OpenShift Container Platform 3.9

リリースノート

Last Updated: 2018-11-14

法律上の通知

概要

目次

第1章 概要

以下の OpenShift Container Platform 3.9 リリースノートでは、新機能のすべて、以前のバージョンからの主な修正、一般公開バージョンの既知の問題についてまとめています。

1.1. バージョン管理ポリシー

OpenShift Container Platform では、アルファ API (通知なしに変更される可能性がある) およびベータ API (後方互換性の対応なしに変更される可能性が時々ある) 以外のサポートのある API はすべて確実に後方互換対応が取られています。

OpenShift Container Platform バージョンは、マスターとノードホストの間で一致する必要があります。クラスターのアップグレード時に、バージョンが一時的に一致しない場合は除きます。たとえば、3.9 のクラスターでは、マスターはすべて 3.9、ノードもすべて 3.9 でなければなりません。ただし、OpenShift Container Platform は、継続して新しいサーバーで以前の **oc** クライアントをサポートします。たとえば、3.4 の **oc** は 3.3、3.4 および 3.5 のサーバーで使用できます。

セキュリティーに関係のない理由で API を変更する場合には、以前の **oc** が更新できるように、最低でもマイナーバージョン 2 つ (例: 3.4、3.5、3.6) に影響が及びます。新機能を使用するには、より新しい **oc** が必要です。3.2 サーバーは、バージョン 3.1 の **oc**で使用できない機能が追加されている場合や、バージョン 3.2 の **oc** に、3.1 サーバーでサポートされていない機能が追加されている場合があります。

表1.1 互換性に関する表

	X.Y (**oc** クライアント)	**X.Y+N** [a] (**oc** クライアント)
X.Y (サーバー)	1	3
X.Y+N [a] (サーバー)	2	1

[a] **N** が 1 より大きい場合

1 完全に互換性があります。

2 **oc** クライアントはサーバー機能にアクセスできない場合があります。

3 **oc** クライアントは、アクセスするサーバーと互換性のないオプションや機能を提供する可能性があります。

第2章 OPENSHIFT CONTAINER PLATFORM 3.9 リリースノート

2.1. 概要

Red Hat OpenShift Container Platform では、設定や管理のオーバーヘッドを最小限に抑えながら、セキュアでスケーラブルなリソースをデプロイするクラウドアプリケーションプラットフォームを、開発者や IT 組織に提供します。OpenShift Container Platform は、Java、Ruby、および PHP など、幅広いプログラミング言語およびフレームワークをサポートしています。

Red Hat Enterprise Linux および Kubernetes でビルドされている OpenShift Container Platform は、現在のエンタープライズレベルのアプリケーションに、セキュアでスケーラブルなマルチテナント対応のオペレーティングシステムを提供するだけでなく、統合アプリケーションランタイムやライブラリーを提供します。OpenShift Container Platform を使用することで、組織はセキュリティー、プライバシー、コンプライアンス、ガバナンスの要件を満たすことができるようになります。

2.2. 本リリースについて

Red Hat OpenShift Container Platform バージョン 3.9 (RHBA-2018:0489) が利用できるようになりました。このリリースは、OpenShift Origin 3.9 をベースにしています。このトピックでは、OpenShift Container Platform 3.9 に含まれる新機能、変更、バグ修正、既知の問題についてまとめています。

Red Hat では、OpenShift Container Platform のバージョンと、Kubernetes をよりよく同期できるように、バージョン 3.7 の後に、OpenShift Container Platform 3.8 リリースを公開せず、直接 OpenShift Container Platform 3.9 をリリースすることにしました。今回のリリースがインストールおよびアップグレードのプロセスにどのような影響を与えるかについては、「インストール」を参照してください。

OpenShift Container Platform 3.9 は、RHEL 7.3、7.4 および 7.5 上でサポートされており、Docker 1.13 を含む Extras の最新パッケージで提供されます。また、Atomic Host 7.4.5 以降のバージョンでもサポートされます。**docker-latest** パッケージは非推奨になりました。

TLSV1.2 は、OpenShift Container Platform バージョン 3.4 以降でサポートされる、唯一のセキュリティーバージョンです。TLSV1.0 または TLSV1.1 をお使いの場合は、更新する必要があります。

初回インストールの場合は、『インストールと設定』の「クラスターのインストール」のトピックを参照してください。

以前のバージョンから今回のリリースにアップグレードする場合は、「クラスターのアップグレード」のトピックを参照してください。

2.3. 新機能および改良された機能

今回のリリースでは、以下のコンポーネントおよびコンセプトに関連する拡張機能が追加されました。

2.3.1. コンテナーのオーケストレーション

2.3.1.1. ソフトイメージプルーニング

今回のリリースでは、イメージをプルーニングする場合に、実際のイメージを削除する必要なく、etcd ストレージを更新するだけになりました。

`--keep-tag-revisions` および **`--keep-younger-than`** を実行するとより安全です。ソフトプルーニングを実行した後に、管理者は、ハードプルーニングを実行することもできます (レジストリーを読み取り専用モードに設定している限り実行しても安全です)。

2.3.1.2. Red Hat CloudForms Management Engine 4.6 コンテナー管理

OpenShift Container Platform 3.9 のインストール Playbook は、現在公開されている Red Hat CloudForms Management Engine (CFME) 4.6 をサポートするように更新されました。詳細情報は、「Red Hat CloudForms の OpenShift Container Platform へのデプロイ」トピックを参照してください。

さらに、今回のリリースには、以下の新機能および更新が追加されています。

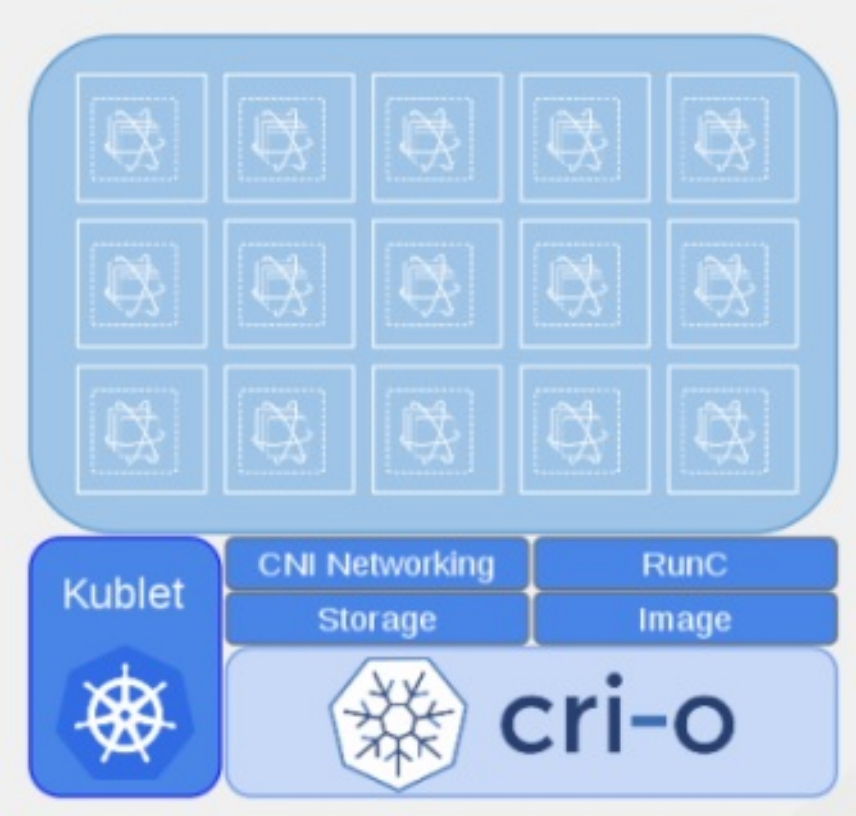
- OpenShift Container Platform テンプレートのプロビジョニング
- オフラインの OpenScapScans
- アラート管理: Prometheus (現時点ではテクノロジープレビュー) を選択して、CloudForms で使用できます。
- レポート機能の拡張
- プロバイダーの更新
- チャージバック機能の拡張
- UX の拡張

2.3.1.3. CRI-O v1.9

CRI-O は、軽量でネイティブの Kubernetes コンテナーのランタイムインターフェースです。CRI-O は設計的に、kubelet で必要なランタイム機能のみを提供します。また、CRI-O は Kubernetes の一部として設計されており、プラットフォームと同様に進化していきます。

CRI-O は以下を提供します。

- 最小限およびセキュアなアーキテクチャー
- 優れたスケーリングおよびパフォーマンス
- Open Container Initiative (OCI) または docker イメージを実行する機能
- 使い慣れた操作ツールおよびコマンド



docker とともに、CRI-O をインストール、実行するには、クラスターのインストール時に、「Ansible インベントリーファイル」の **[OSEv3:vars]** セクションに以下を設定します。

```
openshift_use_crio=true
openshift_crio_use_rpm=true (1)
```

(1) CRI-O は RPM でしかインストールできません。OpenShift Container Platform 3.9 の時点では、以前に提供されていた CRI-O のシステムコンテナーは、テクノロジープレビューから取り除かれました。

注記

atomic-openshift-node サービスは、CRI-O を使用する場合には、**docker** コンテナーベースではなく、RPM またはシステムコンテナーベースでなければなりません。**docker** コンテナーノードを使用する場合には、インストーラーにより CRI-O が使用されないように保護されるので、docker コンテナーがあると、インストールが停止してしまいます。

CRI-O の使用が有効になった場合に、**docker** と一緒にインストールされます。現在、これには、ビルドを実行して操作をレジストリーにプッシュする必要があります。時間が経つにつれ、一時的な **docker** ビルドがノードに累積する可能性があります。オプションで、以下を設定してガーベッジコレクションを有効にし、ビルドを消去する deamonset を追加することも可能です。

```
openshift_crio_enable_docker_gc=true
```

これが有効な場合には、デフォルトで全ノードにガーベッジコレクションが実行されます。以下を設定して、固有のノードに対する deamonset の実行を制限することもできます。

```
openshift_crio_docker_gc_node_selector={'runtime': 'cri-o'}
```

たとえば、上記を設定することで、**runtime: cri-o** のラベルが指定されたノードでのみ実行されるようになります。これは、「一部のノード」でのみ CRI-O を実行していて、他は **docker** のみを実行している場合に便利です。

CRI-O に関する詳細は、「アップストリームのドキュメント」を参照してください。

2.3.2. ストレージ

2.3.2.1. PV のサイズ変更

CNS glusterFS、Cinder および GCE PD 向けに、OpenShift Container Platform からオンラインで Persistent Volume Claim (永続ボリューム要求、PVC) を拡張できます。

1. **allowVolumeExpansion=true** に指定して、ストレージクラスを作成します。
2. PVC はストレージクラスを使用して、要求を送信します。
3. PVC は、増加サイズを新たに指定します。
4. 下層の PV のサイズが変更されます。

2.3.2.2. エンドツーエンドのオンライン拡張およびコンテナー化された GlusterFS PV のサイズ変更

CNS glusterFS ボリューム向けに、OpenShift Container Platform からオンラインで Persistent Volume Claim (永続ボリューム要求、PVC) を拡張できます。

これは、OpenShift Container Platform からオンラインで設定できます。以前のリリースでは、Heketi CLI からしか利用できませんでした。新しいサイズに PVC を設定して、PV のサイズ変更をトリガーします。これは、glusterFs でバックされる PV も完全に対応しています。Gluster ブロックの PV のサイズ変更は、RHEL 7.5 で追加されました。

1. ストレージクラスに **allowVolumeExpansion=true** を追加します。
2. 以下を実行します。

```
$ oc edit pvc claim-name
```

3. **spec.resources.requests.storage** フィールドを編集して、新しい値を入力します。

2.3.2.3. OpenShift Container Platform で利用可能な、Container Native Storage GlusterFS PV の消費量に関するメトリクス

Container Native Storage GlusterFS は、Prometheus または Query 経由で、(消費量を含む) ボリュームメトリクスを提供するように拡張されました。

メトリクスは、PVC エンドポイントから利用できます。これにより、何が割り当てられ、消費されたのかが視覚的に分かるようになります。以前のリリースでは、割り当てられた PV のサイズしか表示されませんでしたが、今回のリリースでは、実際に消費された量が分かるようになったので、必要に応じて、容量が無くなる前に拡張できるようになりました。これにより、管理者は必要に応じて、消費をベースに課金できるようになりました。

追加されたメトリクスの例は以下のとおりです。

- **kubelet_volume_stats_capacity_bytes**

- **kubelet_volume_stats_inodes**
- **kubelet_volume_stats_inodes_free**
- **kubelet_volume_stats_inodes_used**
- **kubelet_volume_stats_used_bytes**

2.3.2.4. OpenShift Container Platform の標準インストールで自動化された CNS デプロイメント

OpenShift Container Platform の標準インストーラーでは、CNS ブロックプロビジョナーデプロイメントが修正されて、CNS のアンインストール Playbook が追加されました。これにより、OpenShift Container Platform での CNS ブロックデプロイメントの問題が解決され、失敗した CNS インストールを削除できるようになりました。

CNS ストレージデバイスの詳細がインストーラーのインベントリーファイルに追加され、標準インストーラーは CNS、ファイル、ブロックプロビジョナー、レジストリー、使用準備のできた PV の設定およびデプロイメントの管理ができるようになりました。

2.3.2.5. テナント駆動型のストレージのスナップショット (テクノロジープレビュー)

テナント駆動型のストレージのスナップショット機能は現在 テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。

テナントは、アプリケーションデータのスナップショットを作成するために割り当てられる、永続ボリューム (PV) をバックする基盤のストレージ技術を活用できるようになりました。またテナントは、以前のアプリケーションから現在のアプリケーションに指定のスナップショットを復元できるようになりました。

外部プロビジョナーは、EBS、GCE pDisk および HostPath、 Cinder スナップショット API にアクセスするために使用します。このテクノロジープレビュー機能では、EBS と HostPath をテストしました。テナントは、手動で pod を停止して、起動する必要があります。

1. 管理者は、クラスター用に外部プロビジョナーを実行します。イメージは、Red Hat Container Catalog からのものです。
2. テナントは、PVC を作成して、サポートされるストレージソリューションの 1 つから PV を所有します。管理者は、以下を設定して、クラスターに新しい **StorageClass** を作成する必要があります。

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: snapshot-promoter
provisioner: volumesnapshot.external-storage.k8s.io/snapshot-
promoter
```

3. テナントは、**gce-pvc** という名前の PVC をもとにスナップショットを作成し、作成されたスナップショットの名前は **snapshot-demo** となります。

```
$ oc create -f snapshot.yaml

apiVersion: volumesnapshot.external-storage.k8s.io/v1
kind: VolumeSnapshot
```

```
metadata:
  name: snapshot-demo
  namespace: myns
spec:
  persistentVolumeClaimName: gce-pvc
```

4. これで、そのスナップショットの状態に、pod を復元できます。

```
$ oc create -f restore.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: snapshot-pv-provisioning-demo
  annotations:
    snapshot.alpha.kubernetes.io/snapshot: snapshot-demo
spec:
  storageClassName: snapshot-promoter
```

2.3.3. スケーリング

2.3.3.1. クラスターの制限

OpenShift Container Platform 3.9 の クラスター制限 に関するガイドが更新され、ご利用いただけるようになりました。

2.3.3.2. デバイスプラグイン (テクノロジープレビュー)

この機能は現在、テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。

デバイスプラグインを使用すると、カスタムコードを作成せずに特定のデバイスタイプ (GPU、InfiniBand、またはベンダー固有の初期化およびセットアップを必要とする他の同様のコンピューティングリソース) を OpenShift Container Platform Pod で使用できます。デバイスプラグインは、クラスター全体でハードウェアデバイスを消費するための一貫性のある移植可能なソリューションを提供します。デバイスプラグインはこれらのデバイスのサポートを拡張メカニズムでサポートします。これにより、これらのデバイスはコンテナーで利用可能となり、デバイスのヘルスチェックやセキュリティーが保護された状態でのデバイスの共有が可能になります。

デバイスプラグインは、ノード上で実行される gRPC サービスで (**`atomic-openshift-node.service`** の外部にあります)、特定のハードウェアリソースを管理します。

デバイスプラグインのコンセプトに関する詳細情報は、『開発者ガイド』を参照してください。

2.3.3.3. CPU マネージャー (テクノロジープレビュー)

CPU マネージャーは現在、テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。

CPU マネージャーは、CPU グループを管理して、ワークロードを固有の CPU に制限します。

CPU マネージャーは、以下のような属性が含まれるワークロードに有用です。

- できるだけ長い CPU 時間が必要な場合

- プロセッサーのキャッシュミスの影響を受ける場合
- レイテンシーが低いネットワークアプリケーションの場合
- 他のプロセスと連携し、単一のプロセッサーキャッシュを共有することで利点がある場合

詳細情報は、「CPU マネージャーの使用」を参照してください。

2.3.3.4. デバイスマネージャー (テクノロジープレビュー)

デバイスマネージャーは現在、テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。

ユーザーによっては、pod の定義にハードウェアデバイスのリソース制限を設定し、スケジューラーを使用して、これらのリソースが指定されたクラスター内でノードを検出する場合があります。同時に、Kubernetes には、ハードウェアベンダーが Kubernetes 内のコアコードを強制的に変更せずに kubelet にリソースを宣言する方法が必要です。

kubelet には、プラグインで拡張可能なデバイスマネージャーが追加されました。ドライバーサポートは、ノードレベルで読み込ませます。次に、ドライバーに表示される要求済みのハードウェアリソースを停止/開始/アタッチ/割り当てる要求をリッスンするプラグインを、ユーザーまたはベンダーが記述します。このプラグインは、daemonSet 経由で全ノードにデプロイされます。

詳細情報は、「デバイスマネージャーの使用」を参照してください。

2.3.3.5. ヒュージページ (テクノロジープレビュー)

ヒュージページは現在、テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。

メモリーは、ページと呼ばれるブロックで管理されます。多くのシステムでは、1 ページは 4Ki です。メモリー 1Mi は 256 ページに、メモリー 1Gi は 256,000 ページに相当します。CPU には、内蔵のメモリー管理ユニットがあり、ハードウェアでこのようなページリストを管理します。トランスレーションルックアサイドバッファー (TLB: Translation Lookaside Buffer) は、仮想から物理へのページマッピングの小規模なハードウェアキャッシュのことです。ハードウェアの指示で渡された仮想アドレスが TLB にあれば、マッピングをすばやく決定できます。ない場合には、TLB ミスが発生し、システムは速度が遅く、ソフトウェアベースのアドレス変換にフォールバックされ、パフォーマンスの問題が発生します。TLB のサイズが固定されているので、ページサイズを増やすしか、TLB ミスの割合を減らす方法はありません。

ヒュージページとは、4Ki より大きいメモリーページのことです。x86_64 アーキテクチャーでは、2Mi と 1Gi の 2 つが一般的なヒュージページサイズです。別のアーキテクチャーではサイズは異なります。ヒュージページを使用するには、アプリケーションが認識できるように、コードを書き込む必要があります。Transparent Huge Pages (THP) は、アプリケーションによる認識なしに、ヒュージページの管理を自動化しようとしますが、制約があります。特に、ページサイズは 2Mi に制限されます。THP では、THP のデフラグが原因で、メモリー使用率が高くなり、断片化が起こり、パフォーマンスの低下に繋がり、メモリーページがロックされてしまう可能性があります。このような理由から、アプリケーションは THP ではなく、事前割り当て済みのヒュージページを使用するように設計 (また推奨) される場合があります。

OpenShift Container Platform では、pod のアプリケーションが事前割り当て済みのヒュージページを割り当て、消費できます。

詳細情報は、「ヒュージページの管理」を参照してください。

2.3.4. ネットワーク

2.3.4.1. namespace 別に半自動的に指定される egress IP 機能

外部のファイアウォールがパケットに関連付けられたアプリケーションを認識できるように、プロジェクトから発信される外部接続はすべて、単一で固定のソース IP アドレスを共有し、この IP を使用して全トラフィックを送信します。

namespace 別の egress IP 自動設定機能を実装するプロセスでは、最初の半分で「トラフィック」側の実装が行われるので、**半自動** とされています。egress IP が自動指定された namespace は、その IP 経由で全トラフィックを送信します。ただし、「管理」側の実装はされません。ノードへの egress IP の設定は自動的に行われないので、管理者は手動でこの設定を行う必要があります。

詳細情報は、「ネットワークの管理」を参照してください。

2.3.4.2. ルーターで消費するための独自の HAProxy RPM のサポート

負荷の多い状態で実行されるルート設定の変更およびプロセスのアップグレードでは通常、特定のサービスを停止、開始するシーケンスが必要となり、一時的にサービスが停止されていました。

OpenShift Container Platform 3.9 では、HAProxy 1.8 は更新とアップグレードを別物とみなさないので、新規設定には新規プロセスが使用され、リッスンするソケットのファイル記述子は、以前のプロセスから新しいプロセスに移行されるので、接続は切断されなくなりました。変更はシームレスに行われ、今後 HTTP/2 などにも対応できるようになっています。

2.3.5. マスター

2.3.5.1. StatefulSets、DaemonSets および Deployments のサポート追加

OpenShift Container Platform では、statefulsets、daemonsets および deployments がテクノロジープレビューではなく、サポートありの安定版になりました。

2.3.5.2. 中央監査機能

管理者が確認を希望する監査項目が追加されました。以下に例を示します。

- イベントのタイムスタンプ
- エントリーを生成したアクティビティー
- 呼び出された API エンドポイント
- HTTP の出力
- アクティビティーが原因で変更された項目と、変更の内容
- アクティビティーを開始したユーザー名
- (できる限り) イベントが発生した namespace の名前
- イベントのステータス (成功または失敗)

管理者が追跡を希望する監査項目が追加されました。以下に例を示します。

- 不正アクセスの試みなど、Web インターフェースからのユーザーログインおよびログアウト (セッションのタイムアウトを含む)
- アカウントの作成、変更または削除

- アカウントロールまたはポリシーの割り当てまたは割り当て解除
- pod のスケーリング
- 新規プロジェクトまたはアプリケーションの作成
- ルートおよびサービスの作成
- ビルドおよび/またはパイプラインのトリガー
- 永続ボリューム要求 (PVC) の追加または削除

master-config file で監査機能を設定して、**master-config** サービスを再起動します。

```
auditConfig:
  auditFilePath: "/var/log/audit-ocp.log"
  enabled: true
  maximumFileRetentionDays: 10
  maximumFileSizeMegabytes: 10
  maximumRetainedFiles: 10
  logFormat: json
  policyConfiguration: null
  policyFile: /etc/origin/master/audit-policy.yaml
  webHookKubeConfig: ""
  webHookMode:
```

ログ出力の例:

```
{"kind":"Event","apiVersion":"audit.k8s.io/v1beta1","metadata":
{"creationTimestamp":"2017-09-
29T09:46:39Z"},"level":"Metadata","timestamp":"2017-09-
29T09:46:39Z","auditID":"72e66a64-c3e5-4201-9a62-
6512a220365e","stage":"ResponseComplete","requestURI":"/api/v1/securitycon
textconstraints","verb":"create","user":
{"username":"system:admin","groups":["system:cluster-
admins","system:authenticated"]},"sourceIPs":["10.8.241.75"],"objectRef":
{"resource":"securitycontextconstraints","name":"scc-
lg","apiVersion":"/v1"},"responseStatus":{"metadata":{},"code":201}}
```

2.3.5.3. oc ステータスへのデプロイメントサポートの追加

oc status コマンドでは、現在のプロジェクトの概要が分かります。これにより、デプロイメントセットがネスト化されているダウンストリームの DeploymentConfigs で見られるのと同様のアップストリームデプロイメントの出力が表示されます。

```
$ oc status
In project My Project (myproject) on server https://127.0.0.1:8443

svc/ruby-deploy - 172.30.174.234:8080
  deployment/ruby-deploy deploys istag/ruby-deploy:latest <-
    bc/ruby-deploy source builds https://github.com/openshift/ruby-ex.git
on istag/ruby-22-centos7:latest
      build #1 failed 5 hours ago - bbb6701: Merge pull request #18 from
```

```
durandom/master (Joe User <joeuser@users.noreply.github.com>)
    deployment #2 running for 4 hours - 0/1 pods (warning: 53 restarts)
    deployment #1 deployed 5 hours ago
```

これを OpenShift Container Platform 3.7 の出力と比較します。

```
$ oc status
In project dc-test on server https://127.0.0.1:8443

svc/ruby-deploy - 172.30.231.16:8080
  pod/ruby-deploy-5c7cc559cc-pvq9l runs test
```

2.3.5.4. Dynamic Admission Controller Follow-up (テクノロジープレビュー)

Dynamic Admission Controller Follow-up は現在、テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。

受付コントローラーは、要求が認証および承認されてから、オブジェクトの永続化される前に Kubernetes API サーバーへの要求をインターセプトするコードです。ユースケースの例として、pod リソースおよびセキュリティーの対応の変更などが挙げられます。

詳細情報は、「Custom Admission Controllers」を参照してください。

2.3.5.5. Feature Gates

プラットフォームの管理者は、プラットフォーム全体で特定の機能をオフにできるようになりました。これにより、実稼働のクラスターでのアルファ、ベータまたはテクノロジープレビュー機能へのアクセスを制御しやすくなります。

Feature gates は、マスターおよび kubelet 設定ファイルで key=value ペアを使用してブロックする機能を記述します。

コントロールプレーン**: master-config.yaml**

```
kubernetesMasterConfig:
  apiServerArguments:
    feature-gates:
    - CPUManager=true
```

kubelet: node-config.yaml

```
kubeletArguments:
  feature-gates:
  - DevicePlugin=true
```

2.3.6. インストール

2.3.6.1. Playbook パフォーマンスの向上

OpenShift Container Platform 3.9 では、パフォーマンスを向上するため、Playbook を大幅にリファクタリングおよび再構成しました。これには以下が含まれます。

- ファクトの収集、共通の依存関係を初期化 play にプッシュするように playbook が再構成され、ロールがコンピュートされた値にアクセスする必要があるたびに呼び出されるのではなく、1 度呼び出すだけで良くなりました。
- playbook がリファクタリングされ、対象の playbook に関連するものだけを処理するように playbook の対応するホストが制限されるようになりました。

2.3.6.2. クイックインストール (非推奨)

クイックインストールは OpenShift Container Platform 3.9 では非推奨となっており、今後のリリースでは完全に削除されます。

クイックインストールでは、3.9 のみをインストールでき、3.7 または 3.8 から 3.9 へのアップグレードには使用できません。

2.3.6.3. 3.7 から 3.9 へのコントロールプレーンのアップグレード自動化

インストーラーは、コントロールプレーンを 3.7 から 3.8 へ、3.8 から 3.9 に順を追って自動的にアップグレードし、ノードは 3.7 から 3.9 にアップグレードします。

コントロールプレーンのコンポーネント (API、コントローラー、コントローラープレーンホストのノード) は 3.7 から 3.8、さらに 3.9 にシームレスにアップグレードされます。データの移行は、OpenShift Container Platform 3.8 および 3.9 のコントロールプレーンのアップグレード前と後に行われます。他のコントロールプレーンのコンポーネント (ルーター、レジストリー、サービスカタログ、ブローカー) は OpenShift Container Platform 3.7 から 3.9 にアップグレードされ、ノード (ノード、docker、ovs) は、一度だけドレインされるだけで、OpenShift Container Platform 3.7 から 3.9 に直接アップグレードされます。この状態でアップグレードプロセスを一時停止する必要がある場合には、OpenShift Container Platform 3.7 ノードは、3.8 のマスターに対していつまでも操作が行われます。ロギングおよびメトリクスは OpenShift Container Platform 3.7 から 3.9 にアップグレードされます。

コントロールプレーンとノードは別個でアップグレードすることを推奨します。オールインワン (all-in-one) の playbook でアップグレードすることも可能ですが、ロールバックがより困難になります。playbook では、OpenShift Container Platform 3.8 のクリーンインストールはできません。

詳細情報は、「クラスターのアップグレード」を参照してください。

2.3.7. メトリクスとロギング

2.3.7.1. システムログ用の Jourald およびコンテナーログ用の JSON ファイル

Docker ログドライバーは、全ノードのデフォルトとして、**`json-file`** に設定されています。Docker **`log-driver`** は、**`journal`** に設定可能ですが、ジャーナルドライバーにはログ回転スロットルがありません。そのため、不正なコンテナーから DoS 攻撃を受けるリスクが常にあります。

Fluentd は自動的に、コンテナーのランタイムが使用するログドライバーを判断します (**`journald`** または **`json-file`**)。Fluentd は journald および **/var/log/containers** (**`log-driver`** が **`json-file`** に設定されている場合) から常にログを読み込みます。Fluentd は **/var/log/messages** からのログの読み込みはなくなりました。

詳細情報は、「コンテナーログの累積」 を参照してください。

2.3.7.2. fluentd 向けの syslog 出力プラグイン (テクノロジープレビュー)

fluentd 向けの syslog 出力プラグインは現在、テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。

システムおよびコンテナーログは OpenShift Container Platform ノードから外部のエンドポイントに syslog プロトコルを使用して送信できます。fluentd syslog 出力プラグインはこの機能をサポートします。

重要

syslog 経由で送信したログは暗号化されていないので、セキュアではありません。

詳細情報は、「外部 syslog サーバーへのログの送信」を参照してください。

2.3.7.3. Prometheus (テクノロジープレビュー)

Prometheus は依然として、テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。Prometheus、AlertManager および AlertBuffer のバージョンは更新され、node-exporter も追加されました。

- prometheus 2.1.0
- Alertmanager 0.14.0
- AlertBuffer 0.2
- node_exporter 0.15.2

OpenShift Container Platform クラスターに Prometheus をデプロイし、Kubernetes およびインフラストラクチャーのメトリクスを収集して、アラートを取得できます。Prometheus web ダッシュボードでメトリクスおよびアラートを表示、照会できます。または、独自の Grafana を用意して、Prometheus に連携させることも可能です。

詳細情報は、「OpenShift 上の Prometheus」を参照してください。

2.3.8. 開発者の体験

2.3.8.1. Jenkins メモリー使用量の改善

以前のリリースでは、Jenkins ワーカー pod はメモリーを過剰または過小に消費することが頻繁にありました。今回のリリースでは、起動スクリプトがインテリジェントに、適切に設定された pod の制限および環境変数を確認して、JVM の起動時に制限が尊重されるようになりました。

2.3.8.2. CLI プラグイン (テクノロジープレビュー)

CLI プラグインは現在、テクノロジープレビュー として提供されており、実稼働環境のワークロードには適していません。

通常、**プラグイン** または **バイナリー拡張** と呼ばれるこの機能は、利用可能なデフォルトの **oc** コマンドを拡張するので、新たなタスクが実行できます。

CLI の拡張のインストールおよび記述の方法に関する情報は、「Extending the CLI」を参照してください。

2.3.8.3. buildconfig Defaulter 経由でデフォルト の tolerations を指定する機能

以前のリリースでは、ビルド固有のノードに配置できるように、ビルドされた pod でデフォルトの toleration を設定する方法がありませんでした。ビルドの Defaulter が更新され、toleration の値を指定できるようになり、作成時にビルド pod に適用されます。

詳細情報は、「グローバルビルドのデフォルト設定および上書きの設定」を参照してください。

2.3.8.4. デフォルトのハードエビクションしきい値

OpenShift Container Platform は、**`eviction-hard`** の以下のデフォルト設定を使用します。

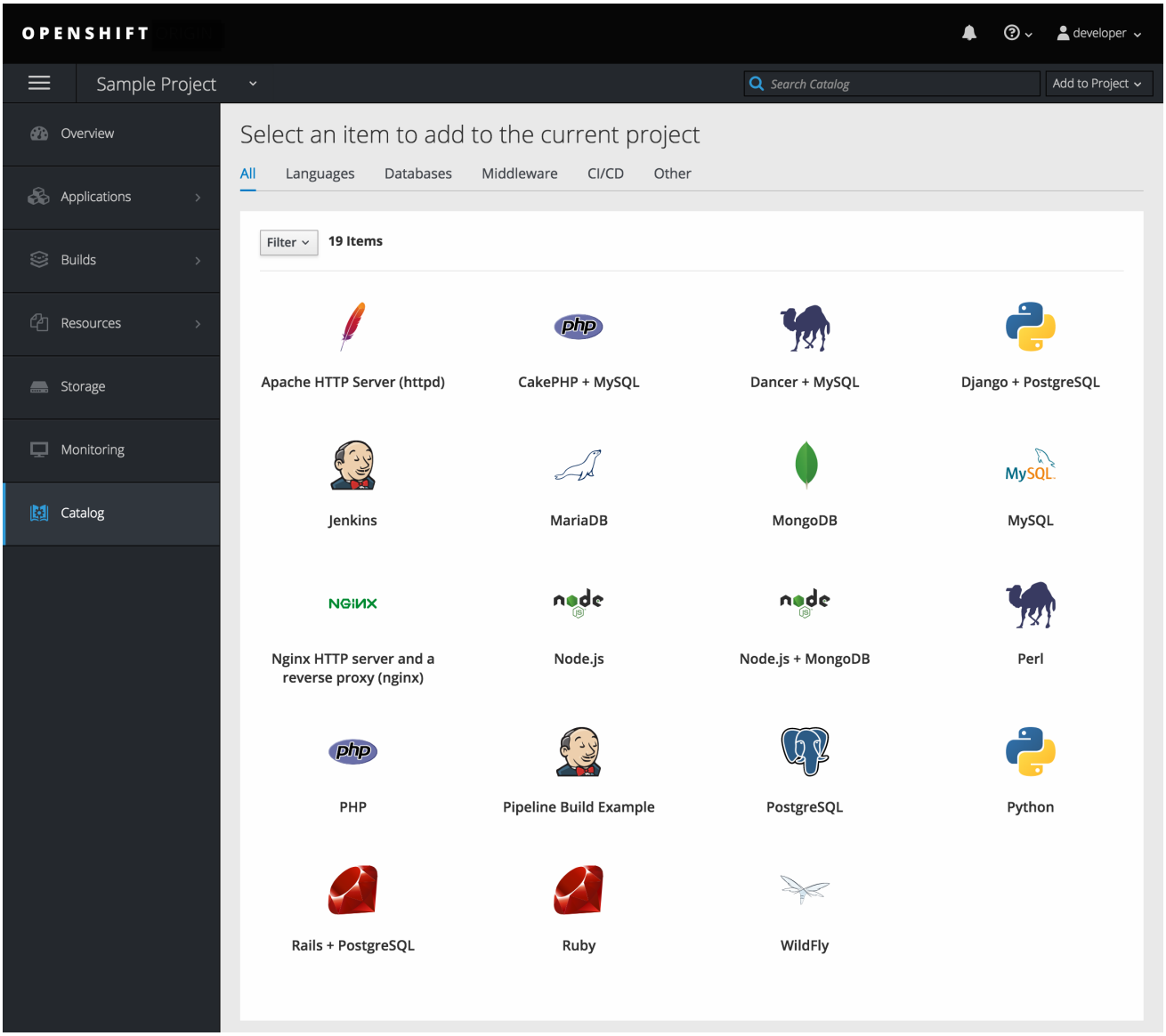
```
...
kubeletArguments:
  eviction-hard:
  - memory.available<100Mi
  - nodefs.available<10%
  - nodefs.inodesFree<5%
  - imagefs.available<15%
...
```

詳細情報は、「Out of Resource (リソース不足) エラーの処理」 を参照してください。

2.3.9. Web コンソール

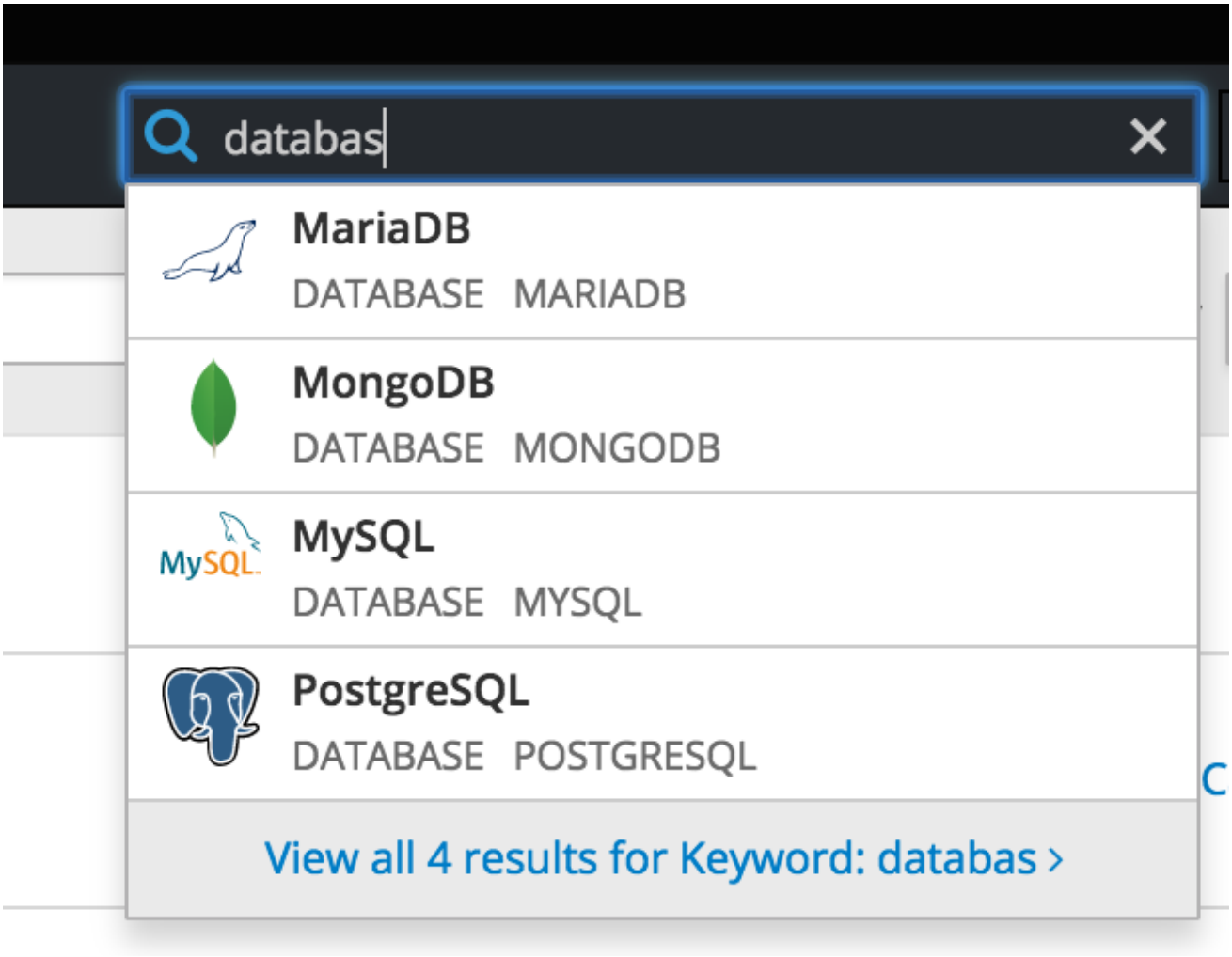
2.3.9.1. プロジェクトビューからカタログへの移動

左側のナビゲーションにある **Catalog** をクリックして、プロジェクト内からカタログにすばやく移動できます。



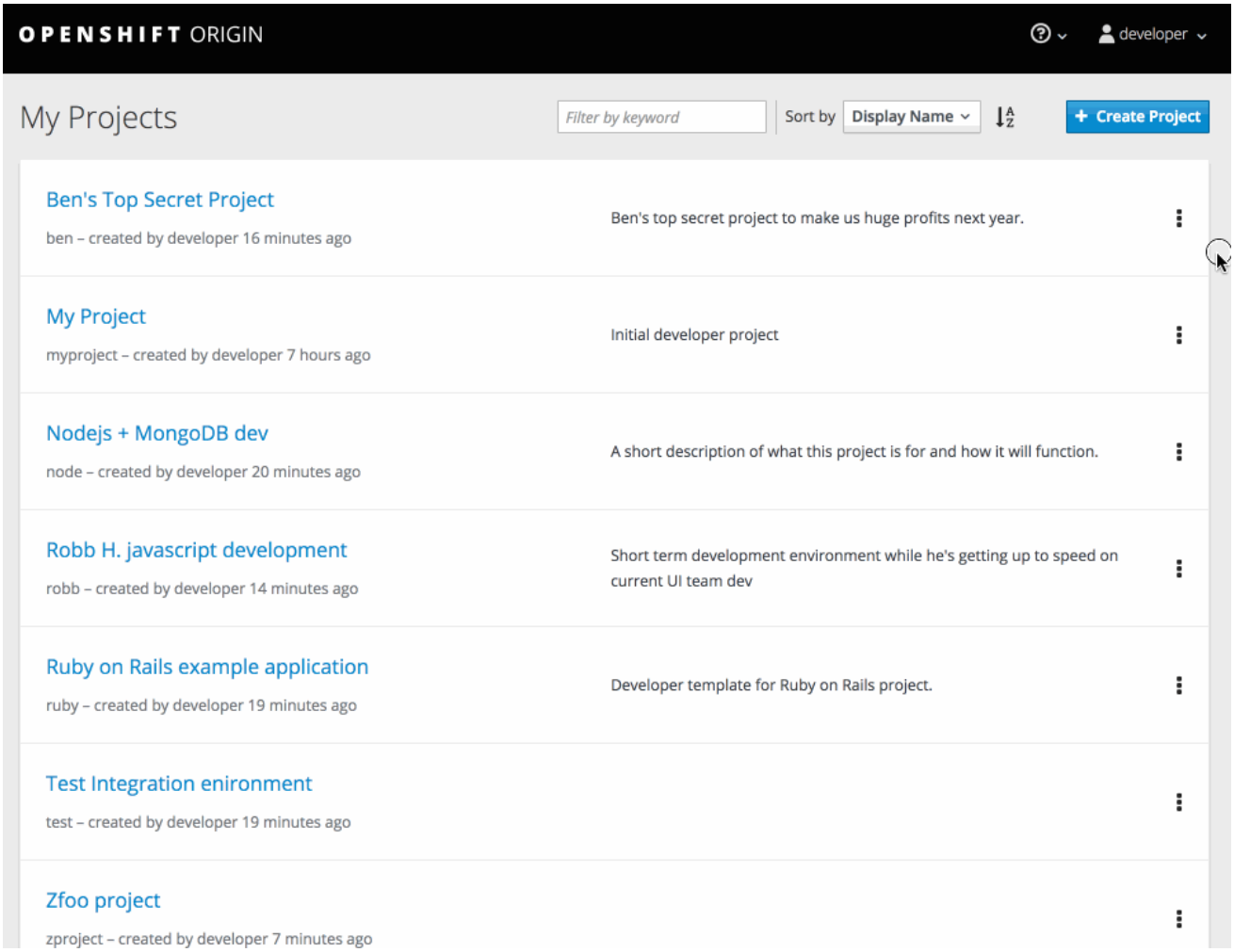
2.3.9.2. プロジェクトビューからカタログのクイック検索

プロジェクトビューからすばやくサービスを検索するには、検索基準を入力します。



2.3.9.3. 任意のホームページの選択

ログイン後に直接特定のページに移動できるようになりました。アカウントのドロップダウンメニューからメニューにアクセスして、希望のオプションを選択してから、再度ログインしなおします。



2.3.9.4. 設定可能な非アクティブタイムアウト

設定したタイムアウトの期間が経過した後にユーザーをログアウトするように、Web コンソールを設定できるようになりました。デフォルトは `0` (タイムアウトなし) です。Ansible の変数 を分単位で設定します。

```
openshift_web_console_inactivity_timeout_minutes=n
```

2.3.9.5. 別の pod としての Web コンソール

Web コンソールは、API サーバーから分離されました。Web コンソールはコンテナーイメージとしてパッケージされ、pod としてデプロイされます。ConfigMap で設定します。変更は自動的に検出されます。

マスターはスケジュール可能で、Web コンソールのデプロイメントが機能するようにスケジュール可能にする必要があります。

2.4. 主な技術的変更

OpenShift Container Platform 3.9 では、主に以下のような技術的な変更が加えられました。

手動でのアップグレードプロセスの非対応化
OpenShift Container Platform 3.9 の時点で、「手動アップグレード」のサポートがなくなりました。今後のリリースでは、このプロセスは削除されます。

スケジュール可能なノードとしてマスターをデフォルトでマーク付け

以前のバージョンの OpenShift Container Platform では、マスターホストには、インストーラーによってデフォルトで、ホストに新しい pod が配置できないように、スケジュール対象外のマークが付けられましたが、OpenShift Container Platform 3.9 では、マスターはインストールおよびアップグレード時に自動的に、マスターにはスケジュール対象のマークが付けられます。これは主に、Web コンソールが、マスターの一部としての実行に使用されるのではなく、マスターにデプロイされている pod として実行できるように変更されました。

デフォルトで設定されるデフォルトノードセレクターセットおよび自動ノードラベル設定

OpenShift Container Platform 3.9 以降、マスターはデフォルトでスケジュール対象ノードとしてマークされるようになり、デフォルトのノードセレクターは、クラスターのインストールおよびアップグレード時に設定されるようになりました (マスター設定ファイルの **`projectConfig.defaultNodeSelector`** フィールドにpod の配置時にデフォルトでプロジェクトがどのノードを使用するかを判断するために定義します。以前のリリースでは、このフィールドはデフォルトで空のままでした)。**`osm_default_node_selector`** Ansible 変数を使用して上書きしない限り、**`node-role.kubernetes.io/compute=true`** に設定されます。

さらに **`osm_default_node_selector`** の設定の有無に拘わらず、以下のように、インストールおよびアップグレード時にインベントリーファイルに定義されたホストに自動的にラベルが付けられます。

- マスター以外で、非専用のインフラストラクチャーノードホスト (デフォルトでは **`region=infra`** ラベルが指定されたもの) は、**`compute`** ノードロールを割り当てる、**`node-role.kubernetes.io/compute=true`** のラベルが付けられます。
- マスターノードは、**`master`** ノードロールを割り当てる **`node-role.kubernetes.io/master=true`** のラベルが付けられます。

これにより、pod の配置を決定するときに、デフォルトのノードセレクターに選択可能なノードがあるようにします。詳細情報は、「ノードホストラベルの設定」を参照してください。

Ansible は rhel-7-server-ansible-2.4-rpms チャネルからインストールする必要がある

OpenShift Container Platform 3.9 以降で、Ansible は RHEL サブスクリプションに含まれる **`rhel-7-server-ansible-2.4-rpms`** チャネル経由でインストールする必要があります。

複数の oc secrets サブコマンドの非推奨化

OpenShift Container Platform 3.9 は、**`oc create secret`** が選択され、以下の **`oc secrets`** サブコマンドが非推奨になりました。

- **`new`**
- **`new-basicauth`**
- **`new-dockercfg`**
- **`new-sshauth`**

インストーラーの template_service_broker_prefix と template_service_broker_image_name in the Installer のデフォルト値の更新

インストーラーの **`template_service_broker_prefix`** および **`template_service_broker_image_name`** のデフォルト値が他の設定と一貫性を保てるように更新されました。

以前の値:

- **`template_service_broker_prefix="registry.example.com/openshift3/"`**

- **template_service_broker_image_name="ose-template-service-broker"**

新しい値:

- **template_service_broker_prefix="registry.example.com/openshift3/ose-"**
- **template_service_broker_image_name="template-service-broker"**

openshift-anisble にある特定のタスクおよび **playbook** 上の **'become: no'** のインスタンスが複数削除される
ユーザーに柔軟性を提供するため、**openshift-anisble** 内の特定のタスクおよび Playbook にある **become: no** のインスタンスが複数削除されました。これらのステートメントは主に、Ansible を実行中のホストに一時ファイルを作成するために、**local_action** および **delegate_to: localhost** コマンドに適用されていました。

パスワードなしの **sudo** が使用できないホストから Ansible を実行する場合に、**ansible-playbook** に **-b** (**become**) コマンドライン切り替えを実行するか、インベントリーまたは **group_vars** でローカルホストに **ansible_become=True** が適用されると、これらのコマンドの一部が失敗する可能性があります。

openshift-ansible Play を実行時にローカルホストでは特権昇格の必要はありません。

ターゲットホスト (OpenShift Container Platform のデプロイ先) で **become** をする必要がある場合には、インベントリーまたは **group_vars**/**host_vars** で、対象のホストまたはグループに **ansible_become=True** を追加することを推奨します。

ユーザーがローカルホストで root として実行している場合、または become を使用せずにリモートホストで root ユーザーに接続している場合には、変化は見られないはずです。

名前指定なしの仕様
名前指定のないイメージの仕様がデフォルトで **docker.io** に設定され、別のレジストリーに対して解決するには API サーバーの設定が必要です。

batch/v2alpha1 ScheduledJob オブジェクエとの非対応化
batch/v2alpha1 ScheduledJob オブジェクトのサポートがなくなりました。代わりに CronJobs を使用してください。

autoscaling/v2alpha1 API グループの削除
autoscaling/v2alpha1 API グループが削除されました。

ノードの起動にはスワップを無効にする必要あり
OpenShift Container Platform 3.9 を新規インストールする場合には、スワップを無効にすることを強く推奨します。OpenShift Container Platform 3.8 では、OpenShift Container Platform の起動ノードではスワップを無効にする必要がありました。Ansible ノードのインストールではすでに無効にされています。

oadm コマンドの非推奨化
oadm コマンドは非推奨になりました。代わりに、**oc adm** を使用してください。

StatefulSets、**DaemonSets** および **Deployments** の完全サポート
DaemonSet、**Deployment**、**ReplicaSet** および **StatefulSet kinds** で構成されるコアのワークロードAPI は **apps/v1** グループバージョンで GA 安定版に昇格したため、**apps/v1beta2** グループバージョンが非推奨になり、全新規コードは apps/v1 group バージョンのものを使用する必要があります。OpenShift Container Platform では、statefulsets、daemonsets および deployments は安定版になり、サポートもされるようになりました。

管理者ソリューションガイドの削除
OpenShift Container Platform 3.9 では、管理者ソリューションガイドが OpenShift Container Platform ドキュメントから削除されました。代わりに 『DAY 2 操作ガイド』を参照してください。

2.5. バグ修正

今回のリリースでは、以下のコンポーネントのバグが修正されました。

ビルド

- 以前のリリースでは、ビルドの起動時に出力イメージのプッシュに使用するシークレットを、ビルドが選択していました。プロジェクトのデフォルトサービスアカウントのシークレットが作成される前に、ビルドが起動された場合には、ビルドによりイメージのプッシュに適したシークレットを見つけられない可能性があり、イメージのプッシュ時にビルドに失敗していました。今回の修正で、デフォルトのサービスアカウントのシークレットが作成されるまでビルドが保持されるようになり、デフォルトのシークレットがイメージのプッシュに適していれば、そのシークレットを使用でき、今後も使用されます。その結果、デフォルトのシークレットが生成される前にビルドが作成されても、新規作成されたプロジェクトの初期ビルドにで失敗するリスクがなくなりました。(**BZ#1333030**)

コマンドラインインターフェース

- マスターの **systemd** ユニットは、診断の更新なしに変更されていたので、診断が存在しないマスターの **systemd** ユニットを警告なしに確認し、問題が報告されませんでした。今回の修正では、診断で正しいマスターユニット名および、マスターの **systemd** ユニットの問題が確認され、ログを見つけることができます。(**BZ#1378883**)

コンテナー

- コンテナーが別のコンテナーと namespace を共有する場合に、namespace のパスが共有されました。最初のコンテナーで **exec** コマンドを実行すると、ファイルに保存された namespace パスのみを読み込み、これらの namespace を連結していたので、2 番目のコンテナーがすでに停止されている場合には、最初のコンテナーの **exec** コマンドに失敗しました。そのため、今回の修正では、コンテナーが namespace を共有している場合でも namespace のパスが保存されるようになりました。(**BZ#1510573**)

イメージ

- Docker には、OpenShift Jenkins イメージに影響を与える既知の「zombie process」現象があり、「zombie process」現象が累積してオペレーティングシステムレベルのリソースがなくなってしまいました。今回の修正では、OpenShift Jenkins イメージは Docker イメージの **init** 実装の 1 つを活用して Jenkins を起動し、「zombie child processes」を監視、処理するようになりました。(**BZ#1528548**)

- スケジューラー実装に問題があるため、**ScheduledImageImportMinimumIntervalSeconds** 設定が正しく準拠されず、OpenShift Container Platform は、スケジュールされたイメージを不正な間隔でインポートしようとしていましたが、この問題は解決されました。(**BZ#1543446**)

- 以前のリリースでは、スケジュール指定の有無に拘わらず、イメージストリームにタグがスケジュール対象とマークされている場合には、OpenShift はイメージストリームで誤ってすべてのタグを再インポートしていましたが、この動作は修正されました。(**BZ#1515060**)

イメージレジストリー

- 署名インポーターは、内部レジストリーから認証情報なしに署名をインポートしようとし、匿

名ユーザーが SAR 要求を使用して署名を取得できるかどうかをレジストリーに確認させていました。今回のバグの修正により、内部レジストリーと署名インポーターは同じストレージで作業するので、署名インポーターは、内部レジストリーをスキップし、SAR の要求がなくなりました。(**BZ#1543122**)

- パス内のコンポーネント数が確認されず、データがストレージに配置されるにも拘わらず、データベースに書き込まれませんでした。今回のバグ修正では、初期の段階でのパスの確認が追加されました。(**BZ#1528613**)

インストーラー

- Kubernetes サービス IP アドレスは、インストール中に docker-registry の **no_proxy** リストに追加されず、内部レジストリーの要求が強制的にプロキシーを使用するため、ログインや内部レジストリーへのプッシュができなくなっていました。インストーラーは、Kubernetes サービス IP を **no_proxy** リストに追加するように変更されました。(**BZ#1504464**)

- インストーラーは、不正な efs-provisioner イメージをプルするため、プロビジョナー pod のインストールでデプロイに失敗していました。インストーラーが正しいイメージをプルするように変更されました。(**BZ#1523534**)

- カスタムのレジストリーを使用して OpenShift Container Platform をインストールする場合には、インストーラーではデフォルトのレジストリーが使用されていました。レジストリーコンソールのデフォルトイメージが、完全修飾イメージ **registry.access.redhat.com/openshift3/registry-console** として定義されるようになったので、カスタムレジストリーが **oreg_url** 経由で指定され、イメージストリームがそのカスタムレジストリーを使用するように変更された場合には、レジストリーコンソールでもこのカスタムレジストリーが使用されるようになりました。(**BZ#1523638**)

- **redeploy-etcd-ca.yml** playbook を実行しても、etcd システムコンテナーが使用する **ca.crt** が更新されませんでした。このコードは、Playbook で **/etc/etcd/ca.crt** の etcd ca.crt が正しく更新されるように、変更されました。(**BZ#1466216**)

- glusterblock で CNS/CRS のデプロイメントに成功した後には、glusterblock を使用して OpenShift Container Platform ロギングおよびメトリックスを、バックエンドストレージとしてデプロイし、フォールトトレラントで、分散型の永続ストレージを実現できるようになりました。(**BZ#1480835**)

- 3.6 から 3.7 にアップグレードすると、Hawkular OpenShift Agent pod を無効にしても、アップグレード後に HOSA pod がまだデプロイされていました。新しい playbook **uninstall_hosa.yaml** が作成され、Ansible インベントリーファイルで **openshift_metrics_install_hawkular_agent=false** が指定されている場合に OpenShift Container Platform クラスターから HOSA が削除されるようになりました。(**BZ#1497408**)

- ブローカーのレジストリー認証情報が ConfigMap に保存されていたため、機密な認証情報がプレーンテキストで公開されていました。認証情報を保存するためのシークレットが作成され、レジストリーの認証情報がプレーンテキストで表示されなくなりました。(**BZ#1509082**)

- 不正な名前が指定されているので、アンインストール playbook では **tuned-profiles-atomic-openshift-node** パッケージが削除されませんでした。この playbook が修正され、OpenShift Container Platform のアンインストール時にこのパッケージが削除されるようになりました。(**BZ#1509129**)

- Jinja コードで **openshift_hosted_registry_storage_volume_size** パラメーター を指定して、インストーラーを実行した場合には、永続ボリュームの作成時にインストールに失敗していました。このコードが修正され、正しく Jinja コードを変換するようになりました。

(BZ#1518386)

- インターネット接続なしでインストールを行う場合に、サービスカタログは、設定済みのレジストリーからイメージを取り除こうとしていました。これにより、インターネット接続なしのインストール時にレジストリーが利用できないので、インストールに失敗していました。インストーラーの **imagePullPolicy** が **ifNotPresent** に変更されました。イメージが存在する場合には、サービスカタログは再度イメージをプルしようとはせずに、サービスカタログがインターネット接続なしで続行されます。(BZ#1524805)

- SSH プロキシーを設定してホストをプロビジョニングする場合には、マスターは UP のマークが付いて起動することはありませんでした。今回のバグ修正で、このタスクは、SSH プロキシー設定に従う Ansible モジュールを使用するように変更されましたので、Ansible はホストに接続でき、UP とマークされるようになりました。(BZ#1541946)

- HTTPS 環境では、playbook が **--noproxy** オプションを指定せずに cURL を使用して API サーバーに問い合わせをしようとするため、サービスカタログのインストールが失敗していました。Playbook のコマンドに **--noproxy** が含まれるように変更され、インストーラーが想定どおりに実行されるようになりました。(BZ#1544645)

- 以前のリリースでは、アップグレードや再実行された場合に、Elasticsearch データセンターのストレージタイプが保持されませんでした。今回のバグ修正では、デフォルトで (インベントリー変数が指定されている場合にはそれを使用) ストレージタイプが保持されるようになりました。(BZ#1496758)

- 以前のリリースでは、ノードの証明書が再デプロイされると、docker デーモンが誤って再起動されていました。そのため、**atomic-openshift-node** しか kubeconfig を読み込むコンポーネントがないので、ノードで不必要なダウンタイムが発生していました。今回のバグ修正で、新規の証明局 (CA) がデプロイされたかを確認するフラグが追加されたので、新しい証明局がデプロイされていない場合には、Docker の再起動が省略されます。(BZ#1537726)

- 以前のリリースでは、**docker_image_availability** のチェックでは、特定のコンテナーイメージを上書きしてコンテナー化されたコンポーネントに使用するための変数が考慮されていませんでした。これにより、上書きされたイメージが実際に利用できるにも拘わらず、デフォルトのイメージが検索されてしまい、可用性チェックで誤って問題が報告されていました。今回のバグ修正の結果、チェックでは、必要なイメージが利用可能かどうか正しく報告されるはずです。(BZ#1538806)

- 永続ボリューム要求 (PVC) が Elasticsearch 用に作成されたかどうか判断する場合に、従来の変数を使用していたため、ネットワークファイルシステム (NFS) がバックする永続ボリューム (PV) の作成時に、PVC が必要かどうかが正しく評価されませんでした。今回のバグ修正では、デプロイメント設定に PVC が必要かどうかが正しく評価されるようになりました。(BZ#1538995)

- 以前のリリースでは、Azure Blob ストレージのレジストリーを設定する場合に、デフォルトで **core.windows.net** のレルムが指定されていました。今回のバグ修正では、**openshift_hosted_registry_storage_azure_blob_realm** の値を、使用する値に変更できるようになりました。(BZ#1491100)

- 既存の GlusterFS デプロイメントをアンインストールする Playbook が新たに導入されました。この Playbook は、pod やサービスなど、既存のリソースすべてを削除します。また、この playbook ではオプションで、GlusterFS pod を実行していたホストからデータや設定もすべて削除されます。(BZ#1497038)

ロギング

- 以前のリリースでは、OpenShift Container Platform ロギングシステムは CRI-O をサポートしていませんでした。今回のバグ修正では、CRI-O 形式のログのパーサーが追加されたので、システムログとコンテナーログの両方を収集できるようになりました。(**BZ#1517605**)

- 以前のリリースでは、ロギングの再デプロイ時に、インストール後に ConfigMap ファイルに加えられた変更を維持しようとしていました。異なる Elasticsearch、Fluentd および Kibana (EFK) のスタックコンポーネントに必要な設定を提供できるようにする必要があるため、ユーザーに ConfigMap ファイルの内容を指定させることが困難でした。今回のバグ修正では、デプロイメント後に加えられた変更をもとにしたパッチが作成され、インストーラーが提供するファイルにそのパッチが適用されます。(**BZ#1519619**)

Web コンソール

- Kibana ページは、OpenShift Container Platform web コンソールの左上隅に **OPENSHIFT ORIGIN** を表示していました。今回のバグ修正で、Origin ヘッダーイメージが OpenShift Container Platform ヘッダーイメージに置き換えられました。そのため、Kibana ページには想定通りのヘッダーが表示されるようになっています。(**BZ#1546311**)

- Web コンソールの Overview ページで、OpenShift Container Platform **`DeploymentConfig`** および Kubernetes extensions/v1beta1 Deployment リソースはいずれも、デプロイメントというラベルが付けられており、これらのリソースを区別することができませんでした。**Overview** ページで **`DeploymentConfig`** のリソースに **`DeploymentConfig`** というラベルが付けられるようになりました。(**BZ#1488380**)

- Web コンソールの pod ステータスフィルターでは、init ステータスエラーなど、エラーにより pod を初期化できない場合に pod init のステータスが正しく表示されませんでした。pod のステータスが **`Init:Error`** の場合に、**Pod Initializing** ではなく、**Init Error** と正しく表示されるようになりました。(**BZ#1512473**)

- 以前のリリースでは、パイプラインのビルド設定の Web コンソールのページでタブを切り替えると、ページの再読み込み中に、そのページの内容の一部が表示されなくなってしまいました。タブを変更してもページ全体が再読み込みされず、コンテンツが正しく表示されるようになりました。(**BZ#1527346**)

- デフォルトでは、ビルダーをプロジェクトに追加すると、ビルダーイメージの以前のバージョンが表示され、デフォルトでビルダー設定中にそのイメージが選択されていました。これにより、以前のバージョンの言語またはフレームワークしか選択できないような、誤った印象を与えていました。ウィザードのタイトルに、バージョン番号が表示されなくなり、最新のバージョンがデフォルトで選択されるようになりました。(**BZ#1542669**)

- ブラウザーを使用する場合に、YAML、Jenkinsfile、Dockerfile エディターなど、ACE エディターライブラリーを使用する内部エディターから、右クリックを使用してテキストをコピーアンドペーストできませんでした。今回の更新では、ACE エディターライブラリーの新規バージョンが使用されるようになり、右クリックメニューオプションがコンソール全体で使用できるようになっています。(**BZ#1463617**)

- 以前のリリースでは、Referrer-Policy ヘッダーがコンソールにより送信されないため、ブラウザーは Referrer-Policy のデフォルトの動作を使用していました。今回のリリースでは、コンソールが正しく **`strict-origin-when-cross-origin`** に設定された Referrer-Policy ヘッダーを送信するようになり、Referrer-Policy ヘッダーをリッスンするブラウザーは Web コンソールの **`strict-origin-when-cross-origin policy`** に従うようになりました。(**BZ#1504571**)

- 以前のリリースでは、ビルドに文字列として Webhook シークレットが保存されていたため、プロジェクトへの読み取りアクセスのあるユーザーに、この Webhook シークレットの値が表示されていました。これらのユーザーは、対象のプロジェクトに対して読み取りアクセスしか

ないにも拘わらず、この値を使用してビルドをトリガーできました。プロジェクトへの読み取りアクセスしかないユーザーには、シークレットの値が表示されず、その値で webhook を使用してビルドをトリガーできないようになりました。(BZ#1504819)

- 以前のリリースでは、Web コンソールで、同じ永続ボリューム要求を複数回、デプロイメントに追加すると、そのデプロイメントの pod が機能しなくなりました。2 番目の PVC をデプロイメントに追加すると、pod テンプレート仕様から既存のボリュームを再利用するのではなく、Web コンソールが不正に新規ボリュームを作成していました。今回のリリースでは、Web コンソールは、同じ PVC が複数回表示された場合には、既存のボリュームを再利用するようになりました。この動作により、必要に応じて、別のマウントパスやサブパスを指定して、同じ PVC を追加できるようになります。(BZ#1527689)

- 以前のリリースでは、新規プロジェクトも作成する場合には、Deploy Image ウィンドウから **Image Name** を選択できないことが明確ではありませんでした。今回のリリースでは、既存のプロジェクトにしか **Image Name** を設定できない旨を説明するヘルプテキストが簡単に見つけられるようになりました。(BZ#1535917)

- 以前のリリースでは、Web コンソールのシークレットページにラベルが表示されませんでした。他のリソースと同様に、シークレットのラベルも表示できるようになりました。(BZ#1545828)

- Web コンソールには、テンプレートを処理するパーミッションがなくても、テンプレートの処理ページが表示される場合がありました。テンプレートを処理しようとすると、エラーが表示されていました。今回のリリースでは、処理できない場合には、テンプレートの処理が表示されなくなりました。(BZ#1510786)

- 以前のリリースでは、**Clear Changes** ボタンを使用しても、Web コンソール環境変数エディターの **Environment From** 変数に加えた編集が正しく消去されませんでした。このボタンで、**Environment From** 変数に対する編集が正しくリセットされるようになりました。(BZ#1515527)

- デフォルトでは、ダイアログの周りのネガティブスペースをクリックすることで、Web コンソールのダイアログを非表示にすることができるため、警告ダイアログが気づかずに終了してしまう可能性がありました。今回のバグ修正では、ダイアログのボタンの 1つをクリックしなければ、ダイアログが閉じないように、警告ダイアログの設定が変更されました。ダイアログを終了するにはダイアログボタンの 1 つをクリックする必要があるため、警告ダイアログはユーザーの不注意で終了されることがなくなりました。(BZ#1525819)

マスター

- スケジューラー実装に問題があるため、**`ScheduledImageImportMinimumIntervalSeconds`** 設定が正しく準拠されず、OpenShift Container Platform は、スケジュールされたイメージを不正な間隔でインポートしようとしていました。今回のバグ修正で、この問題は解決されました。(BZ#1515058)

ネットワーク

- OpenShift Container Platform ノードは、マスターが VNID を割り当てるまでの待機時間が十分になく、ノードの伝搬にしばらく時間がかかることがあり、pod の作成に失敗していました。ノードで VNID をフェッチするまでのタイムアウト期間が 1 から 5 秒に増やしてください。今回のバグ修正により、pod の作成を成功させることができます。(BZ#1509799)

- 今回のリリースでは、Egress ルーターに渡す **`EGRESS_SOURCE`** 変数の一部として、サブネット長を指定できるようになりました (例: **`192.168.1.100`** ではなく **`192.168.1.100/24`**)。ネットワーク設定によっては (ゲートウェイアドレスが、その時々で、複数ある物理 IP の 1 つによりバックされる可能性のある仮想 IP の場合など)、egress ルーターがローカルサブネット上の他のホストにトラフィックを送信できない場合には、egress ルーターとそのゲートウェイ

の間の Egress ARP トラフィックが正しく機能しない場合があります。サブネット長を設定した **EGRESS_SOURCE** を指定する場合には、Egress ルーター設定スクリプトにより、このようなネットワーク設定で機能するように、Egress pod が設定されます。(BZ#1527602)

- 場合によっては、iptables ルールの順番が変更されて、**プロジェクト毎の静的 IP アドレス** 機能が、一部の IP アドレスで機能しなくなる場合がありました (多くの場合、末尾が偶数の egress IP アドレスはそのまま機能しますが、末尾が奇数の egress IP アドレスは失敗します)。そのため、プロジェクト毎の静的 IP アドレス機能を使用する予定のプロジェクトにある pod からの外部トラフィックは、通常のノード IP アドレスを使用することになりました。iptables ルールの順番が変更された場合でも、予定通りの効果があるように iptables ルールが変更されました。今回のバグ修正で、プロジェクト毎の静的 egress IP 機能が確実に機能するようになりました。(BZ#1527642)

- 以前のリリースでは、egress IP の初期化コードは、OpenShift サービスを再起動して既存の実行中の SDN が見つかった場合には実行されずに、完全な SDN 設定を行う場合にのみ実行されていたので、新しいプロジェクト毎の静的 egress IP の作成に失敗していました (**HostSubnet.EgressIPs**)。この問題は修正され、プロジェクト毎の静的 egress IP は、ノードの再起動後に正しく機能するようになりました。(BZ#1533153)

- 以前のリリースでは、OpenShift はホストとサブネットの値が競合し、pod IP ネットワークがノード全体で使用できなくなっていました。これは、ノードの起動時に古い OVS ルールが消去されていなかったことが原因でした。これは修正され、古い OVS ルールがノードの起動時に消去されるようになりました。(BZ#1539187)

- 以前のバージョンでは、静的 IP アドレスがプロジェクトから削除され、同じプロジェクトに追加しなおされた場合に、正しく機能しませんでした。これは修正され、静的な egress IP を削除して追加しなおしても機能するようになりました。(BZ#1547899)

- 以前のリリースでは、OpenShift が OpenStack にデプロイされている場合には、必要な **iptables** ルールが自動的に作成されず、別のノードにある pod 間の通信中にエラーが発生していました。Ansible OpenShift インストーラーは、必要な **iptables** ルールを自動的に設定するようになりました。(BZ#1493955)

- ノードの起動に依存する起動コードで競合が発生し、ユーザー空間のプロキシーが必要とするフィールドを設定していました。ネットワークプラグインが使用されない場合 (または、高速な場合) に、ユーザー空間プロキシーの設定が通常よりも早く実行され、ノードの IP アドレスの値を nil として読み取っていました。後ほどプロキシー (またはプロキシーを使用する **unidler**) が有効化された場合に、IP アドレスの値が nil であるために、競合が発生していました。この問題は修正され、再試行ループが追加され、IP アドレスが設定されるまで待機し、ユーザー空間プロキシーと **unidler** が予想どおりに機能するようになりました。BZ#1519991)

- 場合によって、ノードは、不正な順番の HostSubnet **deleted** イベントをマスターから重複して受信することがありました。重複イベントの処理中に、ノードは、アクティブなノードに対応する OVS フローを削除してしまい、対象の 2 つのノード間の通信を妨害していました。最新のバージョンでは、HostSubnet イベント処理で、重複イベントがないかチェックして、重複イベントを無視するようになったので、OVS フローは削除されず、pod が通常どおりに通信されるようになりました。(BZ#1544903)

- 以前のリリースでは、docker イメージを消去する **openshift ex dockergc** コマンドが失敗することがありました。この問題は修正されました。(BZ#1511852)

- 以前のリリースでは、ネストされたシークレットが pod にマウントされませんでしたが、この問題は修正されました。(BZ#1516569)

- バージョン 1.9 以前の HAproxy では、再読み込み中に接続が切断されることがありましたが、

この問題は修正されました。HAproxy のシームレスな再読み込み機能を使用することで、HAproxy により、再読み込み中に開放されているソケットが渡されるようになり、再読み込みの問題が修正されました。(BZ#1464657)

- システムログに誤ったエラーが表示されていました。**`Stat fs failed. Error: no such file or directory`** のエラーが頻繁にログに表示されていました。これは、パスが存在しない場合に、コードで **`syscall.Statfs`** 関数が呼び出されることが原因です。この問題は修正されました。(BZ#1511576)

- 以前のリリースでは、ルーターのシャードを使用する場合に、拒否ルートのエラーメッセージが表示されていました。この問題は修正され、ルーターシャードを使用する場合に、却下されたルートエラーメッセージは HAproxy では表示されなくなりました。(BZ#1491717)

- 以前のリリースでは、ホストを **`localhost`** に設定したルートを作成する場合や、**`ROUTER_USE_PROXY_PROTOCOL`** 環境変数が **`true`** に設定されていない場合には、ルートの再読み込みに失敗していました。これは、ホスト名がデフォルトに設定されているため、ルート設定が一致しなくなることが原因です。**`curl`** を使用する場合には、**`-H`** オプションが使用されるようになり、ホスト名に「localhost」が設定されている場合にヘルスチェックに合格せず、ルートの再読み込みが成功します。(BZ#1542612)

- 以前のリリースでは、クラスター管理者は、TLS 証明書を更新できませんでした。クラスター管理者が行う必要のあるタスクであるため、TLS 証明書を更新できるように、ロールが変更されました。(BZ#1524707)

サービスブローカー

- 以前のリリースでは、MariaDB、PostgreSQL および MySQL の APB は、「database」ではなく、「databases」のタグ付けされていました。これは、他のサービスと同じ「database」というタグに変更され、検索結果に正しく表示されるようになりました。(BZ#1510804)

- 非同期バインドおよびバインド解除は OpenShift Ansible broker (OAB) の実験的機能で、サポートされておらず、デフォルトで有効になっていません。Red Hat が公式のリリースしている APB (PostgreSQL、MariaDB、MySQL および Mediawiki) は非同期バインドおよびバインド解除をサポートしません。(BZ#1548997)

- 以前のリリースでは、**`etcdctl`** コマンドを使用する場合には、etcd サーバーにはアクセスできませんでした。これは、tcp が **`asb-etcd`** デプロイメント設定で、必要とされる **`--advertise-client-urls`** の値ではなく、「0.0.0.0」に設定されていることが原因です。コマンドが更新され、etcd サーバーがアクセス可能になりました。(BZ#1514417)

- 以前のリリースでは、クラスターの外部で **`apb push -o`** コマンドを使用すると失敗していました。これは、任意のサービスの Docker レジストリーサービスが内部の操作が使用するルートのみに到達するように設定されていたためです。適切なルートを参照するように、該当の Ansible playbook が更新されました。(BZ#1519193)

- 以前のリリースでは、**`asbd --help`** または **`asbd -h`** を入力すると、**`--help`** 引数が返すコードがエラーとして解釈され、エラーが 2 回出力されていました。修正が加えられ、出力が 1 度だけになり、help コマンドのリターンコードが有効と解釈されるようになりました。結果、help コマンドの出力は 1 度のみになりました。(BZ#1525817)

- 以前のリリースでは、RHCC レジストリーに **`white-list`** 変数を設定すると、オプションが設定から削除された後にも、オプションがないかを検索し続けていました。これは、**`white-list`** コードのエラーにより発生していました。今回のバグ修正で、このエラーが修正されました。(BZ#1526887)

- 以前のリリースでは、レジストリーの設定で **config** が **auth_type** に設定されていない場合には、エラーメッセージが表示されていました。今回のバグ修正では、**auth_type** の設定がなくても、レジストリーの設定が正しく機能するようになりました。(**BZ#1526949**)

- 以前のリリースでは、ユーザーにタスクの実行権限がない場合に、ブローカーがステータスコード 403 ではなく、ステータスコード 400 を返していました。今回のバグ修正で、エラーが修正されて、正しいステータスコードが返されるようになりました。(**BZ#1510486**)

- 以前のリリースでは、MariaDB 設定オプションは MySQL オプションで表示されていました。これは、MariaDB がアップストリームでは MySQL 変数を使用するために起こっていました。今回のバグ修正により、OpenShift では変数が MariaDB として呼び出されるようになりました。(**BZ#1510294**)

ストレージ

- 以前のリリースでは、OpenShift は、マウントされた NFS ボリュームのチェックに root squash を使用していました。root で実行している場合には、OpenShift のパーミッションは、マウントされた NFS ボリュームへのアクセス権を持たない「nobody」ユーザー権限に下げられていました。これが原因で、OpenShift のチェックに失敗し、NFS ボリュームがアンマウントされませんでした。今回のリリースでは、OpenShift は、マウントされた NFS ボリュームにアクセスせず、ファイルシステムを解析および処理してマウントがあるか確認します。また、root squash オプションが指定された NFS ボリュームはアンマウントされます。(**BZ#1518237**)

- 以前のリリースでは、OpenStack Cinder タイプの永続ボリュームがアタッチされているノードがシャットダウンまたはクラッシュした場合に、アタッチされていたボリュームのアタッチが解除されませんでした。その結果、永続ボリュームが利用できないので、pod が障害のあるノードから以降されず、他のノードや pod からこれらのボリュームにアクセスできませんでした。ノードに障害が発生すると、アタッチされたボリュームはすべて、タイムアウト期間の経過後に解除されるようになりました。(**BZ#1523142**)

- 以前のリリースでは、Downward API、シークレット、ConfigMap および Projected ボリュームがこれらのコンテンツをすべて管理するため、他のボリュームをその上にマウントできないようになっていました。つまり、ユーザーは、前述のボリューム上に他のボリュームをマウントできませんでした。今回のバグ修正では、作成するファイルのみを操作するようになったので、前述のボリューム上にどのボリュームでもマウントできるようになりました。(**BZ#1430322**)

アップグレード

- 以前のリリースでは、3.6 からアップグレードする場合に、3.6 には 'docker-registry.default.svc' という名前がなく、アップグレードの playbook でレジストリーの証明書が再生成されませんでした。そのため、設定変数は更新されず、DNS 経由でレジストリーにプッシュされませんでした。3.9 のアップグレード playbook では、必要に応じて証明書が再生成され、3.9 にアップグレードされた全環境が DNS 経由でレジストリーにプッシュされるようになりました。(**BZ#1519060**)

- etcd ホストの検証では、1 つまたは複数の etcd ホストに対応するようになり、これまで以上に柔軟に etcd のホスト数を設定できるようになりました。etcd ホストの推奨数は 3 のままです。(**BZ#1506177**)

2.6. テクノロジープレビュー機能

現在、今回のリリースに含まれる機能にはテクノロジープレビューのものがあります。これらの実験的機能は、実稼働環境での使用を目的としていません。これらの機能に関しては、Red Hat カスタマーポータルの以下のサポート範囲を参照してください。

テクノロジープレビュー機能のサポート範囲

以下の表では、**TP** とマークが付いた機能は **テクノロジープレビュー**、**GA** とマークが付いた機能は **一般公開** 機能です。

表2.1 テクノロジープレビューのトラッカー

機能	OCP 3.6	OCP 3.7	OCP 3.9
Prometheus クラスターモニタリング	-	TP	TP
ローカルストレージの永続ボリューム	-	TP	TP
ランタイム pod の CRI-O	-	TP	GA* [a]
テナント駆動形のスナップショット作成	-	TP	TP
oc CLI プラグイン	-	TP	TP
サービスカタログ	TP	GA	-
テンプレートサービスブローカー	TP	GA	-
OpenShift Ansible ブローカー	TP	GA	-
ネットワークポリシー	TP	GA	-
サービスカタログの最初の体験	TP	GA	-
プロジェクト追加に関する新しいフロー	TP	GA	-
検索カタログ	TP	GA	-
CFME インストーラー	TP	GA	-
Cron ジョブ	TP	TP	GA
Kubernetes デプロイメント	TP	TP	GA
StatefulSets	TP	TP	GA

機能	OCP 3.6	OCP 3.7	OCP 3.9
明示的なクォータ	TP	TP	GA
マウントオプション	TP	TP	GA
docker 向けのシステムコンテナー、CRI-O	TP	TP	廃止
インストーラーおよび Kubelet 向けのシステムコントローラー	TP	TP	GA
Hawkular エージェント	TP	廃止	
Pod の PreSet	TP	廃止	
experimental-qos-reserved	-	TP	TP
Pod の sysctls	TP	TP	TP
中央監査	-	TP	GA
外部プロジェクトトラフィックの静的 IP	-	TP	GA
テンプレート完了の検出	-	TP	GA
`replicaSet`	TP	TP	GA
Mux	-	TP	TP
クラスター化された MongoDB のテンプレート	TP	コミュニティー	-
クラスター化された MySQL のテンプレート	TP	コミュニティー	-
Kubernetes リソースを使用したイメージストリーム	TP	TP	GA
デバイスマネージャー	-	-	TP
永続ボリュームのサイズ調整	-	-	TP

機能	OCP 3.6	OCP 3.7	OCP 3.9
ヒュージページ	-	-	TP
CPU マネージャー	-	-	TP
デバイスプラグイン	-	-	TP
fluentd 向けの syslog 出力プラグイン	-	-	TP

[a] * のマークが付いている機能は、z ストリームパッチで提供されることを指します。

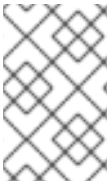
2.7. 既知の問題

- OpenShift Container Platform 3.9 の初期 GA リリースで、インストールおよびアップグレードの playbook が以前のリリースよりもメモリーを多く消費する既知の問題があります。ノードのスケールアップおよびインストール Ansible playbook では、**`include_tasks`** が複数箇所で使用されるので、コントロールホスト (playbook を実行するシステム) で消費するメモリー量が想定される量よりも多い場合があります。この問題は、RHBA-2018:0600 リリースで対処されています。このようなインスタンスの大半は、メモリーがあまり消費されない **`import_tasks`** 呼び出しに変換されました。この変更後には、コントロールホストでのメモリー消費量は、ホストごとに 100MiB 以下に収まるはずです。大規模な環境 (ホストが 100 以上) では、最低でもコントロールホストに 16GiB のメモリーを割り当てることを推奨します。(**BZ#1558672**)

2.8. エラータの非同期更新

OpenShift Container Platform 3.9 のセキュリティー、バグ修正、拡張機能の更新は、Red Hat Network 経由で非同期エラータとして発表されます。OpenShift Container Platform 3.9 の全エラータは Red Hat カスタマーポータル から入手できます。非同期エラータについては OpenShift Container Platform ライフサイクル を参照してください。

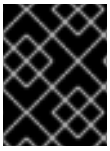
Red Hat カスタマーポータルのユーザーは、Red Hat サブスクリプション管理 (RHSM) のアカウント設定でエラータの通知を有効にすることができます。エラータの通知を有効にすると、登録しているシステムに関連するエラータが新たに発表されるたびに、メールで通知が送信されます。



注記

OpenShift Container Platform のエラータ通知メールを生成させるには、Red Hat カスタマーポータルのユーザーアカウントに登録済みのシステムが含まれており、OpenShift Container Platform エンタイトルメントを消費している必要があります。

以下のセクションは、これからも継続して更新され、今後 OpenShift Container Platform 3.9 バージョンの非同期リリースで発表されたエラータの機能拡張およびバグ修正に関する説明を提供していきます。たとえば、OpenShift Container Platform 3.9.z は、サブセクションで説明します。さらに、エラータの文章がアドバイザリーで提供されたスペースに収まらないリリースについては、その後のサブセクションで説明します。



重要

OpenShift Container Platform のどのバージョンでも、適切な「クラスターのアップグレード」の方法を必ず確認してください。

2.8.1. RHBA-2018:1566: OpenShift Container Platform 3.9.27 バグ修正および機能拡張の更新

発行日: 2018-05-16

OpenShift Container Platform release 3.9.27 が公開されました。この更新に含まれるパッケージおよびバグ修正は、RHBA-2018:1566 アドバイザリーにまとめられています。この更新に含まれるコンテナーイメージは、RHBA-2018:1567 アドバイザリーで提供されます。

アドバイザリーでは、このリリースのバグ修正およびイメージに関する全説明は除外されます。このリリースに含まれるバグ修正およびイメージに関するアップグレードなどの情報については、以下を参照してください。

2.8.1.1. アップグレード

既存の OpenShift Container Platform 3.7 または 3.9 クラスターを最新のこのリリースにアップグレードするには、自動アップグレード playbook を使用します。説明は、「クラスターの自動インプレースアップグレードの実行」を参照してください。

2.8.1.2. バグ修正

- ビルド pod は複数のコンテナーを使用します。バイナリービルドは、コンテンツをストリーミングする先のコンテナーを指定する必要があります。カスタムビルドの場合は、コンテナーの名前はカスタム以外のビルドと異なります。バイナリーコンテンツをカスタムビルドにストリーミングする場合には、必要なコンテナーの git-clone が存在せず、ビルドに失敗します。バイナリーコンテンツをカスタムビルド pod にストリーミングするためのロジックが変更され、正しいコンテナー名のカスタムビルドを参照するようになりました。今回のバグ修正で、バイナリーコンテンツは正常にカスタムビルドのコンテナーにストリーミングされるようになります。(**BZ#1560659**)

- リソースの制約が原因で、Jenkins テンプレート例の Readiness プローブの引用が正しく終了しない可能性があり、Jenkins デプロイメントが不必要に失敗していました。今回のバグ修正では、テンプレートの readiness プローブの制限が緩和されたので、readiness プローブの制約が多いために Jenkins のデプロイメントが不必要に失敗する数が減少しています。(**BZ#1559675**)

- マスター **admin.kubeconfig** ファイルが **oc command** に追加され、操作に必要なリソースに適切な認証およびアクセスを割り当てることができるようになりました。(**BZ#1561247**)

- インストーラーは、存在しない可能性のあるパスに不正に SELinux コンテキストを設定しようとしていました。このタスクは、CRI-O の問題を回避するためのものでしたが、この問題はすでに存在しなくなったのでこのタスクが削除されました。(**BZ#1564949**)

- デフォルトで、サービスカタログ pod にログの詳細レベルが高く設定されていたので、マスターノードのサービスカタログ pod が大量のログデータを作成していました。デフォルトのログ詳細レベルが最小レベルに設定しなおされました。(**BZ#1564179**)

- Elasticsearch サーバーの TLS 証明書には、件名の alt に外部のホスト名が含まれておらず、外部から Elasticsearch にアクセスするクライアントは、MITM サーバー証明書の検証をオンにすることができません。Elasticsearch が外部アクセスを許可するように設定するときに。件名の alt の名前一覧に外部ホスト名を追加してください。TLS クライアントでサーバー証明書の検証をオンできるようになります。(**BZ#1554878**)

- Fluentd プラグインは、障害時にエラー対応すべてをログに記録し、オンディスクのログがいっぱいになってしまいます。デバッグモードの時だけ対応がすべてログ記録されるようになり、オンディスクのログでディスクが消費されないようになりました。(**BZ#1554885**)

- Fluentd の Elasticsearch への書き込み操作は、デフォルトで **`index`** になっています。書き込みにより、不必要な Elasticsearch の **`delete`** 操作がトリガーされて、パフォーマンスに影響を与えるような負荷が余分に発生していました。**`create`** 操作を使用してください。elasticsearch への書き込みが発生すると、記録を作成するだけか、記録が重複する場合に更新を省略して、サーバーの負荷を軽減します。(**BZ#1565909**)

- curator pod は、origin からダウンストリームの dist-git へのマージが不正であったため、エントリーポイントのスクリプトを見つけることができず、クラッシュがループしていました。pod は機能せず、クラッシュループを繰り返していました。今回のバグ修正では、コードがアップストリームで同期されるようになりました。(**BZ#1572419**)

- Fluentd secure-forward プラグインは、設定ファイル内のホスト名のプレースホルダー **`${hostname}`** をサポートします。この値は大文字と小文字を区別するにも拘わらず、大文字の **`${HOSTNAME}`** が設定されており、Fluentd コンテナーの正しいホスト名の選択に失敗していましたが、このバグは修正されました。(**BZ#1553576**)

- 存在しないイメージの URL を手動で入力すると、ページの読み込みが終了しているにも拘わらず、ページの読み込みが進行中であることを示すメッセージがそのページに残り、**The image stream details could not be loaded** の警告が表示されます。イメージが読み込まれた場合もそうでない場合も、**`loaded`** 範囲の変数を設定し、ビューで使用して **loading** メッセージを非表示にしてください。イメージデータの読み込みを試行後には、イメージを読み込むことができなくても、**loading** メッセージが非表示になりました。(**BZ#1550797**)

- 以前のリリースでは、空の ConfigMap を編集する場合に、web コンソールで新しい鍵を追加できませんでした。エディターで **Add Item** をクリックしても何も効果がありませんでした。今回のバグ修正では、何も含まれない ConfigMap を編集する場合に、アイテムを正しく追加できるようになりました。(**BZ#1558863**)

- プロジェクトのデフォルトノードセレクターで、DaemonSet ノードを制限すると、プロジェクトのデフォルトノードセレクターを追加することで制限されたノード上で、DaemonSet pod の削除/作成がループで繰り返されてしまいます。今回のバグ修正では、アップストリームの DaemonSet ロジックが、プロジェクトのデフォルトのノードセレクターを認識するように更新されました。(**BZ#1571093**)

- Hawkular Alerts コンポーネントが Hawkular Metrics から削除されました。この変更は、Hawkular Metrics への機能的な影響はありません。(**BZ#1543647**)

- 以前のリリースでは、OVS フローが正しく管理されていませんでした。2 つのノードが再起動して、再起動後に IP アドレスが交換された場合に、他のノードが、これらの 2 つまたはいずれかのノード上の pod にトラフィックを送信できない可能性があります。OVS フローを管理するコードは、IP の再割当てが行われることを想定し、より慎重に正しい変更が加えられました。Pod-to-pod トラフィックは、ノードで IP アドレスが交換された後でも、そのまま正しく機能するようになりました。(**BZ#1570394**)

- 更新 Egress ポリシーでは、送信トラフィックをブロックして、OVS フローを修正し、トラフィックを再有効化する必要がありましたが、DNS 名の OVS フロー生成に時間がかかっていました。そのため、Egress トラフィックが数秒間ダウンし、許容範囲ではなくなる可能性がありました。今回のバグ修正では、更新 Egress ポリシー処理が更新され、送信トラフィックをブロックする前に、新規の OVS フローが事前に生成されるように更新されており、Egress ポリシーの更新時のダウンタイムが減少しています。(**BZ#1571430**)

- namespace 毎の静的 egress IP を使用する場合に、外部トラフィックすべてが egress IP 経由

でルーティングされます。**External** は、別の pod にダイレクトされない全トラフィックを指すので、これには pod から pod のノードへのトラフィックも含まれることになります。DNS にノードの IP アドレスを使用するように pod が指示された場合には、pod は静的な egress IP を使用して、DNS トラフィックが先に egress ノードにルーティングされてから、元のノードに戻ってきます。これらのノードは、他のホストからの DNS 要求を許可しないように設定されている場合があり、pod が DNS を解決できない可能性があります。Pod からノードへの DNS 要求は、egress IP を回避して、直接ノードに移動することで DNS が機能するようになります。(**BZ#1570398**)

- 今回のバグ修正では、**`cgroups-per-qos`** が有効な場合に **`cpu-cfs-quota`** を無効化しても CPU CFS の制限が pod に設定されていました。(**BZ#1558155**)

- 今回のバグ修正では、該当するインスタンスが停止している場合に、OpenStack クラウド統合を使用して実行しているクラスターからノードが削除される問題に対応しました。インスタンスが停止しているノードリソースがクラスターから削除されなくなりました。(**BZ#1558422**)

- ボリュームが強制的にデタッチされている場合には、ノードが障害状態に入り、再起動しませんでした。新しいボリュームがノードにアタッチされるとアタッチ状態で止まっていました。ノードにアタッチされたボリュームが 21 分以上アタッチ状態になっている場合には、ノードをテイントして、クラスターから削除し、テイントを削除するために追加しなおして、対象ノードの障害状態を修正する必要があります。今回のバグ修正では、スケジュールから障害状態が削除され、OpenShift Container Platform 管理者は、ノードを修正して起動できるようになります。(**BZ#1455680**)

- 以前の OpenShift Container Platform のリリースでは、**`docker`** がセキュアでないと内部のレジストリーにマークを付ける必要がないにも拘わらず、誤ってそのように再設定されていました。OpenShift Container Platform 3.9 でこれは修正され、この現象は発生しなくなっています。(**BZ#1502028**)

2.8.1.3. 機能拡張

- コンテナーのランタイムとして CRI-O を使用できるように、RPM で CRI-O を使用してください。RPM として CRI-O をインストールするには、以下の 2 つのオプションを設定します。

```
openshift_use_crio=True
openshift_crio_use_rpm=True
```

(**BZ#1553186**)

- yedit モジュールは、一意のバックアップファイルを生成します。以前のリリースでは、同じリソースに変更が複数回繰り返されると、最新の相違点のみが保存されていました。(**BZ#1555426**)

- 今回のリリースでは、関連付ける適切な namespace が判断できない場合に管理者にメッセージが表示できるようになりました。表示できないとメッセージがなくなり、確認してレビューできません。存在しない場合には、管理者用に Kibana Index パターンが作成されます。(**BZ#1519522**)

- インベントリーの値がない場合には、現在のデプロイメントに使用する値を再利用して、チューニングした値が保存されます。Elasticsearch の場合には、クラスターのチューニングを行ったにも拘わらず、これらの値を変数に伝搬しな買った場合に、ロギングのアップグレードでロールのデフォルト値を使用し、クラスターが不正な状態に入り、ログデータが失われる可能性がありました。値が EFK の正しい順番を守るようになりました: inventory → existing environment → role defaults。(**BZ#1561196**)

- クラスターの管理者向けの Kibana index パターン数が制限されるようになりました。以前のリ

リースでは、一覧を管理できず、多数の namespace がある大規模なクラスターでは必要ありませんでした。クラスター管理者は、制限を加えた index パターンのサブセットのみが表示されるようになりました。(**BZ#1563230**)

2.8.2. RHBA-2018:1796: OpenShift Container Platform 3.9.30 バグ修正および機能拡張の更新

発行日: 2018-06-06

OpenShift Container Platform release 3.9.30 が公開されました。この更新に含まれるパッケージおよびバグ修正は、RHBA-2018:1796 アドバイザリーにまとめられています。この更新に含まれるコンテナーイメージは、RHBA-2018:1797 アドバイザリーで提供されます。

アドバイザリーでは、このリリースのバグ修正およびイメージに関する全説明は除外されます。このリリースに含まれるバグ修正およびイメージに関するアップグレードなどの情報については、以下を参照してください。

2.8.2.1. バグ修正

- Jenkins no_proxy の処理は、**`".svc"`** などのサフィックスを処理できませんでした。そのため、Jenkins Kubernetes エージェント pod と Jenkins マスターの間の通信が設定済みの **`http_proxy`** を経由しようとして、失敗していました。今回のバグ修正で、OpenShift Container Platform jenkins エージェントイメージは、jenkins マスターと jnlp ホストが自動的に **`no_proxy`** リストに追加されるように更新されました。**`no_proxy`** 処理の Jenkins の制限を回避できるようになりました。(**BZ#1578989**)

- Elasticsearch サーバーの証明書を作成する場合には、外部の Elasticsearch ホスト名が無条件に **`subjectAltName`** に追加されていました。ホスト名のコンポーネントの最初が文字でないと **`subjectAltName`** に追加できないのでインストールに失敗していました。たとえば、**`es.0xdeadbeef.com`** のようなホスト名は追加できず、エラーが発生していました。Elasticsearch のホスト名のコンポーネントが文字以外で始まる場合には、警告が表示されるようになり、**`subjectAltName`** には追加されません。ロギングのインストールが正常に完了するようになりました。(**BZ#1567767**)

- プラグインは、より一般的な例外ではなく、**`KubeException`** のみを検出していました。そのため、API サーバーに問い合わせがされるまで、コンシューマーはそのまま繰り返し処理を行っていました。メタデータの取得が緩和され、例外が正常に検出されるようになり、メタデータが返されないので記録が孤立されるようになりました。(**BZ#1560170**)

- **`logging-elasticsearch-ops`** は、openshift-ansible **`delete_logging`** の delete` configmaps` 一覧から抜けていました。**`logging-elasticsearch-ops`** configmap は、ロギング用の uninstall ansible playbook を実行した後にも存在し続けていました。**`logging-elasticsearch-ops`** は、delete configmaps 一覧に追加され、**`logging-elasticsearch-ops`** などのロギング configmaps すべてが、ロギング用の uninstall ansible playbook を実行することでアンインストールされるようになりました。(**BZ#1549220**)

- Web コンソールのプロジェクト一覧ページにプロジェクトがなく、セルフプロビジョニングが無効になっている場合に、**Create Project** ボタンが誤って表示されていました。このアクションは常に失敗するため、ボタンは非表示にする必要がありました。このバグは修正され、セルフプロビジョニングが無効な場合には **Create Project** は、コンソールで正常に表示されなくなりました。(**BZ#1577359**)

- 今回のバグ修正では、プライベートの docker ハブレジストリーからイメージをプルする問題に対応しています。(**BZ#1578088**)

- 今回のバグ修正では、ノードで **cpu-cfs-quota** が **false** に設定されている場合に、**cfs_quota** が pod に設定された状態になる問題に対応しています。(**BZ#1581860**)

2.8.2.2. 機能拡張

- JSON ペイロード解析を無効にできるようになりました。各ログメッセージを JSON に解析し、最終的なペイロードにアタッチする操作はコストがかかります。Fluentd は、メッセージペイロードを無効にするように設定できるようになりました。これは、fluent-plugin-kubernetes_metadata_filter の非推奨の機能に加えた初回の設定変更です。(**BZ#1569825**)

2.8.2.3. イメージ

今回のリリースでは、以下のイメージで Red Hat コンテナーレジストリー (**registry.access.redhat.com**) を更新しました。

```
openshift3/apb-base:v3.9.30-2
openshift3/container-engine:v3.9.30-2
openshift3/cri-o:v3.9.30-2
openshift3/image-inspector:v3.9.30-2
openshift3/jenkins-2-rhel7:v3.9.30-2
openshift3/jenkins-slave-base-rhel7:v3.9.30-2
openshift3/jenkins-slave-maven-rhel7:v3.9.30-2
openshift3/jenkins-slave-nodejs-rhel7:v3.9.30-2
openshift3/local-storage-provisioner:v3.9.30-2
openshift3/logging-auth-proxy:v3.9.30-2
openshift3/logging-curator:v3.9.30-2
openshift3/logging-elasticsearch:v3.9.30-2
openshift3/logging-eventrouter:v3.9.30-2
openshift3/logging-fluentd:v3.9.30-2
openshift3/logging-kibana:v3.9.30-3
openshift3/mariadb-apb:v3.9.30-2
openshift3/mediawiki-123:v3.9.30-2
openshift3/mediawiki-apb:v3.9.30-2
openshift3/metrics-cassandra:v3.9.30-2
openshift3/metrics-hawkular-metrics:v3.9.30-2
openshift3/metrics-hawkular-openshift-agent:v3.9.30-2
openshift3/metrics-heapster:v3.9.30-2
openshift3/mysql-apb:v3.9.30-2
openshift3/node:v3.9.30-2
openshift3/oauth-proxy:v3.9.30-2
openshift3/openvswitch:v3.9.30-2
openshift3/ose-ansible-service-broker:v3.9.30-2
openshift3/ose-ansible:v3.9.30-3
openshift3/ose-cluster-capacity:v3.9.30-2
openshift3/ose-deployer:v3.9.30-2
openshift3/ose-docker-builder:v3.9.30-2
openshift3/ose-docker-registry:v3.9.30-2
openshift3/ose-egress-http-proxy:v3.9.30-2
openshift3/ose-egress-router:v3.9.30-2
openshift3/ose-f5-router:v3.9.30-2
openshift3/ose-haproxy-router:v3.9.30-2
openshift3/ose-keepalived-ipfailover:v3.9.30-2
openshift3/ose-pod:v3.9.30-2
openshift3/ose-recycler:v3.9.30-2
openshift3/ose-service-catalog:v3.9.30-2
```

```
openshift3/ose-sti-builder:v3.9.30-2
openshift3/ose-template-service-broker:v3.9.30-2
openshift3/ose-web-console:v3.9.30-2
openshift3/ose:v3.9.30-2
openshift3/postgresql-apb:v3.9.30-2
openshift3/prometheus-alert-buffer:v3.9.30-2
openshift3/prometheus-alertmanager:v3.9.30-2
openshift3/prometheus-node-exporter:v3.9.30-2
openshift3/prometheus:v3.9.30-2
openshift3/registry-console:v3.9.30-2
openshift3/snapshot-controller:v3.9.30-2
openshift3/snapshot-provisioner:v3.9.30-2
```

2.8.2.4. アップグレード

既存の OpenShift Container Platform 3.7 または 3.9 クラスターを最新のこのリリースにアップグレードするには、自動アップグレード playbook を使用します。説明は、「クラスターの自動インプレースアップグレードの実行」を参照してください。

2.8.3. RHSA-2018:2013: OpenShift Container Platform 3.9.31 セキュリティー、バグ修正および機能拡張の更新

発行日: 2018-06-27

OpenShift Container Platform release 3.9.31 が公開されました。この更新に含まれるパッケージおよびバグ修正は、RHSA-2018:2013 アドバイザリーにまとめられています。この更新に含まれるコンテナーイメージは、RHBA-2018:2014 アドバイザリーで提供されます。

アドバイザリーでは、このリリースのバグ修正および機能拡張に関する全説明は除外されます。このリリースに含まれるバグ修正および機能拡張に関するアップグレードなどの情報については、以下を参照してください。

2.8.3.1. バグ修正

- webhook ペイロードには、空のコミットアレイが含まれる可能性があり、API サーバーで処理する場合にアレイのインデックス化エラーが発生し、API サーバーがクラッシュしていました。アレイにインデックス化する前に、空のアレイがないか確認する必要がありました。今回のバグ修正では、API サーバーがクラッシュすることなしにコミットペイロードが処理されるようになりました。(**BZ#1586076**)

- 不正なパスワードがシークレットで使用されると、全イメージのプルが失敗してしまい、同じレジストリーからの公開イメージのプルは失敗します。今回のバグ修正では、パスワードが間違っている場合に **401 error** の再試行ロジックが追加され、イメージがパブリックの場合に、イメージがプルされ、不正なシークレットは無視されるようになりました。(**BZ#1506175**)

- **`openshift-jenkins-sync`** プラグインは、OpenShift Container Platform web コンソールのビルド URL を構築する場合に、Jenkins サービスとパイプラインストラテジーのビルドが同じプロジェクトに含まれることを前提としていました。Jenkins とパイプラインストラテジーが別のプロジェクトに含まれている場合に、Jenkins サービス/ルートが見つからないので OpenShift Container Platform web コンソールの表示ログリンクは、不正な URL を参照していました。**`openshift-jenkins-sync`** プラグインは、実行中の namespace で Jenkins サービス/ルートを検索するようになりました。また、Jenkins に root URL を明示的に設定した場合には、優先度が上がるようになりました。OpenShift Container Platform の web コンソールで特定のパイプラインストラテジービルドの URL が正しくレンダリングされるようになりました。(**BZ#1542460**)

- イメージ検証では、古いイメージオブジェクトの検証に使用し、イメージ署名のインポートコントローラーはそのようなイメージを生成していました。そのため、無効なイメージが etcd にプッシュされていました。今回のバグ修正では、新規のイメージオブジェクトを検証するように変更され、無効なイメージを修正するロジックが新たに導入されました。コントローラーは無効なイメージを生成しなくなり、無効なイメージオブジェクトのアップロードができなくなりました。(**BZ#1560311**)

- Jenkins が永続ボリュームを使用して OpenShift Container Platform pod 上にデプロイされている場合には、RPM のインストールの場所から Jenkins のホームディレクトリーへのプラグインの移行が OpenShift Container Platform v2 Jenkins RHEL イメージでは正しく行われませんでした。OpenShift Container Platform v2 Jenkins RHEL イメージにアップグレードすると、より新しいイメージに関連付けられた最新のプラグインがデプロイメントで使用されませんでした。OpenShift Container Platform v2 Jenkins RHEL イメージの **run** スクリプトは、プラグインが正しく移行されるように更新されました。OpenShift Container Platform v2 Jenkins RHEL イメージをアップグレードすると、デプロイメントに、より新しいイメージが関連付けられた最新のプラグインが含まれるようになりました。(**BZ#1550193**)

- Jenkins root URL が Jenkins テンプレートのルートから取得できない場合には、使用できない URL が、パイプラインビルドのさまざまなアノテーションの構築に使用されることがありました。関連のアノテーションリンクは、OpenShift Container Platform web コンソールから参照された場合にはレンダリングされません。このような特別なケースに対応するために、同期プラグインが Jenkins で明示的に設定された root URL を検索するようになり、root URL が正しく設定されている場合にはパイプラインビルドアノテーションに関連付けられたリンクがレンダリングされるようになりました。(**BZ#1558997**)

- インポート構成の設定で許可されたレジストリーは、イメージのインポートのみを考慮しており、イメージストリームを手動で編集してイメージインポートの検証を簡単にすり抜け、希望のイメージを使用できていました。今回のバグ修正では、イメージストリームも検証されるようになり、ホワイトリストされたレジストリーのエントリーとマッチしない外部のイメージを使用できなくなっています。(**BZ#1505315**)

- 特定の場合に、既存の etcd インストールで設定が更新されず、サービスが失敗することがあります。今回のバグ修正では、**etcd.conf** ファイルがアップグレード中に検証され、予想通りに全変数が設定されるようになりました。(**BZ#1529575**)

- Microsoft Azure でストレージデヴァイスのサポートを有効にするには、seboolean **virt_use_samba** が必要です。(**BZ#1537872**)

- ノードの設定ファイルの CRI-O セクションにハードコードされたラベルが含まれるため、インストーラーの他の場所でラベルが設定されている場合にはラベルが重複してしまう可能性があります。必要のないハードコード化されたラベルを削除し、ラベルが重複する可能性をなくします。(**BZ#1553012**)

- 本書に記載されているように、configMap で生成される **secure-forward** テンプレートには **<store>** タグが含まれません。ストアがさらに定義された場合には設定に失敗します。テンプレートには、カッコで括った **<store>** タグを追加します。コメントを削除すると構文が有効な設定になります。(**BZ#1498398**)

- Fluentd のノードにラベルを付ける場合に、スクリプトで **/tmp** が足りなくなっていました。**noexec** オプションが **/tmp** に設定された場合に、playbook は失敗していました。一時停止されている場合にスクリプトを実行するのではなく、**shell** Ansible タスクを使用して一時停止のラベルを付ける必要があります。今回のバグ修正では、一時停止して、完了するまで実行できるようになりました。(**BZ#1588009**)

- アップストリームの Kubernetes で kube-proxy iptables ルールに変更が加えられました。OpenShift Online などの非常に大規模なクラスターで、ネットワークパフォーマンスおよび全

体のシステムパフォーマンスが大きく影響を受けていました。今回のバグ修正では、kube-proxy iptables ルールが複数最適化され、パフォーマンスの問題が解決しました。(**BZ#1514174**)

- SELinux ポリシーが不適切な OVS RPM のバージョンが使用されていため、SELinux が原因で OVS 機能しなくなりました。正しいルールが適用された OVS RPM を取得する必要があります。今回のバグ修正では、OVS が機能するようになりました。(**BZ#1548677**)

- プロジェクト毎の静的 egress IP 機能を使用する場合には、egress IP が別のプロジェクトや別のノードに移動すると、egress IP が機能停止してしまう可能性があります。また、同じ egress IP が 2 つの異なるプロジェクトまたはノードに割り当てられると、重複割り当てが削除された後でも、正常に機能しなくなる可能性があります。今回のバグ修正では、この問題を解決し、プロジェクト毎の静的 egress IP がより確実に機能するはずです。(**BZ#1553294**)

- OpenShift Container Platform のデフォルトのネットワークプラグインは、Kubernetes のアップストリームで導入された新規の NetworkPolicy 機能を実装するように、まだ更新されていません (egress を制御するポリシー、pod や namespace ではなく IP アドレスベースのポリシー)。そのため、OpenShift Container Platform 3.9 では、**`ipBlock`** セクションのある NetworkPolicy を作成すると、ノードがクラッシュし、egress ルールのみが含まれる NetworkPolicy を作成すると誤って受信トラフィックがブロックされてしまいます。NetworkPolicy が実装されていないにも拘わらず、コードは、サポートのない NetworkPolicy 機能を認識してしまいます。NetworkPolicy に **`ipBlock`** ルールが含まれる場合には、これらのルールは無視されます。こうすることで、ポリシー内に **`ipBlock`** ルールしかない場合には、このポリシーは **`deny all`** として処理される可能性があります。NetworkPolicy に egress ルールのみが含まれている場合には、これらのルールは完全に無視されて、受信に影響はありません。(**BZ#1585243**)

- kubelet が使用する docker クライアントは、クライアント側に docker.io のドメインがないイメージパスを適格であるとみなす再発バグがあり、資格のないイメージパスすべてが docker.io からプルを試し、docker デーモンのドメイン検索一覧を無視していました。今回のバグ修正では、この再発バグが解決されました。(**BZ#1588768**)

- テンプレートのサービスインスタンスが削除されている場合に、このサービスインスタンスのバインドを解除すると、エラーが発生しました。TSI が含まれるプロジェクトの削除など、テンプレートのサービスインスタンスを手動で削除した場合には、サービスインスタンスのバインドを解除することができなくなりました。バインドの解除時にテンプレートのサービスインスタンスが存在しない場合には、テンプレートサービスブローカーが **success/gone** を返します。TSI が存在しない場合でも、バインドが解除されるようになりました。(**BZ#1540819**)

- namespace を削除する場合には、namespace 内のオブジェクトが、ユーザーではなく、namespace コントローラーによる削除されます。削除時に、削除を行うユーザーに関連付けられたバインド解除の要求を使用して、サービスのバインドが解除されるので、namespace コントローラーからのバインド解除要求に、バインドを解除する権限すべてが付与されなくなっていました。バインド解除に必要なパーミッションを変更して、namespace コントローラーに割り当てられている権限と同じにする必要があります。バインドを削除する namespace コントローラーがトリガーしたバインド解除も成功するようになりました。(**BZ#1554141**)

- 今回のバグ修正では、3.7 から 3.9 に変更時に発生する API エンドポイントの問題をなくすため、小規模な互換性チェックが追加されました。(**BZ#1554145**)

- ノードのアップグレードプロセス中に任意のタスクを実行するフックセットを定義できるようになりました。これらのフックを実装するには、**`openshift_node_upgrade_pre_hook`**、**`openshift_node_upgrade_hook`** または **`openshift_node_upgrade_post_hook`** を、実行するタスクファイルのパスに設定します。**`The openshift_node_upgrade_pre_hook`** フックは、ノードをドレインして、アップグレードする前に実行します。**`openshift_node_upgrade_hook`** は、ノードをドレインして

パッケージを更新してから、スケジュール可能とマークされるまでに実行されます。また、**openshift_node_upgrade_post_hook** フックは、ノードがスケジュール可能とマークされてから、他のノードに移行される直前に実行されます。(**BZ#1572786**)

- OpenShift Container Platform のルーティング設定に対する入力検証が正しく行われないと、シャード全体がダウンしてしまいます。悪意のあるユーザーがこの脆弱性を利用して、ルーターシャードを使用する他のユーザーに対して DoS 攻撃を行うことができます。(**BZ#1553035**)

- OpenShift および Atomic Enterprise Ansible は、誤った設定の etcd ファイルをデプロイし、SSL クライアントの証明書認証が無効になってしまいました。**etcd.conf** にある **ETCD_CLIENT_CERT_AUTH** と **ETCD_PEER_CLIENT_CERT_AUTH** の値に引用符を付けると、リモートユーザーがマスターノードのネットワークにバインドされている etcd サーバーにアクセスできる場合に、認証なしにこのユーザーの接続を許可するように etcd サーバーが設定されてしまいました。攻撃者はこの欠陥を使用して、etcd データセンター内の OpenShift Container Platform クラスターに関する全データを読み取ることも変更することもでき、別のコンピューターノードの追加や、全クラスターを終了させることも可能な場合があります。(**BZ#1557822**)

- 特権昇格の不具合が OpenShift Container Platform の source-to-image コンポーネントに見つかりました。この不具合は、assemble スクリプトが、特権のないコンテナーで root ユーザーとして実行できてしまいます。攻撃者はこの不具合を利用して、root ユーザーしか通常利用できないホストで、ネットワーク接続や他のアクションを開始できるようになります。(BZ#1579096) (**BZ#1579096**)

2.8.3.2. 機能拡張

- 新しいフラグが **oc adm drain** コマンドに追加され、ラベル別にノードを選択できるようになりました。個別のノードで **drain** 操作を実行する必要なく、複数のノードをドレインできる必要がありました。**oc adm drain** コマンドが **--selector** フラグをサポートするようになり、すべてのノードがドレインされる特定のラベルと一致するようになりました。(**BZ#1466390**)

2.8.3.3. アップグレード

既存の OpenShift Container Platform 3.7 または 3.9 クラスターを最新のこのリリースにアップグレードするには、自動アップグレード playbook を使用します。説明は、「クラスターの自動インプレースアップグレードの実行」を参照してください。

2.8.4. RHBA-2018:2213: OpenShift Container Platform 3.9.33 バグ修正の更新

発行日: 2018-07-18

OpenShift Container Platform release 3.9.33 が公開されました。この更新に含まれるパッケージおよびバグ修正は、RHBA-2018:2213 アドバイザリーにまとめられています。この更新に含まれるコンテナーイメージは、RHBA-2018:2212 アドバイザリーで提供されます。

2.8.4.1. アップグレード

既存の OpenShift Container Platform 3.6 または 3.7 クラスターを最新のこのリリースにアップグレードするには、自動アップグレード playbook を使用します。説明は、「クラスターの自動インプレースアップグレードの実行」を参照してください。

第3章 XPAAS リリースノート

xPaaS ドキュメントのリリースノードは、Red Hat カスタマーポータル で XPaaS 専用のブックに移行されています。

第4章 OPENSHIFT ENTERPRISE 2 との比較

4.1. 概要

OpenShift Container Platform 3 は OpenShift バージョン 3 (v3) のアーキテクチャーをベースにしており、OpenShift バージョン 2 (v2) とは非常に異なる製品です。OpenShift v2 と同じ用語が多く v3 で使用されており、同じ機能が実行されますが、用語が異なる場合もあり、背景で起こっている操作は非常に異なる場合がありますが、OpenShift 自体はアプリケーションプラットフォームのままです。

このトピックでは、バージョンの違いを詳しく説明し、OpenShift v2 から OpenShift v3 に移行する OpenShift のユーザーをサポートします。

4.2. アーキテクチャーの変更

ギア vs コンテナー

ギアは OpenShift v2 のコアコンポーネントです。カーネルの namespace、cGroups および SELinux などのテクノロジーで、スケーラビリティーが高く、セキュアなコンテナーアプリケーションプラットフォームを OpenShift のユーザーに提供します。ギア自体はコンテナー技術形式の 1 つです。

OpenShift v3 は、ギアのアイデアを次のレベルに進化させ、v2 コンテナー技術の次の進化版として Docker を使用します。このコンテナーアーキテクチャーは OpenShift v3 の中核となります。

Kubernetes

OpenShift v2 のアプリケーションは一般的に複数のギアを使用するので、OpenShift v3 のアプリケーションは複数のコンテナーを予想通り使用します。OpenShift v2 では、ギアのオーケストレーション、スケジューリング、配置は、OpenShift ブローカーホストが処理していました。OpenShift v3 は、マスターホストに Kubernetes を統合してコンテナーのオーケストレーションを駆動します。

4.3. アプリケーション

アプリケーションは依然として OpenShift で中核をなします。OpenShift v2 では、アプリケーションが単一のユニットで、カートリッジタイプ 1 つに対して 1 つの Web フレームワークで構成されていました。たとえば、アプリケーションに PHP 1 つと MySQL 1 つを含めることはできますが、Ruby 1 つ、PHP 1 つ、および MySQL 2 つを含めることはできませんでした。また、MySQL などのデータベースカートリッジだけで機能させることもできませんでした。

アプリケーションの範囲に制限をもたせることで、OpenShift が環境変数を使用してアプリケーション内の全コンポーネントをシームレスのリンクすることができます。たとえば、web フレームワークはすべて **`OPENSHIFT_MYSQL_DB_HOST`** および **`OPENSHIFT_MYSQL_DB_PORT`** 変数を使用して MySQL に接続する方法を把握しますが、このリンクは、アプリケーション内に限られており、一緒に機能するように設計されたカートリッジ内でしか機能しませんでした。2 つのアプリケーション間で MySQL インスタンスを共有するなど、アプリケーションのコンポーネント間でリンクしても意味がありませんでした。

他の PaaSes の大半は、Web フレームワークだけに制限され、他の種類のコンポーネントについては外部サービスに依存しますが、OpenShift v3 ではさらに多くのアプリケーショントポロジーや管理を可能にします。

OpenShift v3 は、サービスをリンクするコンセプトとして、「アプリケーション」という用語を使用します。コンポーネントは希望する数だけいくつでも持つことができ、プロジェクト 内でコンテナー化され、柔軟にリンクでき、オプションで、グループ化や構造化するためにラベルを指定できます。この更

新モデルでは、MySQL インスタンスをスタンドアロンで、また、JBoss コンポーネント間で共有することが可能です。

柔軟にリンクすると、任意のコンポーネントを 2 つリンクできます。コンポーネントの 1 つが環境変数をエクスポートでき、他方がこれらの環境変数から値を使用できる限り (環境変数が変わる可能性あり)、ベースにするイメージを変更する必要なく 2 つのコンポーネントをリンクできます。そのため、両方をフォークして、ネットワーク化し、互換性を確保する必要なく、任意のデータベースや web フレームワークを最適にコンテナー化実装したものが直接消費できるようになります。

つまり、OpenShift 上で何でもビルドできるようになり、これが、繰り返されるライフサイクルでアプリケーション全体をビルドできるコンテナーベースのプラットフォーム、OpenShift の主要な目的なのです。

4.4. カートリッジ VS イメージ

OpenShift v2 のカートリッジのコンセプトとして、OpenShift v3 では、「イメージ」が使用されるようになりました。

OpenShift v2 のカートリッジは、アプリケーションをビルドするにあたり中核となっていました。各カートリッジでは、必要なライブラリー、ソースコード、ビルドメカニズム、接続ロジック、ルーティングロジックを事前定義済みの環境に合わせて提供し、アプリケーションのコンポーネントを実行していました。

しかし、カートリッジには欠点がありました。カートリッジでは、開発者のコンテンツとカートリッジのコンテンツの違いが明確でなく、ユーザーにはアプリケーションの各ギアにあるホームディレクトリーの所有権がありませんでした。また、カートリッジは大規模なバイナリーの配信メカニズムとして最適ではありませんでした。カートリッジ内の外部依存関係を使用することもできましたが、使用すると、カプセル化の利点がなくなってしまいます。

パッケージの観点から、イメージはカートリッジよりも多くのタスクを実行し、れたカプセル化機能や柔軟性を提供します。ただし、カートリッジには、イメージには含まれないビルド、デプロイ、ルーティングのロジックが含まれます。OpenShift v3 では、これらの追加のニーズは「Source-to-Image (S2I)」および「テンプレートの設定」で対応しています。

依存関係

OpenShift v2 では、カートリッジの依存関係は、カートリッジのマニフェストの **`Configure-Order`** または **`Requires`** で定義されていました。OpenShift v3 は、pod で事前定義の状態と合うようにする宣言モデルを使用します。適用された明示的な依存関係は、インストール時の順番だけでなく、ランタイム時に実行されます。

たとえば、開始前に別のサービスが必要な場合があります。このような依存関係チェックは、2 つのコンポーネントを作成したときだけでなく、常に適用可能です。そのため、依存関係チェックをランタイムにプッシュして、システムを長期にわたり正常な状態に保つことができます。

コレクション

OpenShift v2 のカートリッジはギア内に共同で配置されますが、OpenShift v3 の イメージ は コンテナー と 1 対 1 でマッピングされます。コンテナーは、共同配置のメカニズムとして pod を使用します。

ソースコード

OpenShift v2 では、アプリケーションは、最低でも Web フレームワーク 1 つに git リポジトリーが 1 つ必要でした。OpenShift v3 では、どのイメージがソースからビルドされているか選択でき、そのソースは OpenShift 以外に存在することもできます。ソースはイメージから切断されているので、イメージ

とソースの選択は、個別の操作となり、ソースは任意となりました。

ビルド

OpenShift v2 では、ビルドはアプリケーションギアで行われていたので、リソースに制約があるのでスケーリングができないアプリケーションにはダウンタイムが発生していました。v3 では、個別のコンテナーで ビルド が行われます。また、OpenShift v2 ビルド結果は、rsync を使用してギアを同期していました。v3 ではビルド結果は、普遍のイメージとして最初にコミットされ、内部レジストリーに公開されます。このイメージが次に、クラスターのノードのいずれかで起動するか、今後ロールバックするときに利用できるようになります。

ルーティング

OpenShift v2 では、アプリケーションがスケーラブルかどうか、さらにアプリケーションのルーティング層が高可用性 (HA) 用に有効化するかどうかを最初に選択する必要がありました。OpenShift v3 では、アプリケーションコンポーネントを 2 つ以上のレプリカにスケールアップすることで、ルート が、HA 対応の第一級オブジェクトとなります。アプリケーションを再作成したり、DNS エントリーを変更したりする必要はありません。

ルート自体はイメージから切断されています。以前のリリースでは、カートリッジによりデフォルトのルートセットが定義され、アプリケーションにエイリアスをさらに追加できました。OpenShift v3 では、テンプレートを使用して、イメージにいくつでもルートを設定できます。これらのルートを使用すると、システムルートとユーザーエイリアスを区別することなく、公開するスキーム、ホスト、パスを任意に変更できます。

4.5. ブローカー VS マスター

OpenShift v3 の マスター は、OpenShift v2 のブローカーホストとよく似ていますが、**etcd** が各マスターホストに通常インストールされるので、OpenShift v2 のブローカーが使用する MongoDB および ActiveMQ の階層は必要なくなりました。

4.6. ドメイン VS プロジェクト

「プロジェクト」は基本的には v2 のドメインに相当します。