OpenShift Container Platform 3.6

Release Notes

Last Updated: 2018-09-27

Legal Notice

Abstract

Table of Contents

CHAPTER 1. OVERVIEW

The following release notes for OpenShift Container Platform 3.6 summarize all new features, major corrections from the previous version, and any known bugs upon general availability.

1.1. VERSIONING POLICY

OpenShift Container Platform provides strict backwards compatibility guarantees for all supported APIs, excluding alpha APIs (which may be changed without notice) and beta APIs (which may occasionally be changed in a non-backwards compatible manner).

The OpenShift Container Platform version must match between master and node hosts, excluding temporary mismatches during cluster upgrades. For example, in a 3.6 cluster, all masters must be 3.6 and all nodes must be 3.6. However, OpenShift Container Platform will continue to support older **oc** clients against newer servers. For example, a 3.6 **oc** will work against 3.3, 3.4, 3.5, and 3.6 servers.

Changes of APIs for non-security related reasons will involve, at minimum, two minor releases (3.1 to 3.2 to 3.3, for example) to allow older **oc** to update. Using new capabilities may require newer **oc**. A 3.2 server may have additional capabilities that a 3.1 **oc** cannot use and a 3.2 **oc** may have additional capabilities that are not supported by a 3.1 server.

Table 1.1. Compatibility Matrix

	X.Y (**oc** Client)	**X.Y+N** [a] (**oc** Client)
X.Y (Server)	1	3
X.Y+N [a] (Server)	2	1

[a] Where **N** is a number greater than 1.

1 Fully compatible.

2 **oc** client may not be able to access server features.

3 **oc** client may provide options and features that may not be compatible with the accessed server.

CHAPTER 2. OPENSHIFT CONTAINER PLATFORM 3.6 RELEASE NOTES

2.1. OVERVIEW

Red Hat OpenShift Container Platform provides developers and IT organizations with a cloud application platform for deploying new applications on secure, scalable resources with minimal configuration and management overhead. OpenShift Container Platform supports a wide selection of programming languages and frameworks, such as Java, Ruby, and PHP.

Built on Red Hat Enterprise Linux and Kubernetes, OpenShift Container Platform provides a secure and scalable multi-tenant operating system for today's enterprise-class applications, while providing integrated application runtimes and libraries. OpenShift Container Platform enables organizations to meet security, privacy, compliance, and governance requirements.

2.2. ABOUT THIS RELEASE

Red Hat OpenShift Container Platform version 3.6 (RHBA-2017:2847) is now available. This release is based on OpenShift Origin 3.6. New features, changes, bug fixes, and known issues that pertain to OpenShift Container Platform 3.6 are included in this topic.

OpenShift Container Platform 3.6 is supported on RHEL 7.3 and newer with the latest packages from Extras, including Docker 1.12.

TLSV1.2 is the only supported security version in OpenShift Container Platform version 3.4 and later. You must update if you are using TLSV1.0 or TLSV1.1.

For initial installations, see the Installing a Cluster topics in the Installation and Configuration documentation.

To upgrade to this release from a previous version, see the Upgrading a Cluster topics in the Installation and Configuration documentation.

2.3. NEW FEATURES AND ENHANCEMENTS

This release adds improvements related to the following components and concepts.

2.3.1. Container Orchestration

2.3.1.1. Kubernetes Upstream

Many core features announced in March for Kubernetes 1.6 were the result of OpenShift Container Platform engineering. Red Hat continues to influence the product in the areas of storage, networking, resource management, authentication and authorization, multi-tenancy, security, service deployments and templating, and controller functionality.

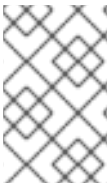
2.3.1.2. CRI Interface for Kublet-to-Docker Interaction

OpenShift Container Platform now uses the CRI interface for kublet-to-Docker interaction.

As the container space matures and choices become more available, OpenShift Container Platform needs an agnostic interface in Kubernetes for container runtime interactions. OpenShift Container Platform 3.6 switches the default configuration to use the Kubernetes Docker CRI interface.

There is a **enable-cri** setting in the ***node-config.yaml*** configuration file. A value of **true** enables the use of the interface. Change it by editing the file and stopping or starting the **atomic-openshift-node.service**.

```
$ cat /etc/origin/node/node-config.yaml
enable-cri:
  - 'true'
```



NOTE

Although the Docker CRI is stable and the default, the overall CRI interface in Kubernetes is still under development. Red Hat does not support crio, rkt, or frakti in this OpenShift Container Platform 3.6 release.

2.3.1.3. Cluster Capacity Utility for Checking True Allocatable Space

Just like a disk drive, a cluster can become fragmented over time. When you ask the cluster how much space is left, the addition of all the free space does not indicate how many actual workloads can run. For example, it might say there is 10 GB left, but it could be that no single node can take more than 512 MB.

OpenShift Container Platform 3.6 introduces a new container that you can launch as a command line or a job. The container allows you to supply a popular workload (image) with a commonly requested CPU and MEM limit and request. The logs from the container will tell you how many of that workload can be deployed.

See Analyzing Cluster Capacity for more information.

2.3.1.4. Quota on How Much (Size and Type) Remote Storage a Project Can Use

You can now control what classes of storage projects are allowed to access, how much (total size) of that class, as well as how many claims.

This feature leverages the **ResourceQuota** object and allows you to call out storage classes by name for size and claim settings.

```
$ oc create quota my-quota-1 --
hard=slow.storageclass.storage.k8s.io/requests.storage=20Gi,slow.storagecl
ass.storage.k8s.io/persistentvolumeclaims=15

$ oc describe quota my-quota-1
Name:                                                     my-quota-1
Namespace:                                                default
Resource                                                  Used   Hard
--------                                                  ----
---
slow.storageclass.storage.k8s.io/persistentvolumeclaims    0    15
slow.storageclass.storage.k8s.io/requests.storage                0
20Gi
```

See Require Explicit Quota to Consume a Resource for more information.

2.3.1.5. Ability to Scope PVC Quotas by Storage Class

In OpenShift Container Platform 3.6, administrators now have the ability to specify a separate quota for persistent volume claims (PVCs) and **`requests.storage`** per storage class.

See Setting Quotas for more information.

2.3.1.6. Project ConfigMaps, Secrets, and Downward API In the Same Directory

When you mount a memory backed volume into a container, it leverages a directory. Now, you can place all sources of the configuration for your application (**`configMaps`**, secrets, and downward API) into the same directory path.

The new projected line in the volume definition allows you to tell multiple volumes to leverage the same mount point while guarding for path collisions.

```
volumes:
  - name: all-in-one
    projected:
      sources:
      - secret:
          name: test-secret
          items:
            - key: data-1
              path: mysecret/my-username
            - key: data-2
              path: mysecret/my-passwd

      - downwardAPI:
          items:
            - path: mydapi/labels
              fieldRef:
                fieldPath: metadata.labels
            - path: mydapi/name
              fieldRef:
                fieldPath: metadata.name
            - path: mydapi/cpu_limit
              resourceFieldRef:
                containerName: allinone-normal
                resource: limits.cpu
                divisor: "1m"

                - configMap:
                    name: special-config
                    items:
                      - key: special.how
                        path: myconfigmap/shared-config
                      - key: special.type
                        path: myconfigmap/private-config
```

2.3.1.7. Init Containers

You run init containers in the same pod as your application container to create the environment your application requires or to satisfy any preconditions the application might have. You can run utilities that you would otherwise need to place into your application image. You can run them in different file system namespaces (view of the same file system) and offer them different secrets than your application container.

Init containers run to completion and each container must finish before the next one starts. The init containers will honor the restart policy. Leverage **`initContainers`** in the **`podspec`**.

```
$ cat init-containers.yaml
apiVersion: v1
kind: Pod
metadata:
  name: init-loop
spec:
  containers:
  - name: hello-openshift
    image: openshift/hello-openshift
    ports:
    - containerPort: 80
    volumeMounts:
    - name: workdir
      mountPath: /usr/share/nginx/html
  initContainers:
  - name: init
    image: centos:centos7
    command:
    - /bin/bash
    - "-c"
    - "while :; do sleep 2; echo hello init container; done"
  volumes:
  - name: workdir
    emptyDir: {}
```

```
$ oc get -f init-containers.yaml
NAME                  READY     STATUS     RESTARTS   AGE
hello-openshift       0/1       Init:0/1   0          6m
```

2.3.1.8. Multiple Schedulers at the Same Time

Kubernetes now supports extending the default scheduler implementation with custom schedulers.

After configuring and deploying your new scheduler, you can call it by name from the **`podspec`** via **`schedulerName`**. These new schedulers are packaged into container images and run as pods inside the cluster.

```
$ cat pod-custom-scheduler.yaml
apiVersion: v1
kind: Pod
metadata:
  name: custom-scheduler
spec:
  schedulerName: custom-scheduler
  containers:
  - name: hello
    image: docker.io/ocpqe/hello-pod
```

See Scheduling for more information.

2.3.1.9. Turn ConfigMap Content into Environment Variables within the Container

Instead of individually declaring environment variables in a pod definition, a **configMap** can be imported and all of its content can be dynamically turned into environment variables.

In the pod specification, leverage the **envFrom** object and reference the desired **configMap**:

```
env:
- name: duplicate_key
  value: FROM_ENV
- name: expansion
  value: $(REPLACE_ME)
envFrom:
- configMapRef:
    name: env-config
```

See **ConfigMaps** for more information.

2.3.1.10. Node Affinity and Anti-affinity

Control which nodes your workload will land on in a more generic and powerful way as compared to **nodeSelector**.

NodeSelectors provide a powerful way for a user to specify which node a workload should land on. However, If the selectors are not available or are conflicted, the workload will not be scheduled at all. They also require a user to have specific knowledge of node label keys and values. Operators provide a more flexible way to select nodes during scheduling.

Now, you can select the label value you would like the operator to compare against (for example, **In**, **NotIn**, **Exists**, **DoesNotExist**, **Gt**, and **Lt**). You can choose to make satisfying the operator required or preferred. Preferred means search for the match, but, if you can not find one, ignore it.

```
affinity:
  nodeAffinity:
    preferredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
        - matchExpressions:
          - key: "failure-domain.beta.kubernetes.io/zone"
            operator: In
            values: ["us-central1-a"]
```

```
affinity:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
        - matchExpressions:
          - key: "failure-domain.beta.kubernetes.io/zone"
            operator: NotIn
            values: ["us-central1-a"]
```

See Advanced Scheduling and Node Affinity for more information.

2.3.1.11. Pod Affinity and Anti-Affinity

Pod affinity and anti-affinity is helpful if you want to allow Kubernetes the freedom to select which zone an application lands in, but whichever it chooses you would like to make sure another component of that application lands in the same zone.

Another use case is if you have two application components that, due to security reasons, cannot be on the same physical box. However, you do not want to lock them into labels on nodes. You want them to land anywhere, but still honor anti-affinity.

Many of the same high-level concepts mentioned in the node affinity and anti-affinity hold true here. For pods, you declare a **`topologyKey`**, which will be used as the boundary object for the placement logic.

```
affinity:
    podAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
          - key: service
            operator: In
            values: [“S1”]
        topologyKey: failure-domain.beta.kubernetes.io/zone


affinity:
    podAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
          - key: service
            operator: In
            values: [“S1”]
        topologyKey: kubernetes.io/hostname
```

See Advanced Scheduling and Pod Affinity and Anti-affinity for more information.

2.3.1.12. Taints and Tolerations

Taints and tolerations allow the **node** to control which **pods** should (or should not) be scheduled on them.

A *taint* allows a node to refuse pod to be scheduled unless that pod has a matching *toleration*.

You apply taints to a node through the node specification (**`NodeSpec`**) and apply tolerations to a pod through the pod specification (**`PodSpec`**). A taint on a node instructs the node to repel all pods that do not tolerate the taint.

Taints and tolerations consist of a key, value, and effect. An operator allows you to leave one of these parameters empty.

In OpenShift Container Platform 3.6, daemon pods do respect taints and tolerations, but they are created with **`NoExecute`** tolerations for the **`node.alpha.kubernetes.io/notReady`** and **`node.alpha.kubernetes.io/unreachable`** taints with no **`tolerationSeconds`**. This ensures that when the **`TaintBasedEvictions`** alpha feature is enabled, they will not be evicted when there are node problems such as a network partition. (When the **`TaintBasedEvictions`** feature is not enabled, they are also not evicted in these scenarios, but due to hard-coded behavior of the **`NodeController`** rather than due to tolerations).

Set the taint from the command line:

```
$ oc taint nodes node1 key=value:NoSchedule
```

Set toleration in the **PodSpec**:

```
tolerations:
- key: "key"
  operator: "Equal"
  value: "value"
  effect: "NoSchedule"
```

2.3.1.13. Using Image Streams with Kubernetes Resources (Technology Preview)

This feature is currently in Technology Preview and not for production workloads.

OpenShift Container Platform has long offered easy integration between continuous integration pipelines that create deployable Docker images and automatic redeployment and rollout with **DeploymentConfigs**. This makes it easy to define a standard process for continuous deployment that keeps your application always running. As new, higher level constructs like deployments and **StatefulSets** have reached maturity in Kubernetes, there was no easy way to leverage them and still preserve automatic CI/CD.

In addition, the image stream concept in OpenShift Container Platform makes it easy to centralize and manage images that may come from many different locations, but to leverage those images in Kubernetes resources you had to provide the full registry (an internal service IP), the namespace, and the tag of the image, which meant that you did not get the ease of use that **BuildConfigs** and **DeploymentConfigs** offer by allowing direct reference of an image stream tag.

Starting in OpenShift Container Platform 3.6, we aim to close that gap both by making it as easy to trigger redeployment of Kubernetes Deployments and **StatefulSets**, and also by allowing Kubernetes resources to easily reference OpenShift Container Platform image stream tags directly.

See Using Image Streams with Kubernetes Resources for more information.

2.3.2. Registry

2.3.2.1. Validating Image Signatures Show Appropriate Metadata

When working with image signatures as the **image-admin** role, you can now see the status of the images in terms of their signatures.

You can now use the **oc adm verify-image-signature** command to save or remove signatures. The resulting **oc describe istag** displays additional metadata about the signature's status.

```
$ oc describe istag origin-pod:latest
Image Signatures:
  Name:
sha256:c13060b74c0348577cbe07dedcdb698f7d893ea6f74847154e5ef3c8c9369b2c@f6
6d720cfaced1b33e8141a844e793be
  Type: atomic
  Status: Unverified

# Verify the image and save the result back to image stream
```

```
$ oc adm verify-image-signature
sha256:c13060b74c0348577cbe07dedcdb698f7d893ea6f74847154e5ef3c8c9369b2c \
  --expected-identity=172.30.204.70:5000/test/origin-pod:latest --save --
as=system:admin
sha256:c13060b74c0348577cbe07dedcdb698f7d893ea6f74847154e5ef3c8c9369b2c
signature 0 is verified (signed by key: "172B61E538AAC0EE")

# Check the image status
$ oc describe istag origin-pod:latest
Image Signatures:
  Name:
sha256:c13060b74c0348577cbe07dedcdb698f7d893ea6f74847154e5ef3c8c9369b2c@f6
6d720cfaced1b33e8141a844e793be
  Type:   atomic
  Status:   Verified
  Issued By:  172B61E538AAC0EE
  Signature is Trusted (verified by user "system:admin" on 2017-04-28
12:32:25 +0200 CEST)
  Signature is ForImage ( on 2017-04-28 12:32:25 +0200 CEST)
```

See Image Signatures and Enabling Image Signature Support for more information.

2.3.2.2. Registry REST Endpoint for Reading and Writing Image Signatures

There is now a programmable way to read and write signatures using only the docker registry API.

To read, you must be authenticated to the registry.

```
PUT /extensions/v2/{namespace}/{name}/signatures/{digest}
$ curl http://<user>:<token>@<registry-
endpoint>:5000/extensions/v2/<namespace>/<name>/signatures/sha256:<digest>

JSON:
{
  "version": 2,
  "type":    "atomic",
  "name":
"sha256:4028782c08eae4a8c9a28bf661c0a8d1c2fc8e19dbaae2b018b21011197e1484@c
ddeb7006d914716e2728000746a0b23",
  "content": "<base64 encoded signature>",
}
```

To write, you must have the **`image-signer`** role.

```
GET /extensions/v2/{namespace}/{name}/signatures/{digest}
$ curl http://<user>:<token>@<registry-
endpoint>:5000/extensions/v2/<namespace>/<name>/signatures/sha256:<digest>


{
  "signatures": [
  {
    "version": 2,
    "type":    "atomic",
    "name":
```

```
"sha256:4028782c08eae4a8c9a28bf661c0a8d1c2fc8e19dbaae2b018b21011197e1484@c
ddeb7006d914716e2728000746a0b23",
    "content": "<base64 encoded signature>",
  }
  ]
}
```

2.3.3. Platform Management

2.3.3.1. Require Explicit Quota to Consume a Resource (Technology Preview)

This feature is currently in Technology Preview and not for production workloads.

If a resource is not managed by quota, a user has no restriction on the amount of resource that can be consumed. For example, if there is no quota on storage related to the gold storage class, the amount of gold storage a project can create is unbounded.

See Setting Quotas for more information.

2.3.4. Storage

2.3.4.1. AWS EFS Provisioner

The AWS EFS provisioner allows you to dynamically use the AWS EFS endpoint to get NFS remote persistent volumes on AWS.

It leverages the external dynamic provisioner interface. It is provided as a **`docker`** image that you configure with a **`configMap`** and deploy on OpenShift Container Platform. Then, you can use a storage class with the appropriate configuration.

Storage Class Example

```
apiVersion: storage.k8s.io/v1beta1
kind: StorageClass
metadata:
  name: slow
provisioner: foobar.io/aws-efs
parameters:
  gidMin: "40000"
  gidMax: "50000"
```

`gidMin` and **`gidMax`** are the minimum and maximum values, respectively, of the GID range for the storage class. A unique value (GID) in this range (**`gidMin`** to **`gidMax`**) is used for dynamically provisioned volumes.

2.3.4.2. VMware vSphere Storage

VMware vSphere storage allows you to dynamically use the VMware vSphere storage options ranging from VSANDatastore, ext3, vmdk, and VSAN while honoring vSphere Storage Policy (SPBM) mappings.

VMware vSphere storage leverages the cloud provider interface in Kubernetes to trigger this in-tree dynamic storage provisioner. Once the cloud provider has the correct credential information, tenants can leverage storage class to select the desired storage.

Storage Class Example

```
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: fast
provisioner: kubernetes.io/vsphere-volume
parameters:
    diskformat: zeroedthick
```

See Configuring for VMWare vSphere and Persistent Storage Using VMWare vSphere Volume for more information.

2.3.4.3. Increased Security with iSCSI CHAP and Mount Operations

You can now use CHAP authentication for your iSCSI remote persistent volumes (PVs). Also, you can annotate your PVs to leverage any mount options that are supported by that underlying storage technology.

The tenant supplies the correct user name and password for the CHAP authentication as a secret in their **podspec**. For mount options, you supply the annotation in the PV.

```
volumes:
  - name: iscsivol
    iscsi:
      targetPortal: 127.0.0.1
      iqn: iqn.2015-02.example.com:test
      lun: 0
      fsType: ext4
      readOnly: true
      chapAuthDiscovery: true
      chapAuthSession: true
      secretRef:
         name: chap-secret
```

Set **`volume.beta.kubernetes.io/mount-options`** to **`volume.beta.kubernetes.io/mount-options: rw,nfsvers=4,noexec`**.

See Mount Options for more information.

2.3.4.4. Mount Options (Technology Preview)

Mount Options are currently in Technology Preview and not for production workloads.

You can now specify mount options while mounting a persistent volume by using the annotation **`volume.beta.kubernetes.io/mount-options`**

See Persistent Storage for more information.

2.3.4.5. Improved and Fully Automated Support for CNS-backed OCP Hosted Registry

Previously, only a few supported storage options existed for a scaled, highly-available integrated OpenShift Container Platform (OCP) registry. Automated container native storage (CNS) 3.6 and the OpenShift Container Platform installer now include an option to automatically deploy a scale-out registry

based on highly available storage, out of the box. When enabled in the installer's inventory file, CNS will be deployed on a desired set of nodes (for instance, infrastructure nodes). Then, the required underlying storage constructs will automatically be created and configured for use with the deployed registry. Moving an existing registry deployment from NFS to CNS is also supported, and requires additional steps for data migration.

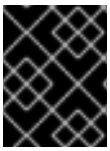
Backing the OpenShift Container Platform registry with CNS enables users to take advantage of the globally available storage capacity, strong read/write consistency, three-way replica, and RHGS data management features.

The feature is provided through integrations in the OpenShift Container Platform advanced installation process. A few dedicated storage devices and a simple change to the inventory file is all that is required.

2.3.4.6. OpenShift Container Platform Commercial Evaluation Subscription Includes CNS and CRS

The OpenShift Commercial Evaluation subscription includes container native storage (CNS), container ready storage (CRS) solutions.

The OpenShift Commercial Evaluation subscription SKU bundles the CNS and CRS features, with additional entitlements to evaluate OpenShift Container Platform with CNS/CRS.



IMPORTANT

Evaluation SKUs are not bundled with OpenShift Container Platform's SKUs or entitlements. Consult your Red Hat account representative for subscription guidance.

2.3.5. Scale

2.3.5.1. Updated etcd Performance Guidance

See Recommended Host Practices for updated etcd performance guidance.

2.3.5.2. Updated Sizing Guidance

In OpenShift Container Platform 3.6 , the maximum number of nodes per cluster is 2000.

2.3.6. Networking

2.3.6.1. Multiple Destinations in egress-router

OpenShift Container Platform 3.6 introduces the ability to connect to multiple destinations from a project without needing to reserve a separate source IP for each of them. Also, there is now an optional fallback IP. Old syntax continues to behave the same and there is no change to **EGRESS_SOURCE** and **EGRESS_GATEWAY** definitions.

Old way:

```
- name: EGRESS_DESTINATION
  value: 203.0.113.25
```

New way:

```
- name: EGRESS_DESTINATION
  value: |
    80 tcp 1.2.3.4
    8080 tcp 5.6.7.8 80
    8443 tcp 9.10.11.12 443
    13.14.15.16
```

```
localport  udp|tcp  dest-ip [dest-port]
```

See Managing Networking for more information.

2.3.6.2. Added HTTP Proxy Mode for the Egress Router

TLS connections (certificate validations) do not easily work because the client needs to connect to the egress router's IP (or name) rather than to the destination server's IP/name. Now, the egress router can be run as a proxy rather than just redirecting packets.

How it works:

1. Create a new project and pod.
2. Create the **`egress-router-http-proxy`** pod.
3. Create the service for **`egress-router-http-proxy`**.
4. Set up **`http_proxy`** in the pod:

```
# export http_proxy=http://my-egress-router-service-name:8080
# export https_proxy=http://my-egress-router-service-name:8080
```

5. Test and check squid headers in response:

```
$ curl  -ILs http://www.redhat.com
$ curl  -ILs https://rover.redhat.com
    HTTP/1.1 403 Forbidden
    Via: 1.1 egress-http-proxy (squid/x.x.x)
$ curl  -ILs http://www.google.com
    HTTP/1.1 200 OK
    Via: 1.1 egress-http-proxy (squid/x.x.x)
$ curl  -ILs https://www.google.com
    HTTP/1.1 200 Connection established
    HTTP/1.1 200 OK
```

See Managing Networking for more information.

2.3.6.3. Use DNS Names with Egress Firewall

There are several benefits of using DNS names versus IP addresses:

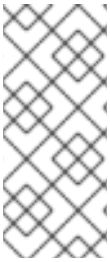
- It tracks DNS mapping changes.
- Human-readable, easily remembered naming.
- Potentially backed by multiple IP addresses.

How it works:

1. Create the project and pod.
2. Deploy egress network policy with DNS names.
3. Validate the firewall.

Egress Policy Example

```
{
    "kind": "EgressNetworkPolicy",
    "apiVersion": "v1",
    "metadata": {
        "name": "policy-test"
    },
    "spec": {
        "egress": [
            {
                "type": "Allow",
                "to": {
                    "dnsName": "stopdisablingselinux.com"
                }
            },
            {
                "type": "Deny",
                "to": {
                  "cidrSelector": "0.0.0.0/0"
                }
            }
        ]
    }
}
```



NOTE

Exposing services by creating routes will ignore the Egress Network Policy. Egress Network policy Service endpoint filtering is performed on the **kubeproxy** node. When the router is involved, **kubeproxy** is bypassed and Egress Network Policy enforcement is not applied. Administrators can prevent this bypass by limiting access and the ability to create routes.

See Managing Pods for more information.

2.3.6.4. Network Policy (Technology Preview)

Network Policy (currently in Technology Preview and not for production workloads) is an optional plug-in specification of how selections of pods are allowed to communicate with each other and other network endpoints. It provides fine-grained network namespace isolation using labels and port specifications.

After installing the Network Policy plug-in, an annotation that flips the namespace from **allow all traffic** to **deny all traffic** must first be set on the namespace. At that point, **NetworkPolicies** can be created that define what traffic to allow. The annotation is as follows:

```
$ oc annotate namespace ${ns} 'net.beta.kubernetes.io/network-policy=
{"ingress":{"isolation":"DefaultDeny"}}'
```

The allow-to-red policy specifies "all red pods in namespace **project-a** allow traffic from any pods in any namespace." This does not apply to the red pod in namespace **project-b** because **podSelector** only applies to the namespace in which it was applied.

Policy applied to project

```
kind: NetworkPolicy
apiVersion: extensions/v1beta1
metadata:
  name: allow-to-red
spec:
  podSelector:
    matchLabels:
      type: red
  ingress:
  - {}
```

See Managing Networking for more information.

2.3.6.5. Router Template Format

OpenShift Container Platform 3.6 introduces improved router customization documentation. Many RFEs could be solved with better documentation around the HAProxy features and functions which are now added, and their customizable fields via annotations and environment variables. For example, router annotations to do per-route operations.

For example, to change the behavior of HAProxy (round-robin load balancing) through annotating a route:

```
$ oc annotate route/ab haproxy.router.openshift.io/balance=roundrobin
```

For more information, see Deploying a Customized HAProxy Router.

2.3.6.6. Use a Different F5 Partition Other than /Common

With OpenShift Container Platform 3.6, there is now the added ability to use custom F5 partitions for properly securing and isolating OpenShift Container Platform route synchronization and configuration.

The default is still **/Common** or global partition if not specified. Also, behavior is unchanged if the partition path is not specified. This new feature ensures all the referenced objects are in the same partition, including virtual servers (**http** or **https**).

2.3.6.7. Support IPv6 Terminated at the Router with Internal IPv4

The router container is able to terminate IPv6 traffic and pass HTTP[S] through to the back-end pod.

The IPv6 interfaces on the router must be enabled, with IPv6 addresses listening (**::80**, **::443**). The client needs to reach the router node using IPv6. IPv4 should be unaffected and continue to work, even if IPv6 is disabled.

NOTE

HAProxy can only terminate IPv6 traffic when the router uses the network stack of the host (default). When using the container network stack (`oc adm router --service-account=router --host-network=false`), there is no global IPv6 address for the pod.

2.3.7. Installation

2.3.7.1. Ansible Service Broker (Technology Preview)

The Ansible service broker is currently in Technology Preview and not for production workloads. This feature includes:

- Implementation of the open service broker API that enables users to leverage Ansible for provisioning and managing of services via the service catalog on OpenShift Container Platform.
- Standardized approach for delivering simple to complex multi-container OpenShift Container Platform services.
- Works in conjunction with Ansible playbook bundles (APB), which is a lightweight meta container comprised of a few named playbooks for each open service broker API operations.

Service catalog and Ansible service broker must be configured during OpenShift Container Platform installation. Once enabled, APB services can be deployed right from Service Catalog UI.

IMPORTANT

In OpenShift Container Platform In OCP 3.6.0, the Ansible Service Broker exposes an unprotected route, which allows unauthenticated users to provision resources in the cluster, namely Mediawiki and Postgres Ansible Playbook Bundles.

See Configuring the Ansible Service Broker for more information.

2.3.7.2. Ansible Playbook Bundles (APB) (Technology Preview)

Ansible playbook bundles (APB) (currently in Technology Preview and not for production workloads) is a short-lived, lightweight container image consisting of:

- Simple directory structure with named action playbooks
- Metadata consisting of:
  - required/optional parameters
  - dependencies (provision versus bind)
- Ansible runtime environment
- Leverages existing investment in Ansible playbooks and roles
- Developer tooling available for guided approach
- Easily modified or extended

- Example APB services included with OpenShift Container Platform 3.6:
  - MediaWiki, PostgreSQL

When a user orders an application from the service catalog, the Ansible service broker will download the associated APB image from the registry and run it. Once the named operation has been performed on the service, the APB image will then terminate.

2.3.7.3. Automated installation of CloudForms 4.5 Inside OpenShift (Technology Preview)

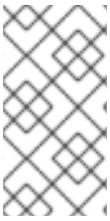
The installation of containerized CloudForms inside OpenShift Container Platform is now part of the main installer (currently in Technology Preview and not for production workloads). It is now treated like other common components (metrics, logging, and so on).

After the OpenShift Container Platform cluster is provisioned, there is an additional playbook you can run to deploy CloudForms into the environment (using the **`openshift_cfme_install_app`** flag in the hosts file).

```
$ ansible-playbook -v -i <INVENTORY_FILE> playbooks/byo/openshift-
cfme/config.yml
```

Requirements:

Type	Size	CPUs	Memory
Masters	1+	8	12 GB
Nodes	2+	4	8 GB
PV Storage	25 GB	N/A	N/A



NOTE

NFS is the only storage option for the Postgres database at this time.

The NFS server should be on the first master host. The persistent volume backing the NFS storage volume is mounted on exports.

2.3.7.4. Automated CNS Deployment with OCP Ansible Advanced Installation

OpenShift Container Platform (OCP) 3.6 now includes an integrated and simplified installation of container native storage (CNS) through the advanced installer. The installer's inventory file is simply configured. The end result is an automated, supportable, best practice installation of CNS, providing ready-to-use persistent storage with a pre-created storage class. The advanced installer now includes automated and integrated support for deployment of CNS, correctly configured and highly available out-of-the-box.

CNS storage device details are added to the installer's inventory file. Examples provided in OpenShift Container Platform advanced installation documentation. The installer manages configuration and deployment of CNS, its dynamic provisioner, and other pertinent details.

2.3.7.5. Installation of etcd, Docker Daemon, and Ansible Installer as System Containers (Technology Preview)

This feature is currently in Technology Preview and not for production workloads.

RHEL System Containers offer more control over the life cycle of the services that do not run inside OpenShift Container Platform or Kubernetes. Additional system containers will be offered over time.

System Containers leverage the OSTree on RHEL or Atomic Host. They are controlled by the kernel init system and therefore can be leveraged earlier in the boot sequence. This feature is enabled in the installer configuration.

For more information, see Configuring System Containers.

2.3.7.6. Running OpenShift Installer as a System Container (Technology Preview)

This feature is currently in Technology Preview and not for production workloads.

To run the OpenShift Container Platform installer as a system container:

```
$ atomic install --system --set INVENTORY_FILE=$(pwd)/inventory
registry:port/openshift3/ose-ansible:v3.6

$ systemctl start ose-ansible-v3.6
```

2.3.7.7. etcd3 Data Model for New Installations

Starting with new installations of OpenShift Container Platform 3.6, the etcd3 v3 data model is the default. By moving to the etcd3 v3 data model, there is now:

- Larger memory space to enable larger cluster sizes.
- Increased stability in adding and removing nodes in general life cycle actions.
- A significant performance boost.

A migration playbook will be provided in the near future allowing upgraded environments to migrate to the v3 data model.

2.3.7.8. Cluster-wide Control of CA

You now have the ability to change the certificate expiration date en mass across the cluster for the various framework components that use TLS.

We offer new cluster variables per framework area so that you can use different time-frames for different framework components. Once set, issue the new **`redeploy-openshift-ca`** playbook. This playbook only works for redeploying the root CA certificate of OpenShift Container Platform. Once you set the following options, they will be effective in a new installation, or they can be used when redeploying certificates against an existing cluster.

New Cluster Variables

```
# CA, node and master certificate expiry
openshift_ca_cert_expire_days=1825
openshift_node_cert_expire_days=730
```

```
openshift_master_cert_expire_days=730

# Registry certificate expiry
openshift_hosted_registry_cert_expire_days=730

# Etcd CA, peer, server and client certificate expiry
etcd_ca_default_days=1825
```

2.3.7.9. General Stability

OpenShift Container Platform engineering and the OpenShift Online operations teams have been working closely together to refactor and enhance the installer. The OpenShift Container Platform 3.6 release includes the culmination of those efforts, including:

- Upgrading from OpenShift Container Platform 3.5 to 3.6
- Idempotency refactoring of the configuration role
- Swap handling during installation
- All BYO playbooks pull from a normalized group source
- A final port of operation's Ansible modules
- A refactoring of excluder roles

2.3.8. Metrics and Logging

2.3.8.1. Removing Metrics Deployer and Removing Logging Deployer

The metrics and logging deployers were replaced with **`playbook2image`** for **`oc cluster up`** so that **`openshift-ansible`** is used to install logging and metrics:

```
$ oc cluster up --logging --metrics
```

Check metrics and logging pod status:

```
$ oc get pod -n openshift-infra
$ oc get pod -n logging
```

2.3.8.2. Expose Elasticsearch as a Route

By default, the Elasticsearch instance deployed with OpenShift Container Platform aggregated logging is not accessible from outside the deployed OpenShift Container Platform cluster. You can now enable an external route for accessing the Elasticsearch instance via its native APIs to enable external access to data via various supported tools.

Direct access to the Elasticsearch instance is enabled using your OpenShift token. You have the ability to provide the external Elasticsearch and Elasticsearch Operations host names when creating the server certificate (similar to Kibana). The provided Ansible tasks simplify route deployment.

2.3.8.3. Mux (Technology Preview)

mux is a new Technology Preview feature for OpenShift Container Platform 3.6.0 designed to facilitate better scaling of aggregated logging. It uses a smaller set of from Fluentd instances (called *muxes*) kept near the Elasticsearch instance pod to improve the efficiency of indexing log records into Elasticsearch.

See Aggregating Container Logs for more information.

2.3.9. Developer Experience

2.3.9.1. Service Catalog Experience in the CLI (Technology Preview)

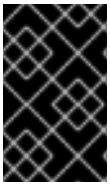
This feature (currently in Technology Preview and not for production workloads) brings the Service Catalog experience to the CLI.

You can run **`oc cluster up --version=latest --service-catalog=true`** to get the Service Catalog experience in OpenShift Container Platform 3.6.

2.3.9.2. Template Service Broker (Technology Preview)

The template service broker (currently in Technology Preview) exposes OpenShift templates through a open service broker API to the Service Catalog.

The template service broker (TSB) matches the lifecycles of provision, deprovision, bind, unbind with existing templates. No changes are required to templates, unless you expose bind. Your application will get injected with configuration details (bind).



IMPORTANT

The TSB is currently a Technology Preview feature and should not be used in production clusters. Enabling the TSB currently requires opening unauthenticated access to the cluster; this security issue will be resolved before exiting the Technology Preview phase.

See Configuring the Template Service Broker for more information.

2.3.9.3. Automatic Build Pruning

Previously, only **`oc adm prune`** could be used. Now, you can define how much build history you want to keep per build configuration. Also, you can set **`successful`** versus **`failed`** history limits separately.

See Advanced Build Operations for more information.

2.3.9.4. Easier Custom Slave Configuration for Jenkins

In OpenShift Container Platform 3.6, it is now easier to make images available as slave pod templates.

Slaves are defined as image-streams or image-stream tags with the appropriate label. Slaves can also be specified via a **`ConfigMap`** with the appropriate label.

See Using the Jenkins Kubernetes Plug-in to Run Jobs for more information.

2.3.9.5. Detailed Build Timing

Builds now record timing information based on more granular steps.

Information such as how long it took to pull the base image, clone the source, build the source, and push the image are provided. For example:

```
$ oc describe build nodejs-ex-1
Name:        nodejs-ex-1
Namespace:    myproject
Created:    2 minutes ago

Status:            Complete
Started:        Fri, 07 Jul 2017 17:49:37 EDT
Duration:        2m23s
  FetchInputs:       2s
  CommitContainer:   6s
  Assemble:          36s
  PostCommit:           0s
  PushImage:         1m0s
```

2.3.9.6. Default Hard Eviction Thresholds

OpenShift Container Platform uses the following default configuration for **`eviction-hard`**.

```
...
kubeletArguments:
  eviction-hard:
  - memory.available<100Mi
  - nodefs.available<10%
  - nodefs.inodesFree<5%
  - imagefs.available<15%
...
```

See Handling Out of Resource Errors for more information.

2.3.9.7. Other Developer Experience Changes

- Webhook triggers for Github and Bitbucket.
- HTTPD 2.4 s2i support.
- Separate build events for **`start`**, **`canceled`**, **`success`**, and **`fail`**.
- Support for arguments in Docker files.
- Environment variables in pipeline builds.
- Credential support for Jenkins Sync plug-in for ease of working external Jenkins instance.
- **`ValueFrom`** Support in build environment variables.
- Deprecated Jenkins v1 image.
- **`oc cluster up`**: support launching service catalog
- Switch to nip.io from xip.io, with improved stability

2.3.10. Web Console

2.3.10.1. Service Catalog (Technology Preview)

You can now opt into the service catalog (currently in Technology Preview and not for production workloads) during installation or upgrade.

When developing microservices-based applications to run on cloud native platforms, there are many ways to provision different resources and share their coordinates, credentials, and configuration, depending on the service provider and the platform.

To give developers a more seamless experience, OpenShift Container Platform includes a Service Catalog, an implementation of the open service broker API (OSB API) for Kubernetes. This allows users to connect any of their applications deployed in OpenShift Container Platform to a wide variety of service brokers.

The service catalog allows cluster administrators to integrate multiple platforms using a single API specification. The OpenShift Container Platform web console displays the service classes offered by brokers in the service catalog, allowing users to discover and instantiate those services for use with their applications.

As a result, service users benefit from ease and consistency of use across different types of services from different providers, while service providers benefit from having one integration point that gives them access to multiple platforms.

This feature consists of:

- The Service Consumer: The individual, application , or service that uses a service enabled by the broker and catalog.
- The Catalog: Where services are published for consumption.
- Service Broker: Publishes services and intermediates service creation and credential configuration with a provider.
- Service Provider: The technology delivering the service.
- Open Service Broker API: Lists services, provisions and deprovisions, binds, and unbinds.

See Enabling the Service Catalog for more information.

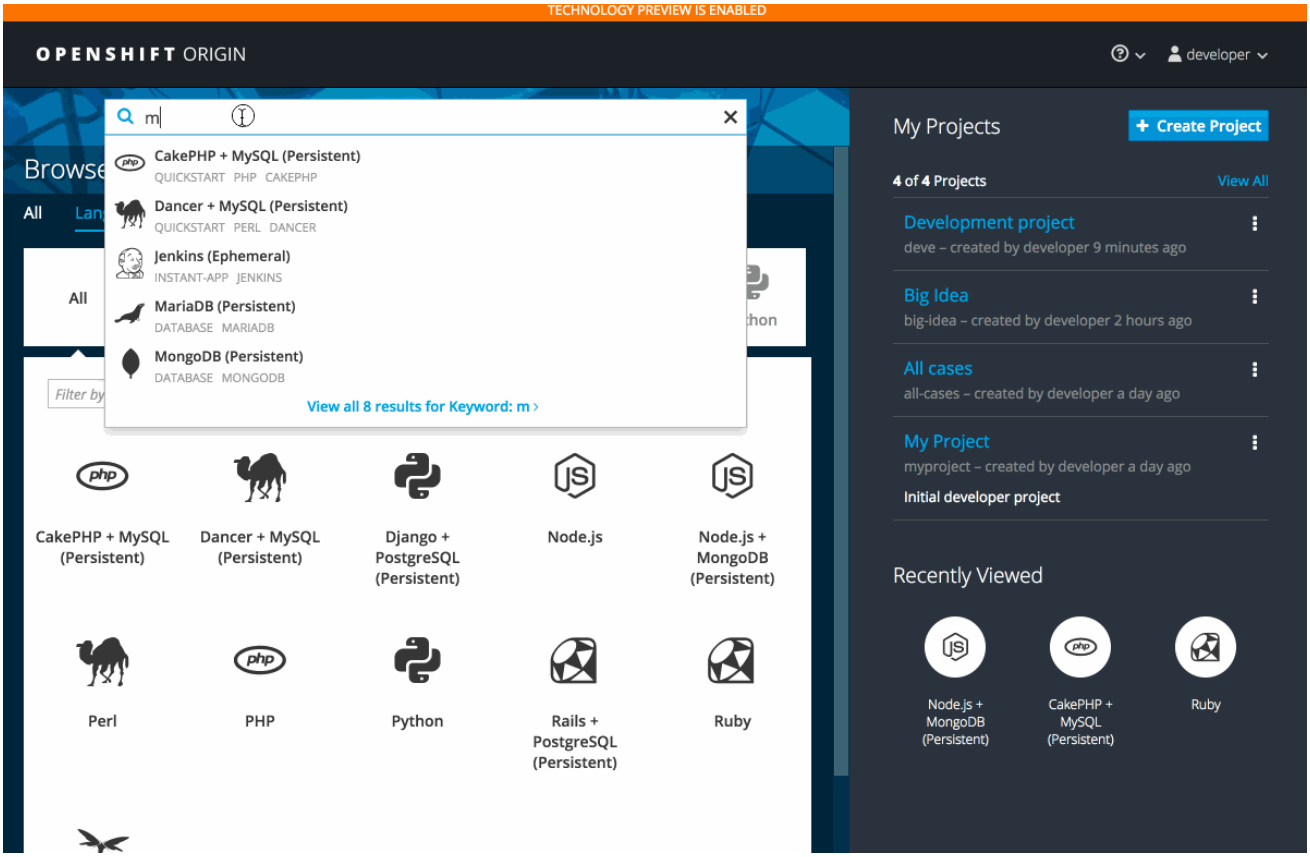
2.3.10.2. Initial Experience (Technology Preview)

In OpenShift Container Platform 3.6, a better initial user experience (currently in Technology Preview and not for production workloads) is introduced, motivated by service catalog. This includes:

- A task-focused interface.
- Key call-outs.
- Unified search.
- Streamlined navigation.

2.3.10.3. Search Catalog (Technology Preview)

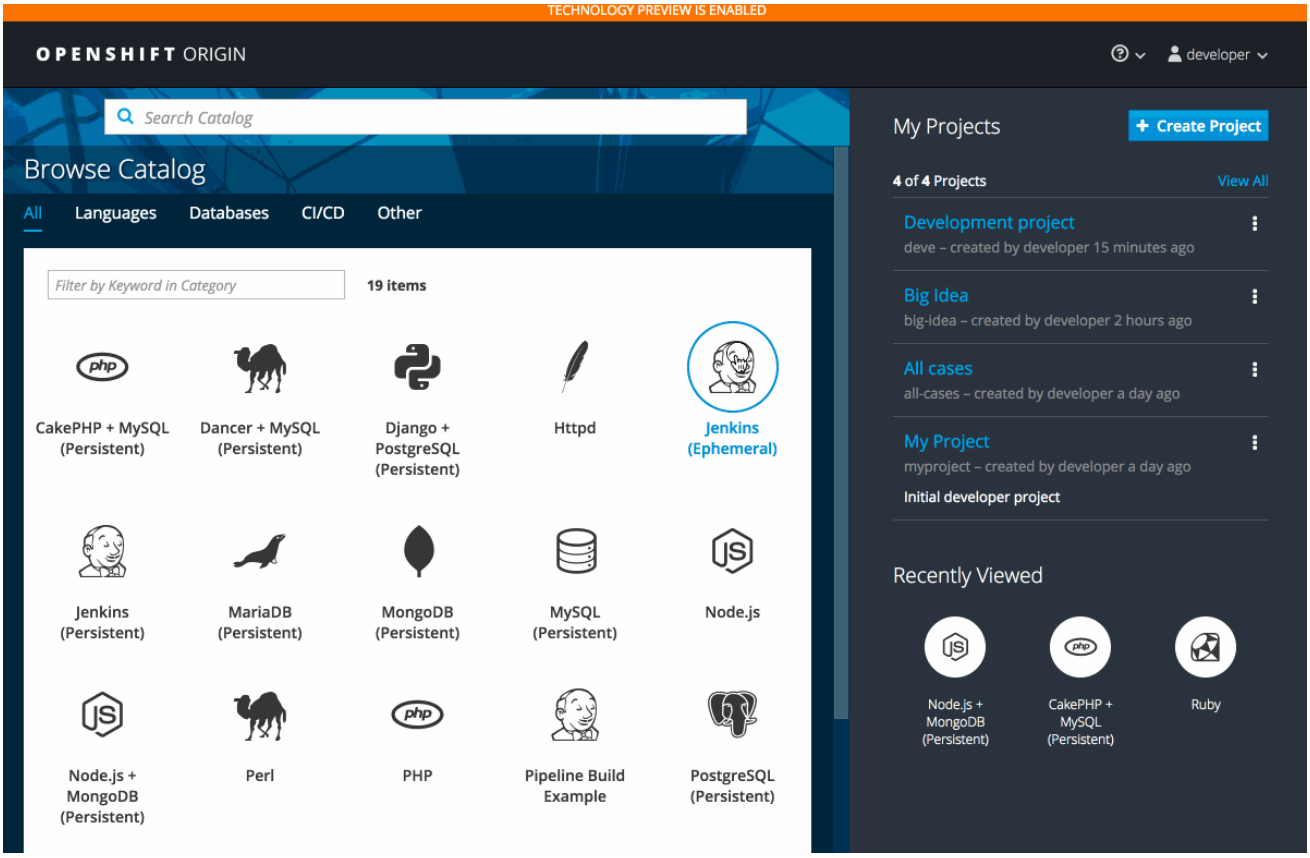
The search catalog feature (currently in Technology Preview and not for production workloads) provides a single, simple way to quickly get what you want.



2.3.10.4. Add from Catalog (Technology Preview)

The add from catalog feature (currently in Technology Preview and not for production workloads) allows you to provision a service from the catalog.

Select the desired service, then follow prompts for your desired project and configuration details.



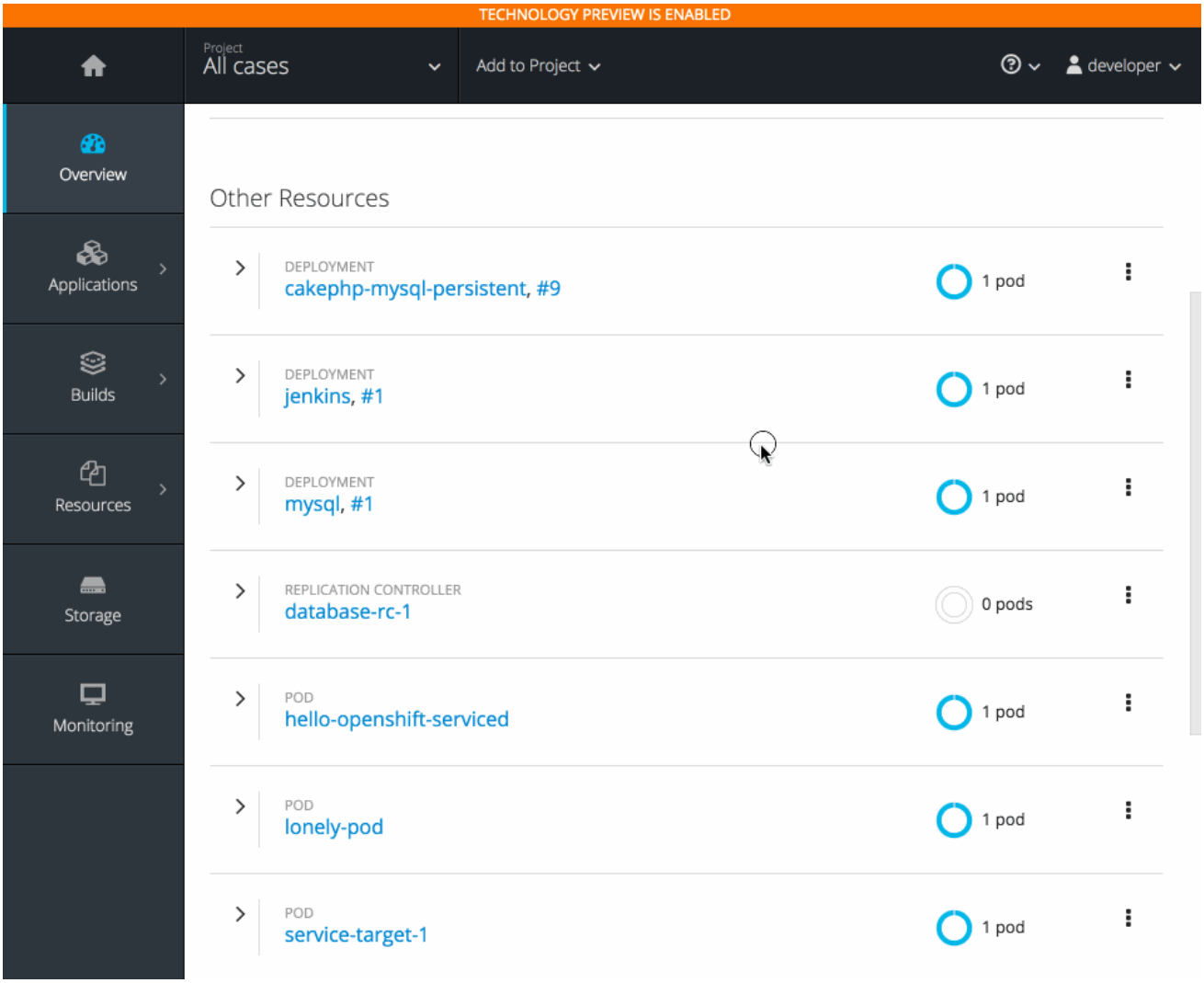
2.3.10.5. Project Overview Redesign

In OpenShift Container Platform 3.6, the Project Overview was resigned based on feedback from customers.

In this redesign, there are three focused views:

- Applications
- Pipelines
- Resource types

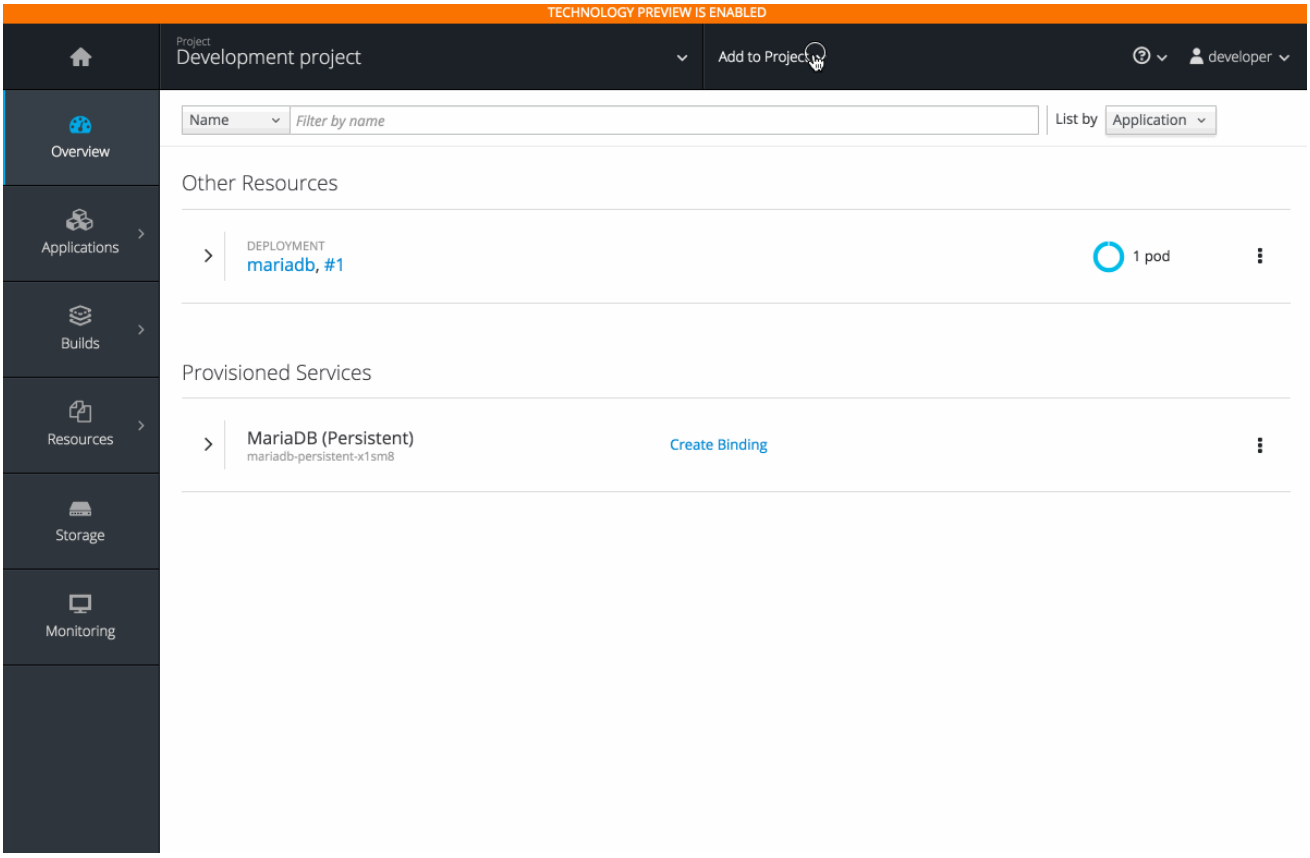
There are now more contextual actions and rolled up metrics across multiple pods.



2.3.10.6. Add to Project (Technology Preview)

The add to project feature (currently in Technology Preview and not for production workloads) allows you to provision a service without having to leave the Project Overview.

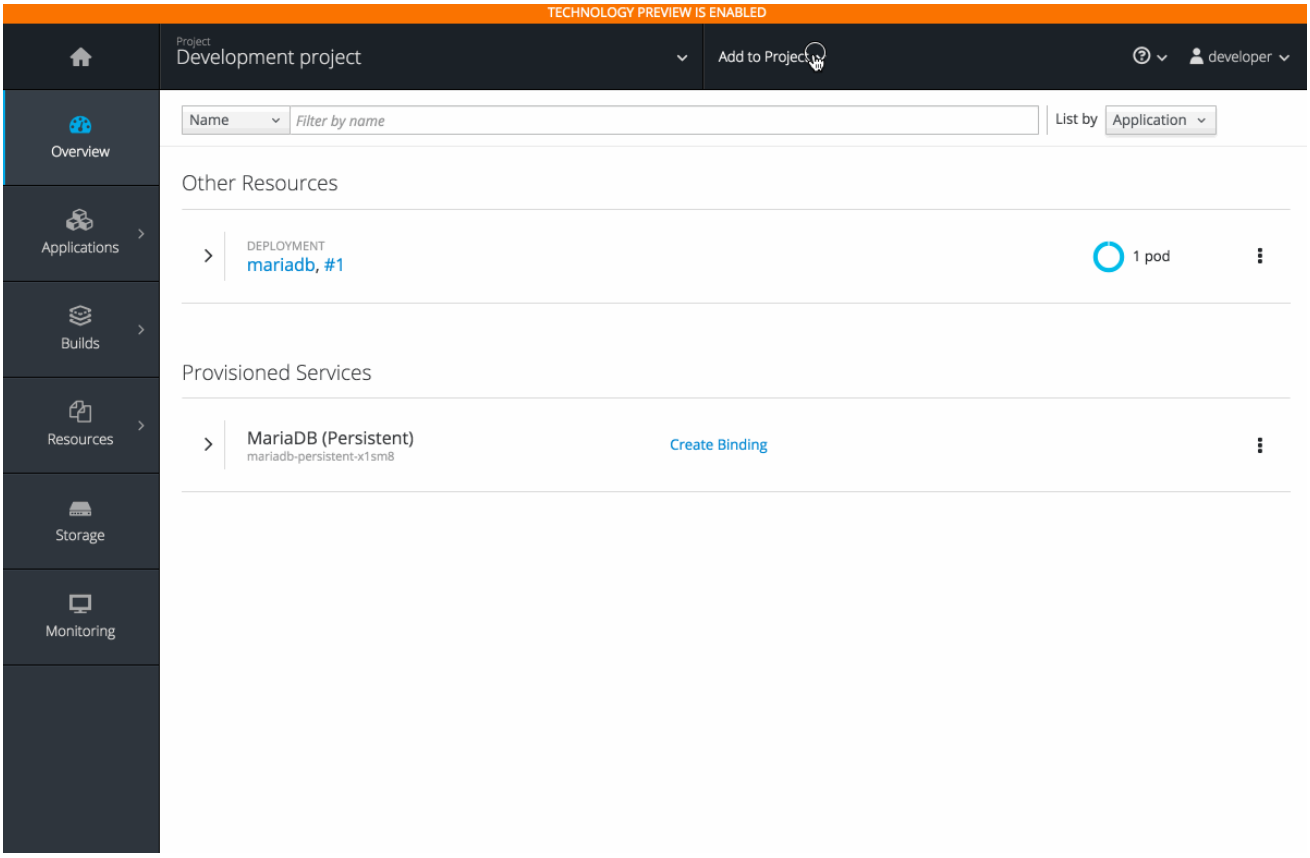
When you go directly to the catalog from project, the context is preserved. You can directly provision, then bind.



2.3.10.7. Bind in Context (Technology Preview)

The bind in context feature (currently in Technology Preview and not for production workloads) allows you to provision a service and bind without having to leave the Project Overview.

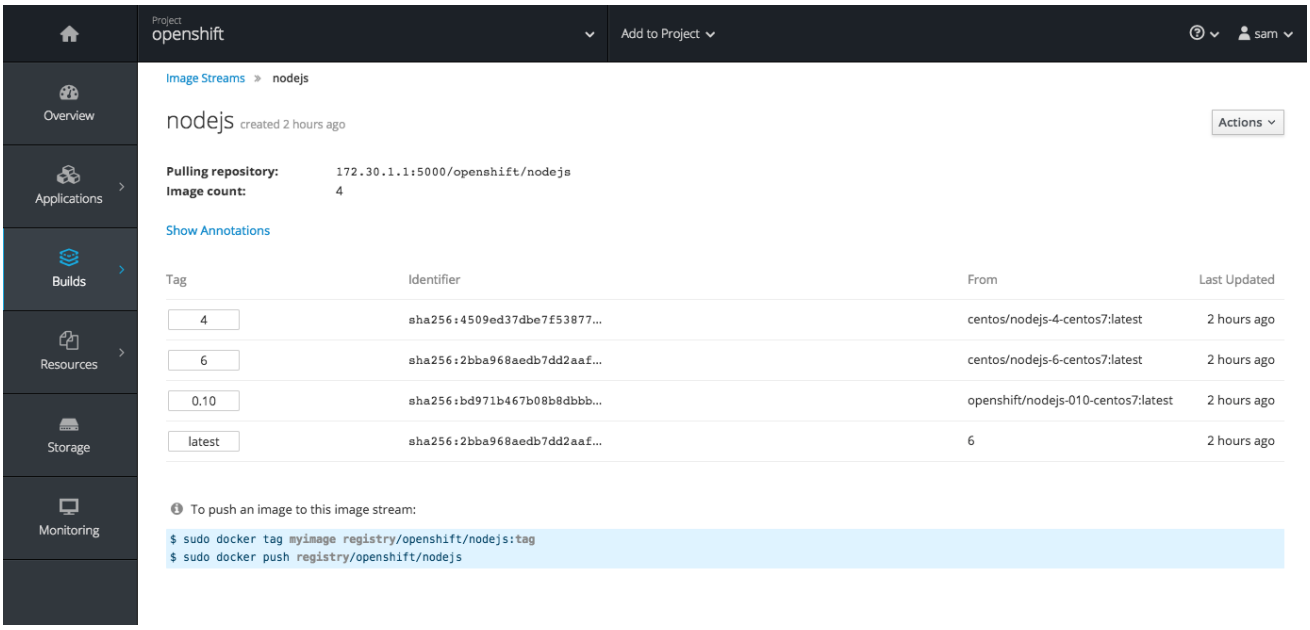
- Select deployment and initiate a bind.
- Select from bindable services.
- Binding is created and the user stays in context
- See relationships between bound applications and services in the Project Overview section.



2.3.10.8. Image Stream Details

In OpenShift Container Platform 3.6, additional details are provided about image streams and their tags.

This feature leverages Cockpit views from image streams. It details tags and provide information about each.



2.3.10.9. Better Messages for Syntax Errors in JSON and YAML Files

With OpenShift Container Platform 3.6, better messages for syntax errors in JSON and YAML files are provided. This includes details of the syntax problem and the line number containing the issue.

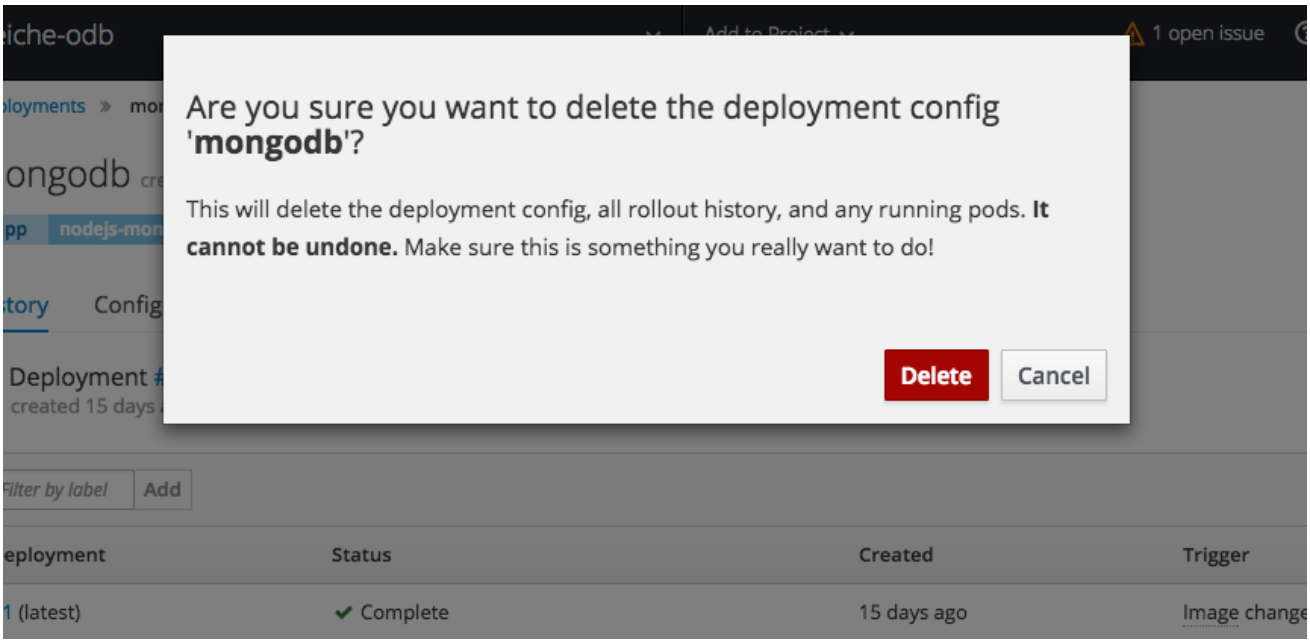
This feature validates input on commands such as **`oc create -f foo.json`** and **`oc new-app -f template.yaml`**. For example:

```
$ oc create -f dc.json
error: json: line 27: invalid character 'y' looking for beginning of value
```

2.3.10.10. Cascading Deletes

When deleting a resource, this feature ensures that all generated or dependent resources are also deleted.

For example, when selecting a deployment configuration and deleting will delete the deployment configuration, deployment history, and any running pods.



2.3.10.11. Other User Interface Changes

- Pod details now should show information about init containers.
- You can now add or edit environment variables that are populated by data in secrets or configuration maps.
- You can now create cluster-wide resources from JSON and YAML files.
- There is now an alignment of notification designs.

2.4. NOTABLE TECHNICAL CHANGES

OpenShift Container Platform 3.6 introduces the following notable technical changes.

Use the Ansible Version Shipping with OpenShift Container Platform
OpenShift Container Platform 3.6 and 3.7 were developed and tested using Ansible 2.3, which ships in the OpenShift Container Platform channels. Subsequently, the RHEL 7 Extras channel added Ansible 2.4, which has known issues with OpenShift Container Platform 3.6 and 3.7. If you experience any problems with the installer, downgrade to Ansible 2.3 by running **`yum downgrade ansible-2.3\*`**
See BZ#1575063 for additional information.

Payment Card Industry Data Security Standard (PCI DSS) Compliance

Red Hat has worked with a PCI DSS Qualified Assessor (QSA) and has determined that OpenShift Container Platform running on either Red Hat Enterprise Linux or Red Hat Enterprise Linux Atomic Host could be deployed in a way that it would pass a PCI assessment. Ultimately, compliance and validation is the responsibility of the organization deploying OpenShift Container Platform and their assessor. Implementation of proper configuration, rules, and policies is paramount to compliance, and Red Hat makes no claims or guarantees regarding PCI assessment.

Federation Decision Deliberation

In the upstream federation special interest group (SIG), there are two primary ideas being discussed. The current control plane model is an intelligent controller that duplicates API features and functions at a high level. The client is agnostic and the controller handles the inter-cluster relationships, policy, and so on. The control plane model may be difficult to maintain.

In the client model, multiple controllers would exist for various features and functions, and the client would maintain the intelligence to understand how to affect change across clusters. Red Hat is currently soliciting feedback on these two models. Customers, partners, and community members are encouraged to participate in the upstream SIGs.

DNS Changes

Prior to OpenShift Container Platform 3.6, cluster DNS was provided by the API server running on the master and the use of **dnsmasq** could be disabled by setting **`openshift_use_dnsmasq=false`**. Starting with OpenShift Container Platform 3.6, the use of **dnsmasq** is now mandatory and upgrades will be blocked if this variable is set to false.

Also, when upgrading to version 3.6, the playbooks will configure the node service to serve DNS requests on **`127.0.0.1:53`** and **dnsmasq** will be reconfigured to route queries for **`cluster.local`** and **`in-addr.arpa`** to **`127.0.0.1:53`** rather than to the Kubernetes service IP. Your node must not run other services on port 53. Firewall rules exposing port 53 are not necessary, as all queries will originate from the local network.

Deprecated API Types

The **`ClusterPolicy`**, **`Policy`**, **`ClusterPolicyBinding`** and **`PolicyBinding`** API types are deprecated. Users will need to switch any interactions with these types to instead use **`ClusterRole`**, **`Role`**, **`ClusterRoleBinding`**, or **`RoleBinding`** as appropriate. The following **`oc adm policy`** commands can be used to help with this process:

```
add-cluster-role-to-group
add-cluster-role-to-user
add-role-to-group
add-role-to-user
remove-cluster-role-from-group
remove-cluster-role-from-user
remove-role-from-group
remove-role-from-user
```

The following **`oc create`** commands can also help:

```
clusterrole
clusterrolebinding
role
rolebinding
```

The use of **`oc create policybinding`** is also deprecated and no longer a perquisite for creating a **`RoleBinding`** to a **`Role`**.

OpenShift Resources Registered to API groups

Custom roles that reference OpenShift resources should be updated to include the appropriate API groups.

Ambiguous CIDR Values Rejected

OpenShift Container Platform will now reject **`EgressNetworkPolicy`**, **`ClusterNetwork`**, **`HostSubnet`**, and **`NetNamespace`** objects with ambiguous CIDR values. Before, an **`EgressNetworkPolicyRule`** such as the following would be interpreted as "allow to `192.168.1.**0**/**24**`".

```
type: Allow
to:
  cidrSelector: 192.168.1.15/24
```

However, the user most likely meant "allow to 192.168.1.**15**/**32**". In OpenShift Container Platform 3.6, trying to create such a rule (or to modify an existing rule without fixing it) will result in an error.

The same validation is also now performed on CIDR-valued fields in **`ClusterNetwork`**, **`HostSubnet`**, and **`NetNamespace`** objects, but these are normally only created or modified by OpenShift Container Platform itself.

Volumes Removed at Pod Termination

In prior versions, pod volumes remained attached until the pod resource was deleted from the master. This prevented local disk and memory resources from being reclaimed as a result of pod eviction. In OpenShift Container Platform 3.6, the volume is removed when the pod is terminated.

Init Containers

Pod authors can now use init containers to share volumes, perform network operations, and perform computation prior to the start of the remaining containers.

An init container is a container in a pod that is started before the pod's application containers are started. Init containers can also block or delay the startup of application containers until some precondition is met.

Pod Tolerations and Node Taints No Longer Defined in Annotations

Pod tolerations and node taints have moved from annotations to API fields in pod specifications (PodSpec) and node specification (NodeSpec) files, respectively. Pod tolerations and node taints that are defined in the annotations will be ignored. The annotation keys **`scheduler.alpha.kubernetes.io/tolerations`** and **`scheduler.alpha.kubernetes.io/taints`** are now removed.

Router Does Not Allow SSLv3

The OpenShift router will no longer allow SSLv3 (to prevent the POODLE attack). No modern web browser should require this.

Router Cipher List Updates

The router cipher list has changed to reflect the current *intermediate* cipher suite recommendations from Mozilla. It is now also possible to set the cipher suite explicitly, or choose from a list of named preset security levels.

NetworkPolicy Objects Have NetworkPolicy v1 Semantics from Kubernetes 1.7

When using the **`redhat/openshift-ovs-networkpolicy`** plug-in, which is still in Technology Preview, **`NetworkPolicy`** objects now have the **`NetworkPolicy`** v1 semantics from Kubernetes 1.7. They are still in the **`extensions/v1beta1`** API group; the new **`networking.k8s.io/v1`** API group is not yet available.

In particular, the **net.beta.kubernetes.io/network-policy** annotation on namespaces to opt in to isolation has been removed. Instead, isolation is now determined at a per-pod level, with pods being isolated if there is any **NetworkPolicy** whose **spec.podSelector** targets them. Pods that are targeted by **NetworkPolicies** accept traffic that is accepted by any of the **NetworkPolicies** (and nothing else), and pods that are not targeted by any **NetworkPolicy** accept all traffic by default.

To preserve compatibility when upgrading:

1. In namespaces that previously had the **DefaultDeny** annotation, you can create equivalent v1 semantics by creating a **NetworkPolicy** that matches all pods but does not allow any traffic:

```
kind: NetworkPolicy
apiVersion: extensions/v1beta1
metadata:
  name: default-deny
spec:
  podSelector:
```

   This will ensure that pods that are not matched by any other **NetworkPolicy** will continue to be fully-isolated, as they were before.

2. In namespaces that previously did not have the **DefaultDeny** annotation, you should delete any existing **NetworkPolicy** objects. These would have had no effect before, but with v1 semantics they might cause some traffic to be blocked that you did not intend to be blocked.

Metadata volumeSource Now Deprecated

The metadata **volumeSource** is now deprecated for multiple releases and will be removed in OpenShift Container Platform 3.7.

Breaking API Change

Unless explicitly documented otherwise, API fields containing lists of items no longer distinguish between null and **[]**, and may return either null or **[]** regardless of the original value submitted.

Atomic Command on Hosts

When using system containers with OpenShift Container Platform, the **atomic** command on hosts must be **1.17.2** or later.

Containers Run Under Build Pod's Parent cgroup

Containers launched by the build pod (the s2i assemble container or the **docker build** process) now run under the build pod's parent cgroup.

Previously, the containers had their own cgroup and the memory and CPU limits were mirrored from the pod's cgroup limits. With this change, the secondary containers will now be sharing the memory limit that is consumed by the build pod, meaning the secondary containers will have slightly less memory available to them.

SecurityContextConstraints Available via Groupified API

SecurityContextConstraints are now also available via a groupified API at ***/apis/security.openshift.io/v1/securitycontextconstraints***. They are still available at ***/api/v1/securitycontextconstraints***, but using the groupified API will provide better integration with tooling.

Openshift Volume Recycler Now Deprecated

Openshift Volume Recycler is being deprecated. Anyone using recycler should use dynamic provision and volume deletion instead.

2.5. BUG FIXES

This release fixes bugs for the following components:

Authentication

- Nested groups now sync between OpenShift Container Platform and Active Directory. It is common to have nested groups in Active Directory. Users wanted to be able to sync such groups with OpenShift Container Platform. This feature was always supported, but lacked any formal documentation and was difficult to discover. Documentation is now added. (**BZ#1437324**)

Builds

- When a build is started from a webhook, the server response does not contain a body Therefore, the CLI cannot easily determine the generation of the created build, and cannot report it to the user. Change webhook response to contain the created build object in the body. The CLI can now report the correct build generation when created. (**BZ#1373441**)

- Build durations are recorded as part of a storage hook. Build duration is sometimes calculated incorrectly and reported with an invalid value. Calculate build duration when recording build time of build completion. As a result, build durations are reported correctly and align with the build start and completion times. (**BZ#1443687**)

- The code was not setting the status reason and status message for certain failures. Therefore, there were missing status reasons and status messages for certain failures. With this bug fix, code was added that sets the status reason and status message and the status reason and message are now set. (**BZ#1436391**)

- A debug object type is used when high levels of logging are requested. Client code did not anticipate the alternative object type and experienced a typecast error. With this bug fix, the client code is updated to handle the debug object type. The typecast error will not occur and builds now proceed as expected. (**BZ#1441929**)

- When resources were specified in the build default configuration, the resource values were not applied to the build pods. They were only applied to the build object. Builds ran without the default resource limits being applied to them because the pod was created before the build was updated with the default resource limits. With this bug fix, the build resource defaults are applied to the build pod. Build pods now have the default resource limits applied, if they do not already specify resource limits. (**BZ#1443187**)

- The **`new-app`** circular dependency code did not account for **`BuildConfig`** sources pointing to the **`ImageStreamImage`** type. As a result, an unnecessary warning was logged about not being able to follow the reference type **`ImageStreamImage`**. This bug fix enhances the **`new-app`** circular dependency code to account for the **`ImageStreamImage`** type. The unnecessary warning no longer appears. (**BZ#1422378**)

Command Line Interface

- Previously, pod headers were only being printed once for all sets of pods when listing pods from multiple nodes. Executing **`oc adm manage-node <node-1> <node-2> … --evacuate --dry-run`** with multiple nodes would print the same output multiple times (once per each specified node). Therefore, users would see inconsistent or duplicate pod information. This bug fix resolves the issue. (**BZ#1390900**)

- The **`--sort-by`** in the **`oc get`** command fails when any object in the list contains an empty value in the field used to sort, causing a failure. With this bug fix, empty fields in **`--sort-by`** are

now correctly handled. The output of **`oc get`** is printed correctly and empty fields are considered in sorting. (BZ#1409878)

- A Golang issue (in versions up to 1.7.4) adds an overhead of around four seconds to the TLS handshake on macOS. Therefore, the **oc** calls time out intermittently on macOS. This bug fix backported the existing fix to 1.7.5 and upgraded the Golang that we use to build **oc** to that version. The TLS handshake time is now reduced by about four seconds on macOS. (BZ#1435261)

- When the master configuration specified a default **`nodeSelector`** for the cluster, test projects created by **`oc adm diagnostics`** NetworkCheck got this **`nodeSelector`** and, therefore, the test pods were also confined to this **`nodeSelector`**. NetworkCheck test pods could only be scheduled on a subset of nodes, preventing the diagnostic covering the entire cluster; in some clusters, this might even result in too few pods running for the diagnostic to succeed even if the cluster health is fine. NetworkCheck now creates the test projects with an empty **`nodeSelector`** so they can land on any schedulable node. The diagnostic should now be more robust and meaningful. (BZ#1459241)

Installer

- OpenShift Ansible facts were splitting a configuration parameter incorrectly. Therefore, invalid **`NO_PROXY`** strings were generated and inserted into user ***sysconfig/docker*** files. The logic that generates the NO_PROXY strings was reviewed and fixed. Valid Docker **`NO_PROXY`** settings are enerated and inserted into the ***sysconfig/docker*** file now. (1414748)

- The OpenShift CA redeployment playbook (***playbooks/byo/openshift-cluster/redeploy-openshift-ca.yml***) would fail to restart services if certificates were previously expired. Service restarts are now skipped within the OpenShift CA redeployment playbook when expired certificates are detected. Expired cluster certificates may be replaced with the certificate redeployment playbook (***playbooks/byo/openshift-cluster/redeploy-certificates.yml***) once the OpenShift CA certificate has been replaced via the OpenShift CA redeployment playbook. (1452367)

- Previously, installation would fail in multi-master environments in which the load balanced API was listening on a different port than that of the OpenShift API and console. This difference is now accounted for and and the master loopback client configuration is configured to interact with the local master. (1454321)

- A readiness probe is introduced with OpenShift Container Platform 3.6, but the timeout threshold was not high enough. This bug fix increases the timeout threshold. (1456139)

- Elasticsearch heap dump should not be written to the root partition. Specify a location to write a heap dump other than the root partition. (1369914)

- Previously, the upgrade playbooks would use the default **`kubeconfig`**, which may have been modified since creation to use a non-admin user. Now the upgrade playbooks use the admin **`kubeconfig`**, which avoids this problem. (1468572)

- A fix for a separate PROXY related issue was merged. Therefore, various proxy related operations began to fail.A correct fix for the original PROXY-related issue was merged and functionality is now restored. (1470165)

- **`NO_PROXY`** setting logic was incorrectly indented in the openshift-ansible facts module, causing **`NO_PROXY`** settings to always be generated and added to service configuration files. The logic indentation was moved into the correct conditional. (BZ#1468424)

- Image streams now reference the DNS hostname of **`docker-registry.default.svc:5000`**, which allows the installer to ensure that the hostname is appended to **`NO_PROXY`** environment variables so image pushes work properly in an environment that requires a proxy. (**BZ#1414749**)

- Starting with OpenShift Container Platform 3.4, the software-defined networking (SDN) plug-ins no longer reconfigure the docker bridge maximum transmission unit (MTU), rather pods are configured properly on creation. Because of this change, non-OpenShift containers may have a MTU configured that is too large to allow access to hosts on the SDN. The installer has been updated to align the MTU setting for the docker bridge with the MTU used inside the cluster, thus avoiding the problem. (**BZ#1457062**)

- As part of the RFE to be able to label **`PersistentVolume`** (PV) for **`PersistentVolumeClaim`** (PVC) selectors, the default PVC selector was set to null but should have been an empty string. This caused the playbook to fail if the user did not provide a label. This fix leaves the default label blank, allowing the playbook to run to completion if the user does not provide a PV label. (**BZ#1462352**)

- Metrics were not consistently able to install correctly when using a non-root user. This caused the playbook to fail due to lack of permissions, or files not visible due to permissions. With this fix, any local action within the metrics role added a **`become: false`** so it ensured it was using the local actions as the same user running the playbook. The playbook no longer fails to complete due to permissions. (**BZ#1464977**)

- This feature grants the ability to provide **`PersistentVolume`** (PV) selectors for PVs created during installation. Previously when installing logging and metrics with the installer, a PV created for logging could be bound to a metrics PVC, creating confusion. Now you can provide a PV selector in your inventory when installing logging and metrics and the PVs created will contain the appropriate label so that the generated PVCs will correctly bind. (**BZ#1442277**)

- Hosts missing an OpenSSL python library caused large serial numbers to not be parsed using the existing manual parser workaround for missing OpenSSL libraries. This bug fix updates the manual parser to account for certificate formats with large serial numbers. As a result, certificates with large serials on hosts missing the OpenSSL python library can now be parsed, such as during certificate expiration checking or certificate redeployment. (**BZ#1464240**)

- The master configuration parameter **`serviceAccountConfig.limitSecretReferences`** may now be set via the installation playbooks by setting the variable **`openshift_master_saconfig_limitsecretreferences`** to **`true`** or **`false`**. (**BZ#1442332**)

- Older logic was missing a condition in which the **`systemd`** unit files should be reloaded, causing updated or changed service unit files to not be identified. This bug fix updates the Ansible installer master and node roles to ensure the **`reload system units`** action is triggered. As a result, updated service unit files are correctly detected and users no longer receive a “Could not find the requested service” error anymore. (**BZ#1451693**)

- An incorrect check for python libraries was used for the metrics role, causing playbooks to fail when checking whether **python2-passlib** was installed. This bug fix updates the query for checking the availability of the library. As a result, the playbook no longer incorrectly fails when **python2-passlib** is installed. (**BZ#1455310**)

- The default persistent volume (PV) selector for the logging persistent volume claim (PVC) generation was **`None`** and was being interpreted as a variable. This caused the playbook to fail because it could not find a variable of the name **`None`**. This bug fix updates the default to be **`''`**.

As a result, the playbook is able to correctly run to completion when not providing a PV selector. (BZ#1463055)

- The installer now creates a default **`StorageClass`** whenever AWS or GCE cloud providers are configured, allowing for out-of-the-box dynamic volume creation. (BZ#1393207)
- The example inventory files have been amended to illustrate all available audit logging configuration options. (BZ#1447774)
- The default templates have been updated to the latest available for OpenShift Container Platform 3.6. (BZ#1463553)
- Previously, all certificates for an OpenShift cluster have a validity of one year. This was not practical for enterprise-level installations. The installer tool was modified to allow configuration of certificates, meaning the validity period can be extended. (BZ#1275176)
- The service accounts that belonged in the **`openshift-infra`** namespace were being created in **`default`** after a different fix to create them before role bindings. Therefore, pods were not able to find their SA for running. With this bug fix, SAs are created in the correct namespace and pods are able to start. (BZ#1477440)

Image

- When Kubernetes settings are updated, Jenkins is restarted and reloaded. This causes all of the configurations to be reloaded, including OpenShift Container Platform settings. Therefore, **`credentialsId`** becomes null and causes NPE's to be thrown, stopping the watchers, which can not recover. When Kubernetes is updated, synchronization with OpenShift Container Platform is stopped. With this bug fix, the getter for **`credentialsId ` check for null, and returns `""`** if found. Kubernetes can now be updated without NPE. (BZ#1451223)
- Proxy values are logged during builds. Previously, proxy values that contained user credentials were exposed to anyone who can view build logs. With this bug fix, credentials that are part of proxy values (for example, **`http://user:password@proxy.com`**) will be redacted from the proxy value being logged. Proxy credentials are now no longer exposed in build logs. (BZ1366795)
- Previously, the PHP **`latest`** image stream tag did not point to the latest available PHP image (7.0). Therefore, users of the **`latest`** image stream tag did not get the most recent PHP image available. With this bug fix, the **`latest`** tag is updated to point to the most recent image stream tag for PHP. Now, users who select the **`latest`** tag will get the PHP 7.0 image. (BZ#1421982) (BZ1421982)

Image Registry

- There was a logic error in how weak and strong references were identified when searching images eligible for pruning. Therefore, some images having both strong and weak references in pruning graph could be removed during pruning. The logic responsible for finding which images have strong references is now fixed. Pruning now correctly recognizes and prunes images. (BZ440177)
- Only aliases within single Image streams were being resolved. If an update was done to the source image, cross-image-stream aliases were not resolved properly, pointing to the old image. This bug fix forbids the creation of cross-image-stream aliases. Users creating a cross-image-stream alias now get an error. (1435588)

Kubernetes

- Previously, if the pod restarted due to exceeding **`failureThreshold`** on a probe, the restarted pod was only allowed a single probe failure before being restarted, regardless of the **`failureThreshold`** value. This caused restarted pods not to get the expected number of probe attempts before being restarted. This fix allows the reset the failure counter when the pod is restarted, therefore the restarted pod gets **`failureThreshold`** attempts for the probe to succeed. (**BZ#1455056**)

- When attempting to connect to an **`etcd`** server to acquire a leader lease, the master controllers process only tried to reach a single **`etcd`** cluster member even if multiple are specified. If the selected **`etcd`** cluster member is unavailable, the master controllers process is not able to acquire the leader lease, which means it will not start up and run properly. This fix enables attempts to connect to all of the specified **`etcd`** cluster members until a successful connection is made, and as a result the master controllers process can acquire the leader lease and start up properly. (**BZ#1426183**)

- Previously, the same error message was being output for each node in a cluster. With this fix, the error will include its message and its repeat count. (BZ#1462345)

Logging

- A change in the **`authproxy`** was keeping it from finding dependent files, causing the **`authproxy`** to terminate. With this fix, environment variables were added to the **`deploymentconfig`** with the correct path to the files. As a result, the **`openshift-auth-proxy`** finds dependent files and starts correctly. (BZ#1439451)

- The Aggregated Logging diagnostic was not updated to reflect updates made to logging deployment. Therefore, the diagnostic incorrectly reported errors for an unnecessary Service Account and (if present) the **`mux`** deployment. With this bug fix, these errors are no longer reported. In addition, warnings about missing optional components were all downgraded to Info level. The diagnostic no longer needlessly alarms the user for these issues. (**1421623**)

Web Console

- Previously, there were issues viewing logs for pods with multiple containers caused, especially when switching between containers. You should now be able to switch between container logs without issue and the Follow link should work as expected. (**1421287**)

- It was difficult to find the underlying reason for a failed deployment from the project overview. The overview will now link to the Events page in these scenarios, which typically contains useful information about what went wrong. (**1365525**)

- Previously, the OpenShift namespace appeared at the top of the list of namespaces for the image stream tag picker, which was confusing in long lists of namespaces if the user was expecting to find it alphabetically in the drop-down menu. This happened because the image stream tag picker was adding the OpenShift namespace to the list after the list was already sorted. The list of namespaces the user can pick from is now sorted after the OpenShift namespace is added to the list. Now the list of namespaces a user can pick from, when selecting an image stream tag for build configuration, options have OpenShift sorted alphabetically with the other namespaces the user can access. (**BZ#1436819**)

- The web console now better uses the screen space when displaying services. (**BZ#1401134**)

Metrics

- Previously, partitions in the **`metrics_idx`** table cause Cassandra to write into the table packets that are as large as 496 MB and even 700 MB, causing client requests to Hawkular Metrics to

fail. A workaround of changing the compaction strategy for the **metrics_idx** table from **LCS** to **STCS** was created, leading to a new, persisting Hawkular image. (BZ#1422271)

- The internal metadata around the Cassandra schema was out of date, leading to the data being a mix of old and new schema information. The version has been updated. (BZ#1466086)

Networking

- Previously, the OpenShift Container Platform node proxy did not support using a specified IP address. This prevented correct operation on hosts with multiple network interface cards. The OpenShift Container Platform node process already accepts a **--bind-address=<ip address>:<port>** command-line flag and **bindAddress:** configuration file option for the multiple network interface card case. The proxy functionality is now fixed to respect these options. When **--bind-address** or **bindAddress** are used, the OpenShift Container Platform node proxy should work correctly when the OpenShift Container Platform node host has multiple network interface cards. (1462428)

- Previously, when an IP address was re-used, it would be generated with a random MAC address that would be different from the previous one, causing any node with an ARP cache that still held the old entry for the IP to not communicate with the node. Now, generating the MAC address deterministically from the IP address now results in a re-used IP address always having the same MAC address, so the ARP cache can not be out of sync. This ensures the traffic will now flow. (BZ#1451854)

- Previously, the VNID allow rules were removed before they were really unused. This meant that if there were still pods in that namespace on the node, they could not communicate with one another. The way that the tracking is done was changed so to avoid the edge cases around pod creation or deletion failures. This meant that the VNID tracking does not fail, so traffic flows. (BZ#1454948)

- Previously, running **oc adm diagnostics NetworkCheck** would result in a timeout error. Changing the script to run from the pod definition fixed the issue. (BZ#1421643)

- Previously, using an F5 router did not work with re-encrypt routes. Adding the re-encrypt routes to the same vserver fixed the problem. (BZ#1431655)

- Previously, there was a missing **iptables** rule to block **INVALID** packets, causing packets to escape cluster. The missing rule was added missing rule resulting in no more leaks. (BZ#1438762)

- Minor enhancements have been made to the **iptables** proxier to reduce node CPU usage when many pods and services exist. (BZ#1387149)

- Previously, some fragmented IP packets were mistakenly dropped by **openshift-node** instead of being delivered to pods, causing large UDP and TCP packets to have some or all fragments dropped instead of being delivered. The relevant fragments are now correctly evaluated and sent to their destination, meaning large UDP and TCP packets should now be delivered to the intended pods in the cluster. (BZ#1419692)

- Previously, the ARP cache was not compatible with OpenShift clusters with a large number of routes (more than the default value of **1024**). The default has been changed to **65536**, meaning clusters with many routes will function. (BZ#1425388)

- Previously, using **oc expose svc** picked up the service port instead of the target port, meaning the route would not work. The command is now picked up from the port number. (BZ#1431781)

- Previously, the hybrid proxy was not correctly protecting access to internal data. This meant that, when it was enabled, it could terminate the **openshift-node** process with a runtime panic due to concurrent data accesses. As a fix, all internal data is correctly protected against concurrent access, meaning the **openshift-node** process should no longer panic with concurrent data access failures when the hybrid proxy is enabled. (**BZ#1444125**)

- Previously, after adding the **netnamespace.network.openshift.io/multicast-enabled=true** annotation to **netnamespace**, it will create one open-flow rule in table 110, but the annotation is still there after deletion. The problem has now been fixed. (**BZ#1449058**)

- Previously, the CLI help text was not clear about what worked on the F5 versus the HAProxy routers. The CLI help text has been updated with clearer expectations. (**BZ#1427774**)

- Previously, having multiple node IP addresses reported in random order by node status. This led to the SDN controller picking up a random IP each time. IP stickiness is now maintained, meaning the IP is valid when chosen. (**BZ#1438402**)

- Previously, cluster-external traffic was handled incorrectly when using the Technology Preview **NetworkPolicy** plug-in. Pods could not connect to IP addresses outside the cluster. The issue has been resolved and external traffic now works correctly. (**BZ#1443765**)

- Previously, the code to set up multicast was not run when only one node was in the cluster, leading to multicast traffic dropping when on a single-node cluster. The rules have been changed so the multicast setup is performed for a single-node. (**BZ#1445501**)

- Previously, the initialization order of the SDN plug-in set up the event handler too late, causing early events to have no handler, so the SDN would panic. The SDN initialization has been re-ordered so that the event handler is in place before it can be called. (**BZ#1445504**)

- Previously, the iptables rules were logged at too low of a log level, causing the logs to fill with iptables noise. The level at which they are logged has changed. (**BZ#1455655**)

- Previously, the **NetworkPolicy** plug-in (currently in Tech Review) in OpenShift Container Platform 3.5 did not implement all features of **NetworkPolicy**. When using certain **NetworkPolicy** resources that used **PodSelectors**, pods would be accessible by pod IP, but not by service IP. These issues have been addressed. All connections that should be allowed by a **NetworkPolicy** are now allowed whether made directly (pod-to-pod) or indirectly via a service IP. (**BZ#1419430**)

REST API

- **maxScheduledImageImportsPerMinute** was previously documented as accepting **-1** as a value to allow unlimited imports. This would cause the cluster to panic. **maxScheduledImageImportsPerMinute** now correctly accepts **-1** as an unlimited value. Administrators who have set **maxScheduledImageImportsPerMinute** to an extremely high number as a workaround may leave the existing setting or now use **-1**. (**BZ#1388319**)

- Previously, deleting created resources from a project failed to delete the route and an error message was shown on the web console. The issue has been resolved. (**BZ#1452569**)

Routing

- This enhancement strips HTTP **Proxy** headers to prevent the **httpoxy** (**https://httpoxy.org/**) vulnerability. Applications behind the router are now protected from **httpoxy**. (**1469633**)

- Previously, when quickly adding then deleting a route using the CLI, routes are queued up to be processed, saving the request data in a store, then acts on them. The problem is the store is empty when the last request is popped, causing an issue. This bug fix resolves the issue. (BZ#1447928)

- This bug fixes the matching logic change, which made the trailing slashed inadvertently break, meaning that subpaths with trailing `/`s no longer worked. The code that matches them has been corrected. (BZ#1448944)

- Previously, the logic in the HAProxy router template did not account for **`Allow`** as **`InsecureEdgeTerminationPolicy`** for re-encrypt routes, because the cookie object was set as secure. Logic has been added to correctly tag the cookie as insecure when **`InsecureEdgeTerminationPolicy`** is **`Allow`** for re-encrypt routes. (BZ#1428720)

- Previously, the command to create a list of routes was incorrect, meaning the route statuses did not get deleted. The logic enumerating routes has been improved. (BZ#1429364)

- Previously, the script did not check the version of **`jq`** and does not populate its array of routes correctly, leading to the script failing when using **`-r`**. The fix was to check to make sure the user has an appropriate version of **`jq`** and populate the array of target routes properly. Then, the script correctly clears the routes specified of status messages. (BZ#1429398)

- Previously, the router template did not add **`option forwardfor`** to re-encrypt type routes, causing the **`X-Forwarded-For`** section of ***http header*** file to go missing. This bug fix adds **`option forwardfor`** in the router template for the re-encrypt type routes. Now the **`X-Forwarded-For`** section of the ***http header*** file will correctly populate. (BZ#1449022)

- Version 3.6 router introduced a new port named **`router-stats`**. This bug created an option for **`oc adm router`** command to allow a user to specify customized a router-stats port, such as **`--stats-port=1936`**, so that user could easily create an customized router. (BZ#1452019)

- This bug tracks the changing matching logic leading to trailing slashed inadvertently breaking, leading to subpaths with trailing `/`s no longer working. The code that matches them has been corrected. (BZ#1446627)

- This bug added the feature that using **`ROUTER_BACKEND_PROCESS_ENDPOINTS=shuffle`** will randomize the order of back-ends in the HAProxy configuration. With long running sessions and a router that reloads regularly, the first endpoint in the configuration may receive significantly more load than other back-ends. Setting the environment variable will randomize the order of the back-ends on every reload and, thus, help spread the load. (BZ#1447115)

Storage

- When an OpenShift node crashed before unmapping a RBD volume, the advisory lock held on the RBD volume was not released. This prevented other nodes from using the RBD volume till the advisory lock is manually removed. Now, if no RBD client is using the RBD volume, the advisory lock is removed automatically. Thus, the RBD volume can be used by other nodes without manually removing the lock. (BZ#1365867)

- Attach operations on AWS were slow because of duplicate API calls and frequent polling of volume information. In the latest version, the duplicate API calls are removed from the code and bulk polling of AWS volumes is implemented, to avoid API quota problems. (BZ#1392357)

- For persistent volumes, the default mount options provided by OpenShift were not customizable. Users can now tweak mount options for persistent volumes (including NFS and other volume types that support it) depending on their storage configuration. (BZ#1405307)

- The **`recycle`** reclaim policy is deprecated in favor of dynamic provisioning and it will be removed in future releases. (**BZ#1432281**)
- OpenStack Cinder v1 API got deprecated in recent OpenStack release. OpenShift now supports OpenStack v2 API. (**BZ#1427701**)
- In kubelet logs, a running pod was sometimes reported as *'cannot start, time out waiting for the volume'*. Because the kubelet's volumemanager reconstructor for actual state of world was running before the desired state of world was populated, which caused the pods in the actual state of world, to have incorrect volume information. This issue is now fixed. (**BZ#1444096**)
- The OpenStack Cinder StorageClass ignored availability zones because of an issue in the **`gophercloud/gophercloud`** library. OpenStack Cinder StorageClass now provisions volumes in the specified availability zone and fails if the specified availability zone does not exist. (**BZ#1444828**)
- When mounting volumes using a subpath, the subpath did not receive correct permissions. This issue is now fixed. (**BZ#1445526**)
- Volumes failed to detach after unmounting them from the node. Because Openshift did not attempt detach operation for pods that were completed (or terminated) but were not deleted from API server. Thus preventing reuse of volume in other pods. This bug is fixed and volumes for terminated or completed pods are detached automatically. (**BZ#1446788**)
- If the availability optional parameter was not provided for the OpenStack Cinder StorageClass, all Persistent Volumes provisioned for the Persistent Volume Claims that used the specified StorageClass were provisioned in the **`nova`** zone. Now, such Persistent Volumes are provisioned in an active zone where OpenShift has a node. (**BZ#1447568**)
- Pods failed to start, if they specified a file as a volume subPath to mount. This is now fixed. (**BZ#1451008**)
- OpenShift failed to attach disks to the Azure F-Series VMs. This issue is now fixed. (**BZ#1451039**)
- Previously, when a node stopped (or rebooted) the ones using EBS volumes were failing because the volume was not detached from the stopped node. Now the volume gets successfully unmounted and detached from node. (**BZ#1457510**)
- High OpenShift process CPU utilization is now fixed. (**BZ#1460280**)
- Previously, the **`AccessModes`** field of a PVC was ignored when a PV was dynamically provisioned for it. This caused users to receive a PV with inaccurate **`AccessModes`**. Now the dynamic provisioning of PVs with inaccurate **`AccessModes`** are not provisioned when PVCs ask for **`AccessModes`** that can't be satisfied by the PVs' underlying volume plugin. (**BZ#1462275**)
- Dynamically created Azure blob containers were accessible on public internet. This happened because the default access permissions for Persistent Volumes (PVs) were set to **`container`** which exposed a publically accessible URI. The container permissions are now set to **`private`** for provisioned Azure Volumes. (**BZ#1462551**)
- Sometimes, even after the PV and volume are proivisioned successfully, there was a failed volume creation event in the logs. This issue is now fixed. (**BZ#1395547**)
- This bug made it possible to specify multiple **`targetPortals`** to make use of iSCSI multipath, which is the de-facto standard in environments that use iSCSI. (**BZ#1376022**)

Upgrades

- Previously, when 3.1 version of **etcd** was available, the **etcd** RPM did not get upgraded to the version during the control plane upgrade. The playbook responsible for **etcd** upgrading is now extended and the **etcd** RPM (and **etcd** docker images) are properly upgraded to **3.1.***. (**BZ#1421587**)

- To minimize the attack surface for containers escaping namespace isolation, the label **svirt_sandbox_file_t** on **/var/lib/origin/openshift.local.volumes/** was removed. (**BZ#1450167**)

- Previously, when named certificates were added to ansible hosts file, and certificate redeploy playbook was run, certificates were not added to **master-config.yaml**. This issue is now fixed. (**BZ#1454478**)

- Previously, the certificate redeployment playbook would not update master configuration when named certificates were provided. Named certificates will now be replaced and master configuration will be updated during certificate redeployment. (**BZ#1455485**)

- The OpenShift upgrade got applied to all nodes if **openshift_upgrade_nodes_label** fits no label. Now the installer verifies the provided label and matches a set of hosts prior to upgrading. If the label does not match hosts, the upgrade would silently proceed with upgrading all nodes given the logic for creating the **oo_nodes_to_upgrade** group. (**BZ#1462992**)

- If the version of etcd used to produce the etcd backup was version 3.x the backup can only be loaded by etcd 3.x. This occurs when running etcd in a containerized install and the version of the rpm installed on the host differs from that running inside the container. We have updated the backup playbooks to use the version of etcd from within the container which ensures that a matching version of etcd is used. (**BZ#1402771**)

- Previously, the registry-console would use an older image version even after upgrade. Since registry-console was installed by default, the upgrade playbook did not update the registry-console deployment configuration to use the same version as docker-registry. This issue is now fixed. (**BZ#1421987**)

2.6. TECHNOLOGY PREVIEW FEATURES

Some features in this release are currently in Technology Preview. These experimental features are not intended for production use. Note the following scope of support on the Red Hat Customer Portal for these features:

Technology Preview Features Support Scope

The following new features are now available in Technology Preview:

- Require Explicit Quota to Consume a Resource
- Mount Options
- Automated installation of CloudForms 4.5 Inside OpenShift
- Installation of etcd, Docker Daemon, and Ansible Installer as System Containers
- Running OpenShift Installer as a System Container
- Service Catalog

- Ansible Service Broker
- Ansible Playbook Bundles (APB)
- Initial Experience
- Search Catalog
- Add from Catalog
- Add to Project
- Bind in Context
- Template Service Broker
- Service Catalog Experience in the CLI
- **`mux`**

The following features that were formerly in Technology Preview from a previous OpenShift Container Platform release are now fully supported:

- Init containers

The following features that were formerly in Technology Preview from a previous OpenShift Container Platform release remain in Technology Preview:

- Cron Jobs (formerly called Scheduled Jobs)
- Network Policy
- Kubernetes Deployments Support
- Pod Distribution Budgets
- **`StatefulSets`** formerly known as **`PetSets`**

2.7. KNOWN ISSUES

- When running an upgrade with **`` `--tags pre_upgrade ``**, the upgrade failed with:

  ```
  "the file_name '/usr/share/ansible/openshift-
  ansible/playbooks/common/openshift-cluster/upgrades/etcd/noop.yml'
  does not exist, or is not readable"
  ```

  (**BZ#1464025**)

- When de-provisioning an Ansible Playbook Bundle (APB) via the **`service-catalog`** and **`ansible-service-broker`**, the **Provisioned Service** entry will linger for longer than the assets created by the APB. The service itself will have correctly been de-provisioned. This is due to the service catalog eventually confirming with the Ansible Service Broker that the service is actually gone. It was patched upstream. (**BZ#1475251**)

- The Ansible playbooks for running OpenShift pre-installation and health checks may have unexpected side-effects, as they have dependencies on code from the installer for configuring

hosts. This may result in changes to configuration for yum repositories, Docker, or the firewall for hosts where configuration differs from the settings specified by the Ansible inventory. Therefore, users should avoid running these playbooks with an inventory configuration that could result in changing the cluster in these areas. (**BZ#1476890**)

- When upgrading from a release of OpenShift Container Platform less than 3.5.5.18, the upgrade process may remove data on persistent volumes that fail to unmount correctly. If you are running a version less than 3.5.5.18, perform the following steps prior to performing the normal upgrade process:

```
# atomic-openshift-excluder unexclude
# yum upgrade atomic-openshift-node
# systemctl restart atomic-openshift-node
```

(**BZ#1463393**)

- In OpenShift Container Platform 3.5 and earlier, the Fluentd image included **`fluent-plugin-elasticsearch`** version 1.9.2 and earlier. This version will silently drop records sent in a bulk index request when the queue size is full. In OpenShift Container Platform 3.6, which uses version 1.9.5, an error log message was added, which is why the **`Error: status=429`** message in the Fluentd logs occurs.
To reduce the frequency of this problem, you can increase the Fluentd buffer chunk size. However, testing does not give consistent results. You will need to stop, configure, and restart Fluentd running on all of your nodes.

  1. Edit the daemonset:

```
# oc edit -n logging daemonset logging-fluentd
```

  2. In the **`env:`** section, look for **`BUFFER_SIZE_LIMIT`**. If the value is less than **`8Mi`** (8 megabytes), change the value to **`8Mi`**. Otherwise, use a value of **`16Mi`** or **`32Mi`**. This will roughly increase the size of each bulk index request, which should decrease the number of such requests made to Elasticsearch, thereby allowing Elasticsearch to process them more efficiently.

  3. Once the edit is saved, the Fluentd daemonset trigger should cause a restart of all of the Fluentd pods running in the cluster.
  You can monitor the Elasticsearch bulk index thread pool to see how many bulk index requests it processes and rejects.

  4. Get the name of an Elasticsearch pod:

```
# oc get -n logging pods -l component=es

# espod=$name_of_es_pod
```

  5. Run the following command:

```
# oc exec -n logging $espod -- \
  curl -s -k --cert /etc/elasticsearch/secret/admin-cert \
  --key /etc/elasticsearch/secret/admin-key \
https://localhost:9200/_cat/thread_pool?
v\&h=host,bulk.completed,bulk.rejected,bulk.queue,bulk.active,bul
k.queueSize
```

The output looks like this:

```
host       bulk.completed bulk.rejected bulk.queue bulk.active
bulk.queueSize
10.128.0.6           2262             0          0           0
50
```

The term **`completed`** means the number of bulk indexing operations that have been completed. There will be many (hundreds or thousands of) log records per bulk index request.

The term **`queue`** is the number of pending requests that have been queued up for the server to process. Once this queue is full, additional operations are rejected.

Note the number of **`bulk.rejected`** operations. These correspond to **`error status=429`** in your Fluentd pod logs. Rejected operations means that Fluentd dropped these records, and you might need to increase the chunk size again.

If you have multiple nodes running Elasticsearch, they will each be listed in the **`curl`** output. (**BZ#1470862**)

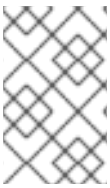
- When performing builds using an image source input, directories within the input content are injected with permissions of 0700 with the default image user as the owner. This means the content is unlikely to be accessible when the application image is run under a random UID. This can be worked around by performing a **`chmod`** operation in either the assemble script (for S2I builds) or the Dockerfile (for Docker builds). Most OpenShift Container Platform S2I builder images already perform this **`chmod`** operation, but custom built S2I builder images or builds using custom assemble scripts may not. (**BZ#1479130**)

- There is a known issue affecting logging behavior when using the non-default **`json-file`** log driver. As a workaround, remove line **`/var/lib/docker/containers`** from ***/etc/oci-umount.conf***, then restart **`docker`**, OpenShift services, and the Fluentd pod. (**BZ#1477787**)

2.8. ASYNCHRONOUS ERRATA UPDATES

Security, bug fix, and enhancement updates for OpenShift Container Platform 3.6 are released as asynchronous errata through the Red Hat Network. All OpenShift Container Platform 3.6 errata is available on the Red Hat Customer Portal. See the OpenShift Container Platform Life Cycle for more information about asynchronous errata.

Red Hat Customer Portal users can enable errata notifications in the account settings for Red Hat Subscription Management (RHSM). When errata notifications are enabled, users are notified via email whenever new errata relevant to their registered systems are released.



NOTE

Red Hat Customer Portal user accounts must have systems registered and consuming OpenShift Container Platform entitlements for OpenShift Container Platform errata notification emails to generate.

This section will continue to be updated over time to provide notes on enhancements and bug fixes for future asynchronous errata releases of OpenShift Container Platform 3.6. Versioned asynchronous releases, for example with the form OpenShift Container Platform 3.6.z, will be detailed in subsections.

In addition, releases in which the errata text cannot fit in the space provided by the advisory will be detailed in subsections that follow.

IMPORTANT

For any OpenShift Container Platform release, always review the instructions on upgrading your cluster properly.

2.8.1. RHEA-2017:2475 - OpenShift Container Platform 3.6.173.0.5-4 Images Update

Issued: 2017-08-15

OpenShift Container Platform release 3.6.173.0.5-4 is now available. The list of container images included in the update are documented in the RHEA-2017:2475 advisory.

The container images in this release have been updated using the latest base images.

2.8.1.1. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.2. RHBA-2017:1829 - OpenShift Container Platform 3.6.173.0.5 Bug Fix Update

Issued: 2017-08-31

OpenShift Container Platform release 3.6.173.0.5 is now available. The container images included in the update are provided by the RHBA-2017:1829 advisory and listed in Images.

Space precluded documenting all of the images for this release in the advisory. See the following sections for notes on upgrading and details on the images included in this release.

2.8.2.1. Images

This release updates the Red Hat Container Registry (**`registry.access.redhat.com`**) with the following images:

```
openshift3/ose-pod:v3.6.173.0.5-5
rhel7/pod-infrastructure:v3.6.173.0.5-5
openshift3/ose-ansible:v3.6.173.0.5-5
openshift3/ose:v3.6.173.0.5-5
openshift3/ose-docker-registry:v3.6.173.0.5-5
openshift3/ose-egress-router:v3.6.173.0.5-5
openshift3/ose-keepalived-ipfailover:v3.6.173.0.5-5
openshift3/ose-f5-router:v3.6.173.0.5-5
openshift3/ose-deployer:v3.6.173.0.5-5
openshift3/ose-haproxy-router:v3.6.173.0.5-5
openshift3/node:v3.6.173.0.5-5
openshift3/ose-sti-builder:v3.6.173.0.5-5
openshift3/ose-docker-builder:v3.6.173.0.5-5
openshift3/logging-deployer:v3.6.173.0.5-5
openshift3/logging-curator:v3.6.173.0.5-5
openshift3/metrics-deployer:v3.6.173.0.5-5
openshift3/logging-auth-proxy:v3.6.173.0.5-5
openshift3/logging-elasticsearch:v3.6.173.0.5-5
```

```
openshift3/logging-fluentd:v3.6.173.0.5-9
openshift3/logging-kibana:v3.6.173.0.5-7
openshift3/metrics-cassandra:v3.6.173.0.5-5
openshift3/metrics-hawkular-metrics:v3.6.173.0.5-5
openshift3/metrics-hawkular-openshift-agent:v3.6.173.0.5-5
openshift3/metrics-heapster:v3.6.173.0.5-5
openshift3/jenkins-1-rhel7:v3.6.173.0.5-5
openshift3/jenkins-5-rhel7:v3.6.173.0.5-5
openshift3/jenkins-slave-base-rhel7:v3.6.173.0.10-5

openshift3/jenkins-slave-maven-rhel7:v3.6.173.0.10-5
openshift3/jenkins-slave-nodejs-rhel7:v3.6.173.0.10-5
openshift3/registry-console:v3.6.173.0.5-5
openshift3/mediawiki-123:v3.6.173.0.10-5
openshift3/apb-base:v3.6.173.0.10-5
openshift3/ose-ansible-service-broker:v3.6.173.0.10-5
openshift3/mediawiki-apb:v3.6.173.0.10-5
openshift3/postgresql-apb:v3.6.173.0.10-5
openshift3/ose-federation:v3.6.173.0.5-5
openshift3/openvswitch:v3.6.173.0.5-5
```

2.8.2.2. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.3. RHBA-2017:2639 - atomic-openshift-utils Bug Fix and Enhancement Update

Issued: 2017-09-05

OpenShift Container Platform bug fix and enhancement advisory RHBA-2017:2639, providing updated **atomic-openshift-utils**, **ansible**, and **openshift-ansible** packages that fix several bugs and add enhancements, is now available.

2.8.3.1. Upgrading

To apply this update, run the following on all hosts where you intend to initiate Ansible-based installation or upgrade procedures:

```
# yum update atomic-openshift-utils
```

2.8.4. RHBA-2017:2642 - OpenShift Container Platform 3.6.1 Bug Fix and Enhancement Update

Issued: 2017-09-08

OpenShift Container Platform release 3.6.1 is now available. The packages and bug fixes included in the update are documented in the RHBA-2017:2642 advisory. The list of container images included in the update are documented in the RHEA-2017:2644 advisory.

The container images in this release have been updated using the latest base images.

2.8.4.1. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.5. RHBA-2017:2847 - OpenShift Container Platform 3.6.173.0.21 Images Update

Issued: 2017-10-03

OpenShift Container Platform release 3.6.173.0.21 is now available. The list of container images included in the update are documented in the RHBA-2017:2847 advisory.

The container images in this release have been updated using the latest base images.

2.8.5.1. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.6. RHBA-2017:3049 - OpenShift Container Platform 3.6.173.0.49 Bug Fix and Enhancement Update

Issued: 2017-10-25

OpenShift Container Platform release 3.6.173.0.49 is now available. The list of packages included in the update are documented in the RHBA-2017:3049 advisory. The container images included in the update are provided by the RHBA-2017:3050 advisory.

Space precluded documenting all of the bug fixes, enhancements, and images for this release in the advisories. See the following sections for notes on upgrading and details on the bug fixes, enhancements, and images included in this release.

2.8.6.1. Bug Fixes

Image Registry

- There was no way to prune orphaned blobs from the OpenShift Container Registry's storage. Orphaned blobs could pile up and consume a considerable amount of free space. This bug fix provides a new low-level utility that is run inside of registry's container and removes the orphaned blobs. As a result, administrators are now able to remove orphaned blobs to retrieve storage space. (**BZ#1479340**)

- The OpenShift Container Registry used to append the forwarded target port to redirected location URLs. The registry client would get confused by the received location containing a superfluous port, and could not match it against the original host. This happened when exposed with TLS-termination other than passthrough. The client's new request to the target location lacked credentials, and as a consequence, the image push failed due to authorization error. This bug fix rebases the registry to a newer version, which fixes forwarding processing logic. As a result, the registry no longer confuses its clients; clients can push images successfully to the exposed registry using arbitrary TLS-termination. (**BZ#1489042**)

- Images younger than the threshold were not added to the dependency graph. Blobs used by a young image and by a prunable image were deleted because they had no references in the graph. This bug fix adds young images to the graph and marks them as non-prunable. As a result, blobs now have references and are not deleted. (**BZ#1498124**)

- Neither documentation nor CLI help talked about insecure connections to the secured registry.

Errors used to be hard to decipher when users attempted to prune the secured registry with a bad CA certificate. This bug fix ensures that errors are now printed with hints, CLI help has been updated, and new flags have been provided to allow for insecure fallback. As a result, users can now easily enforce both secure and insecure connections and understand any HTTPS errors and how to resolve them. (**BZ#1476779**)

- The ClusterRegistry diagnostic checks the registry given to ImageStreams by default with the known registry service. It compares the IP. However, with OpenShift Container Platform 3.6, the ImageStream now gets a cluster hostname for the registry instead of an IP. Therefore, the diagnostic reports a false error condition because the IP is not the same as the hostname. With this bug fix, the diagnostic now checks if either of the hostname and IP version of the registry matches. The diagnostic now reports correctly against either old or new deployments. (**BZ#1488059**)

Logging

- Messages were previously read into Fluentd's memory buffer and were lost if the pod was restarted. Because Fluentd considers them read even though they have not been pushed to storage, any message not stored but already read by Fluentd was lost. This bug fix replaces the memory buffer with a file-based buffer. As a result, file-buffered messages are pushed to storage once Fluentd restarts. (**BZ#1483114**)

- The pattern for container logs in the journal field **`CONTAINER_NAME`** changed. The pattern was not matching for logs from pods in the **`default`**, **`openshift`**, or **`openshift-infra`** namespaces. Logs from these namespaces were being stored in indices matching **`project.default.*`**, for example rather than **`.operations.*`**. This bug fix updates the pattern matcher to match the correct pattern. As a result, logs from pods in the affected namespaces are correctly written to the **`.operations.*`** indices. (**BZ#1493188**)

- Fluentd could not write the files it uses for buffering records due to a problem converting values from ASCII-8BIT to UTF-8, causing Fluentd to emit numerous errors and be unable to add records to Elasticsearch. This bug fix removes the patch that forced the UTF-8 conversion. As a result, Fluentd can write ASCII-8BIT encoded files for its buffer. (**BZ#1482002**)

- Non-operations users were not given the proper access to scroll data. This caused users to see a 403 action denied error in Kibana. This bug fix provides cluster-level permissions for the failed action for ordinary users. As a result, users are now able to export UI objects. (**BZ#1463436**)

- Fluentd was not removing the Kubernetes metadata filter configuration when being used as a mux client. This caused Fluentd to continue opening connections to the OpenShift API server. This bug fix ensures that the Kubernetes metadata filter configuration file is removed when Fluentd is being used as a mux client. As a result, there is no longer a connection from Fluentd running as a mux client to the OpenShift API server. (**BZ#1464024**)

- In OpenShift Container Platform 3.5 and earlier, the Fluentd image included **fluent-plugin-elasticsearch** version 1.9.2 and earlier. This version will silently drop records sent in a bulk index request when the queue size is full. In OpenShift Container Platform 3.6, which uses version 1.9.5, an error log message was added, causing the "Error: status=429" messages in the Fluentd logs when this occurs. With this bug fix (**fluent-plugin-elasticsearch** version 1.9.5.1), when Fluentd gets an error response from Elasticsearch, Fluentd will retry the operation until it succeeds. However, it will also retry successful operations too in some cases, which will lead to duplicate records. See BZ#1491401 for more details. (**BZ#1470862**)

- Fluentd does not stop reading from the journal when the output queue is full. Records are dropped until Fluentd can empty the queue by sending records to Elasticsearch. This bug fix introduces a new configuration parameter, **`buffer_queue_full_action`**, to all output plug-

ins. If using the journal as input, Fluentd will use a value of **`block`** for this parameter, which will cause Fluentd to stop reading from the journal until Fluentd is able to flush the queue. As a result, records are no longer dropped in this scenario. (**BZ#1473788**)

- The collectd schema was using **`int`** instead of **`long`** for the field values, causing long integer values not to be stored. This bug fix uses **`long`** instead of **`int`** in the schema for those fields. As a result, all long integer values can now be stored. (**BZ#1479645**)

- There was previously no configuration to distinguish which Kibana instance to use, and users were always routed to the same instance. This bug fix adds an annotation to the operations namespace which has the Kibana host name. As a result, users are routed to the Kibana instance specified in the annotation if it exists on the project. (**BZ#1480988**)

- The Fluentd processing pipeline to format journald records (system and container logs) into the viaq data model format was using dozens of embedded ruby evaluations per record. The record processing was very slow, with excessive CPU usage. This bug fix moves the processing and formatting into dedicated ruby code in the viaq filter plug-in. As a result, the record processing is much faster, with less CPU usage. (**BZ#1489976**)

- Previously, the **`k8s_meta_filter_for_mux_client`** ruby code was missing from Flentd's Dockerfile, causing Fluentd to complain about the missing filter. This bug fix adds the file to the Dockerfile, and the errors no longer occur. (**BZ#1490395**)

- Previously, the openshift-elasticsearch-plugin improperly handled user names with backslashes. This caused users to be unable to access Elasticsearch. This bug fix modifies requests to convert backslashes to forwardslashes. As a result, users are able to access Elasticsearch with user names that contain backslashes. (**BZ#1491227**)

- Fluentd logs previously filled up with warnings about "log unreadable. It is excluded and would be examined next time." when using the json-file log driver. This bug fix adds an exclude pattern to configuration to exclude messages with this priority level. As a result, the messages are no longer visible. (**BZ#1478821**)

Master

- In some failure cases, the etcd client used by OpenShift will not rotate through all the available etcd cluster members. The client will end up repeatedly trying the same server. If that server is down, then requests will fail for an extended time until the client finds the server invalid. If the etcd leader goes away when it is attempted to be contacted for something like authentication, then the authentication fails and the etcd client is stuck trying to communicate with the etcd member that does not exist. User authentication would fail for an extended period of time. With this bug fix, the etcd client now rotates to other cluster members even on failure. If the etcd leader goes away, the worst that should happen is a failure of that one authentication attempt. The next attempt will succeed because a different etcd member will be used. (**BZ#1490427**)

Networking

- Previously, the iptables proxy was not properly locking its use of iptables. Therefore, the iptables proxy could conflict with **`docker`** and the **`openshift-node`** process and cause a failure to start containers. The iptables proxy now locks its use of iptables. Pod creation failures due to improper locking of iptables no longer occur. (**BZ#1417234**)

- There was a locking issue with the egress DNS policy. This prevented the syncing of egress policies, which could lead to invalid egress access from the pods. This bug fix addresses the egress locking issue. Egress policies are now synced and work as expected. (**BZ#1445694**)

- Conntrack entries for UDP traffic were not cleared when an endpoint was added for a service

that previously had no endpoints. The system could end up incorrectly caching a rule that would cause traffic to that service to be dropped rather than being sent to the new endpoint. With this bug fix, the relevant conntrack entries are now deleted at the right time and UDP services work correctly when endpoints are added and removed. (**BZ#1497767**)

Pod

- After an **`atomic-openshift-node`** restart, pods with init containers may have had their status reset when an attempt was made to reread the init container status. If the container was deleted, this would fail, resetting the status. This bug fix prevents rereading the init container status if the current status on the pod resource indicates the init container is terminated. (**BZ#1491040**)

- Pod anti-affinity is respected across projects. As a result, pod A from project 1 will not land on a node where pod B from project 2 is running, if pod anti-affinity is enabled when scheduling pod A. While scheduling pod A, check for pod anti-affinity only within the project of pod A. Pod anti-affinity will not be respected across projects. (**BZ#1483119**)

Routing

- HAProxy was sending IPv6 addresses in X-Forwarded headers when in ipv6 mode. The behavior changed, so clients that did not expect ipv6 would break. IPv6 mode must be enabled manually rather than defaulting to on. Now, the headers do not change unexpectedly. (**BZ#1472976**)

2.8.6.2. Enhancements

- Using Fluentd as a component for aggregating logs from Fluentd node agents, called *mux*, is a Technology Preview feature for OpenShift Container Platform 3.6. (**BZ#1469859**)

2.8.6.3. Images

This release updates the Red Hat Container Registry (**`registry.access.redhat.com`**) with the following images:

```
openshift3/ose-f5-router:v3.6.173.0.49-4
openshift3/container-engine:v3.6.173.0.49-4
openshift3/jenkins-slave-maven-rhel7:v3.6.173.0.49-5
openshift3/logging-auth-proxy:v3.6.173.0.49-4
openshift3/logging-deployer:v3.6.173.0.49-5
openshift3/logging-fluentd:v3.6.173.0.49-4
openshift3/metrics-cassandra:v3.6.173.0.49-5
openshift3/metrics-hawkular-metrics:v3.6.173.0.49-5
openshift3/metrics-hawkular-openshift-agent:v3.6.173.0.49-4
openshift3/apb-base:v3.6.173.0.49-4
openshift3/ose-cluster-capacity:v3.6.173.0.49-4
openshift3/ose-docker-builder:v3.6.173.0.49-4
openshift3/ose-docker-registry:v3.6.173.0.49-4
openshift3/ose-federation:v3.6.173.0.49-4
openshift3/ose-keepalived-ipfailover:v3.6.173.0.49-4
openshift3/node:v3.6.173.0.49-5
openshift3/ose-pod:v3.6.173.0.49-4
openshift3/ose-service-catalog:v3.6.173.0.49-4
openshift3/jenkins-2-rhel7:v3.6.173.0.49-5
openshift3/ose-egress-http-proxy:v3.6.173.0.49-4
openshift3/ose-ansible:v3.6.173.0.49-5
openshift3/jenkins-slave-base-rhel7:v3.6.173.0.49-5
```

```
openshift3/jenkins-slave-nodejs-rhel7:v3.6.173.0.49-5
openshift3/logging-curator:v3.6.173.0.49-4
openshift3/logging-elasticsearch:v3.6.173.0.49-5
openshift3/logging-kibana:v3.6.173.0.49-5
openshift3/metrics-deployer:v3.6.173.0.49-5
openshift3/metrics-heapster:v3.6.173.0.49-4
openshift3/ose-ansible-service-broker:v3.6.173.0.49-4
openshift3/ose-deployer:v3.6.173.0.49-4
openshift3/ose:v3.6.173.0.49-4
openshift3/ose-egress-router:v3.6.173.0.49-4
openshift3/ose-haproxy-router:v3.6.173.0.49-4
openshift3/mediawiki-123:v3.6.173.0.49-4
openshift3/openvswitch:v3.6.173.0.49-5
openshift3/ose-recycler:v3.6.173.0.49-4
openshift3/ose-sti-builder:v3.6.173.0.49-4
openshift3/jenkins-1-rhel7:v3.6.173.0.49-5
openshift3/registry-console:v3.6.173.0.49-4
```

2.8.6.4. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.7. RHSA-2017:3389 - Moderate: OpenShift Container Platform 3.6.173.0.63 Security, Bug Fix, and Enhancement Update

Issued: 2017-12-06

OpenShift Container Platform release 3.6.173.0.63 is now available. The list of packages included in the update are documented in the RHSA-2017:3389 advisory. The container images included in the update are provided by the RHBA-2017:3390 advisory.

Space precluded documenting all of the bug fixes, enhancements, and images for this release in the advisories. See the following sections for notes on upgrading and details on the bug fixes and images included in this release.

2.8.7.1. Images

This release updates the Red Hat Container Registry (**`registry.access.redhat.com`**) with the following images:

```
openshift3/apb-base:v3.6.173.0.63-10
openshift3/container-engine:v3.6.173.0.63-11
openshift3/logging-auth-proxy:v3.6.173.0.63-11
openshift3/logging-curator:v3.6.173.0.63-11
openshift3/logging-deployer:v3.6.173.0.63-11
openshift3/logging-elasticsearch:v3.6.173.0.63-11
openshift3/logging-fluentd:v3.6.173.0.63-11
openshift3/logging-kibana:v3.6.173.0.63-11
openshift3/mediawiki-123:v3.6.173.0.63-11
openshift3/mediawiki-apb:v3.6.173.0.63-10
openshift3/metrics-cassandra:v3.6.173.0.63-11
openshift3/metrics-deployer:v3.6.173.0.63-11
openshift3/metrics-hawkular-metrics:v3.6.173.0.63-11
openshift3/metrics-hawkular-openshift-agent:v3.6.173.0.63-11
```

```
openshift3/metrics-heapster:v3.6.173.0.63-11
openshift3/node:v3.6.173.0.63-11
openshift3/openvswitch:v3.6.173.0.63-11
openshift3/ose-ansible-service-broker:v3.6.173.0.63-11
openshift3/ose-ansible:v3.6.173.0.63-11
openshift3/ose-base:v3.6.173.0.63-11
openshift3/ose-cluster-capacity:v3.6.173.0.63-11
openshift3/ose-deployer:v3.6.173.0.63-11
openshift3/ose-docker-builder:v3.6.173.0.63-11
openshift3/ose-docker-registry:v3.6.173.0.63-11
openshift3/ose-egress-http-proxy:v3.6.173.0.63-11
openshift3/ose-egress-router:v3.6.173.0.63-11
openshift3/ose-f5-router:v3.6.173.0.63-11
openshift3/ose-federation:v3.6.173.0.63-11
openshift3/ose-haproxy-router:v3.6.173.0.63-11
openshift3/ose-keepalived-ipfailover:v3.6.173.0.63-11
openshift3/ose-pod:v3.6.173.0.63-11
openshift3/ose-recycler:v3.6.173.0.63-11
openshift3/ose-service-catalog:v3.6.173.0.63-11
openshift3/ose-sti-builder:v3.6.173.0.63-11
openshift3/ose:v3.6.173.0.63-11
openshift3/postgresql-apb:v3.6.173.0.63-10
openshift3/registry-console:v3.6.173.0.63-11
```

2.8.7.2. Bug Fixes

Authentication

- During upgrades, reconciliation happens only for cluster roles automatically, but this role needs to be adjusted in 3.6 due to enablement of API groups in this release. The Ansible upgrade code has been changed to address this role upgrade. (**BZ#1493213**)

Image Registry

- The size of a cached layer did not get counted. Therefore, the layer size for cached layers was zero. Counting the size for cached layers now allows images to have proper layer sizes. (**BZ#1457042**)

Logging

- **`openshift-elasticsearch-plugin`** was creating ACL roles based on the provided name, which could include slashes and commas. This caused the dependent library to not properly evaluate roles. With this bug fix, hash the name when creating ACL roles so they no longer contain the invalid characters. (**BZ#1494239**)

- If the logging system is under a heavy load, it may take longer than the five-second timeout for Elasticsearch to respond, or it may respond with an error indicating that Fluentd needs to back off. In the former case, Fluentd will retry to send the records again, which can lead to having duplicate records. In the latter case, if Fluentd is unable to retry, it will drop records, leading to data loss. For the former case, the fix is to set the **`request_timeout`** to 10 minutes, so that Fluentd will wait up to 10 minutes for the reply from Elasticsearch before retrying the request. In the latter case, Fluentd will block attempting to read more input, until the output queues and buffers have enough room to write more data. This bug fix greatly reduces chances of duplicate data (though it is not entirely eliminated). Also, there is no data loss due to back pressure. (**BZ#1497836**, **BZ#1501948**, **BZ#1506854**)

Management Console

- The management console was defaulting to the legacy API group **`extensions`** for jobs. As a result, the legacy API group appeared in the UI in places such as **Edit YAML**. With this bug fix, the console snow uses the new **`batch`** API group as the default for job resources. The API group and version on a job resource now appear as **`batch/v1`** wherever it is visible in the console. (**BZ#1506233**)

Metrics

- Extra, unnecessary queries were being performed on each request. The GET **`/hawkular/metrics/metrics`** endpoint could fail with timeouts. With this bug fix, the extra queries are only performed when explicitly requested. By default, do not execute the extra queries that provide optional data. The endpoint is now more stable and not as susceptible to timeouts. (**BZ#1458186**)

- When either a certificate within the chain at **`serviceaccount/ca.crt`** or any of the certificates within the provided truststore file contained a white space after the **`BEGIN CERTIFICATE`** declaration, the Java keytool rejected the certificate with an error, causing Origin Metrics to fail to start. As a workaround, Origin Metrics will now attempt to remove the spaces before feeding the certificate to the Keytool, but administrators should ensure their certificates do not contain such spaces. (**BZ#1471251**, **BZ#1500464**, **BZ#1500471**)

Networking

- A slow image pull made the network diagnostics fail. With this bug fix, the timeout for the image pull was increased. The diagnostics now run in slow environments. (**BZ#1481550**)

- The OpenShift node proxy previously did not support using a specified IP address. This could prevent correct operation on hosts with multiple network interface cards. The OpenShift node process already accepts a **`--bind-address=<ip address>:<port>`** command-line flag and **`bindAddress:`** configuration file option for the multiple network interface card case. The proxy functionality has been fixed to respect these options. When **`--bind-address`** or **`bindAddress`** are used, the OpenShift node proxy should work correctly when the OpenShift node host has multiple network interface cards. (**BZ#1489023**, **BZ#1489024**)

- Iptables called too often and unnecessarily. Therefore, time-outs would wait for iptables operations to finish. This bug fix changes the code so that it skips reloads when the iptables rules are unchanged. There are now fewer calls to iptables and, therefore, less time-outs. (**BZ#1501517**)

Pod

- There was a symbolic link error for the log file of every pod started when the docker log driver was journald. Log symlink creation that fails when using journald logging driver was skipped. This bug fix resolves the issue. (**BZ#1434942**)

- Currently, pod anti-affinity is respected across projects. Pod A from Project 1 will not land on node where Pod B from Project 2 is running, if pod anti-affinity is enabled when scheduling Pod A. While scheduling Pod A, check for pod anti-affinity only within the project of Pod A. Pod anti-affinity will not be respected across projects. (**BZ#1492194**)

Storage

- The volumePath that included the datastore name was parsed incorrectly. The same applies to volumePath that included datacluster and datastore names. It is not possible to attach persistent volumes that have the above described volumePath values. volumePath is now parsed

correctly. Persistent volumes that have the above described volumePath values are attached correctly. (**BZ#1497042**)

Security

- An attacker with knowledge of the given name used to authenticate and access Elasticsearch can later access it without the token, bypassing authentication. This attack also requires that the Elasticsearch be configured with an external route, and the data accessed is limited to the indices. (**BZ#1501986**)

2.8.7.3. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.8. RHBA-2017:3438 - OpenShift Container Platform 3.6.173.0.83 Bug Fix and Enhancement Update

Issued: 2017-12-14

OpenShift Container Platform release 3.6.173.0.83 is now available. The list of packages included in the update are documented in the RHBA-2017:3438 advisory. The container images included in the update are provided by the RHBA-2017:3439 advisory.

Space precluded documenting all of the bug fixes, enhancements, and images for this release in the advisories. See the following sections for notes on upgrading and details on the bug fixes and images included in this release.

2.8.8.1. Images

This release updates the Red Hat Container Registry (**`registry.access.redhat.com`**) with the following images:

```
openshift3/apb-base:v3.6.173.0.83-2
openshift3/container-engine:v3.6.173.0.83-2
openshift3/jenkins-2-rhel7:v3.6.173.0.83-2
openshift3/jenkins-slave-base-rhel7:v3.6.173.0.83-2
openshift3/jenkins-slave-maven-rhel7:v3.6.173.0.83-2
openshift3/jenkins-slave-nodejs-rhel7:v3.6.173.0.83-2
openshift3/logging-curator:v3.6.173.0.83-2
openshift3/logging-elasticsearch:v3.6.173.0.83-2
openshift3/logging-fluentd:v3.6.173.0.83-2
openshift3/logging-kibana:v3.6.173.0.83-2
openshift3/mediawiki-123:v3.6.173.0.83-2
openshift3/mediawiki-apb:v3.6.173.0.83-2
openshift3/metrics-cassandra:v3.6.173.0.83-2
openshift3/metrics-deployer:v3.6.173.0.83-2
openshift3/metrics-hawkular-metrics:v3.6.173.0.83-2
openshift3/metrics-hawkular-openshift-agent:v3.6.173.0.83-2
openshift3/metrics-heapster:v3.6.173.0.83-2
openshift3/node:v3.6.173.0.83-2
openshift3/openvswitch:v3.6.173.0.83-4
openshift3/ose-ansible:v3.6.173.0.83-2
openshift3/ose-cluster-capacity:v3.6.173.0.83-2
openshift3/ose-deployer:v3.6.173.0.83-4
```

```
openshift3/ose-docker-builder:v3.6.173.0.83-4
openshift3/ose-docker-registry:v3.6.173.0.83-2
openshift3/ose-egress-http-proxy:v3.6.173.0.83-2
openshift3/ose-f5-router:v3.6.173.0.83-4
openshift3/ose-federation:v3.6.173.0.83-2
openshift3/ose-haproxy-router:v3.6.173.0.83-4
openshift3/ose-keepalived-ipfailover:v3.6.173.0.83-2
openshift3/ose-pod:v3.6.173.0.83-2
openshift3/ose-recycler:v3.6.173.0.83-4
openshift3/ose-service-catalog:v3.6.173.0.83-2
openshift3/ose:v3.6.173.0.83-2
openshift3/postgresql-apb:v3.6.173.0.83-2
openshift3/registry-console:v3.6.173.0.83-2
```

2.8.8.2. Bug Fixes

- The **imagetrigger-controller** was missing the permission to create custom **`Build`** objects. Builds with **`customStrategy`** and **`ImageStreamTag`** triggers were not started whenever the **`ImageStreamTag`** was updated. This bug fix adds the permission to the **imagetrigger-controller**, and as a result builds with **`customStrategy`** and **`ImageStreamTag`** triggers are now started when the **`ImageStreamTag`** is updated. (**BZ#1519296**)

- Container Native Storage (CNS) installations would fail with an error when **`groups.glusterfs_registry`** was undefined. This bug has been fixed. (**BZ#1462143**)

- CNS installations would fail if the namespace for CNS pods was different from the namespace **`default`**. This bug fix ensures that proper namespaces are used for the **`heketi`** command and service account, and the failure no longer occurs. (**BZ#1464393**)

- The GlusterFS example inventory files had unsupported parameters. This bug fix cleans up these examples. As a result, unsupported parameters are no longer suggested. (**BZ#1474692**)

- In verifying sufficient disk space available under the ***/var*** directory, the **disk_availability** check only counted storage mounted directly at ***/var*** (or */* if ***/var*** is not a separate file system). Extra storage mounted below ***/var***, for instance in ***/var/lib/docker***, was not counted toward the required available storage, and thus the check could fail erroneously. This bug fix ensures that storage mounted below a file system is included in its total availability. As a result, the check should accurately account for availability of storage. (**BZ#1491566**)

- Fluentd input plug-ins treated messages as ASCII-8BIT by default. As a result, if one of these messages or any other message with UTF-8 characters was read in and later needed to be converted to UTF-8, the conversion failed with an "UndefinedConversionError". This bug fix adds a new **record_modifier** filter, which treats the messages as UTF-8. (**BZ#1501993**)

- The **`_source`** was not disabled for metrics records being stored in Elasticsearch. The records were taking up much more CPU, RAM, and disk resources than necessary. This bug fix completely disables the **`_source`** for **`project.ovirt-metrics*`** records. As a result, metrics records are much smaller and require fewer resources to handle. (**BZ#1512132**)

- OCP 3.6 rejected certain invalid master configuration values which OCP 3.5 silently accepted. When upgrading from 3.5 to 3.6, the master would fail to start if the **`clusterNetworkCIDR`** or **`serviceNetworkCIDR`** value in the master configuration was "invalid" (for example, if it had "172.30.1.1/16" instead of "172.30.0.0/16"). This bug fix ensures that OCP 3.6 accepts the same invalid values that OCP 3.5 accepted, but logs a warning about it. As a result, upgrades will now work, and the administrator is notified about the incorrect configuration values. (**BZ#1508445**)

2.8.8.3. Enhancements

- Loading the Java console can sometimes take a while, and it is important to show to users that the Java console is still loading and not frozen. With this enhancement, a spinning image is shown under the top menu bar while the Java console is loading. (**BZ#1426615**)

2.8.8.4. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.9. RHBA-2018:0076 - OpenShift Container Platform 3.6.173.0.83-10 Images Update

Issued: 2018-01-10

OpenShift Container Platform release 3.6.173.0.83-10 is now available. The list of container images included in the update are documented in the RHBA-2018:0076 advisory.

The container images in this release have been updated using the latest base images.

2.8.9.1. Images

This release updates the Red Hat Container Registry (**`registry.access.redhat.com`**) with the following images:

```
openshift3/node:v3.6.173.0.83-10
openshift3/logging-kibana:v3.6.173.0.83-9
openshift3/openvswitch:v3.6.173.0.83-11
```

2.8.9.2. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.10. RHBA-2018:0113 - OpenShift Container Platform 3.6.173.0.96 Bug Fix and Enhancement Update

Issued: 2018-01-22

OpenShift Container Platform release 3.6.173.0.96 is now available. The packages and bug fixes included in the update are documented in the RHBA-2018:0113 advisory. The container images included in the update are provided by the RHBA-2018:0114 advisory.

Space precluded documenting all of the bug fixes and images for this release in the advisory. See the following sections for notes on upgrading and details on the bug fixes, enhancements, and images included in this release.

2.8.10.1. Bug Fixes

- A regression bug was reported whereby source-to-image builds would fail if the source repository filesystem contained a broken symlink (pointing to a non-existent item). This was resolved with this bug fix. (**BZ#1519822**)

- Categories like **all** were moved to the server, but some of them were only moved after the upstream cut for the rebase, causing an incomplete list of resources. Therefore, some resources could not be found in **`oc get all`** and some other **`oc get`** calls. With this bug fix, the remaining upstream commits were picked to include all needed resources and **`oc get all`** and the other problematic calls were fixed. (**BZ#1515878**)

- Node selectors were incorrectly set on template service broker daemonset object. Consequently, the looping failed the deployment of template service broker pods and there was excessive CPU usage on master and nodes. The node selectors are now set correctly on the template service broker daemonset object and the template service broker pods now deploy correctly. (**BZ#1524219**)

- Fluentd fails to properly process messages when it is unable to determine the namespace and pod UUIDs.The logging pipeline outputs a lot of messages and sometimes blocks log flow to Elasticsearch. Check for the missing fields and orphan the record if needed. With this bug fix, logs continue to flow and orphaned records end up in an orphaned namespace. (**BZ#1494612**)

- Elasticsearch clusters that take a long time recovering data do not reach the **YELLOW** state fast enough. The OpenShift Container Platform cluster restarts the pod because the readiness probe fails, which starts the Elasticsearch node recovery again. Check only for the Elasticsearch cluster to be listening on the desired port. The OpenShift Container Platform cluster does not terminate the Elasticsearch node early, which allows it to complete its recovery. The cluster may be in the **RED** state at this time, but is able to accept queries and writes. (**BZ#1510697**)

- When trying to alias an index that does not exist, the bulk alias operation failed. Only alias ***.operations*** if there is at least one ***.operations*** index. As a result, there will be an alias for all indices. (**BZ#1519705**)

- The **remote-syslog** plug-in in fluentd takes a configuration parameter **`tag_key`** to use the field specified in **`tag_key`** from the record to set the syslog key. When a field specified in **`tag_key`** does not exist, it caused a Ruby exception, which was not caught. With this bug fix, the field no longer exists and **`tag_key`** is ignored and the default tag is used. (**BZ#1519213**)

- There was a logic error in the fluentd startup script. When an ops cluster was first disabled then enabled, the proper ops configuration file was not enabled. As a result, Sub-configuration files starting with **`output-ops-extra-`** did not have a chance to be called from the ops configuration file. The logic error s now fixed When an ops cluster is first disabled then enabled, the proper ops configuration file is enabled and its sub-configuration files are also enabled. (**BZ#1519679**)

- The annotation to identify the proper Kibana instance to use for ops namespaces was being set regardless of if an ops logging cluster is used or not. Users were directed to non-existent service from the web console when trying to view operations logs. For existing deployments, manually run **`oc annotate ns $NAMESPACE openshift.io/logging.ui.hostname-`** for each affected namespace. New deployments will only have this annotation set by the installer if the`openshift_logging_use_ops` variable is set to **`true`**. With this bug fix, users will be directed to the correct version of Kibana when viewing ops logs from the web console. (**BZ#1519808**)

- Previously, when importing a template as YAML in the web console, then clicking **Back** in the wizard, the **Next** button would stop working. The problem has been fixed so that the **Next** button works correctly after clicking **Back** in the **Import YAML** dialog. (**BZ#1526215**)

- The Kubernetes resource quota controller had a fatal a race condition. Therefore, the master controller process occasionally crashes, writes a stack dump, and restarts. With this bug fix, the race condition is resolved and the crash no longer occurs. (**BZ#1519277**)

- When a Network Egress DNS policy was used, a bug may have prevented further correct operation of the proxy, resulting in new pods not handling service requests. That bug is fixed and Egress DNS policies can now be used without triggering this bug. (**BZ#1502602**)

- A bug in the node container garbage collection and network setup prevented pod sandboxes from being properly garbage collected. Nodes could exhaust the available pool of pod IP addresses, especially if they are restarted and/or containers were removed while the node was not running. Nodes now properly garbage collect and tear down pod sandboxes, ensuring that IP addresses are released to the pool for subsequent re-use. Newly installed nodes should no longer experience IP address exhaustion due to pod sandbox teardown errors. Upgraded nodes should remove all files in ***/var/lib/cni/networks/openshift-sdn/*** during the upgrade, or after upgrade when no pods are running on the node. (**BZ#1516782**)

- This bug fix corrects an interaction between runc and systemd where the sandbox creation fails when the pod specifies CPU limit in 1000s of millicores and the last digit is not 0. (**BZ#1509467**)

- This bug fix corrects an issue where **`oc logs`** exits with error **unexpected stream type**. (**BZ#1521026**)

- When running the etcd v2 to v3 migration playbooks as included in the OpenShift Container Platform 3.7 release, the playbooks incorrectly assumed that all services were HA services (**`atomic-openshift-master-api`** and **`atomic-openshift-master-controllers`** rather than **`atomic-openshift-master`**), which is the norm on version 3.7. However, the migration playbooks would be executed prior to upgrading to version 3.7, so this was incorrect. The migration playbooks have been updated to start and stop the correct services ensuring proper migration. (**BZ#1523814**)

2.8.10.2. Enhancements

- Elasticsearch **`deploymentconfigs`** are now modified to disable OpenShift Container Platform rollback behavior. Upgrades of the logging stack that do not result in the readiness probe succeeding are rolled back to the last previous successful deployment. This can result in the deployed instances being out-of-date with their configuration and old images. This change will make it so the deployments will not be rolled back and operators can manually intervene without the image and configuration mismatches. (**BZ#1519622**)

- An ***.operations*** index-mapping in a non-ops Elasticsearch cluster is no longer displayed because operations indices will never exist in a non-ops Elasticsearch cluster. (**BZ#1519706**)

2.8.10.3. Images

This release updates the Red Hat Container Registry (***registry.access.redhat.com***) with the following images:

```
openshift3/apb-base:v3.6.173.0.96-2
openshift3/container-engine:v3.6.173.0.96-2
openshift3/jenkins-1-rhel7:v3.6.173.0.96-2
openshift3/jenkins-2-rhel7:v3.6.173.0.96-2
openshift3/jenkins-slave-base-rhel7:v3.6.173.0.96-2
openshift3/jenkins-slave-maven-rhel7:v3.6.173.0.96-2
openshift3/jenkins-slave-nodejs-rhel7:v3.6.173.0.96-2
openshift3/logging-auth-proxy:v3.6.173.0.96-2
openshift3/logging-curator:v3.6.173.0.96-2
openshift3/logging-elasticsearch:v3.6.173.0.96-3
openshift3/logging-fluentd:v3.6.173.0.96-2
openshift3/logging-kibana:v3.6.173.0.96-2
```

```
openshift3/mediawiki-123:v3.6.173.0.83-2
openshift3/mediawiki-apb:v3.6.173.0.96-2
openshift3/metrics-cassandra:v3.6.173.0.96-2
openshift3/metrics-deployer:v3.6.173.0.83-11.1513170044
openshift3/metrics-hawkular-metrics:v3.6.173.0.96-2
openshift3/metrics-hawkular-openshift-agent:v3.6.173.0.96-2
openshift3/metrics-heapster:v3.6.173.0.96-2
openshift3/node:v3.6.173.0.96-2
openshift3/openvswitch:v3.6.173.0.96-2
openshift3/ose-ansible-service-broker:v3.6.173.0.78-1
oopenshift3/ose-ansible:v3.6.173.0.96-3
openshift3/ose-base:v3.6.173.0.96-2
openshift3/ose-cluster-capacity:v3.6.173.0.96-2
openshift3/ose-deployer:v3.6.173.0.96-2
openshift3/ose-docker-builder:v3.6.173.0.96-2
openshift3/ose-docker-registry:v3.6.173.0.96-2
openshift3/ose-egress-http-proxy:v3.6.173.0.96-2
openshift3/ose-egress-router:v3.6.173.0.96-2
openshift3/ose-f5-router:v3.6.173.0.96-2
openshift3/ose-federation:v3.6.173.0.96-2
openshift3/ose-haproxy-router:v3.6.173.0.96-2
openshift3/ose-keepalived-ipfailover:v3.6.173.0.96-2
openshift3/ose-pod:v3.6.173.0.96-2
openshift3/ose-recycler:v3.6.173.0.96-2
openshift3/ose-service-catalog:v3.6.173.0.96-2
openshift3/ose-sti-builder:v3.6.173.0.96-2
openshift3/ose:v3.6.173.0.96-2
openshift3/postgresql-apb:v3.6.173.0.96-2
openshift3/registry-console:v3.6.173.0.96-2
```

2.8.10.4. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.11. RHBA-2018:1106 - OpenShift Container Platform 3.6.173.0.112 Bug Fix Update

Issued: 2018-04-12

OpenShift Container Platform release 3.6.173.0.112 is now available. The list of packages included in the update are documented in the RHBA-2018:1106 advisory. The list of container images included in the update are documented in the RHBA-2018:1107 advisory.

Space precluded documenting all of the bug fixes for this release in the advisories. See the following sections for notes on upgrading and details on the bug fixes included in this release.

2.8.11.1. Bug Fixes

Builds

- The OpenShift Docker builder invokes the Docker build API without the **`ForceRmTemp`** flag. Containers from failed builds remain on the node where the build ran. These containers are not recognized by the kubelet for **gc** and are therefore accumulated until the node runs out of space.

This bug fix modifies the Docker build API call from the OpenShift Docker builder to force the removal of temporary containers. As a result, failed containers no longer remain on the node where a Docker build ran. (BZ#1533181)

- The login plug-in servlet filter was not getting re-registered in Jenkins after a restart until a login via the web console occurred. This made HTTP access to an OpenShift Jenkins pod with OAuth via bearer token impossible until the servlet filter was re-registered. This bug fix forces the re-register of the login plug-in servlet filter after pod restarts. As a result, HTTP access via bearer token is possible without having to login via the Jenkins console first. (BZ#1533938)

Installer

- Amazon EC2 C5 instances use different identifiers for `bios_vendor`. Code which uses this information for identifying the host as an AWS instance was not being run. This bug fix adds logic to use the new `bios_vendor` identifier. As a result, AWS C5 instances are properly identified by **openshift-ansible** code. (BZ#1538778)

- OpenShift Container Platform requires that host names conform to standards which preclude the use of uppercase letters. With this bug fix, the installer now ensures that the host names for node objects are created with lowercase letters. (BZ#1543749)

- The `OPTIONS` value in */etc/sysconfig/atomic-openshift-node* now works properly when multiple options are specified with containerized installations. (BZ#1539091)

- HAProxy services files were not included in the uninstall playbook. This caused HAProxy service files to remain after running the uninstall playbook. This bug fix adds HAProxy service files to the uninstall playbook, and as a result they are remove after running the playbook. (BZ#1511294)

- Alternative names in certificates were not being properly parsed. Alternatives with `email:` were being added as additional host names. This bug fix updates the logic to only add alternative names which begin with `DNS:`. As a result, `namedCertificates` are properly parsed and updated. (BZ#1538895)

- The **docker_image_availability** check did not take into account variables that override specific container images used for containerized components. The check could incorrectly report failure looking for the default images when the overridden images to be used are actually available. This bug fix ensures the check now looks for override variables, and as a result the check should now accurately report whether the necessary images are available. (BZ#1539150)

Logging

- The Fluentd input plug-ins treat the logs as ASCII-8BIT by default. Then, they are converted to UTF-8 to forward to ElasticSearch, where `UndefinedConversionError` is rarely observed. With this bug fix, if the environment variable `ENABLE_UTF8_FILTER` is set to `true`, the incoming logs are treated as UTF-8 and it eliminates the chance of the conversion error. (BZ#1505959)

- Messages for which the unique namespace ID could not be determined could not be properly indexed. Messages could be lost and the error message appears in the logs. This bug fix modifies the cache algorithm to provide the necessary data or default the value to `orphaned`. As a result, the error message is resolved and messages are stored in an `orphaned` index when a unique namespace ID can not be determined. (BZ#1506286, BZ#1525415)

- The Kibana container image was not built with the required header image. This bug fix replaces the OpenShift Origin header image with the OpenShift Container Platform header image. As a result, the Kibana page displays the desired header. (BZ#1547347)

- Fluentd inserts documents (logs) into Elasticsearch using the bulk insert API but relies upon Elasticsearch to generate UUIDs for each document. It does not remove successfully indexed documents from the bulk payload when the bulk operation fails. This causes the initial payload to be resubmitted and documents that were successfully indexed are submitted again which results in duplicate documents with different UUIDs. This bug fix ensures that document IDs are generated before submitting bulk insert requests. As a result, Elasticsearch will now disregard insert of documents that already exist in the data store and insert documents that do not. (**BZ#1556897**)

- The link generation code assumes all project logs are written to indices that have a common naming pattern. This can cause users to be linked to non-existent indices. With this bug fix, project logs that will be archived to different indices are annotated with the required information to properly build the link. As a result, users are routed using a link that will query the data store correctly and return data. (**BZ#1547688**)

Web Console

- Use of javascript Array/String methods not supported by IE11 (**`Array.find`**, **`Array.findIndex`**, **`String.startsWith`**, **`String.endsWith`**) caused an error that prevents some A-MQ pages to show content when the user clicks on a tree node. This bug fix replaces the unsupported methods with the Lodash library equivalent methods. As a result, A-MQ pages show their content as expected. (**BZ#1543435**)

- When multiple roles with long names are assigned to a user, the roles do not wrap below. This causes the page layout to break and pushes the "Add another role" form select input off the screen, preventing additional roles from being assigned to the user. This bug fix allows roles to vertically stack, and as a result the "Add another role" select input field now remains in place. (**BZ#1545089**)

Master

- Rate limiting is causing some connections to masters from nodes to be dropped, causing node status to not be updated properly. This bug fix uses a separate (non-limited) client for node status. As a result, the node status should be properly saved in the master. (**BZ#1527389**)

- This update fixes an issue in an event reported by the deployment configuration controller where the old replica count would be an unreasonably large and incorrect number. (**BZ#1536189**)

Pod

- This update fixes an issue where when querying the Azure API for the existence of a instance and there was an error, the error was dropped and views as the instance not existing. This caused unintended side effects like the node to going into **NotReady** state or the pods being evicted from the node. (**BZ#1550992**)

- This update fixes an issue where slow pod deletion on a node under eviction pressure could result in the eviction of all pods. (**BZ#1509289**)

Storage

- Dynamic provisioning of Cinder volumes stops working after certain amount of time. Restarting OpenShift Container Platform resolves this problem temporarily because it is unable to re-authenticate to OpenStack Keystone V3. This made it impossible to create new Cinder volumes using dynamic provisioning. This bug fixes OpenShift Container Platform re-authentication to OpenStack Keystone V3, and as a result dynamic provisioning of new Cinder volumes works as expected. (**BZ#1527973**)

2.8.11.2. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.12. RHBA-2018:1579 - OpenShift Container Platform 3.6.173.0.117 Bug Fix and Enhancement Update

Issued: 2018-05-17

OpenShift Container Platform release 3.6.173.0.117 is now available. The packages, bug fixes, and enhancements included in the update are documented in the RHBA-2018:1579 advisory. The list of container images included in the update are documented in the RHBA-2018:1578 advisory.

2.8.12.1. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.13. RHBA-2018:1801 - OpenShift Container Platform 3.6.173.0.123 Bug Fix and Enhancement Update

Issued: 2018-06-07

OpenShift Container Platform release 3.6.173.0.123 is now available. The packages and bug fixes included in the update are documented in the RHBA-2018:1801 advisory. The list of container images included in the update are documented in the RHBA-2018:1800 advisory.

The container images in this release have been updated using the latest base images.

2.8.13.1. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.14. RHBA-2018:2007 - OpenShift Container Platform 3.6.173.0.124 Bug Fix Update

Issued: 2018-06-28

OpenShift Container Platform release 3.6.173.0.124 is now available. The list of packages included in the update are documented in the RHBA-2018:2007 advisory. The list of container images included in the update are documented in the RHBA-2018:2008 advisory.

Space precluded documenting all of the bug fixes for this release in the advisories. See the following sections for notes on upgrading and details on the bug fixes included in this release.

2.8.14.1. Bug Fixes

- Jenkins core/remoting has subpar handling of the **`no_proxy`** environment variable. This affects communication between Jenkins agents and master when starting a build using the Kubernetes plug-in in the OpenShift Jenkins image. Pipelines using the Kubernetes plug-in were therefore unable to start agents with that plug-in when HTTP proxies were defined. This bug fix updates the sample Maven and Node.js OpenShift Jenkins images to automatically add the server URL

and tunnel hosts to the **no_proxy** list, ensuring that communication works when HTTP proxies are defined. As a result, Jenkins Pipelines can now leverage the Kubernetes plug-in to start pods based on the OpenShift Jenkins Maven and Node.js images. (**BZ#1573648**)

- Certain incoming data to Elasticsearch has a field **record["event"]** that is a String value and not the Hash value expected by the **transform_eventrouter** code. This caused the code to throw an error and fluentd to emit an error like:

```
    error_class=NoMethodError error="undefined method `key?' for
\"request\":String"
```

This bug fix changes the **transform_eventrouter** code to only process the **record["event"]** field if it is a Hash. As a result, records can flow through to Elasticsearch again. (**BZ#1588772**)

2.8.14.2. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.15. RHBA-2018:2232 - OpenShift Container Platform 3.6.173.0.126 Bug Fix Update

Issued: 2018-07-24

OpenShift Container Platform release 3.6.173.0.126 is now available. The list of packages and bug fixes included in the update are documented in the RHBA-2018:2232 advisory. The list of container images included in the update are documented in the RHBA-2018:2233 advisory.

2.8.15.1. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.16. RHBA-2018:2340 - OpenShift Container Platform 3.6.173.0.128 Bug Fix Update

Issued: 2018-08-09

OpenShift Container Platform release 3.6.173.0.128 is now available. The list of packages and bug fixes included in the update are documented in the RHBA-2018:2339 advisory. The list of container images included in the update are documented in the RHBA-2018:2340 advisory.

2.8.16.1. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.17. RHBA-2018:2545 - OpenShift Container Platform 3.6.173.0.129 Bug Fix Update

Issued: 2018-08-28

OpenShift Container Platform release 3.6.173.0.129 is now available. The list of packages and bug fixes included in the update are documented in the RHBA-2018:2545 advisory. The list of container images included in the update are documented in the RHBA-2018:2544 advisory.

2.8.17.1. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

2.8.18. RHSA-2018:2654 - OpenShift Container Platform 3.6.173.0.130 Security, Bug Fix, and Enhancement Update

Issued: 2018-09-26

OpenShift Container Platform release 3.6.173.0.130 is now available. The list of packages and bug fixes included in the update are documented in the RHSA-2018:2654 advisory. The list of container images included in the update are documented in the RHBA-2018:2655 advisory.

2.8.18.1. Bug Fixes

- This update adds retries to the `shared-resource-viewer` update logic to avoid problems with object contention. (**BZ#1507119**)

2.8.18.2. Upgrading

To upgrade an existing OpenShift Container Platform 3.5 or 3.6 cluster to this latest release, use the automated upgrade playbook. See Performing Automated In-place Cluster Upgrades for instructions.

CHAPTER 3. XPAAS RELEASE NOTES

The release notes for xPaaS docs have migrated to their own book on the Red Hat customer portal.

CHAPTER 4. COMPARING WITH OPENSHIFT ENTERPRISE 2

4.1. OVERVIEW

OpenShift Container Platform 3 is based on the OpenShift version 3 (v3) architecture, which is very different product than OpenShift version 2 (v2). Many of the same terms from OpenShift v2 are used in v3, and the same functions are performed, but the terminology can be different, and behind the scenes things may be happening very differently. Still, OpenShift remains an application platform.

This topic discusses these differences in detail, in an effort to help OpenShift users in the transition from OpenShift v2 to OpenShift v3.

4.2. ARCHITECTURE CHANGES

Gears Versus Containers

Gears were a core component of OpenShift v2. Technologies such as kernel namespaces, cGroups, and SELinux helped deliver a highly-scalable, secure, containerized application platform to OpenShift users. Gears themselves were a form of container technology.

OpenShift v3 takes the gears idea to the next level. It uses Docker as the next evolution of the v2 container technology. This container architecture is at the core of OpenShift v3.

Kubernetes

As applications in OpenShift v2 typically used multiple gears, applications on OpenShift v3 will expectedly use multiple containers. In OpenShift v2, gear orchestration, scheduling, and placement was handled by the OpenShift broker host. OpenShift v3 integrates Kubernetes into the master host to drive container orchestration.

4.3. APPLICATIONS

Applications are still the focal point of OpenShift. In OpenShift v2, an application was a single unit, consisting of one web framework of no more than one cartridge type. For example, an application could have one PHP and one MySQL, but it could not have one Ruby, one PHP, and two MySQLs. It also could not be a database cartridge, such as MySQL, by itself.

This limited scoping for applications meant that OpenShift performed seamless linking for all components within an application using environment variables. For example, every web framework knew how to connect to MySQL using the **`OPENSHIFT_MYSQL_DB_HOST`** and **`OPENSHIFT_MYSQL_DB_PORT`** variables. However, this linking was limited to within an application, and only worked within cartridges designed to work together. There was nothing to help link across application components, such as sharing a MySQL instance across two applications.

While most other PaaSes limit themselves to web frameworks and rely on external services for other types of components, OpenShift v3 makes even more application topologies possible and manageable.

OpenShift v3 uses the term "application" as a concept that links services together. You can have as many components as you desire, contained and flexibly linked within a project, and, optionally, labeled to provide grouping or structure. This updated model allows for a standalone MySQL instance, or one shared between JBoss components.

Flexible linking means you can link any two arbitrary components together. As long as one component can export environment variables and the second component can consume values from those

environment variables, and with potential variable name transformation, you can link together any two components without having to change the images they are based on. So, the best containerized implementation of your desired database and web framework can be consumed directly rather than you having to fork them both and rework them to be compatible.

This means you can build anything on OpenShift. And that is OpenShift's primary aim: to be a container-based platform that lets you build entire applications in a repeatable lifecycle.

4.4. CARTRIDGES VERSUS IMAGES

In OpenShift v3, an image has replaced OpenShift v2's concept of a cartridge.

Cartridges in OpenShift v2 were the focal point for building applications. Each cartridge provided the required libraries, source code, build mechanisms, connection logic, and routing logic along with a preconfigured environment to run the components of your applications.

However, cartridges came with disadvantages. With cartridges, there was no clear distinction between the developer content and the cartridge content, and you did not have ownership of the home directory on each gear of your application. Also, cartridges were not the best distribution mechanism for large binaries. While you could use external dependencies from within cartridges, doing so would lose the benefits of encapsulation.

From a packaging perspective, an image performs more tasks than a cartridge, and provides better encapsulation and flexibility. However, cartridges also included logic for building, deploying, and routing, which do not exist in images. In OpenShift v3, these additional needs are met by Source-to-Image (S2I) and configuring the template.

Dependencies

In OpenShift v2, cartridge dependencies were defined with **`Configure-Order`** or **`Requires`** in a cartridge manifest. OpenShift v3 uses a declarative model where pods bring themselves in line with a predefined state. Explicit dependencies that are applied are done at runtime rather than just install time ordering.

For example, you might require another service to be available before you start. Such a dependency check is always applicable and not just when you create the two components. Thus, pushing dependency checks into runtime enables the system to stay healthy over time.

Collection

Whereas cartridges in OpenShift v2 were colocated within gears, images in OpenShift v3 are mapped 1:1 with containers, which use pods as their colocation mechanism.

Source Code

In OpenShift v2, applications were required to have at least one web framework with one Git repository. In OpenShift v3, you can choose which images are built from source and that source can be located outside of OpenShift itself. Because the source is disconnected from the images, the choice of image and source are distinct operations with source being optional.

Build

In OpenShift v2, builds occurred in application gears. This meant downtime for non-scaled applications due to resource constraints. In v3, builds happen in separate containers. Also, OpenShift v2 build results used rsync to synchronize gears. In v3, build results are first committed as an immutable image and

published to an internal registry. That image is then available to launch on any of the nodes in the cluster, or available to rollback to at a future date.

Routing

In OpenShift v2, you had to choose up front as to whether your application was scalable, and whether the routing layer for your application was enabled for high availability (HA). In OpenShift v3, routes are first-class objects that are HA-capable simply by scaling up your application component to two or more replicas. There is never a need to recreate your application or change its DNS entry.

The routes themselves are disconnected from images. Previously, cartridges defined a default set of routes and you could add additional aliases to your applications. With OpenShift v3, you can use templates to set up any number of routes for an image. These routes let you modify the scheme, host, and paths exposed as desired, with no distinction between system routes and user aliases.

4.5. BROKER VERSUS MASTER

A master in OpenShift v3 is similar to a broker host in OpenShift v2. However, the MongoDB and ActiveMQ layers used by the broker in OpenShift v2 are no longer necessary, because **etcd** is typically installed with each master host.

4.6. DOMAIN VERSUS PROJECT

A project is essentially a v2 domain.

CHAPTER 5. REVISION HISTORY: RELEASE NOTES

5.1. MON JAN 22 2018

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHBA-2018:0113 - OpenShift Container Platform 3.6.173.0.96 Bug Fix and Enhancement Update.

5.2. WED JAN 10 2018

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHBA-2018:0076 - OpenShift Container Platform 3.6.173.0.83-10 Images Update.

5.3. THU DEC 14 2017

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHBA-2017:3438 - OpenShift Container Platform 3.6.173.0.83 Bug Fix and Enhancement Update.

5.4. WED DEC 06 2017

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHSA-2017:3389 - Moderate: OpenShift Container Platform 3.6.173.0.63 Security, Bug Fix, and Enhancement Update.

5.5. WED OCT 25 2017

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHBA-2017:3049 - OpenShift Container Platform 3.6.173.0.49 images update.

5.6. WED OCT 11 2017

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHBA-2017:2847 - OpenShift Container Platform 3.6.173.0.21 images update.

5.7. FRI SEP 08 2017

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHBA-2017:2642 - OpenShift Container Platform 3.6.1 bug fix and enhancement update.

5.8. THU AUG 31 2017

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHBA-2017:1829 - OpenShift Container Platform 3.6.173 Bug Fix Update.

5.9. THU AUG 17 2017

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for RHEA-2017:2475 - OpenShift Container Platform 3.6.173.0.5-4 Images Update.

5.10. WED AUG 09 2017

OpenShift Container Platform 3.6 Initial Release

Affected Topic	Description of Change
OpenShift Container Platform 3.6 Release Notes	Added release notes for initial release.