Pivotal

PRODUCT DOCUMENTATION

Pivotal Cloud Cache for PCF® Documentation

Version 1.0.8

Published: 15 Nov 2018

Table of Contents

Pivotal Cloud Cache

Note: Pivotal Cloud Cache (PCC) v1.0.8 is no longer supported. The support period for PCC has expired. To stay up-to-date with the latest software and security updates, upgrade to a supported version.

Overview

Pivotal Cloud Cache (PCC) is a high-performance caching layer for Pivotal Cloud Foundry (PCF). PCC offers an in-memory key-value store. It delivers low-latency responses to a large number of concurrent data access requests.

PCC provides a service broker to create in-memory data clusters on demand. These clusters are dedicated to the PCF space and tuned for specific use cases defined by the plan. Service operators can create multiple plans to support different use cases.

PCC uses Pivotal GemFire. The Pivotal GemFire API Documentation details the API for client access to data objects within Pivotal GemFire.

This documentation performs the following functions:

- Describes the features and architecture of PCC
- Provides the PCF operator with instructions for installing, configuring, and maintaining PCC
- Provides app developers instructions for choosing a service plan, creating and deleting PCC service instances, and binding apps

Product Snapshot

Current PCC details:

- **Version:** v1.0.8
- **Release Date:** November 15, 2017
- **Software Component Version:** GemFire v9.1.1
- **Compatible Ops Manager Version:** v1.9.x and v1.10.x
- **Compatible Elastic Runtime Version:** v1.9.21+ and v1.10.x
- **vSphere Support:** Yes
- **Azure Support:** Yes
- **GCP Support:** Yes
- **OpenStack Support:** Yes
- **AWS Support:** Yes
- **IPsec Support:** No

Known Issues

Service Instance Upgrade Fails When Pulse Has Open Connections

Affected Versions: 1.0.0 - 1.0.5

The `upgrade-all-service-instances` errand fails. On the PCC service instance deployment, this may appear with an error similar to the following:

```
19:38:27 | Updating instance locator: locator/d6d98feb-f005-49ca-b1c1-3e2803811cc8 (0) (canary) (00:10:20)
        L Error: Action Failed get_task: Task f425d048-954e-424c-642f-3ba2f2b3bec4 result: Unmounting persistent disk: Running command: 'umount /dev/sdc1', stdout: '', stderr: 'umount: /var/vcap/store: device
     (In some cases useful info about processes that use
      the device is found by lsof(8) or fuser(1))
      ': exit status 1
```

This is due to a bug in GemFire where the locators cannot stop successfully when there is an open connection to the Pulse dashboard. This issue can be tracked in the Apache Geode Jira here.

Mitigation Option 1

If possible, close all connections to the Pulse dashboards before upgrading the PCC tile.

Mitigation Option 2

If you have not yet run the `upgrade-all-service-instances` errand, follow the steps below:

1. SSH onto the Operations Manager VM.
2. Identify each PCC deployment using the BOSH CLI.
3. For each PCC deployment, use `bosh ssh` to SSH onto a locator VM.
4. On the locator VM, change yourself to the root user by running `sudo su`.
5. On the locator VM, use `monit` to stop the `route_registrar` job by running `monit stop route_registrar`.
6. Repeat steps 4-5 for all the locators for each PCC deployment.
7. Navigate back to the Operations Manager dashboard and click **Apply Changes**.
8. The `upgrade-all-service-instances` errand should complete successfully.

Mitigation Option 3

If you have already run the `upgrade-all-service-instances` errand and now have a stopped locator in one of your PCC deployments, follow the steps below:

1. SSH onto the Operations Manager VM.
2. Identify the locator which is in a stopped state using the BOSH CLI. This can be done by running `bosh vms`.
3. On the locator VM, change yourself to the root user by running `sudo su`.
4. (Optional) On the locator VM, run `monit summary` to confirm that the `gemfire-locator` process is not monitored.
5. Kill the process by running `kill -9 "$(ps -ef | grep LocatorLauncher | head -n 1 | awk '{print $2}')"`.

After killing the LocatorLauncher process on the stopped Locator VM, proceed with option 2 above.

Pulse Monitoring Tool Issue

The topology diagram might not be accurate and might display more members in a cluster than the cluster actually contains. However, the numerical value displayed on the top bar is accurate.

Updating a PCC Service Instance Fails if syslog TLS Is Disabled

Affected Versions: 1.0.0 - 1.0.6

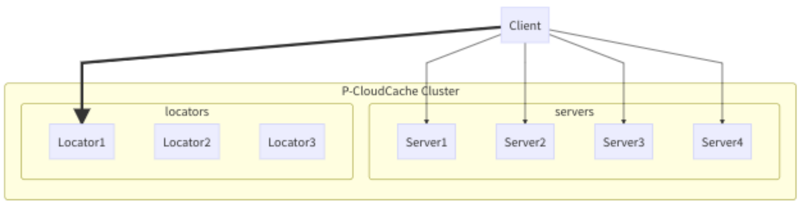
If you have syslog enabled but want to disable syslog TLS, you will not be able to upgrade PCC. You will encounter the following error:

```
Service broker error: previous manifest locator job must contain syslog.tls properties if syslog properties exist
```

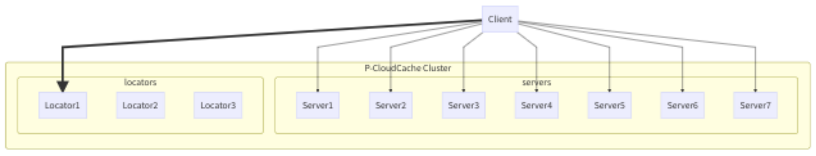
Architecture

PCC deploys cache clusters that use Pivotal GemFire to provide high availability, replication guarantees, and eventual consistency.

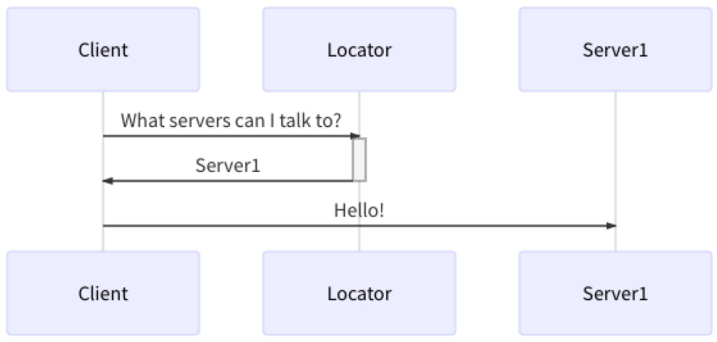
When you first spin up a cluster, you'll have three locators and at least four servers.



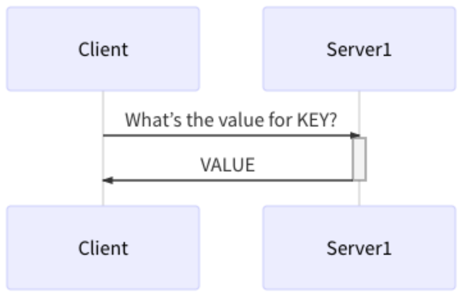
If you scale the cluster up, you'll have more servers, increasing the capacity of the cache. There always will be three locators.



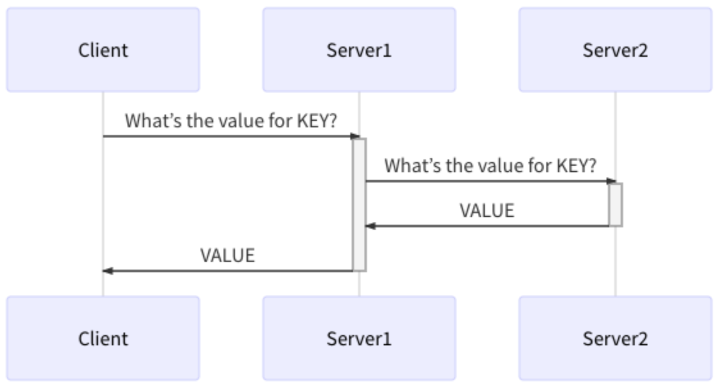
When a client connects to the cluster, it first connects to a locator. The locator replies with the IP address of a server for it to talk to. The client then connects to that server.



When the client wants to read or write data, it sends a request directly to the server.

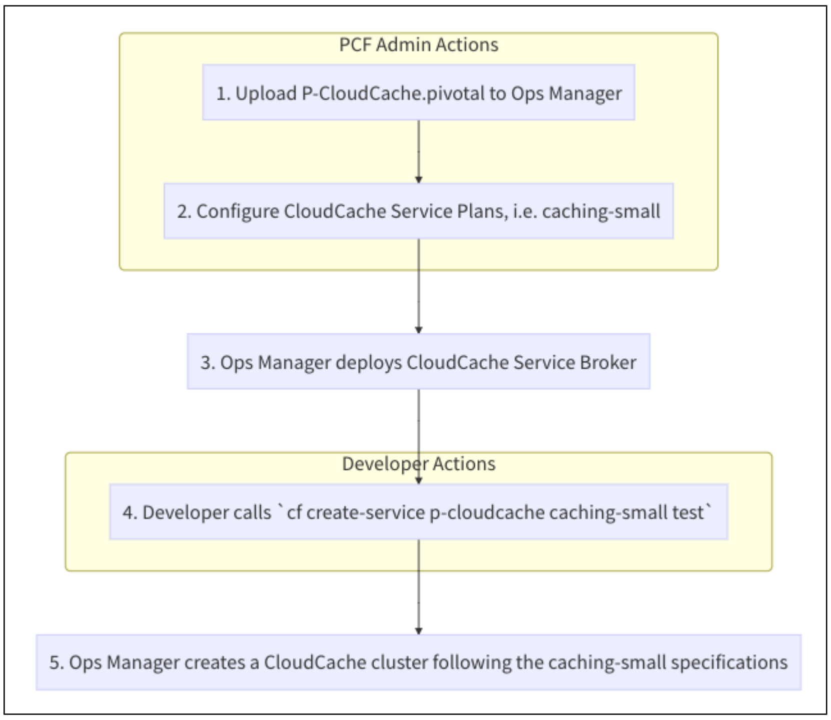


If the server doesn't have the data locally, it fetches it from another server.



Workflow

The workflow for the PCF admin setting up a PCC service plan:


PCF Admin Actions
1. Upload P-CloudCache.pivotal to Ops Manager
2. Configure CloudCache Service Plans, i.e. caching-small
3. Ops Manager deploys CloudCache Service Broker
Developer Actions
4. Developer calls `cf create-service p-cloudcache caching-small test`
5. Ops Manager creates a CloudCache cluster following the caching-small specifications

Recommended Usage and Limitations

- PCC can be used as a cache. It supports the look-aside cache pattern.
- PCC can be used to store objects in key/value format, where the value can be any object.
- PCC works with gfsh. You can only use the gfsh version that matches this release's version of GemFire. For the version of GemFire supported in this release, see Product Snapshot, above.
- PCC supports basic OQL queries, with no support for joins.

Limitations

- Scale down of the cluster is not supported.
- Plan migrations, for example `-p` flag with the `cf update-service` command, are not supported.
- WAN (Cross Data Center) replication is not supported.
- Persistent regions are not supported.

Security

Pivotal recommends that you do the following:

- Run PCC in its own network
- Use a load balancer to block direct, outside access to the Gorouter

To allow PCC network access from apps, you must create application security groups that allow access on the following ports:

- 1099
- 8080
- 40404
- 55221

For more information, see the PCF Application Security Groups topic.

Authentication

Clusters are created with two default users: `cluster_operator` and `developer`. A cluster can only be accessed using one of these two users. All client applications, gfsh, and JMX clients need to use one of these users to access the cluster.

Authorization

Default user roles `cluster_operator` and `developer` have different permissions:

- `cluster_operator` role has `CLUSTER:WRITE`, `CLUSTER:READ`, `DATA:MANAGE`, `DATA:WRITE`, and `DATA:READ` permissions.
- `developer` role has `CLUSTER:READ`, `DATA:WRITE`, and `DATA:READ` permissions.

You can find more details about these permissions in the Pivotal GemFire Implementing Authorization topic.

Feedback

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

Pivotal Cloud Cache Operator Guide

This document describes how a Pivotal Cloud Foundry (PCF) operator can install, configure, and maintain Pivotal Cloud Cache (PCC).

Requirements for Pivotal Cloud Cache

Service Network

You must have access to a Service Network in order to install PCC.

Minimum Version Requirements

PCC requires PCF with PCF Elastic Runtime v1.9.11 or later.

Beginning Your Installation of Pivotal Cloud Cache

Follow the steps below to start installing PCC on PCF:

1. Download the tile from the Pivotal Network.
2. Click **Import a Product** to import the tile into Ops Manager.
3. Click the **+** symbol next to the uploaded product description.
4. Click on the Cloud Cache tile. From here, you can click through the **Settings**, **Status**, **Credentials**, and **Logs** tabs to start configuring PCC.

Configure Tile Properties

Assign Availability Zones and Networks

1. Under **Place singleton jobs in**, select the Availability Zone (AZ) where your singleton virtual machines (VMs) will reside.
2. Under **Balance other jobs in**, select the AZ(s) you want to use for distributing other GemFire VMs. We recommend selecting all of them.
3. For **Network**, select your Elastic Runtime network.
4. For **Service Network**, select the network to be used for GemFire VMs.
5. Click **Save**.

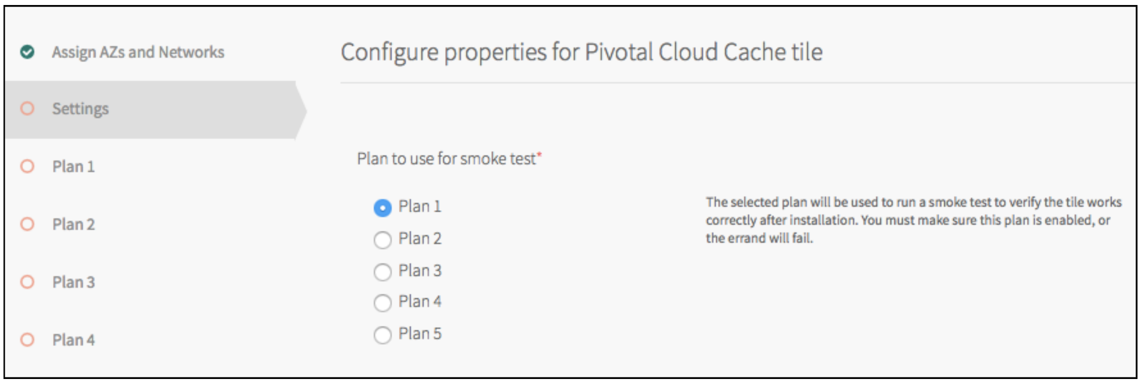
Settings: Smoke Tests

Click the **Settings** heading on the the left side of the page to go to the next set of configurable properites. Here, you can choose a plan to use for smoke tests.

This section allows you to configure the `smoke-tests` errand that runs after tile installation. The errand verifies that your installation was successful.

From the list shown, select a plan to use when the `smoke-tests` errand runs. The `smoke-tests` errand uses the selected plan. Ensure the selected plan is enabled on the plan-configuration page. If the selected plan is not enabled, the `smoke-tests` errand will fail.

Pivotal recommends that you use the smallest four-server plan for smoke tests. Because smoke tests create and later destroy this plan, using a very small plan reduces installation time.



Configure properties for Pivotal Cloud Cache tile

Plan to use for smoke test*

[✓ Plan 1]
Plan 2
Plan 3
Plan 4
Plan 5

The selected plan will be used to run a smoke test to verify the tile works correctly after installation. You must make sure this plan is enabled, or the errand will fail.

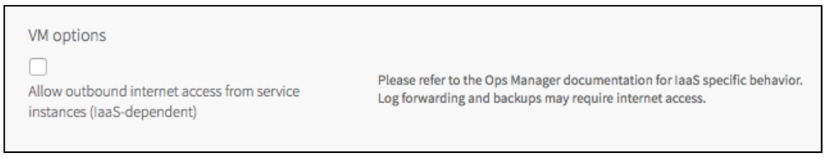
By default, the `smoke-test` errand runs on the `system` org and the `p-cloudcache-smoke-test` space.

Settings: Allow Outbound Internet Access

The **Allow outbound internet access from service instances** checkbox is unchecked by default.

Check this box if you want to allow outbound internet traffic (this is IaaS dependent; see Ops Manager documentation for details). For configuring an external syslog endpoint, this box may need to be checked.

Check **Allow outbound internet access from service instances** if BOSH is configured to use an external blob store; this enables the PCC tile to allow external internet access.



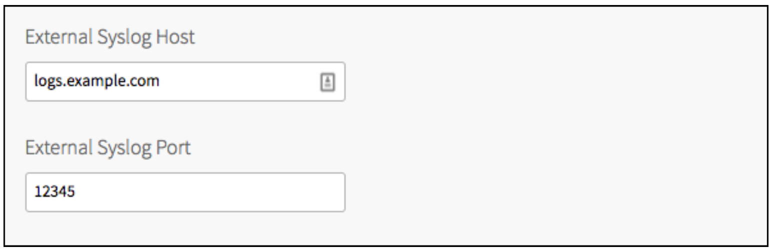
VM options

☐ Allow outbound internet access from service instances (IaaS-dependent)

Please refer to the Ops Manager documentation for IaaS specific behavior. Log forwarding and backups may require internet access.

Settings: External Syslog

PCC supports forwarding syslog to an external log management service (e.g. papertrail, splunk, or your custom enterprise log sink). To enable remote syslog for the service broker, provide a host and port in the **External Syslog Host** and **External Syslog Port** fields.

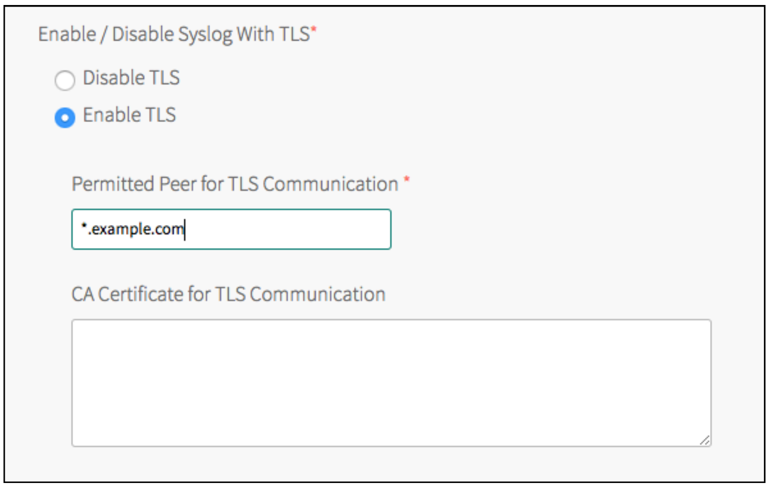


External Syslog Host: logs.example.com

External Syslog Port: 12345

The broker logs are useful for debugging problems creating, updating, and binding service instances.

Remote syslog sends unencrypted logs by default. To secure log transmission, you can enable TLS by selecting the **Enable TLS** option button. We recommend enabling TLS, as most syslog endpoints such as Papertrail and Logsearch require TLS. If there are several peer servers that may respond to remote syslog connections, you need to provide a regex in the **Permitted Peer for TLS Communication** field (shown here as *.example.com). You may need to provide the CA certificate of the log management service endpoint if the server certificate is not signed by a known authority (for exampe, an internal syslog server).



Enable / Disable Syslog With TLS*

○ Disable TLS

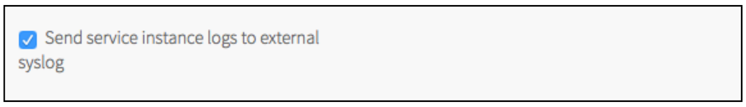
● Enable TLS

Permitted Peer for TLS Communication *

*.example.com

CA Certificate for TLS Communication

By default, only the broker logs will be forwarded to your configured log management service. If you would like to forward server and locator logs from all service instances please check the "Send service instance logs to external syslog" checkbox.

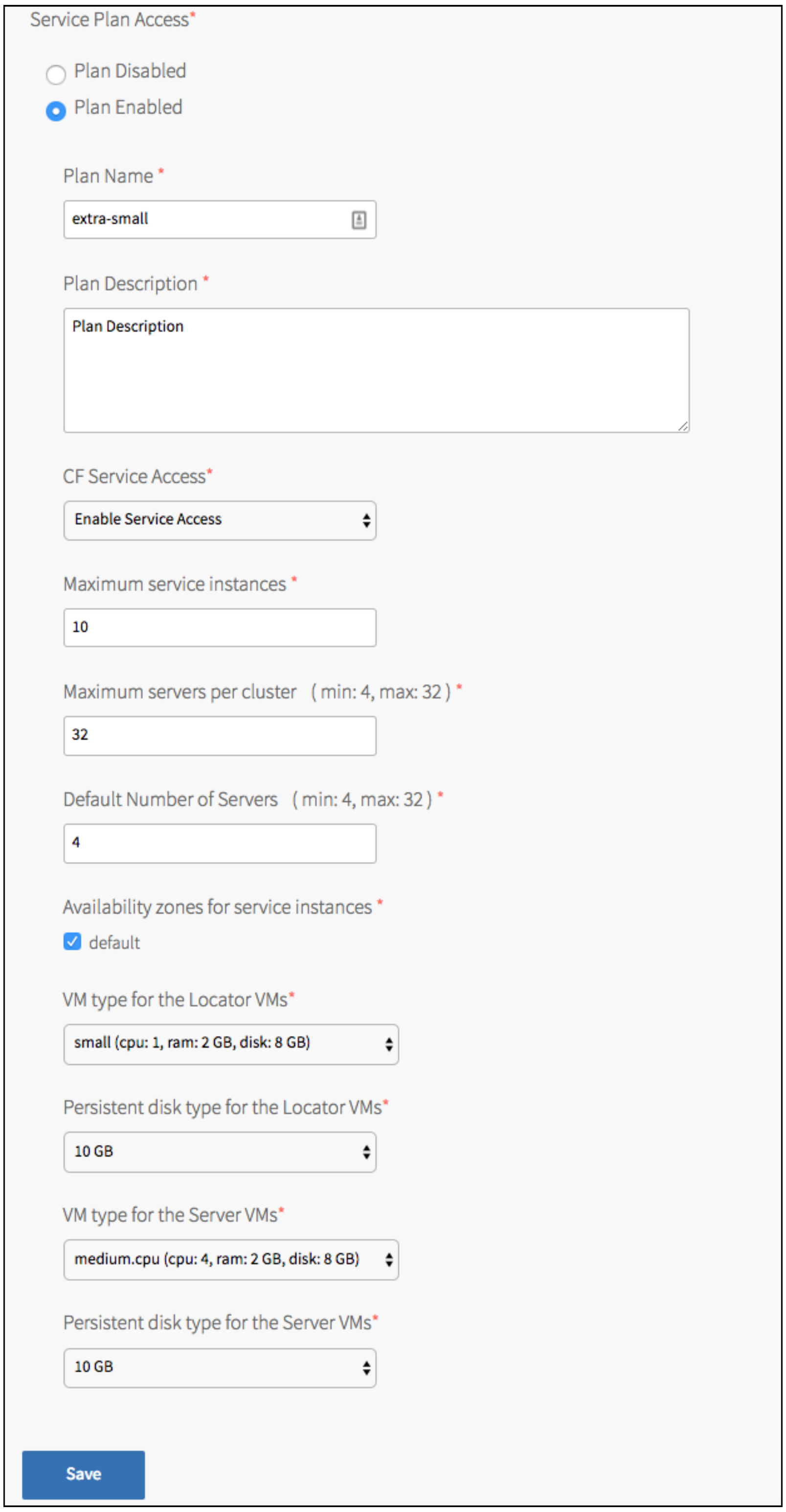


✓ Send service instance logs to external syslog

You may want to enable remote syslog for service instances if you would like to monitor the health of the clusters. However, this will generate a large volume of logs, which is why it is disabled by default. The broker logs only include information about service instance creation, not on-going cluster health. Please note that service instance logs will be sent to the same host and port configured for the broker logs.

Configure Service Plans

You can configure five individual plans for your developers. Select the **Plan 1** through **Plan 5** tabs to configure each of them.


Service Plan Access*

( ) Plan Disabled
(●) Plan Enabled

Plan Name *
extra-small

Plan Description *
Plan Description

CF Service Access*
Enable Service Access

Maximum service instances *
10

Maximum servers per cluster (min: 4, max: 32) *
32

Default Number of Servers (min: 4, max: 32) *
4

Availability zones for service instances *
✓ default

VM type for the Locator VMs*
small (cpu: 1, ram: 2 GB, disk: 8 GB)

Persistent disk type for the Locator VMs*
10 GB

VM type for the Server VMs*
medium.cpu (cpu: 4, ram: 2 GB, disk: 8 GB)

Persistent disk type for the Server VMs*
10 GB

The **Enable Plan** toggle is checked by default. If you do not want to add this plan to the CF service catalog, select **Disable Plan**. You must enable at least one plan.

The **CF Service Access** dropdown allows you to configure the plan's visibility in the CF Marketplace. **Enable Service Access** will display the service plan to all developers in the CF marketplace. **Disable Service Access** will not display the service plan to developers in the CF marketplace and cannot be enabled at a later time. **Leave Service Access Unchanged** will not display the service plan in the CF marketplace by default but can be enabled at a later time.

The **Plan Name** text field allows you to customize the name of the plan. This plan name is displayed to developers when they view the service in the Marketplace.

The **Plan Description** text field allows you to supply a plan description. The description is displayed to developers when they view the service in the Marketplace.

The **Service Instance Quota** sets the maximum number of PCC clusters that can exist simultaneously.

When developers create or update a service instance, they can specify the number of servers in the cluster. The **Maximum servers per cluster** field allows operators to set an upper bound on the number of servers developers can request. If developers do not explicitly specify the number of servers in a service instance, a new cluster has the number of servers specified in the **Default Number of Servers** field.

The **Availability zones for service instances** setting determines which AZs are used for a particular cluster. The members of a cluster are distributed evenly across AZs.

WARNING After you've selected AZs for your service network, you cannot add additional AZs; doing so causes existing service instances to lose data on update.

The remaining fields control the VM type and persistent disk type for servers and locators. The total size of the cache is directly related to the number of servers and the amount of memory of the selected server VM type. We recommend the following configuration:

- For the **VM type for the Locator VMs** field, select a VM that has at least 1 GB of RAM and 4 GB of disk space.
- For the **Persistent disk type for the Locator VMs** field, select 10 GB or higher.
- For the **VM type for the Server VMs** field, select a VM that has at least 4 GB of RAM and 8 GB of disk space.
- For the **Persistent disk type for the server VMs** field, select 10 GB or higher.

When you finish configuring the plan, click **Save** to save your configuration options.

Stemcell

Ensure you import the correct type of stemcell indicated on this tab.

You can download the latest available stemcells from Pivnet.

BOSH System Health Metrics

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

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

Upgrading Pivotal Cloud Cache

Follow the steps below to upgrade PCC on PCF:

1. Download the new version of the tile from the Pivotal Network.
2. Upload the product to Ops Manager.
3. Click **Add** next to the uploaded product.
4. Click on the Cloud Cache tile and review your configuration options.
5. Click **Apply Changes**.

Updating Pivotal Cloud Cache Plans

Follow the steps below to update plans in Ops Manager.

1. Click on the Cloud Cache tile.
2. Click on the plan you want to update under the **Information** section.
3. Edit the fields with the changes you want to make to the plan.

4. Click **Save** button on the bottom of the page.

5. Click on the **PCF Ops Manager** to navigate to the **Installation Dashboard**.

6. Click **Apply Changes**.

Plan changes are not applied to existing services instances until you run the `upgrade-all-service-instances` BOSH errand. You must use the BOSH CLI to run this errand. Until you run this errand, developers cannot update service instances.

Changes to fields that can be overridden by optional parameters, for example `num_servers` or `new_size_percentage`, will change the default value of these instance properties, but will not affect existing service instances.

If you change the allowed limits of an optional parameter, for example the maximum number of servers per cluster, existing service instances in violation of the new limits will not be modified.

When existing instances are upgraded, all plan changes will be applied to them.

Uninstalling Pivotal Cloud Cache

To uninstall PCC, follow the steps from below from the **Installation Dashboard**:

1. Click the trash can icon in the bottom-right-hand corner of the tile.

2. Click **Apply Changes**.

Troubleshooting

View Statistics Files

You can visualize the performance of your cluster by downloading the statistics files from your servers. These files are located in the persistent store on each VM. To copy these files to your workstation, run the following command:

```
bosh scp server/0:/var/vcap/store/gemfire-server/statistics.gfs /tmp
```

See the Pivotal GemFire Installing and Running VSD topic for information about loading the statistics files into Pivotal GemFire VSD.

Smoke Test Failures

Error: “Creating p-cloudcache SERVICE-NAME failed”

The smoke tests could not create an instance of GemFire. To troubleshoot why the deployment failed, use the cf CLI to create a new service instance using the same plan and download the logs of the service deployment from BOSH.

Error: “Deleting SERVICE-NAME failed”

The smoke test attempted to clean up a service instance it created and failed to delete the service using the `cf delete-service` command. To trobleshoot this issue, run `bosh logs` to view the logs on the broker or the service instance to see why the deletion may have failed.

Error: Cannot connect to the cluster SERVICE-NAME

The smoke test was unable to connect to the cluster.

To troubleshoot the issue, review the logs of your load balancer, and review the logs of your CF Router to ensure the route to your PCC cluster is properly registered.

You also can create a service instance and try to connect to it using the gfsh CLI. This requires creating a service key.

Error: “Could not perform create/put on Cloud Cache cluster”

The smoke test was unable to write data to the cluster. The user may not have permissions to create a region or write data.

Error: “Could not retrieve value from Cloud Cache cluster”

The smoke test was unable to read back the data it wrote. Data loss can happen if a cluster member improperly stops and starts again or if the member machine crashes and is resurrected by BOSH. Run `bosh logs` to view the logs on the broker to see if there were any interruptions to the cluster by a service update.

General Connectivity

Client-to-server Communication

PCC Clients communicate to PCC servers on port 40404 and with locators on port 55221. Both of these ports must be reachable from the Elastic Runtime network to service the network.

Membership Port Range

PCC servers and locators communicate with each other using UDP and TCP. The current port range for this communication is `49152-65535`.

If you have a firewall between VMs, ensure this port range is open.

BOSH Director and VMs on Different Networks

A deployment will fail if the VMs in the service network cannot reach the BOSH Director VM on another network. VMs on a different service network need to be able to talk to the BOSH Director.

If the VMs cannot communicate with the BOSH Director, the following error message appears:

```
Director task 1257
Started preparing deployment > Preparing deployment. Done (00:00:00)

Started preparing package compilation > Finding packages to compile. Done (00:00:00)

Started creating missing vms
Started creating missing vms > locator/d328ac30-fd48-424f-a28f-5087d94c07fd (0)
Started creating missing vms > locator/c41e4988-257c-47a4-bb97-1828dae15df4 (2)
Started creating missing vms > server/54f3690f-2366-405e-aca0-e0d630753e91 (0)
Started creating missing vms > server/cbbb4739-aae0-472f-9f7f-713d6ac15d07 (3)
Started creating missing vms > locator/36758e47-6f89-4003-968d-55620ca28e8a (1)
Started creating missing vms > server/abc4f156-1403-4187-a1f7-35f1ff4d961c (1)
Started creating missing vms > server/5a9c74e8-881e-4f49-99e3-0a708b2b583c (2). Failed: Timed out pinging to 37ebdd5e-4fe8-49cd-8a0f-cc72837a361c after 600 seconds (00:10:35)
Failed creating missing vms > server/cbbb4739-aae0-472f-9f7f-713d6ac15d07 (3): Timed out pinging to db1d846b-a2cd-4c49-942e-3cf1b48edac1 after 600 seconds (00:10:37)
Failed creating missing vms > locator/d328ac30-fd48-424f-a28f-5087d94c07fd (0): Timed out pinging to 578a74a3-5511-4263-adbd-5028c6b7d1ab after 600 seconds (00:10:38)
Failed creating missing vms > server/54f3690f-2366-405e-aca0-e0d630753e91 (0): Timed out pinging to 73aec92f-b563-4f50-b22a-283072455b6e after 600 seconds (00:10:39)
Failed creating missing vms > locator/36758e47-6f89-4003-968d-55620ca28e8a (1): Timed out pinging to 634738c2-2338-4d32-973a-63f91dcfd922 after 600 seconds (00:10:39)
Failed creating missing vms > locator/c41e4988-257c-47a4-bb97-1828dae15df4 (2): Timed out pinging to 35b63dff-a7ff-44e2-984d-481dd9d1337b after 600 seconds (00:10:41)
Failed creating missing vms > server/abc4f156-1403-4187-a1f7-35f1ff4d961c (1): Timed out pinging to e6a59079-974b-4c05-8ce3-d250b582a571 after 600 seconds (00:10:54)
Failed creating missing vms (00:10:54)

Error 450002: Timed out pinging to 37ebdd5e-4fe8-49cd-8a0f-cc72837a361c after 600 seconds

Task 1257 error
```

Pivotal Cloud Cache Developer Guide

This document describes how a Pivotal Cloud Foundry (PCF) app developer can choose a service plan, create and delete Pivotal Cloud Cache (PCC) service instances, and bind an app.

You must install the Cloud Foundry Command Line Interface (cf CLI) to run the commands in this topic.

Viewing All Plans Available for Pivotal Cloud Cache

Run `cf marketplace -s p-cloudcache` to view all plans available for PCC. The plan names displayed are configured by the operator on tile installation.

```
$ cf marketplace -s p-cloudcache

Getting service plan information for service p-cloudcache as admin...
OK

service plan   description      free or paid
extra-small    Caching Plan 1   free
small          Caching Plan 2   free
medium         Caching Plan 3   free
large          Caching Plan 4   free
extra-large    Caching Plan 5   free
```

Creating a Pivotal Cloud Cache Service Instance

Run `cf create-service p-cloudcache PLAN-NAME SERVICE-INSTANCE-NAME` to create a service instance. Replace `PLAN-NAME` with the name from the list of available plans. Replace `SERVICE-INSTANCE-NAME` with a name of your choice. You use this name to refer to your service instance with other commands. Service instance names can include alpha-numeric characters, hyphens, and underscores.

```
$ cf create-service p-cloudcache extra-large my-cloudcache
```

Service instances are created asynchronously. Run the `cf services` command to view the current status of the service creation, and of other service instances in the current org and space:

```
$ cf services
Getting services in org my-org / space my-space as user...
OK

name            service        plan    bound apps   last operation
my-cloudcache   p-cloudcache   small                create in progress
```

When completed, the status changes from `create in progress` to `create succeeded`.

Provide Optional Parameters

You can create a customized service instance by passing optional parameters to `cf create-service` using the `-c` flag. The `-c` flag accepts a valid JSON object containing service-specific configuration parameters, provided either in-line or in a file.

The PCC service broker supports the following parameters:

- `num_servers`: An integer that specifies the number of server instances in the cluster. The minimum value is `4`. The maximum and default values are configured by operator.
- `new_size_percentage`: An integer that specifies the percentage of the heap to allocate to young generation. This value must be between `5` and `83`. By default, the new size is 2 GB or 10% of heap, whichever is smaller.

The following example creates the service with five service instances in the cluster:

```
$ cf create-service p-cloudcache small my-cloudcache -c '{"num_servers": 5}'
```

Enable Session State Caching with the Java Buildpack

When the service instance name is followed by the `session-replication` tag, the Java buildpack will download all the required resources for session state caching.

To enable session state caching, do one of the following:

- When creating your service instance name, append it with the `session-replication` tag. For example, for the `p-cloudcache` service:

  ```
  $ cf create-service p-cloudcache my-service-instance -t session-replication
  ```

- When updating your service instance name (for example, if the updated name is `new-service-instance`), append it with the `session-replication` tag:

  ```
  $ cf update-service new-service-instance -t session-replication
  ```

- End the service instance name with the text `-session-replication` : `my-service-instance-session-replication` .

Updating a Pivotal Cloud Cache Service Instance

You can apply all optional parameters to an existing service instance using the `cf update-service` command. You can, for example, scale up a cluster by increasing the number of servers.

Previously specified optional parameters are persisted through subsequent updates. To return the service instance to default values, you must explicitly specify the defaults as optional parameters.

For example, if you create a service instance with five servers using a plan that has a default value of four servers:

```
$ cf create-service p-cloudcache small my-cloudcache -c '{"num_servers": 5}'
```

And you set the `new_size_percentage` to 50%:

```
$ cf update-service my-cloudcache -c '{"new_size_percentage": 50}'
```

Then the resulting service instance has `5` servers and `new_size_percentage` of 50% of heap.

Cluster Rebalancing

When updating a cluster to increase the number of servers, the available heap size is increased. When this happens, PCC automatically rebalances data in the cache to distribute data across the cluster.

This automatic rebalancing does not occur when a server leaves the cluster and later rejoins, for example when a VM is re-created, or network connectivity lost and restored. In this case, you must manually rebalance the cluster using the gfsh `rebalance` command while authenticated as a cluster operator.

About Changes to the Service Plan

Your PCF operator can change details of the service plan available on the Marketplace. If your operator changes the default value of one of the optional parameters, this does not affect existing service instances.

However, if your operator changes the allowed values of one of the optional parameters, existing instances that exceed the new limits are not affected, but any subsequent service updates that change the optional parameter must adhere to the new limits.

For example, if the PCF operator changes the plan by decreasing the maximum value for `num_servers` , any future service updates must adhere to the new `num_servers` value limit.

You may see the following error message when attempting to update a service instance:

```
$ cf update-service  my-cloudcache -c '{"num_servers": 5}'
Updating service instance my-cloudcache as admin...
FAILED
Server error, status code: 502, error code: 10001, message: Service broker error: Service cannot be updated at this time, please try again later or contact your operator for more information
```

This error message indicates that the operator has made an update to the plan used by this service instance. You must wait for the operator to apply plan changes to all service instances before you can make further service instance updates.

Accessing a Service Instance

Once you have created a service instance, you can start accessing it. Usually you should set up cache regions before using your service instance from a deployed CF app. You can do this with the gfsh command line tool. To connect, you must set up a service key.

Create Service Keys

Service keys provide a way to access your service instance outside the scope of a deployed CF app. Run `cf create-service-key SERVICE-INSTANCE-NAME KEY-NAME` to create a service key. Replace `SERVICE-INSTANCE-NAME` with the name you chose for your service instance. Replace `KEY-NAME` with a name of your choice. You use this name to refer to your service key with other commands.

```
$ cf create-service-key my-cloudcache my-service-key
```

Run `cf create-service-key SERVICE-INSTANCE-NAME KEY-NAME` to view the newly created service key.

```
$ cf service-key my-cloudcache my-service-key
```

The `cf service-key` returns output in the following format:

```
{
  "locators": [
    "10.244.0.66[55221]",
    "10.244.0.4[55221]",
    "10.244.0.3[55221]"
  ],
  "urls": {
    "gfsh": "",
    "pulse": ""
  },
  "users": [
    {
      "password": "",
      "username": "developer"
    },
    {
      "password": "",
      "username": "cluster_operator"
    }
  ]
}
```

This returned structure contains the following information:

- `cluster_operator` and `developer` credentials
- `gfsh-url` : A URL usable to connect the gfsh client to the service instance
- `pulse-url` : A URL usable to view the Pulse dashboard in a web browser, which allows monitoring of the service-instance status. You can use the developer credentials to authenticate.

The `cluster_operator` and `developer` roles provide access to different functions on the service instance.

You can use the `cluster_operator` credentials to administer the service instance. For example, you can use the `cluster_operator` credentials to define or delete cache regions.

You can use the `developer` credentials to perform create, retrieve, update, and delete (CRUD) data operations. You can also use the `cluster_operator` credentials to perform the same functions the `developer` credentials. As a best practice, you should use the role with the least amount of permissions necessary.

Connect with gfsh

You can manage your cluster and data with the `gfsh` command-line interface. PCC works with `gfsh`.

Check the product snapshot of the PCC version you're using to determine which version of `gfsh` you need in order to connect to your cluster. For example, if your tile package uses GemFire v9.0.4, you should be using a 9.0.4 version of the `gfsh` client.

To install gfsh:

1. Download the Pivotal GemFire ZIP archive from Pivotal Network
2. Unzip and locate the correct binary for your architecture (`gfsh` for unix or `gfsh.bats` for windows)
3. Add the `gfsh` binary to your path

Start the `gfsh` command-line tool:

```
$ gfsh
    _________________________     __
   / _____/ ______/ ______/ /____/ /
  / /  __/ /___  /_____  / _____  /
 / /__/ / ____/  _____/ / /    / /
/______/_/      /______/_/    /_/    9.0.1

Monitor and Manage Pivotal GemFire
gfsh>
```

From the `gfsh>` prompt, connect to your service instance as `cluster_operator`:

```
gfsh>connect --use-http --url= --user=cluster_operator --password=
```

Workaround Connect with gfsh over HTTPS

Connecting over HTTPS is a bit more work than HTTP, but it is required for some network setups. The additional steps are:

1. Creating a truststore.
2. Setting an environment variable before you start gfsh.

Before you start this process, you'll need to have your ERT environment certificate handy.

Create a truststore that contains the ERT certificate

We suggest using the `keytool` command line utility to create a truststore file. You'll pass in your ERT certificate file with the `-file` flag, and the path for the output truststore file with the `-keystore` flag.

When you run this command, you will be prompted to enter a keystore password. Create a password and remember it! You will then be prompted with the certificate details. Type **yes** to trust the certificate.

Here is an example of how to run `keytool` and what the output looks like:

```
$ keytool -import -alias <env>-ssl -file <path-to-ERT-cert> -keystore <env>.truststore
Enter keystore password:
Re-enter new password:
Owner: CN=*.url.example.com, OU=Cloud Foundry, O=Pivotal, L=New York, ST=New York, C=US
Issuer: CN=*.url.example.com, OU=Cloud Foundry, O=Pivotal, L=New York, ST=New York, C=US
Serial number: bd84912917b5b665
Valid from: Sat Jul 29 09:18:43 EDT 2017 until: Mon Apr 07 09:18:43 EDT 2031
Certificate fingerprints:
     MD5:  A9:17:B1:C9:6C:0A:F7:A3:56:51:6D:67:F8:3E:94:35
     SHA1: BA:DA:23:09:17:C0:DF:37:D9:6F:47:05:05:00:44:6B:24:A1:3D:77
     SHA256: A6:F3:4E:B8:FF:8F:72:92:0A:6D:55:6E:59:54:83:30:76:49:80:92:52:3D:91:4D:61:1C:A1:29:D3:BD:56:57
     Signature algorithm name: SHA256withRSA
     Version: 3

Extensions:

#1: ObjectId: 2.5.29.10 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:0
]

#2: ObjectId: 2.5.29.11 Criticality=false
SubjectAlternativeName [
  DNSName: *.sys.url.example.com
  DNSName: *.apps.url.example.com
  DNSName: *.uaa.sys.url.example.com
  DNSName: *.login.sys.url.example.com
  DNSName: *.url.example.com
  DNSName: *.ws.url.example.com
]

Trust this certificate? [no]:  yes
Certificate was added to keystore
```

Setting the Environment Variable

Now that you've created a truststore file, you'll need to tell gfsh to use it. The way to do that is to set the `JAVA_ARGS` environment variable, before starting gfsh.

```
$ export JAVA_ARGS="-Djavax.net.ssl.trustStore=<path to generated truststore>"
```

Connect over HTTPS

This part is nearly the same as over HTTP; just make sure to use a url starting with `https`.

Open the gfsh console and connect via https:

```
$ gfsh
    _________________________     __
   / _____/ ______/ ______/ /____/ /
  / /  __/ /___  /_____  / _____  /
 / /__/ / ____/  _____/ / /    / /
/______/_/      /______/_/    /_/    9.0.1

Monitor and Manage Pivotal GemFire

gfsh>connect --use-http --url=https-gfsh-url --user=cluster_operator --password=cluster_operator-password
```

Using Pivotal Cloud Cache

Create Regions with gfsh

After connecting with gfsh as a `cluster_operator`, you can define a new cache region.

The following command creates a partitioned region with a single redundant copy:

```
gfsh>create region --name=my-cache-region --type=PARTITION_HEAP_LRU --redundant-copies=1
     Member      | Status
---------------- | ------------------------------------------------------
cacheserver-z2-1 | Region "/my-cache-region" created on "cacheserver-z2-1"
cacheserver-z3-2 | Region "/my-cache-region" created on "cacheserver-z3-2"
cacheserver-z1-0 | Region "/my-cache-region" created on "cacheserver-z1-0"
cacheserver-z1-3 | Region "/my-cache-region" created on "cacheserver-z1-3"
```

The following region types are recommended for PCC:

- `PARTITION_HEAP_LRU`
- `REPLICATE_HEAP_LRU`

To achieve optimal performance, you should understand the type of region that is relevant to your situation. For more information, see the Pivotal GemFire Region Types documentation.

You can test the newly created region by writing and reading values with gfsh:

```
gfsh>put --region=/my-cache-region --key=test --value=thevalue
Result      : true
Key Class   : java.lang.String
Key         : test
Value Class : java.lang.String
Old Value   :


gfsh>get --region=/my-cache-region --key=test
Result      : true
Key Class   : java.lang.String
Key         : test
Value Class : java.lang.String
Value       : thevalue
```

In practice, you should perform these get/put operations from a deployed PCF app. To do this, you must bind the service instance to these apps.

Bind an App to a Service Instance

Binding your apps to a service instance enables the apps to connect to the service instance and read or write data to the region. Run `cf bind-service APP-NAME SERVICE-INSTANCE-NAME` to bind an app to your service instance. Replace `APP-NAME` with the name of the app. Replace `SERVICE-INSTANCE-NAME` with the name you chose for your service instance.

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

Binding an app to the service instance provides connection information through the `VCAP_SERVICES` environment variable. Your app can use this information to configure components, such as the GemFire client cache, to use the service instance.

The following is a sample `VCAP_SERVICES` environment variable:

```
{
  "p-cloudcache": [
    {
      "credentials": {
    "locators": [
      "10.244.0.4[55221]",
      "10.244.1.2[55221]",
      "10.244.0.130[55221]"
    ],
    "urls": {
      "gfsh": "http://cloudcache-5574c540-1464-4f9e-b56b-74d8387cb50d.local.pcfdev.io/gemfire/v1",
      "pulse": "http://cloudcache-5574c540-1464-4f9e-b56b-74d8387cb50d.local.pcfdev.io/pulse"
    },
    "users": [
      {
        "password": "some_developer_password",
        "username": "developer"
      },
      {
        "password": "some_password",
        "username": "cluster_operator"
      }
    ]
      },
      "label": "p-cloudcache",
      "name": "test-service",
      "plan": "caching-small",
      "provider": null,
      "syslog_drain_url": null,
      "tags": [],
      "volume_mounts": []
    }
  ]
}
```

A Sample Java Client App for Pivotal Cloud Cache

A sample app in pure Java has been written to demonstrate how to connect an app to the service instance.

The code is available at https://github.com/cf-gemfire-org/cloudcache-sample-app.git

The code assumes that you have already created a region named `test` in your cluster using `gfsh` as described in the section above.

Apache Geode 1.0.0 is currently the only version that works to connect to our version of GemFire, which is listed in the `build.gradle`.

Follow these steps to deploy the sample app:

1. Modify the sample manifest, replacing `service0` with the name of your service.

```
---
applications:
- name: sample_app
path: build/libs/cloudcache-sample-app-1.0-SNAPSHOT-all.jar
no-route: true
health-check-type: none
services:
- service0
```

2. To deploy the app, run the `cf push` command:

```
$ cf push -f /path/to/manifest
```

After the app is pushed and started, there should be an entry in the `test` region where `key=1` and `value=one`.

Use the Pulse Dashboard

You can access the Pulse dashboard for a service instance by accessing the pulse-url you obtained from a service key in a web browser.

Use either the `cluster_operator` or `developer` credentials to authenticate.

Export gfsh Logs

You can get logs and `.gfs` stats files from your PCC service instances using the `export logs` command in gfsh.

1. Use the Connect with gfsh procedure to connect to the service instance for which you want to see logs.
2. Run `export logs`.
3. Find the ZIP file in the directory where you started gfsh. This file contains a folder for each member of the cluster. The member folder contains the associated log files and stats files for that member.

For more information about the gfsh export command, see the gfsh export documentation.

Deleting a Service Instance

You can delete service instances using the cf CLI. Before doing so, you must remove any existing service keys and app bindings.

1. Run `cf delete-service-key SERVICE-INSTANCE-NAME KEY-NAME` to delete the service key.
2. Run `cf unbind-service APP-NAME SERVICE-INSTANCE-NAME` to unbind your app from the service instance.
3. Run `cf delete-service SERVICE-INSTANCE-NAME` to delete the service instance.

```
$ cf delete-service-key my-cloudcache my-service-key
$ cf unbind-service my-app my-cloudcache
$ cf delete-service my-cloudcache
```

Deletions are asynchronous. Run `cf services` to view the current status of the service instance deletion.

Pivotal Cloud Cache Release Notes

v1.0.8

Release Date: November 15, 2017

Features:

- Stemcell upgraded to 3445
- GemFire version upgraded to 9.1.1

v1.0.7

Release Date: August 10, 2017

Features:

- Improved password requirement of the testing utility
- Increased monit for smoke tests to be successful
- Improved kill command to force kill in the event of gfsh stop failure
- Fixed upgrading a Pivotal Cloud Cache (PCC) service failure when syslog tls is disabled
- Send service instance logs to external syslog is enabled

v1.0.6

Features:

- Fixed service instance upgrade failure when pulse has open connections

v1.0.5

Release Date: June 29, 2017

Features:

- This version is based on the On-Demand Services SDK (ODB) v0.16. It fixes the issue with the*upgrade-all-service-instances* errand returning success when it has actually failed.

v1.0.4

Release Date: June 9, 2017

Features:

- Rewrote smoke test using Go to fix timeout issues on some environments

v1.0.3

Features:

Release Date: May 22, 2017

- Upgraded to GemFire v9.0.4

- Upgraded to v0.15.3 of On-Demand Service Broker
- Fixed JVM parameters for garbage collection
- URL routes were updated to read “cloudcache” rather than “gemfire”

v1.0.2

Release Date: May 3, 2017

Features:

- Upgrades to v0.15.2 of On-Demand Service Broker
- Increased smoke test canary timeout to five minutes
- Sets global service instance quota to 50 instances
- Provides option to disable a plan and service access separately
- Validates that the plan description doesnt contain any spaces

v1.0.1

Release Date: April 13, 2017

Features:

- Upgrades to v0.15.1 of On-Demand Service Broker
- Smoke test now runs in the `system` organization
- Improved reliability of Smoke Test errand
- Adds support for gfsh `export logs` command
- Optimized size of tile to make downloading and uploading faster
- Ensured no data loss with when performing a stemcell upgrade for regions with redundancy
- Add post deployment checks that all locators properly join the cluster
- PCC broker can be configured to send logs to a syslog server
- PCC service instances will now send logs to configured syslog endpoint if enabled
- Added option to provide public IP addresses to VMs, which is necessary to allow egress on Google Cloud Platform
- Added errand when deleting tile that deletes all PCC service instances

v1.0.0

Release Date: March 31, 2017

Requirement: PCC v1.0.0 requires PCF with Elastic Runtime v1.9.11 or above.

Features:

- Support for application look-aside cache pattern
- Multi AZ support
- On-demand provisioning
- MVP role-based access control
- Quotas, both for service instances and for number of servers in a cluster
- GemFire v9.0.2
- Manage and configure clusters using the Cloud Foundry Command Line Interface (cf CLI) service commands and the GemFire gfsh CLI

Known issues:

- The `Upgrade all service instances` errand can only upgrade the first 50 instances created.

- On PCF for Google Cloud Platform deployments, Ops Manager service network VMs are not assigned the correct firewall rules. As a result, these VMs cannot communicate with the BOSH Director, and the PCC broker will fail to create service instances. As a workaround, modify your firewall to use subnet CIDR-based rules.
- The PCC broker will accept requests to modify instances that have ongoing BOSH operations, resulting in an asynchronous failure.