

```
Getting services in org system / space apps-manager as admin...
OK

name              service   plan      bound apps   last operation
my-redis-instance p-redis   shared-vm              create succeeded
```

2. Run the following command to bind the application to the service instance.

```
$ cf bind-service <application-name> <service-instance-name>
```

For example:

```
$ cf bind-service my-application redis
```

3. Restage your application.

```
$ cf restage <application-name>
```

Deleting a Redis Instance

When you delete a Redis service instance, all applications that are bound to that service are automatically unbound and any data in the service instance is cleared.

Using Pivotal Apps Manager

1. Locate the row under Services that contains the service instance you want to delete and click Delete.

2. If you had applications that were bound to this service, you may need to restage or re-push your application for the application changes to take effect.

```
$ cf restage <application-name>
```

Using the CLI

1. Run the following command.

```
$ cf delete-service <service-instance-name>
```

The service-instance-name is that of the service instance that you would like to delete. Enter ‘y’ when prompted.

For example:

```
$ cf delete-service my-redis-instance

Really delete the service my-redis-instance?> y
Deleting service my-redis-instance in org system / space apps-manager as admin...
OK
```

2. If you had applications that were bound to this service, you may need to restage or re-push your application for the application changes to take effect.

```
$ cf restage <application-name>
```

Sample Redis Configuration

The following is a `redis.conf` file from a Dedicated-VM plan instance:

```
daemonize yes
pidfile /var/vcap/sys/run/redis.pid
port 6379
tcp-backlog 511
timeout 0
tcp-keepalive 0
loglevel notice
logfile /var/vcap/sys/log/redis/redis.log
syslog-enabled yes
syslog-ident redis-server
syslog-facility local0
databases 16
save 900 1
save 300 10
save 60 10000
stop-writes-on-bgsave-error yes
rdbcompression yes
rdbchecksum yes
dbfilename dump.rdb
dir /var/vcap/store/redis
slave-serve-stale-data yes
slave-read-only yes
repl-diskless-sync no
repl-diskless-sync-delay 5
repl-ping-slave-period 10
repl-timeout 60
repl-disable-tcp-nodelay no
slave-priority 100
maxmemory-policy noeviction
appendonly yes
appendfilename appendonly.aof
appendfsync everysec
no-appendfsync-on-rewrite no
auto-aof-rewrite-percentage 100
auto-aof-rewrite-min-size 64mb
aof-load-truncated yes
lua-time-limit 5000
slowlog-log-slower-than 10000
slowlog-max-len 128
latency-monitor-threshold 0
notify-keyspace-events ""
hash-max-ziplist-entries 512
hash-max-ziplist-value 64
list-max-ziplist-entries 512
list-max-ziplist-value 64
set-max-intset-entries 512
zset-max-ziplist-entries 128
zset-max-ziplist-value 64
hll-sparse-max-bytes 3000
activerehashing yes
client-output-buffer-limit normal 0 0 0
client-output-buffer-limit slave 256mb 64mb 60
client-output-buffer-limit pubsub 32mb 8mb 60
hz 10
aof-rewrite-incremental-fsync yes
rename-command CONFIG "A-B-Ab1AZec_-AaC1A2bAbB22a_a1Baa"
rename-command SAVE "SAVE"
rename-command BGSAVE "BGSAVE"
rename-command DEBUG ""
rename-command SHUTDOWN ""
rename-command SLAVEOF ""
rename-command SYNC ""
requirepass 1a1a2bb0-0ccc-222a-444b-1e1e1e1e2222
maxmemory 1775550873
```