Configuration and Deployment

Dell EMC ECS with Pivotal Cloud Foundry

Configuration and Best Practices Guide

Abstract

This document is a reference guide for configuring Pivotal Cloud Foundry® with Dell EMC™ ECS™.

November 2019

H17569

Revisions

Date	Description
January 2019	Initial release
November 2019	Component version updates; NSX-T Load Balancer option, versioning and lifecycle configuration examples

Acknowledgments

This paper was produced by the Unstructured Technical Marketing Engineering and Solution Architects team.

Author: Rich Paulson

Table of contents

Executive summary

Cloud-native is a modern approach to application architecture, development and delivery that has emerged as a natural response to the changes in business needs and infrastructure capabilities. This new model directly increases the speed and agility of application delivery for IT organizations and has proven its benefits for startups and established enterprises alike.

Pivotal Cloud Foundry simplifies and enables organizations to deploy new code as soon as it is available, reliably run applications at cloud scale, and improve your security posture with built-in capabilities.

Dell EMC ECS provides a complete software-defined strongly-consistent, indexed, cloud storage platform that supports the storage, manipulation, and analysis of unstructured data on a massive scale. Client access protocols include an S3 compatible API (with additional Dell EMC extensions for retention, byte range append/update/overwrite, and indexed metadata).

Objectives

This document is meant to be a reference guide for configuring various components in the Pivotal Cloud Foundry ecosystem with Dell EMC ECS.

- Introduce the key components
- Describe how each component interfaces and connects with ECS
- Describe how to configure each component with ECS
- Include a solution verification test when applicable

Audience

This document is intended for Pivotal Cloud Foundry administrators interested in the configuration and best practices to use with Dell EMC ECS.

This guide assumes a high level of technical knowledge for the devices and technologies described.

Terminology

Buildpack - Buildpacks provide framework and runtime support for apps. Buildpacks typically examine your apps to determine what dependencies to download and how to configure the apps to communicate with bound services.

Droplet - An archive within the Pivotal Application Service that contains the application ready to run on Diego. A droplet is the result of the application staging process.

Versioning - Versioning is a feature of the S3 protocol that allows a bucket to retain older versions of object data.

Lifecycle - Lifecycle controls the lifetime of objects in a bucket

1 Solution overview

This section provides an overview of both Dell EMC ECS and Pivotal Cloud Foundry key technologies and solution architecture.

1.1 Pivotal Cloud Foundry

Pivotal Cloud Foundry is an application deployment platform that delivers software and software updates to a public or private cloud. It is a family of products that includes Pivotal Application Service (PAS), used to build and run applications; Pivotal Container Service (PKS), used to build and run containers; and Pivotal Function Service (PFS), used to build and run functions. These features are powered by BOSH (BOSH Outer shell), an open source tool for release engineering, deployment, lifecycle management, and monitoring of distributed systems such as PCF. BOSH helps leading companies deploy and scale PCF across clouds with only a handful of operators. PCF is the proven solution for companies seeking software-led, digital transformation.

- Use PCF's portfolio of modern runtimes to deliver features faster.
- Designed for zero-downtime deployments.
- Protect systems from attackers using Pivotal's 3 Rs of security: repair, repave, and rotate.
- Rely on built-in high availability to keep customer-facing systems online under even the most challenging circumstances.

1.2 Dell EMC ECS

ECS provides a complete software-defined strongly-consistent, indexed, cloud storage platform that supports the storage, manipulation, and analysis of geographically distributed unstructured data on a massive scale. Client access protocols include S3, with additional Dell EMC extensions to the S3 protocol. Object access for S3 is achieved via REST APIs. Objects are written, retrieved, updated and deleted via HTTP or HTTPS calls using REST verbs such as GET, POST, PUT, DELETE, and HEAD.

ECS was built as a completely distributed system following the principle of cloud applications. In this model, all hardware nodes provide the core storage services. Without dedicated index or metadata nodes the system has limitless capacity and scalability.

Service communication ports are integral when configuring ECS with Pivotal Cloud Foundry. See Table 1 below for a list of the S3 ports used with ECS. For more information on ECS ports refer to the ECS Security Configuration Guide.

Note that ECS requires an external IP traffic load balancer, the default ports shown in Table 1 may be mapped to ports 80 and 443 for client/application-facing connections.

Table 1 ECS S3 Transport Protocol and Ports

Protocol	Transport Protocol	Port
S3	HTTP	9020
	HTTPS	9021

For a more thorough ECS overview, please review ECS Overview and Architecture whitepaper.

1.3 ECS Multi-Site Deployments

In a multisite deployment, more than one VDC is managed as a federation and/or geo-replicated. In a geo-replicated deployment, data can be read or written from any active site within the defined replication group.

Geo-replication provides enhanced protection against site failures by having multiple copies of the data, i.e., a primary copy of the data at the original site and a secondary copy of the data at a remote site/VDC.

Reference the ECS Overview and Architecture document for detailed information.

Managing application traffic both locally and globally can provide high availability (HA) and efficient use of ECS storage resources. HA is obtained by directing application traffic to known-to-be-available local or global storage resources. Optimal efficiency can be gained by balancing application load across local storage resources. An IP Load Balancer is required to direct traffic to an alternate site when the primary site is unavailable. This can be accomplished using Global Site Load Balancing.

The ECS General Best Practices document describes several approaches to configure a load balancer with failover capability for ECS.

1.4 Solution architecture

Figure 1 shows the logical architecture of the Pivotal Cloud Foundry components with Dell EMC ECS storage defined as the external blobstore, ECS Service Broker endpoint, image repository and managed services backup repository. The load balancer in this case is VMware NSX-T which directs traffic and monitors the health of the ECS nodes.

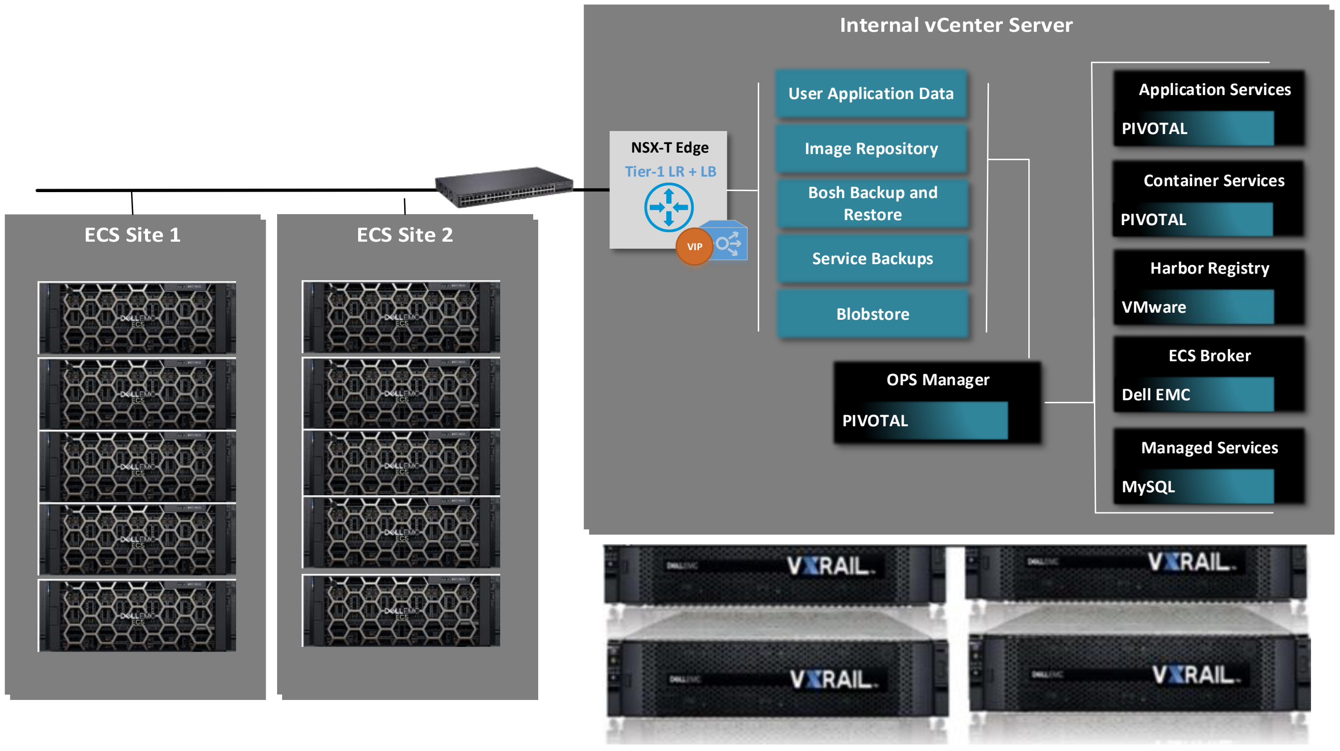


Figure 1 Logical Architectural Overview

1.5 Key components

The following components and versions were used for the implementation of this solution.

Table 2 Components

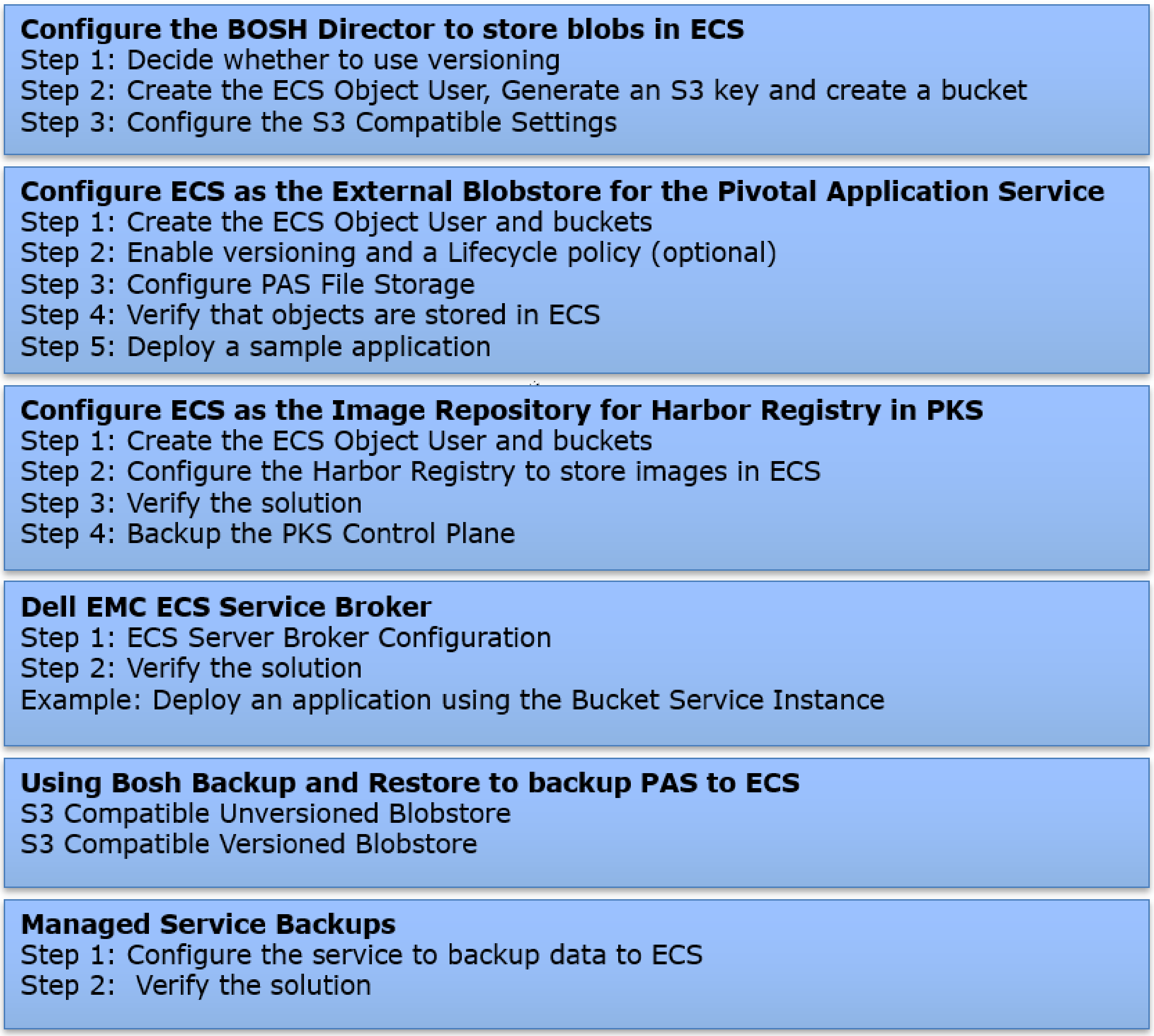
Product Information	Version
VMware ESXi	6.7
VMware vCenter	6.7
BOSH Director for vSphere	2.7.1
Pivotal Small Footprint PAS	2.7.2
Pivotal Enterprise PKS	1.5.1
VMware Harbor Registry	1.7.5
Dell EMC ECS Service Broker	1.2.3
MySQL for Pivotal Cloud Foundry v2	2.6.2
Dell EMC ECS EX300 Appliance	3.4

2 Solution implementation

This section describes the high-level steps required to configure Pivotal Cloud Foundry with Dell EMC ECS.

2.1 Implementation workflow

This below describes the steps take to configure each Pivotal component with Dell EMC ECS.



Configure the BOSH Director to store blobs in ECS
Step 1: Decide whether to use versioning
Step 2: Create the ECS Object User, Generate an S3 key and create a bucket
Step 3: Configure the S3 Compatible Settings

Configure ECS as the External Blobstore for the Pivotal Application Service
Step 1: Create the ECS Object User and buckets
Step 2: Enable versioning and a Lifecycle policy (optional)
Step 3: Configure PAS File Storage
Step 4: Verify that objects are stored in ECS
Step 5: Deploy a sample application

Configure ECS as the Image Repository for Harbor Registry in PKS
Step 1: Create the ECS Object User and buckets
Step 2: Configure the Harbor Registry to store images in ECS
Step 3: Verify the solution
Step 4: Backup the PKS Control Plane

Dell EMC ECS Service Broker
Step 1: ECS Server Broker Configuration
Step 2: Verify the solution
Example: Deploy an application using the Bucket Service Instance

Using Bosh Backup and Restore to backup PAS to ECS
S3 Compatible Unversioned Blobstore
S3 Compatible Versioned Blobstore

Managed Service Backups
Step 1: Configure the service to backup data to ECS
Step 2: Verify the solution

2.2 Installation and configuration steps

2.2.1 Configure the BOSH Director to store blobs in ECS

The BOSH Director is used to package, deploy and manage cloud software.

BOSH Director can be configured to store its data as an internal server or an external endpoint. Because the internal server is less scalable and secure, Pivotal recommends that you configure an external blobstore.

Note: You cannot change the blobstore location once the changes are applied.

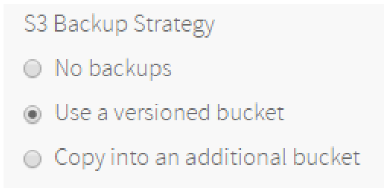
When configuring the BOSH Director, an option exists to store its blobs to an external blobstore. This option exists in the **Director Config** setting under the **Blobstore location** heading. With later versions of BOSH Director, Blobs can be backed up using a versioned or non-versioned bucket.

Selecting the 'S3 Compatible Blobstore' radio button allows you to configure the ECS settings.

Note: It is recommended to create a separate ECS namespace for organizing and segregating the space for the Pivotal application.

S3 Endpoint: The URL of the ECS cluster. This is typically the load balancer in front of the ECS cluster.
Bucket Name: Enter the name of the ECS bucket to store the Blobs. Note that the bucket must be created prior to saving the configuration.
Access Key: The name of the ECS Object User
Secret Key: The object users S3 Key
Signature: ECS supports both V2 and V4 signatures
Region: If V4 signature is selected then a region must be specified. Regions don't apply to ECS so using 'Standard' is sufficient

In the S3 Backup Strategy section (BOSH Director version 2.6 and later), you are provided with several options to backup data.


S3 Backup Strategy
No backups [✓ Use a versioned bucket] Copy into an additional bucket

If you choose to use a versioned bucket, then versioning must be enabled on the ECS bucket (See Appendix A.2 for examples on enabling versioning).

If you choose to copy backups into an additional ECS bucket, then specify the **Backup Bucket Region** and the **Backup Bucket Name**. Note that the bucket must already exist.

Note: BOSH director blobs are uploaded to the object store using multi-part upload (MPU). The MPU part size and number of upload threads are not configurable.

2.2.3 Configure ECS as the External Blobstore for the Pivotal Application Service

Pivotal Application Service is a cloud application platform, providing a choice of clouds, developer frameworks, and application services.

Pivotal recommends using highly resilient and redundant external file stores for Pivotal Application Service (PAS) storage such as ECS.

Note: These installation instructions reflect version 2.7 of PAS. Earlier versions may require that an add-on be installed to support an S3-compatible Blobstore.

This section outlines the steps to configure ECS as the external Blobstore for Pivotal Cloud Foundry installations. The steps for this procedure are:

1. Create the ECS Object User and buckets
2. Enable versioning and a Lifecycle policy
3. Configure PAS File Storage
4. Verify that objects are stored in ECS
5. Deploy a sample application

Step 1: Create the ECS Object User and Buckets

Create an object user using the ECS Web Portal or Management API and generate an S3 key. Reference the ECS Administration Guide for details on creating users and buckets.

PAS File Storage requires that 4 separate buckets be created which are referred to as the live buckets:

Buildpacks Bucket Name - This bucket stores application buildpacks.
Droplets Bucket Name - This bucket stores application droplets.
Packages Bucket Name - This bucket stores application packages.
Resources Bucket Name - This bucket stores application resources.

Note: If versioning is not going to be used for backup and restore purposes, then 3 additional buckets will need to be created to store the backups for Buildpacks, Droplets and Packages. These bucket names must be unique.

Step 2: Enable Versioning and Lifecycle Policy (optional)

If versioning is going to be used for backups and restores, then versioning must be enabled on the 4 buckets which were created in section 5.1. This can be done using the ECS S3 Object API or a client tool such as S3 Browser.

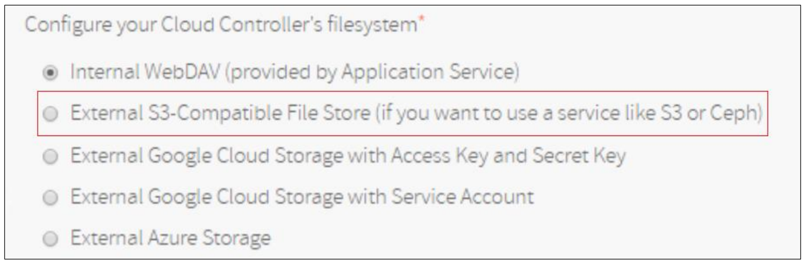
Note: It's strongly recommended to enable a lifecycle policy to limit the number of revisions in the bucket and reduce capacity. Please reference the ECS S3 Object Service API for information on enabling versioning and lifecycle policies.

Note: When using the ECS S3 Object API to enable a lifecycle policy, if versioning is enabled on the bucket then it must be suspended before the lifecycle policy is enabled. Re-enable versioning once the lifecycle is in place.

Step 3: Configure PAS File Storage

Login to OPS Manager, select the PAS tile and navigate to the 'File Storage' section.

Select 'External S3-Compatible File Store (if you want to use a service like S3 or Ceph)'



Configure your Cloud Controller's filesystem*

- ● Internal WebDAV (provided by Application Service)
- ○ External S3-Compatible File Store (if you want to use a service like S3 or Ceph)
- ○ External Google Cloud Storage with Access Key and Secret Key
- ○ External Google Cloud Storage with Service Account
- ○ External Azure Storage

Enter the following information

- **URL Endpoint -** The URL of the ECS cluster. This is typically the load balancer in front of the ECS cluster.
- **Access Key** – The name of the ECS Object User
- **Secret Key –** The object users S3 Key
- **S3 Signature Version** – ECS supports both V2 and V4. Note: if the option to use versioning is enabled then V4 must be selected.
- **Region** – Regions don't apply to ECS, but this field is required so using 'Standard' is sufficient.
- **Buildpacks Bucket Name** – Enter the name of the buildpacks bucket which was created in Step 1
- **Droplets Bucket Name** – Enter the name of the droplets bucket which was created in Step 1
- **Packages Bucket Name** – Enter the name of the packages bucket which was created in Step 1
- **Resources Bucket Name** – Enter the name of the resources bucket which was created in Step 1
- **Use versioning for backup and restore:** File Storage can be configured with versioning enabled or disabled. The differences between the two are:
    - **Versioned S3 buckets**: Enable the Use versioning for backup and restore checkbox to archive each bucket to a version.
    - **Unversioned S3 buckets**: Disable the Use versioning for backup and restore checkbox and enter a backup bucket name for each live bucket. Backups will be stored in the backup bucket. The backup bucket name must be different from the name of the active bucket it backs up.

Note: Prior to ECS Release 3.3.x, bucket listing operations were susceptible to load and can be impacted by concurrent bucket listing and other workloads running on the same system. Therefore, backing up S3 versioned blobstores may experience performance issues depending on system load. If ECS is being used for other workloads that place a high additional amount of system load, then an Unversioned blobstore configuration should be used due to the frequent small file changes that PCF creates.

If versioning is enabled, then click the 'Save' button otherwise enter the information for the backup buckets

- **Backup Buildpacks Bucket Name –** Enter the name of the backup buildpacks bucket
- **Backup Droplets Bucket Name –** Enter the name of the backup droplets bucket
- **Backup Packages Bucket Name –** Enter the name of the packages bucket name

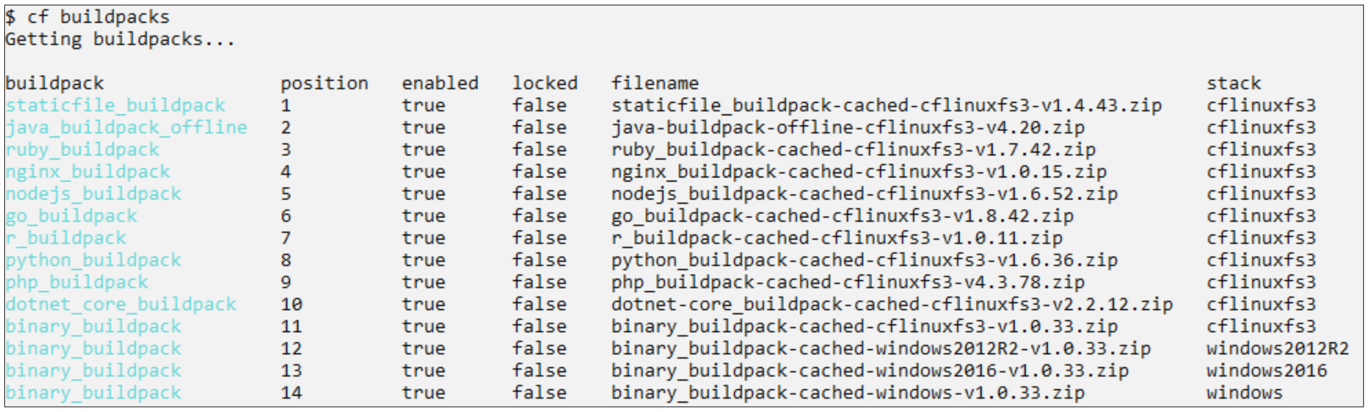
Click Save and continue configuring PAS. Once complete, apply the changes.

Note: PAS will verify connectivity to ECS once you click save. If there are any problems, then the save will fail, and PAS will display the issue(s) on screen to troubleshoot further.

Step 4: Post-installation Verification

Upon a successful installation, several objects will be created under each of the 4 ECS buckets. For example, using the 'cf buildpacks' CLI command we see that the following buildpacks are installed:

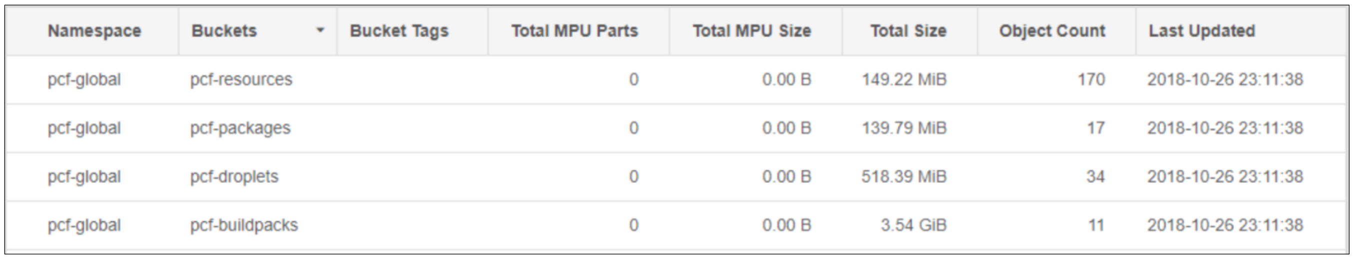
Figure 1. Initial buildpacks



```
$ cf buildpacks
Getting buildpacks...

buildpack                 position   enabled   locked   filename                                                stack
staticfile_buildpack      1          true      false    staticfile_buildpack-cached-cflinuxfs3-v1.4.43.zip      cflinuxfs3
java_buildpack_offline    2          true      false    java-buildpack-offline-cflinuxfs3-v4.20.zip             cflinuxfs3
ruby_buildpack            3          true      false    ruby_buildpack-cached-cflinuxfs3-v1.7.42.zip            cflinuxfs3
nginx_buildpack           4          true      false    nginx_buildpack-cached-cflinuxfs3-v1.0.15.zip           cflinuxfs3
nodejs_buildpack          5          true      false    nodejs_buildpack-cached-cflinuxfs3-v1.6.52.zip          cflinuxfs3
go_buildpack              6          true      false    go_buildpack-cached-cflinuxfs3-v1.8.42.zip              cflinuxfs3
r_buildpack               7          true      false    r_buildpack-cached-cflinuxfs3-v1.0.11.zip               cflinuxfs3
python_buildpack          8          true      false    python_buildpack-cached-cflinuxfs3-v1.6.36.zip          cflinuxfs3
php_buildpack             9          true      false    php_buildpack-cached-cflinuxfs3-v4.3.78.zip             cflinuxfs3
dotnet_core_buildpack     10         true      false    dotnet-core_buildpack-cached-cflinuxfs3-v2.2.12.zip     cflinuxfs3
binary_buildpack          11         true      false    binary_buildpack-cached-cflinuxfs3-v1.0.33.zip          cflinuxfs3
binary_buildpack          12         true      false    binary_buildpack-cached-windows2012R2-v1.0.33.zip       windows2012R2
binary_buildpack          13         true      false    binary_buildpack-cached-windows2016-v1.0.33.zip         windows2016
binary_buildpack          14         true      false    binary_buildpack-cached-windows-v1.0.33.zip             windows
```

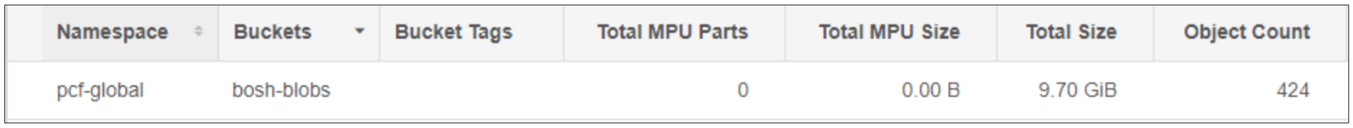
Our installation contained the following number of objects in each bucket:

Figure 2. Initial PAS blobstore objects



Namespace	Buckets	Bucket Tags	Total MPU Parts	Total MPU Size	Total Size	Object Count	Last Updated
pcf-global	pcf-resources		0	0.00 B	149.22 MiB	170	2018-10-26 23:11:38
pcf-global	pcf-packages		0	0.00 B	139.79 MiB	17	2018-10-26 23:11:38
pcf-global	pcf-droplets		0	0.00 B	518.39 MiB	34	2018-10-26 23:11:38
pcf-global	pcf-buildpacks		0	0.00 B	3.54 GiB	11	2018-10-26 23:11:38

Figure 3. BOSH Blobstore objects



Namespace	Buckets	Bucket Tags	Total MPU Parts	Total MPU Size	Total Size	Object Count
pcf-global	bosh-blobs		0	0.00 B	9.70 GiB	424

Step 5: Sample Application Deployment

Deploy a sample app to validate the configuration. The application we'll be deploying can be downloaded from Pivotal using the following URL: https://github.com/cloudfoundry-samples/hello-spring-cloud

This is a sample application built with Spring Boot which uses Spring Cloud Connectors to connect to cloud services and get information about the cloud environment.

After downloading the app on a system which has the cloud foundry CLI installed, login to the API endpoint, choose an organization and space then navigate to the app directory and issue the 'push' command: cf push

Once the application starts we can open the URL in a browser.

```
name               requested state   instances   memory   disk   urls
hello-spring-cloud     started          1/1        1G      1G    hello-spring-cloud.apps.example.local
```

2.2.4 Configure ECS as the Image Repository for Harbor Registry in PKS

Harbor is a docker image registry from VMware which can be used with the Pivotal Container Service (PKS). The container registry can be configured to use ECS as an image store.

This section outlines the steps to configure ECS as the image repository for Harbor Registry. The steps for this procedure are:

1. Create the ECS Object User and buckets
2. Configure the Harbor Registry to store images in ECS
3. Verify that images are stored in ECS
4. Backup the PKS Control Plane

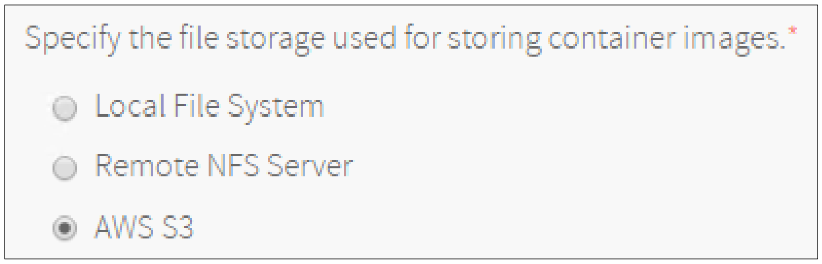
Step 1: Create the ECS Object User and Bucket

Create an object user using the ECS Web Portal or Management API and generate an S3 key. Reference the ECS Administration Guide for details on creating users and buckets.

A separate bucket is required to store the container registry. Create the object user and bucket to be used to configuring the Container Registry storage for the VMware Harbor Registry,

Step 2: Configure Harbor to store images in ECS

Select the AWS S3 option to configure ECS.



Specify the file storage used for storing container images.*

( ) Local File System

( ) Remote NFS Server

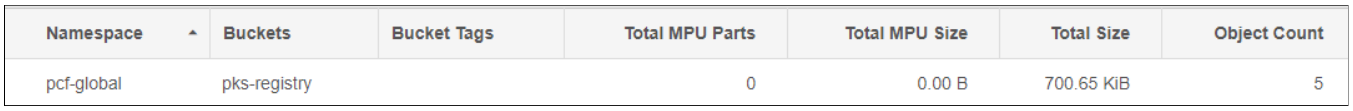
(●) AWS S3

Enter the following information

- **Access Key** – The name of the ECS Object User which was created in the above section
- **Secret Key –** The object users S3 Key which was generated in the above section
- **Region –** Regions don't apply to ECS, but this field is required so using 'Standard' is sufficient
- **Endpoint URL of your S3-compatible file store -** The URL of the ECS cluster. This is typically the load balancer in front of the ECS cluster
- **Bucket Name –** Enter the name of the bucket which was created in Section 7.1
- **Root Directory in the Bucket –** (optional). A prefix/path can be created under the bucket to store container registry
- **Chunk Size –** The default size is 5MB. Dell EMC recommends increasing the chunk size to 128MB or larger for increased performance and to reduce overhead.
- **Enable v4auth -** Access to your S3 bucket is authenticated by default. Deselect this checkbox for anonymous access. ECS supports both V2 and V4.
- **Secure Mode -** Access to your S3 bucket is secure by default. Deselect this checkbox to disable secure mode.

Click Save and continue configuring the VMWare Harbor Registry tile. Once complete, apply the changes.

Upon a successful installation, several objects will be created under the ECS bucket.



Namespace	Buckets	Bucket Tags	Total MPU Parts	Total MPU Size	Total Size	Object Count
pcf-global	pks-registry		0	0.00 B	700.65 KiB	5

Step 3: Solution Verification

The verification we'll use to verify that images are being stored in ECS is to authenticate using the Harbor registry, pull an image and then push it to Harbor.

Login to docker using the Harbor endpoint

```
$ sudo docker login harbor.richp.local
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /home/richp/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Pull an image

```
$ sudo docker pull busybox
Using default tag: latest
latest: Pulling from library/busybox
7c9d20b9b6cd: Pull complete
Digest: sha256:fe301db49df08c384001ed752dff6d52b4305a73a7f608f21528048e8a08b51e
Status: Downloaded newer image for busybox:latest
$
$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
busybox             latest              19485c79a9bb        11 days ago         1.22MB
```

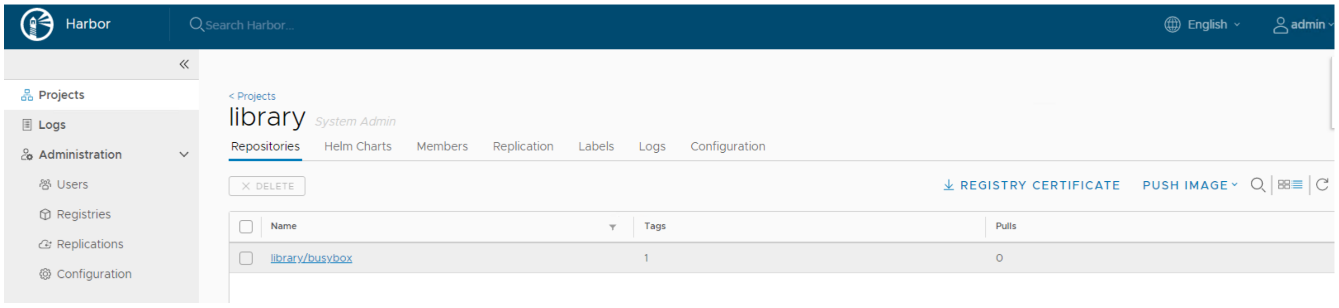
Tag and Push the image to the Harbor Registry to the 'Library' project

```
$ sudo docker tag busybox:latest harbor.richp.local/library/busybox:harbor
$
$ sudo docker images
REPOSITORY                           TAG                 IMAGE ID            CREATED             SIZE
busybox                              latest              19485c79a9bb        11 days ago         1.22MB
harbor.richp.local/library/busybox   harbor              19485c79a9bb        11 days ago         1.22MB
```

Push the image to the Harbor Registry

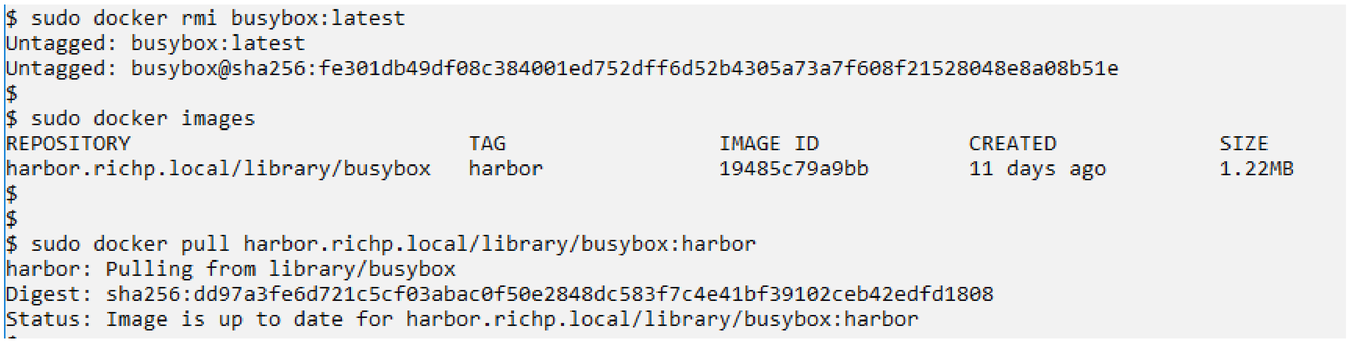
```
$ sudo docker push harbor.richp.local/library/busybox:harbor
The push refers to repository [harbor.richp.local/library/busybox]
6c0ea40aef9d: Pushed
harbor: digest: sha256:dd97a3fe6d721c5cf03abac0f50e2848dc583f7c4e41bf39102ceb42edfd1808 size: 527
```

The image is displayed in the Harbor UI



	Name	Tags	Pulls
☐	library/busybox	1	0

Delete the local (latest) image and pull it from the Harbor Registry



```
$ sudo docker rmi busybox:latest
Untagged: busybox:latest
Untagged: busybox@sha256:fe301db49df08c384001ed752dff6d52b4305a73a7f608f21528048e8a08b51e
$
$ sudo docker images
REPOSITORY                          TAG                 IMAGE ID            CREATED             SIZE
harbor.richp.local/library/busybox   harbor              19485c79a9bb        11 days ago         1.22MB
$
$
$ sudo docker pull harbor.richp.local/library/busybox:harbor
harbor: Pulling from library/busybox
Digest: sha256:dd97a3fe6d721c5cf03abac0f50e2848dc583f7c4e41bf39102ceb42edfd1808
Status: Image is up to date for harbor.richp.local/library/busybox:harbor
```

Step 4: Backing up the PKS Control Plane

The PKS control plane can be backed up and restored using BOSH Backup and Restore (BBR). The artifacts which are backed up are stored on the BBR jumpbox in a subdirectory named DEPLOYMENT-TIMESTAMP. The backups consist of a folder with the backup artifacts and metadata files.

Note: It is strongly recommended to back up the PKS artifacts to a separate highly durable storage location such as ECS.

2.2.5 Dell EMC ECS Service Broker

Dell EMC ECS Service Broker for PCF registers a service broker on Pivotal and exposes its service plans in the Marketplace. Developers can provision buckets by creating instances of service plans using the Pivotal Apps Manager or the Command Line Interface (CF).

This section outlines the steps to configure the ECS Broker Service. The steps for this procedure are:

1. ECS Server Broker Configuration (Administrator function)
2. Consume ECS Broker services (Developer)

Step 1: ECS Service Broker Installation and Configuration

The ECS Broker service is installed as a tile in the Pivotal Ops Manager and can be downloaded from the Pivotal Network at https://network.pivotal.io/products/ecs-service-broker/. Once the broker is downloaded it can be imported into the BOSH Director

Note: If a stemcell that the broker requires has not been imported yet then the broker tile will contain a link to the stemcell library. Import and associate the required stemcell with the broker before continuing.

Click the tile to configure the broker settings.

Navigate to **Assign AZs and Networks** and confirm that the default Availability Zones and Network fields are correct.

Navigate to the **Dell EMC ECS Connectivity** tab

- **The ECS Management Endpoint** is the URI/endpoint for the ECS HTTPS management API which typically runs on port 4443. Note that ECS requires an IP Load Balancer with ECS so the port may be mapped to 80 or 443 for client/application-facing connections.
- **The Replication Group** is the name of the ECS replication group to be used for created buckets.
- **The Namespace** is the name of the ECS namespace to be used for created buckets and users. It's recommended to use a separate namespace.
- **The ECS Admin Username** is the name of the ECS Management API user. In ECS terminology, this is a Management User (not an Object User), which should be configured as either a Namespace Administrator or System Administrator.
- **The ECS Admin Password** is the password for the ECS Management API username.
- **The ECS Base URL** is an optional field. This is the name of the ECS configured base URL that should be used with bucket credentials. If no value is entered, the broker will attempt to discover a default value. If this feature is enabled, ensure that the management user has System Monitor access on the ECS. Reference the ECS Administration Guide for more information on Object Base URLs.
- If the ECS system has a certificate installed which is not signed by a trust authority, then select the No radio button under '**Is the ECS SSL certificate is signed by a trust authority?**' and enter the certificate in the ECS Management Certificate field.
- **ECS Object Endpoint is the** URI/endpoint for the ECS S3 API. This is typically the IP load balancer in front of the ECS system accessed using port 9020 (HTTP) or port 9021 (HTTPS).

Note: If no Base URL was provided then the Object Endpoint must be configured. The URI/endpoints may be mapped to port 80 or 443 for client/application-facing connections.

Click save to apply the settings.

Navigate to the **Service Broker Settings** tab
These are optional settings which allow you to override the brokers default values.

- **Broker Prefix**, enter a string used to prefix broker-created buckets and users.
- **Repository Endpoint**, enter the URI/endpoint for the ECS S3 API that should be used for persisting broker metadata. This defaults to the object endpoint or base URL.
- **Repository User**, enter the name of the user that should be created or used for managing broker metadata. The default user name is ecs-cf-broker-user.
- **Repository Bucket**, enter the name of the bucket that should be created or used for managing broker metadata. The default bucket name is ecs-cf-broker-repository.

Navigate to the **Catalog Service Settings** tab
Catalog services and plans define the ECS services which will be available in the Pivotal Application Services marketplace. Note that Catalog Service 1 is enabled by default.

Note: The 'Enable as a repository Service' should only be enabled for one service, as the broker may pick from repository enabled services at random on startup

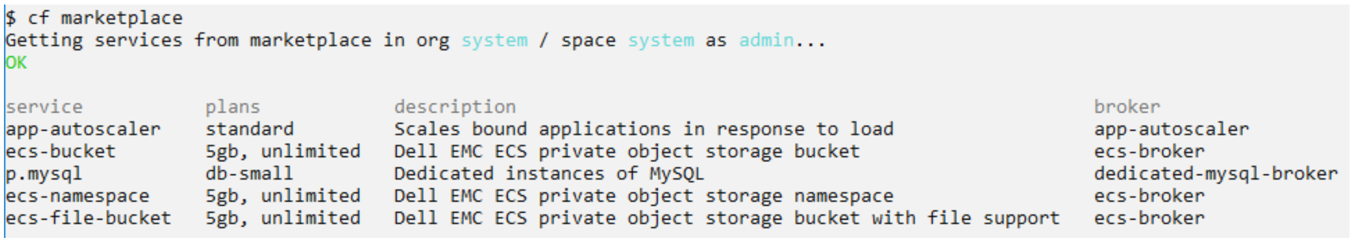
Step 2: Deploying Applications

The below examples illustrate an application deployment that a developer may use to consume the various service options. Catalog Services 1 and 3 will be used to deploy the sample application. The sample application we'll be using for these examples is the ECS Cloud Foundry S3 Java Demo App which can be downloaded from here. Note that the below examples assume the demo application has already been compiled.

After the tile is installed, the ECS services will show up in the marketplace and can be used to deploy applications.

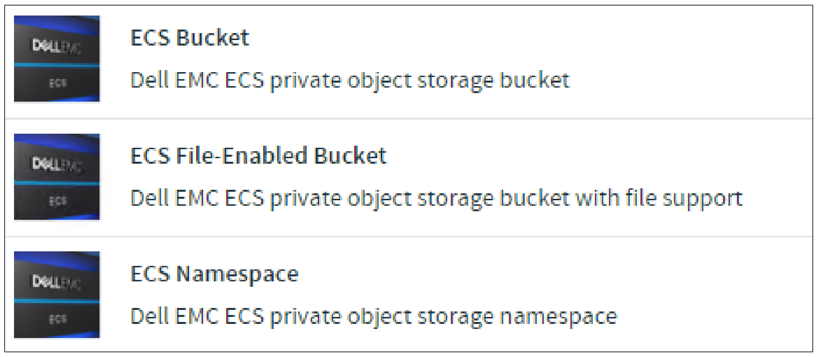
Note: Catalog services may need to be manually enabled in the desired organization and space using the 'cf enable-service-access' command.

ECS Services listing from the CLI



```
$ cf marketplace
Getting services from marketplace in org system / space system as admin...
OK

service           plans           description                                           broker
app-autoscaler    standard        Scales bound applications in response to load         app-autoscaler
ecs-bucket        5gb, unlimited  Dell EMC ECS private object storage bucket            ecs-broker
p.mysql           db-small        Dedicated instances of MySQL                          dedicated-mysql-broker
ecs-namespace     5gb, unlimited  Dell EMC ECS private object storage namespace         ecs-broker
ecs-file-bucket   5gb, unlimited  Dell EMC ECS private object storage bucket with file support   ecs-broker
```

Figure 4. ECS Services listing from Apps Manager



ECS Bucket
Dell EMC ECS private object storage bucket

ECS File-Enabled Bucket
Dell EMC ECS private object storage bucket with file support

ECS Namespace
Dell EMC ECS private object storage namespace

The steps we'll follow to deploy the demo application using the ECS Broker Service instances are:

1. Create the ECS service instance
2. Bind the application to the ECS service instance
3. Start the application
4. Launch and test the application

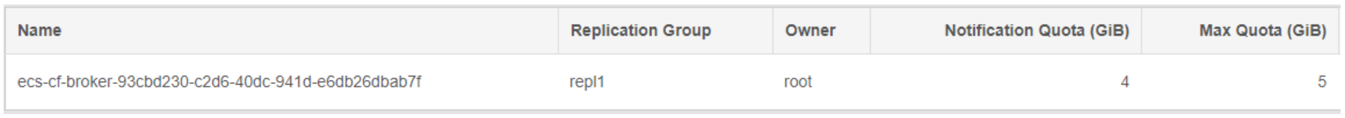
Example: Deploy an application using the Bucket Service Instance

The bucket service instance allows users to store data in an ECS bucket using S3.

Step 1: Create the ECS service instance using the 5gb service plan. This is the instance we'll bind the demo app to so that data is stored in ECS.

```
$ cf create-service ecs-bucket 5gb mybucket
Creating service instance mybucket in org system / space system as admin...
OK

$ cf services
Getting services in org system / space system as admin...

name       service      plan   bound apps   last operation     broker       upgrade available
mybucket   ecs-bucket   5gb                 create succeeded   ecs-broker
```

From the ECS Web Portal, we can see that our bucket was successfully created. The quota has been defined per the 5gb plan we used from the Marketplace.



Name	Replication Group	Owner	Notification Quota (GiB)	Max Quota (GiB)
ecs-cf-broker-93cbd230-c2d6-40dc-941d-e6db26dbab7f	repl1	root	4	5

Step 2: Bind the application to the ECS service instance

Push the demo application with the '—no-start' flag

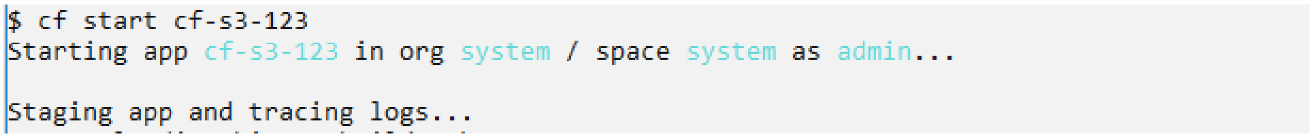
```
$ cf push cf-s3-123 --no-start -p build/libs/cf-ecs-demo.jar
Pushing app cf-s3-123 to org system / space system as admin...
Getting app info...
Creating app with these attributes...
+ name:       cf-s3-123
  path:       /home/richp/cf-apps/cf-ecs-demo/build/libs/cf-ecs-demo.jar
  routes:
+   cf-s3-123.apps.richp.local

Creating app cf-s3-123...
Mapping routes...
Comparing local files to remote cache...
Packaging files to upload...
Uploading files...
 666.99 KiB / 666.99 KiB [==================================================================================

Waiting for API to complete processing files...

name:              cf-s3-123
requested state:   stopped
routes:            cf-s3-123.apps.richp.local
last uploaded:
stack:
buildpacks:

type:           web
instances:      0/1
memory usage:   1024M
     state   since                  cpu    memory   disk     details
#0   down    2019-09-16T17:17:34Z   0.0%   0 of 0   0 of 0
```

Bind the application to our ECS bucket

```
$ cf bind-service cf-s3-123 mybucket
Binding service mybucket to app cf-s3-123 in org system / space system as admin...
OK
```

Step 3: Start the application

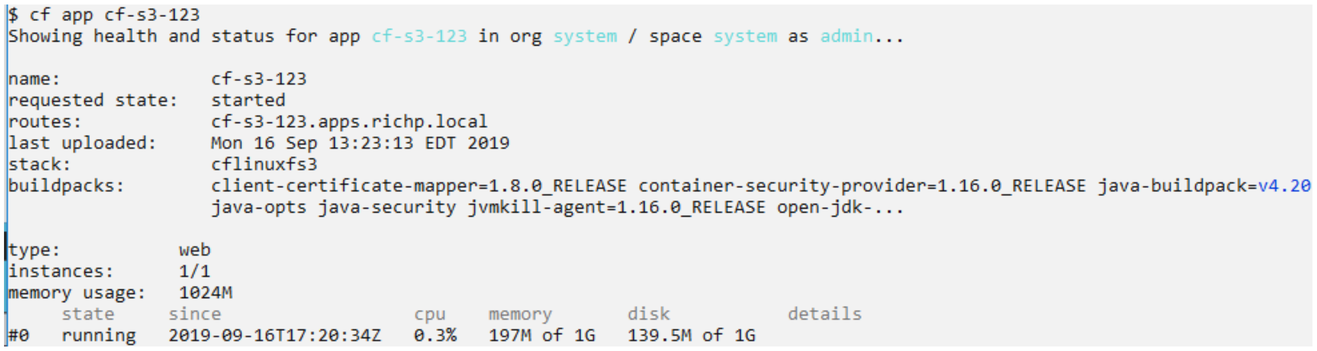
Start the application using the command 'cf start cf-s3-123'



```
$ cf start cf-s3-123
Starting app cf-s3-123 in org system / space system as admin...

Staging app and tracing logs...
```

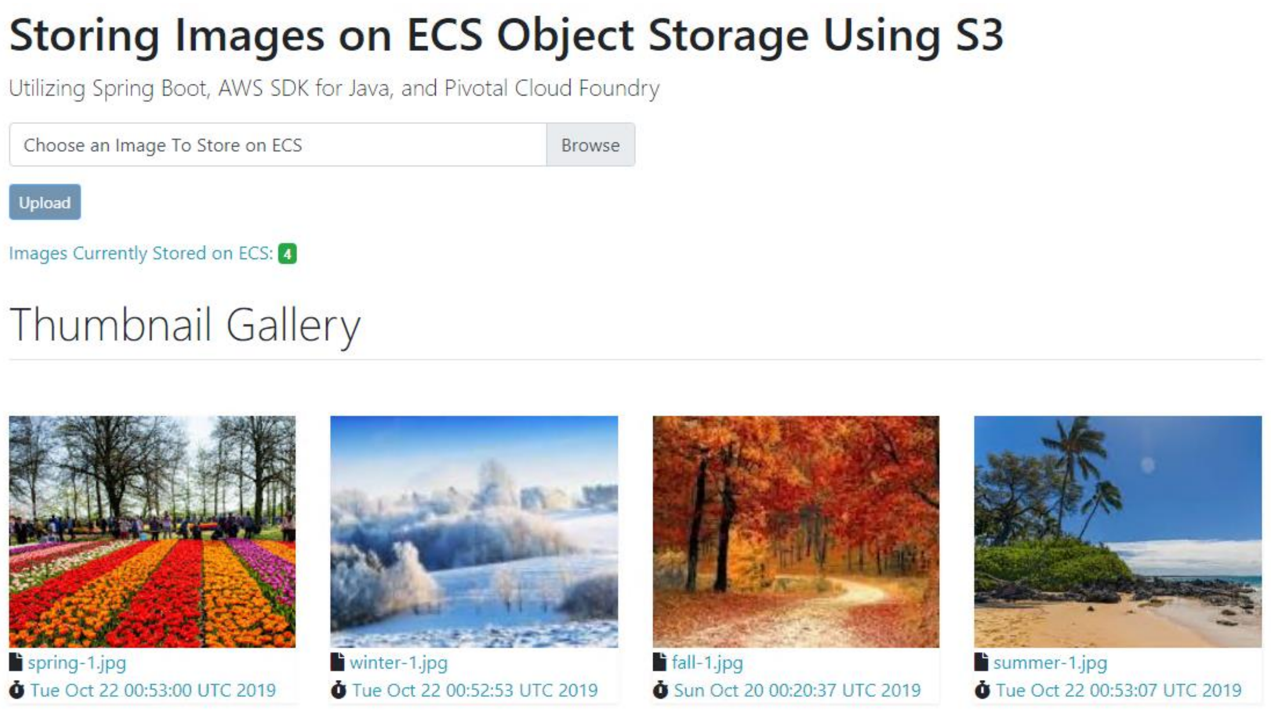
Step 4: Launch and test the application

Launch the application once it's started. To find the route/URL use the 'cf apps' command



```
$ cf app cf-s3-123
Showing health and status for app cf-s3-123 in org system / space system as admin...

name:              cf-s3-123
requested state:   started
routes:            cf-s3-123.apps.richp.local
last uploaded:     Mon 16 Sep 13:23:13 EDT 2019
stack:             cflinuxfs3
buildpacks:        client-certificate-mapper=1.8.0_RELEASE container-security-provider=1.16.0_RELEASE java-buildpack=v4.20
                   java-opts java-security jvmkill-agent=1.16.0_RELEASE open-jdk-...

type:           web
instances:      1/1
memory usage:   1024M
     state     since                  cpu    memory      disk          details
#0   running   2019-09-16T17:20:34Z   0.3%   197M of 1G   139.5M of 1G
```

The demo application allows a user to upload images which will be stored in the ECS bucket.



Storing Images on ECS Object Storage Using S3

Utilizing Spring Boot, AWS SDK for Java, and Pivotal Cloud Foundry

Choose an Image To Store on ECS | Browse

Upload

Images Currently Stored on ECS: 4

Thumbnail Gallery

spring-1.jpg
Tue Oct 22 00:53:00 UTC 2019

winter-1.jpg
Tue Oct 22 00:52:53 UTC 2019

fall-1.jpg
Sun Oct 20 00:20:37 UTC 2019

summer-1.jpg
Tue Oct 22 00:53:07 UTC 2019

The ECS S3 user credentials are contained in the environmental variables of the app listed under the VCAP_SERVICES key:

```
$ cf env cf-s3-123
Getting env variables for app cf-s3-123 in org system / space system as admin...
OK

System-Provided:
{
 "VCAP_SERVICES": {
  "ecs-bucket": [
   {
    "binding_name": null,
    "credentials": {
     "accessKey": "ecs-cf-broker-3aebc1fa-2851-413d-a75d-21a1849587c6",
     "bucket": "ecs-cf-broker-93cbd230-c2d6-40dc-941d-e6db26dbab7f",
     "endpoint": "http://ecs.example.com",
     "path-style-access": true,
     "s3Url": "http://ecs-cf-broker-3aebc1fa-2851-413d-a75d-
21a1849587c6:pLCMtmZ8NMqPkQZUG3CYsDPIq47ezLbp+ixEvVZH@10.246.25.216/ecs-cf-broker-
93cbd230-c2d6-40dc-941d-e6db26dbab7f",
     "secretKey": "pLCMtmZ8NMqPkQZUG3CYsDPIq47ezLbp+ixEvVZH"
    },
    "instance_name": "mybucket",
    "label": "ecs-bucket",
    "name": "mybucket",
    "plan": "5gb",
    "provider": null,
    "syslog_drain_url": null,
    "tags": [
     "s3",
     "bucket"
    ],
    "volume_mounts": []
   }
  ]
 }
}
```

The S3 credentials can be used to list the contents of our ECS bucket. This example is using s3curl.

```
$ s3curl --id=pcfapp -- -s http://ecs.example.com/ecs-cf-broker-93cbd230-c2d6-40dc-941d-
e6db26dbab7f | xmllint --format -
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>ecs-cf-broker-93cbd230-c2d6-40dc-941d-e6db26dbab7f</Name>
  <Prefix/>
  <Marker/>
  <MaxKeys>1000</MaxKeys>
  <IsTruncated>false</IsTruncated>
  <ServerSideEncryptionEnabled>false</ServerSideEncryptionEnabled>
  <Contents>
    <Key>5f2c6c9a-d1ab-467d-a054-698dd03f59b7-Dell-EMC-ECS-Node-EX3000-1000x450-IMG-
XXL.png</Key>
    <LastModified>2019-09-16T17:25:46.073Z</LastModified>
    <ETag>"736de58cff1247fd30ffea90ea196611"</ETag>
    <Size>227153</Size>
    <StorageClass>STANDARD</StorageClass>
    <Owner>
      <ID>ecs-cf-broker-3aebc1fa-2851-413d-a75d-21a1849587c6</ID>
      <DisplayName>ecs-cf-broker-3aebc1fa-2851-413d-a75d-21a1849587c6</DisplayName>
    </Owner>
  </Contents>
</ListBucketResult>
```

2.2.6 Using Bosh Backup and Restore to backup PAS to ECS

BBR is a binary that can back up and restore BOSH deployments and BOSH Directors. BBR requires that the backup targets supply scripts that implement the backup and restore functions. BBR can communicate securely with external blobstores and databases, using TLS, if these are configured accordingly.

BBR can back up the following artifacts to ECS:

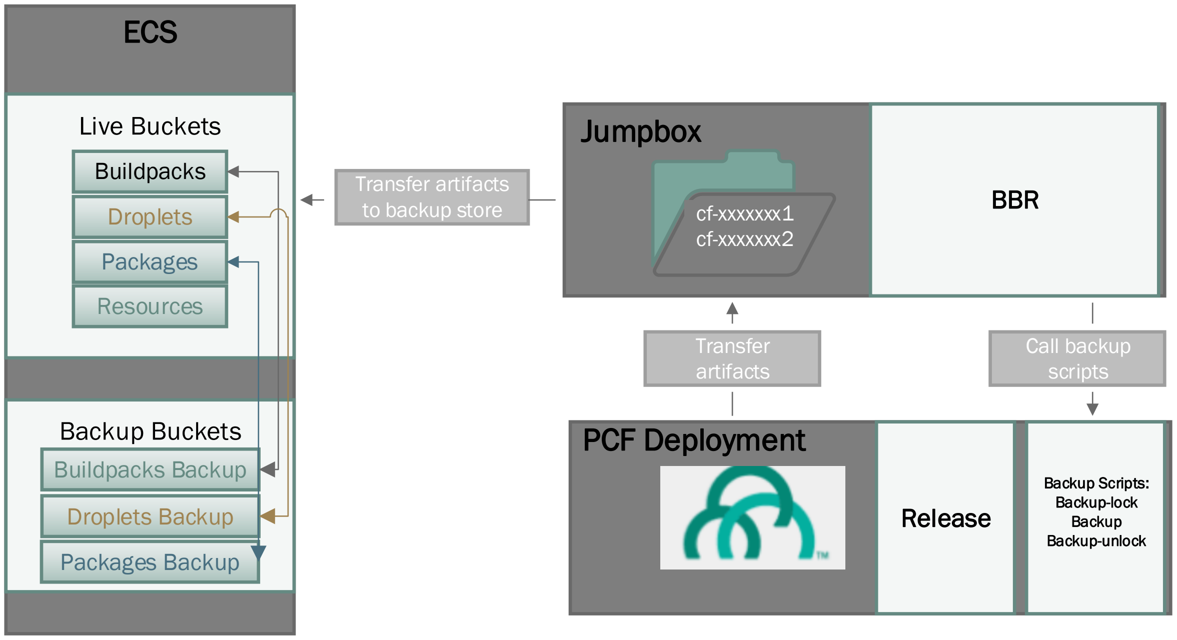
- Pivotal Application Service (PAS)
- BOSH Director

As mentioned in Section 5.3, PAS File Storage can be configured with versioning enabled or disabled. BBR will back up the external blobstores differently depending on which method was chosen.

S3 Compatible Unversioned Blobstores

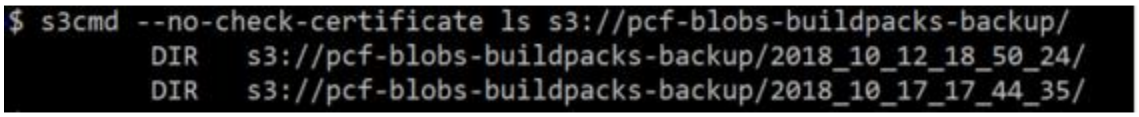
When versioning is not enabled, BBR backs up external blobstores by copying blobs from live buckets to backup buckets.

Figure 5. Unversioned BBR Workflow



The backups are stored using the date and time as the bucket prefix for each backup.

Figure 6. BBR Unversioned backup prefixes



```
$ s3cmd --no-check-certificate ls s3://pcf-blobs-buildpacks-backup/
         DIR   s3://pcf-blobs-buildpacks-backup/2018_10_12_18_50_24/
         DIR   s3://pcf-blobs-buildpacks-backup/2018_10_17_17_44_35/
```

S3 Compatible Versioned Blobstores

BBR does not download the blobs to the backup artifact when performing a backup using versioned buckets. Instead, BBR notes the *current version identifier* of each blob and stores the identifiers in the artifact.

When restoring, BBR reverts the blobs to the original versions using the version identifiers. Since the backup artifact contains only identifiers, not blobs, BBR can only restore the blobs if the original bucket containing the blob versions still exists.

Figure 7. Versioned BBR Workflow

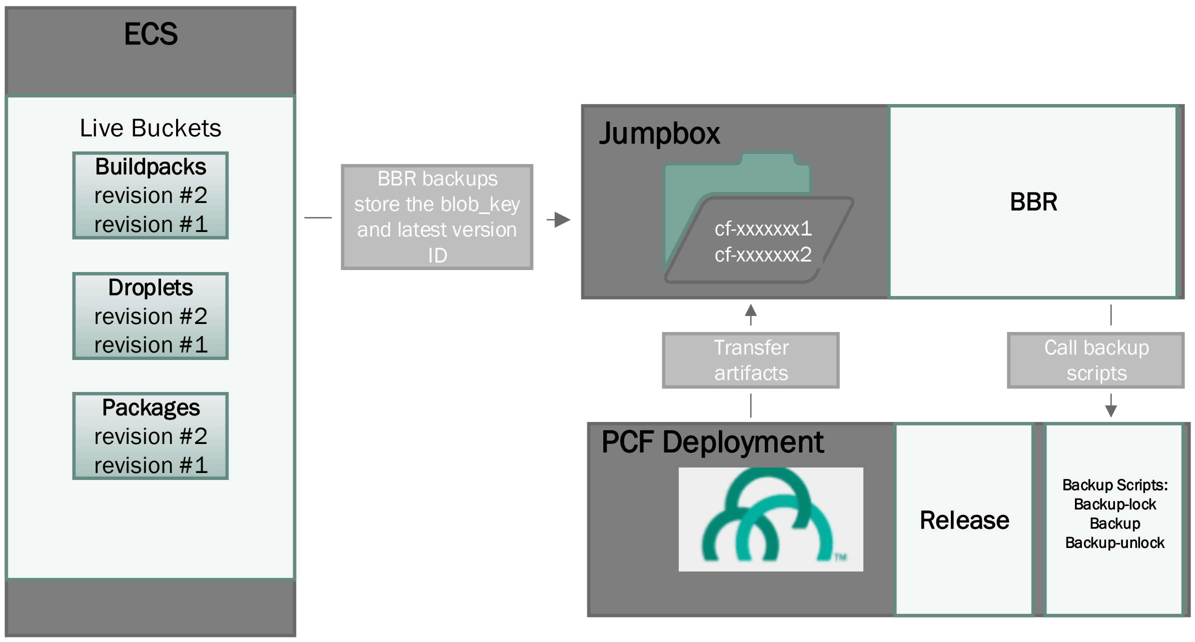


Figure 8. BBR versioned backup snapshots

Backup snapshots are stored in a directory using the naming convention ‘<deploymentID>_Date+Time’

For example: cf-2208d32e24dfe489edac_20181201T032853Z

Note: It is strongly recommended to back up the versioned artifacts to a separate highly durable storage location such as ECS.

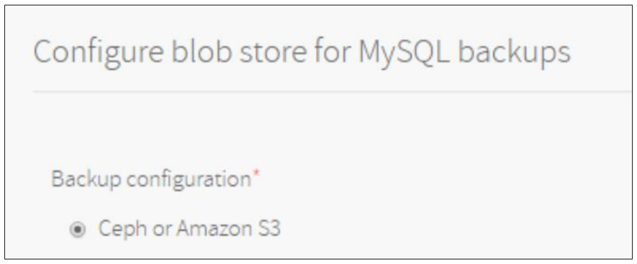
2.2.7 Managed Service Backups

BOSH backup and Restore does not backup managed or on-demand services such as MySQL. These services are configured to backup to ECS from within the service tile.

The following is an example of configuring MySQL for Pivotal Cloud Foundry v2 to backup to ECS. The MySQL for PCF product delivers dedicated instances on demand, “Database as a Service”, to PCF users. When installed, the tile deploys and maintains a single Service Broker that is responsible for PCF integration.

Once the product has been imported and added to OPS Manager, open the tile and navigate to ‘Backups’. Select the option to backup to ‘**Ceph or Amazon S3’**.

Note: This service should automatically create the bucket if it doesn’t exist, but this did not work in the version we used. We recommend creating the bucket and path prior to applying the settings.


Configure blob store for MySQL backups

Backup configuration*

● Ceph or Amazon S3

Enter the following information from the ECS Site the backups will be stored on:

- **Access Key ID:** The name of the ECS Object User.
- **Secret Access Key:** The object users S3 Key.
- **Endpoint URL:** The URL of the ECS cluster. This is typically the load balancer in front of the ECS cluster.
- **Bucket Name:** The name of the ECS bucket to store the backups.
- **Bucket Path:** The prefix (folder) to store backups. Example: backups.
- **Cron Schedule:** The service backup schedule.

Note: It’s recommend enabling a lifecycle policy on the bucket storing the managed service backups to automatically remove backups that may no longer be required.

3 Best practices

This section contains recommendations and best practices to use when configuring Dell EMC ECS with Pivotal Cloud Foundry.

Table 3 Best practices

Recommendation	Details
Managing traffic flow to ECS	Best practice for distribution of REST traffic among the ECS nodes requires an external IP load balancer. The ECS General Best Practices document contains information for several IP Load Balancers that can be configured with ECS. An NSX-T Load Balancer can also be used to distribute traffic among the ECS nodes. The configuration guide can be found here.
Use Versioning and Lifecycle Policies	If versioning is enabled for Pivotal Application Service File Storage using external blobstore then the live buckets in ECS should have versioning enabled prior to applying any changes. - Versioning can be enabled using the ECS S3 Object Access API or an S3 client with versioning support such as S3 Browser. It's strongly recommended to set lifecycle policies on the ECS buckets to minimize data and object metadata protection overhead by permanently removing revisions and backups that may no longer be required. - Note that Lifecycle is not supported on file system-enabled buckets. - Lifecycle policies can be set using the ECS S3 Object Access API or an S3 client with Lifecycle support such as S3 Browser.
ECS Service Broker recommendations	When utilizing the ECS Service Broker, it's advisable to restrict use as an origin server for static content and files. ECS should not be used as a persistent volume store for PKS data services, e.g. databases or other middleware with the ECS Service Broker due to the high-transactional / high-throughput performance profile of these operations.

Temporary Site Outage (TSO) In a multi-site ECS deployment	ECS offers an access during outage (ADO) feature that would allow access to data when there is a temporary disconnect or site outage between two sites or a failure of one site due to a power failure or natural disaster. If access is required by an application in case of temporary site outage, it is best to enable ADO when creating the bucket. However, there some things to consider when enabling ADO: - File System enabled buckets (NFS/HDFS) are read-only during TSO. - During rejoin, conflict resolution favors secondary site. - Listing of some buckets may fail during a TSO. - If possible, use a Global Server Load Balancer to handle failover so that requests are automatically directed to an available site in case of failure.
General	Dell EMC recommends that capacity and performance planning be conducted prior to deployment even if the ECS environment is dedicated to Pivotal Cloud Foundry. These areas should be revisited as the environment grows and additional workloads are introduced. We strongly recommend that any BBR artifacts which are backed up to the jumpbox be backed up to separate highly durable storage location such as ECS. A utility such as ECS-Sync can be used to automate syncing the artifacts to ECS. It’s recommended to create a separate ECS namespace for organizing and segregating the space for the Pivotal application. The buckets must exist prior to configuring File Storage for the Pivotal Application Service. The buckets and paths must exist prior to configuring managed services that store its backups in ECS

A Technical support and resources

Dell.com/support is focused on meeting customer needs with proven services and support.

Storage technical documents and videos provide expertise that helps to ensure customer success on Dell EMC storage platforms.

A.1 Related resources

The following list of documents may be helpful as a reference when configuring Dell EMC ECS with Pivotal Cloud Foundry.

Document	**Location**
Dell EMC ECS product documentation	https://www.dell.com/support/article/us/en/19/sln319451/dell-emc-ecs-3-4-x-product-documentation-index?lang=en
Dell EMC ECS Architecture and Overview	https://www.dellemc.com/resources/en-us/asset/white-papers/products/storage-1/h14071-ecs-architectural-guide-wp.pdf
Dell EMC ECS Networking and Best Practices	https://dellemc.com/resources/en-us/asset/white-papers/products/software/h15718-ecs-networking-bp-wp.pdf
Dell EMC ECS General Best Practices	https://www.dellemc.com/resources/en-us/asset/white-papers/products/storage/h16016-ecs-best-practices-guide-wp.pdf
Dell EMC ECS High Availability Design	https://www.dellemc.com/resources/en-us/asset/white-papers/products/storage-2/h16344-elastic-cloud-storage-ha-design.pdf
Pivotal Architecture and Installation Overview	https://docs.pivotal.io/platform/2-7/installing/index.html
Backing up Pivotal Cloud Foundry with BBR	https://docs.pivotal.io/platform/2-7/customizing/backup-restore/backup-pcf-bbr.html
PCF Planning and Installation	https://docs.pivotal.io/platform/2-7/customizing/index.html

A.2 Enabling Versioning and Lifecycle policies

There two most common options available to enable versioning on an ECS bucket and create a lifecycle policy include using the S3 API or an S3 client such as S3 Browser.

Versioning
Versioning is a feature of the S3 protocol that allows a bucket to retain older versions of object data. Versioning is off by default on new buckets. When you write an object, it becomes the “current version” of the object and the “older” version of the object becomes known as the “noncurrent version”. Deleting the current version will create a delete marker to indicate the object is deleted.

There’s a two-step process when using the ECS S3 API to enable versioning on a bucket.

Step 1 - Create an XML file which contains the verb to enable versioning. The name of the file can be anything, for example EnableVersioning.txt:

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>Enabled</Status>
</VersioningConfiguration>
```

Step 2 - Put the file to the bucket where versing is being enabled using an S3 command line client such as s3cmd

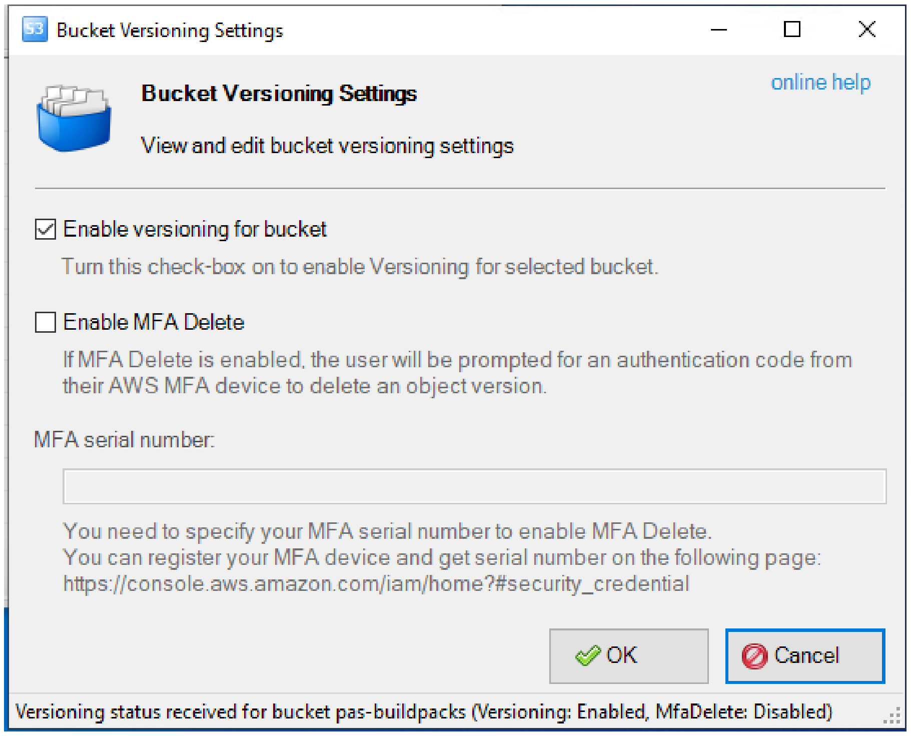
```
s3curl --id pcf -- -X PUT -d @EnableVersioning.txt -ks https://ecs.example.com/pcf-blobs-buildpacks/?versioning
```

Use the following call to confirm versioning is enabled:

```
s3curl --id pcf -- -ks https://ecs.example.com/pcf-blobs-buildpacks/?versioning | xmllint --format -
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>Enabled</Status>
```

An alternative would be to use an S3 client such as S3 Browser to enable versioning.
Right-click on the bucket and select ‘**Edit Versioning Settings..**”



Bucket Versioning Settings

View and edit bucket versioning settings

☑ Enable versioning for bucket

Turn this check-box on to enable Versioning for selected bucket.

☐ Enable MFA Delete

If MFA Delete is enabled, the user will be prompted for an authentication code from their AWS MFA device to delete an object version.

MFA serial number:

You need to specify your MFA serial number to enable MFA Delete.
You can register your MFA device and get serial number on the following page:
https://console.aws.amazon.com/iam/home?#security_credential

Versioning status received for bucket pas-buildpacks (Versioning: Enabled, MfaDelete: Disabled)

Enable the check box to enable versioning and click the OK button.

Lifecycle policy
Lifecycle controls the lifetime of objects in a bucket. Create policies to “expire” objects.

In a versioning enabled bucket:

- Expiring current versions makes them a noncurrent version
- Expiring noncurrent versions purges them from the bucket
  - Use ExpiredObjectDeleteMarker to prune Delete Markers once there are no more noncurrent versions

The following steps use the ECS S3 API to create a lifecycle policy for a bucket

Step 1 - Create the lifecycle policy Xml file and define the expiration for non-current versions

```
<LifecycleConfiguration>
  <Rule>
     <ID>expires in 90 days</ID>
     <Prefix></Prefix>
     <Status>Enabled</Status>
     <Expiration>
        <Days>1000000</Days>
     </Expiration>
     <NoncurrentVersionExpiration>
        <NoncurrentDays>90</NoncurrentDays>
     </NoncurrentVersionExpiration>
  </Rule>
</LifecycleConfiguration>
```

Step 2 - Calculate the MD5 of the payload

```
cat VersioningExpiration30.xml | openssl dgst -md5 -binary | base64
kQzCvtDClFBKVRobJG1FRw==  ← The MD5 checksum
```

Step 3 - Set the lifecycle policy on the bucket

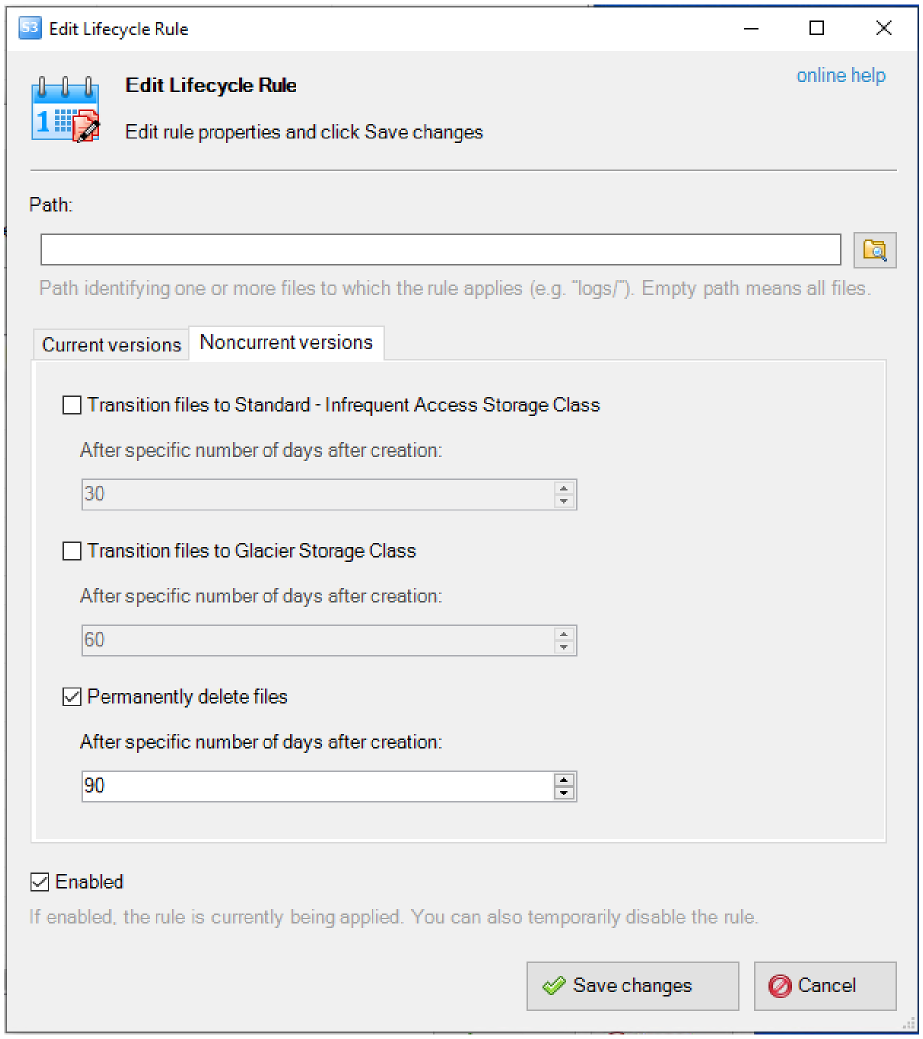
```
s3curl -id pcf --put=VersioningExpiration30.xml --contentMD5 "kQzCvtDClFBKVRobJG1FRw==" -- -ks
https://ecs.example.com/pcf-blobs-buildpacks?lifecycle | xmllint --format -
```

Use the following call to confirm a lifecycle policy is enabled:

```
s3curl --id=pcf -- -ks https://ecs.example.com/pcf-blobs-buildpacks/?lifecycle | xmllint --format –
```

As with versioning, an S3 client such as S3 browser can be used to create a policy:
Right-click the bucket and select "**Lifecycle Configuration**.."

Click 'Add' and click on the "**Noncurrent versions**" tab and define the number of days before expiring noncurrent versions



Edit Lifecycle Rule

Edit Lifecycle Rule

Edit rule properties and click Save changes

online help

Path:

Path identifying one or more files to which the rule applies (e.g. "logs/"). Empty path means all files.

Current versions | Noncurrent versions

☐ Transition files to Standard - Infrequent Access Storage Class

After specific number of days after creation: 30

☐ Transition files to Glacier Storage Class

After specific number of days after creation: 60

☑ Permanently delete files

After specific number of days after creation: 90

☑ Enabled

If enabled, the rule is currently being applied. You can also temporarily disable the rule.