Red Hat OpenShift Service on AWS 4

Les tutoriels

Le Red Hat OpenShift Service sur AWS tutoriels

Last Updated: 2025-05-27

Red Hat OpenShift Service on AWS 4 Les tutoriels

Le Red Hat OpenShift Service sur AWS tutoriels

Notice légale

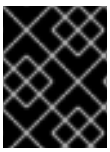
Résumé

Tutoriels sur la création de votre premier cluster Red Hat OpenShift Service sur AWS (ROSA).

Table des matières

CHAPITRE 1. APERÇU DES TUTORIELS

Faites appel aux tutoriels étape par étape des experts Red Hat pour tirer le meilleur parti de votre cluster Managed OpenShift.

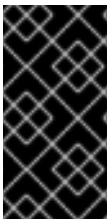


IMPORTANT

Ce contenu est rédigé par des experts de Red Hat mais n'a pas encore été testé sur toutes les configurations prises en charge.

CHAPITRE 2. TUTORIEL: ROSA AVEC ACTIVATION HCP ET LIAISON DE COMPTE

Ce tutoriel décrit le processus d'activation du service Red Hat OpenShift sur AWS (ROSA) avec des plans de contrôle hébergés (HCP) et la connexion à un compte AWS, avant de déployer le premier cluster.



IMPORTANT

Lorsque vous avez reçu une offre privée pour le produit, assurez-vous de procéder conformément aux instructions fournies avec l'offre privée avant de suivre ce tutoriel. L'offre privée est conçue soit pour un cas où le produit est déjà activé, qui remplace un abonnement actif, soit pour la première fois des activations.

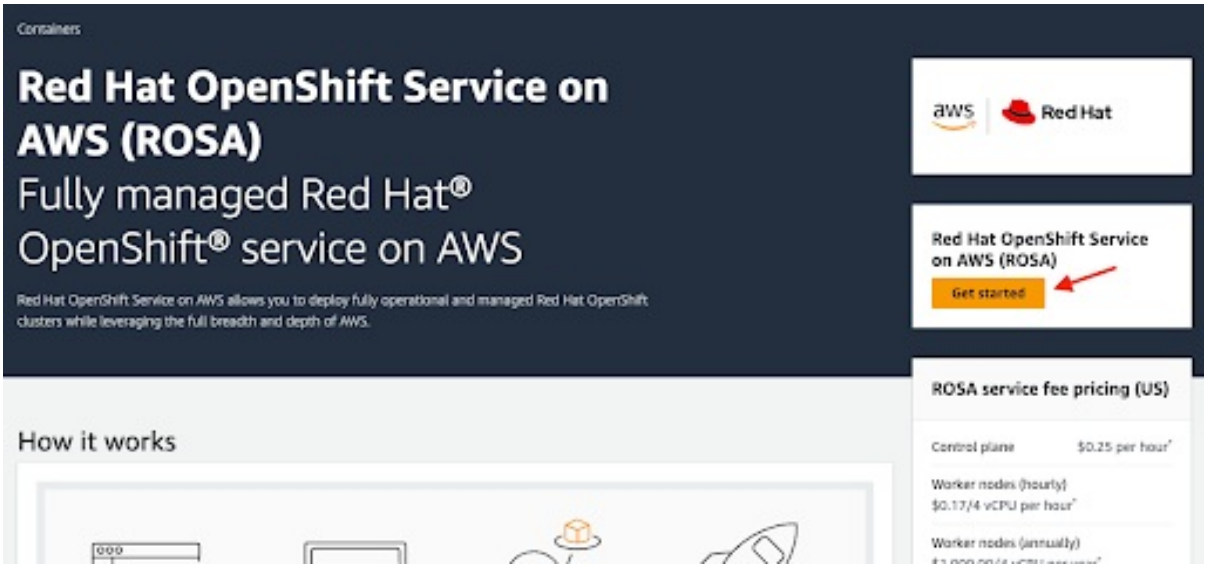
2.1. CONDITIONS PRÉALABLES

- Assurez-vous de vous connecter au compte Red Hat que vous envisagez d'associer au compte AWS où vous avez activé ROSA avec HCP dans les étapes précédentes.
- Le compte AWS utilisé pour la facturation de service ne peut être associé qu'à un seul compte Red Hat. Généralement, un compte de payeur AWS est celui qui est utilisé pour s'abonner à ROSA et utilisé pour le lien de compte et la facturation.
- Les membres de l'équipe appartenant à la même organisation Red Hat peuvent utiliser le compte AWS lié pour la facturation de service tout en créant ROSA avec des clusters HCP.

2.2. ACTIVATION DE L'ABONNEMENT ET CONFIGURATION DU COMPTE AWS

1. Activez le ROSA avec le produit HCP sur la page de la console AWS en cliquant sur le bouton Démarrer:

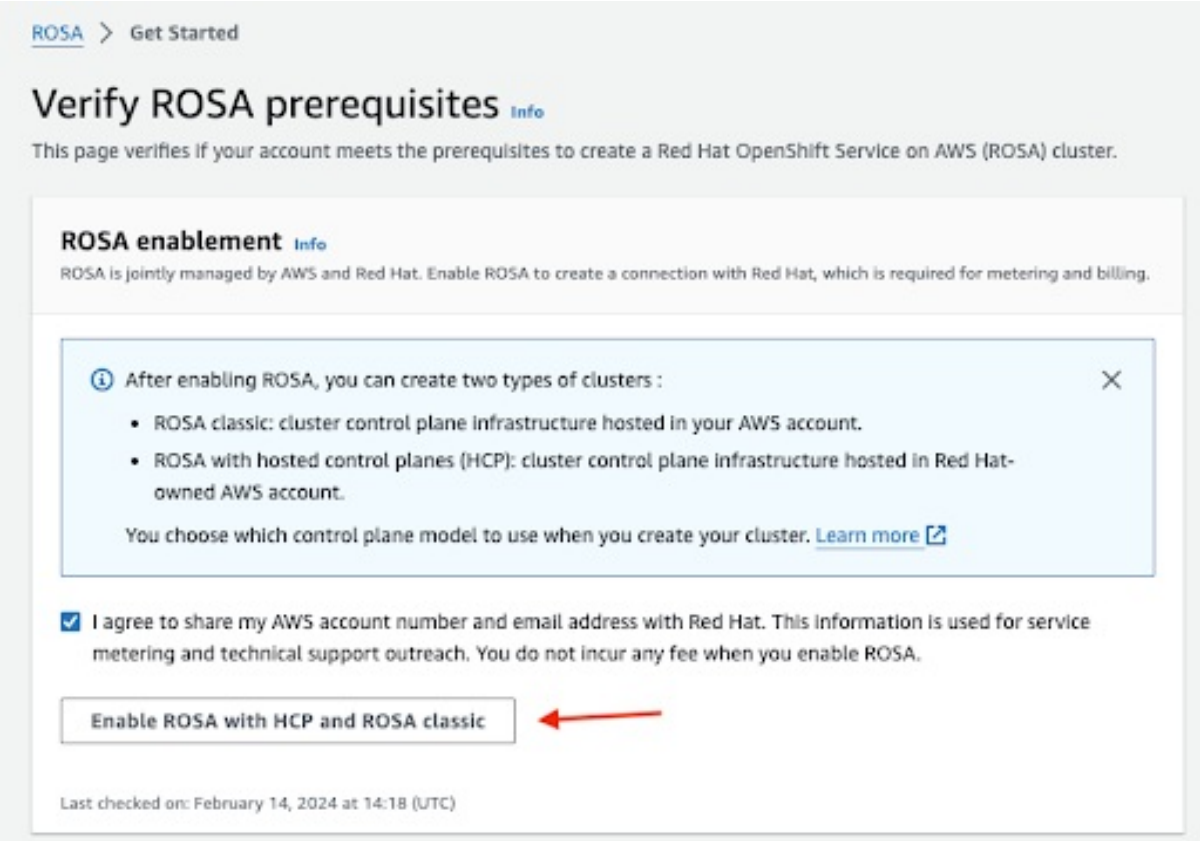
Figure 2.1. Démarrez-vous



Lorsque vous avez activé ROSA avant mais n'avez pas terminé le processus, vous pouvez cliquer sur le bouton et compléter le lien de compte comme décrit dans les étapes suivantes.

2. Confirmez que vous souhaitez que vos coordonnées soient partagées avec Red Hat et activez le service:

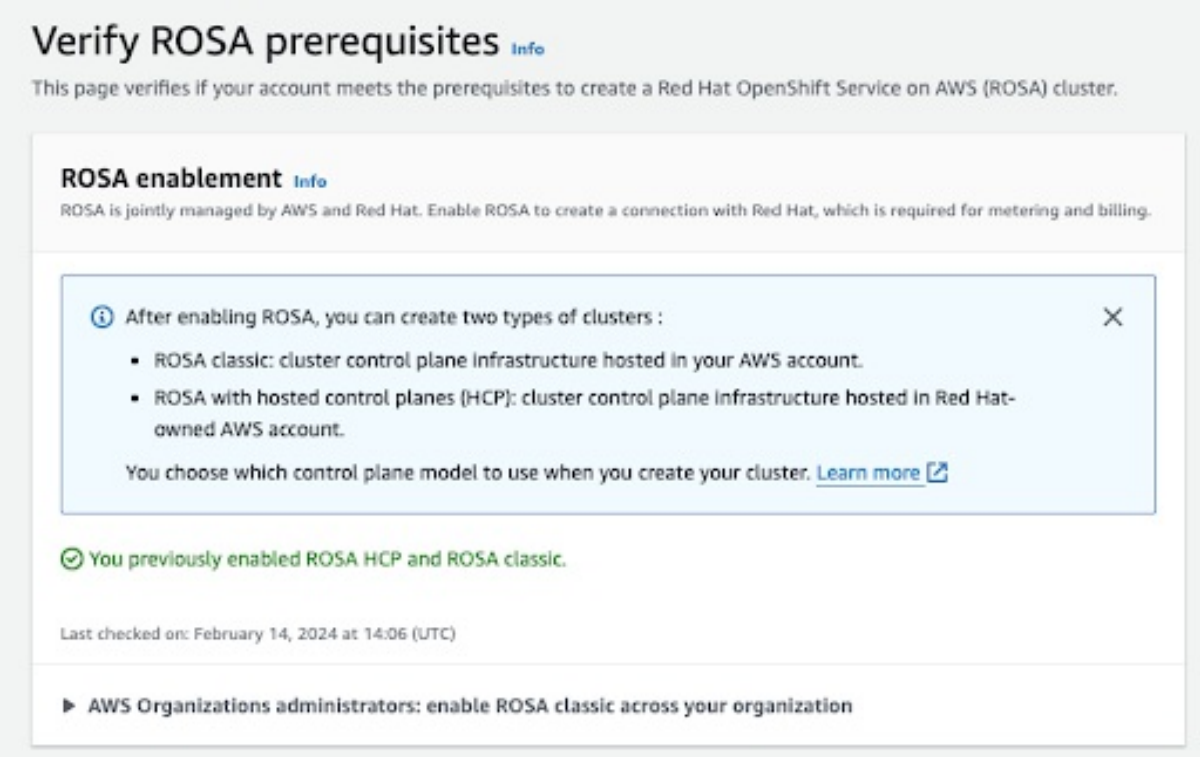
Figure 2.2. Activer ROSA



- Il ne vous sera pas facturé en activant le service dans cette étape. La connexion est faite pour la facturation et le comptage qui n'aura lieu qu'après le déploiement de votre premier cluster. Cela pourrait prendre quelques minutes.

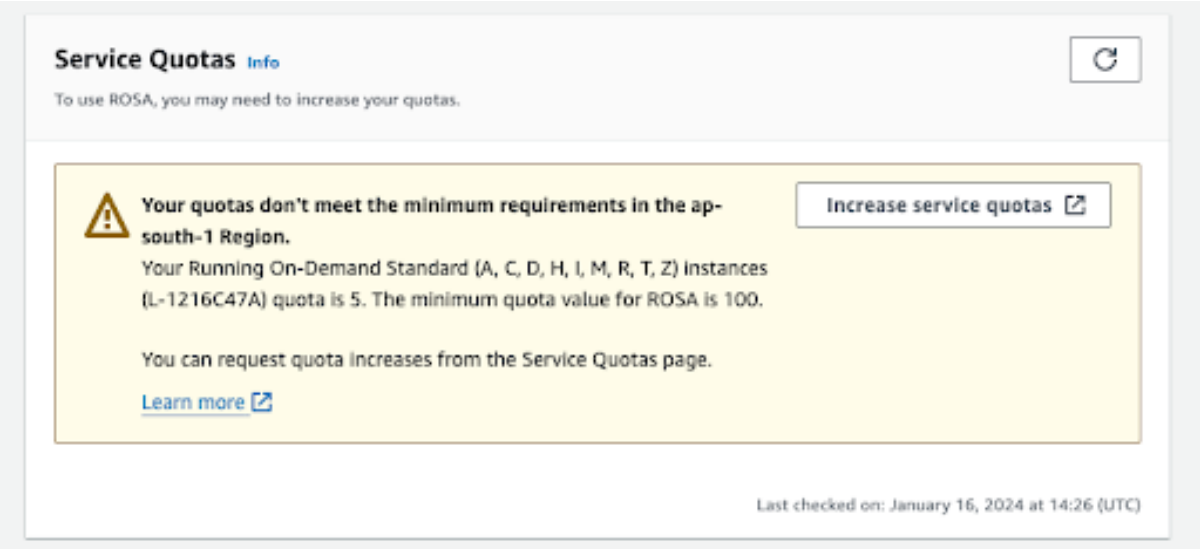
3. Après la fin du processus, vous verrez une confirmation:

Figure 2.3. Confirmation de l'activation ROSA



4. D'autres sections de cette page de vérification montrent l'état des conditions préalables supplémentaires. Dans le cas où l'une de ces conditions préalables ne serait pas remplie, un message correspondant est affiché. En voici un exemple d'insuffisance des quotas dans la région sélectionnée:

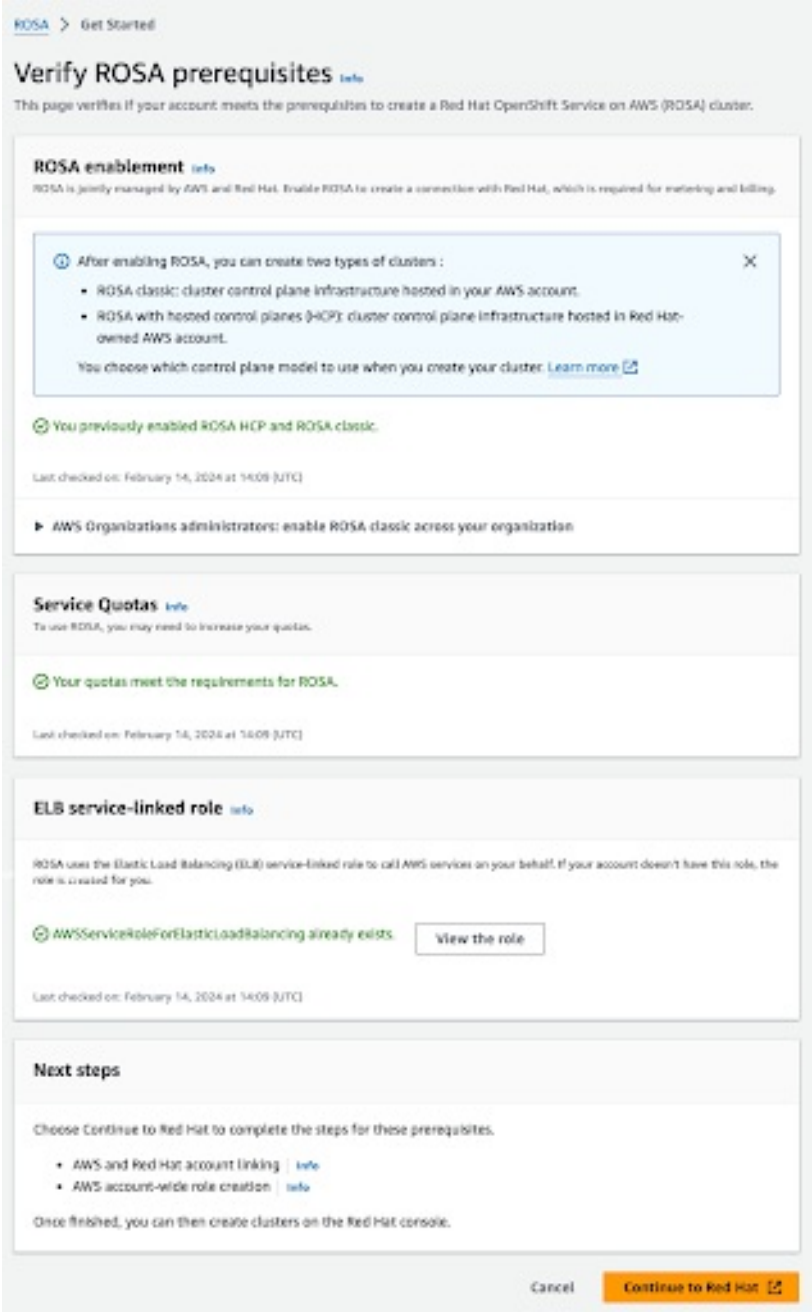
Figure 2.4. Quotas de service



   a. Cliquez sur le bouton Augmenter les quotas de service ou utilisez le lien En savoir plus pour obtenir plus d'informations sur la façon de gérer les quotas de service. Dans le cas d'un quota insuffisant, notez que les quotas sont spécifiques à la région. Dans le coin supérieur droit de la console Web, vous pouvez utiliser le commutateur de région pour redémarrer la vérification des quotas pour n'importe quelle région qui vous intéresse, puis soumettre des demandes d'augmentation de quota de service au besoin.

5. Lorsque toutes les conditions préalables sont remplies, la page ressemblera à ceci:

Figure 2.5. Vérifier les prérequis de ROSA

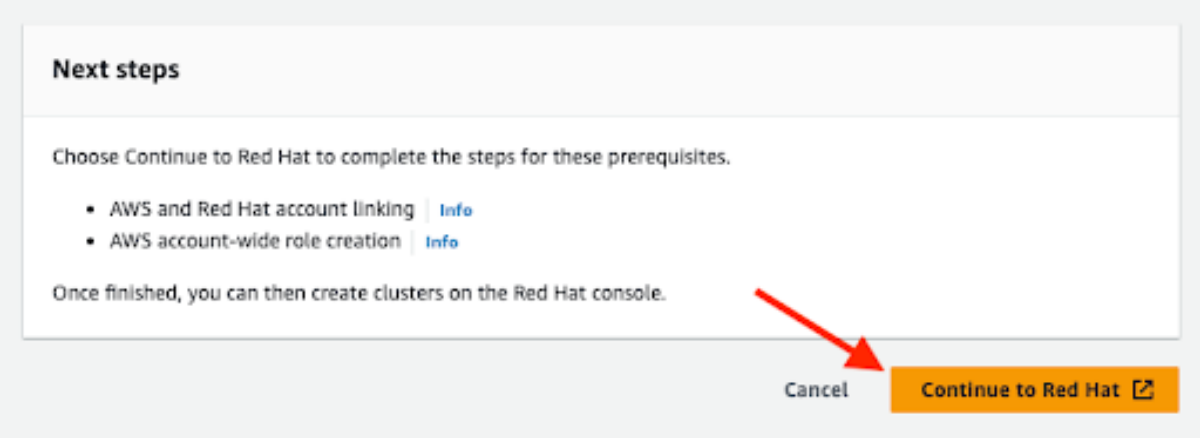


Le rôle ELB lié au service est créé automatiquement pour vous. Cliquez sur l'un des petits liens en bleu Info pour obtenir de l'aide et des ressources contextuelles.

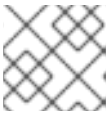
2.3. COMPTE AWS ET RED HAT ET LIEN D'ABONNEMENT

1. Cliquez sur le bouton orange Continuer à Red Hat pour continuer avec le lien de compte:

Figure 2.6. Continuer à Red Hat



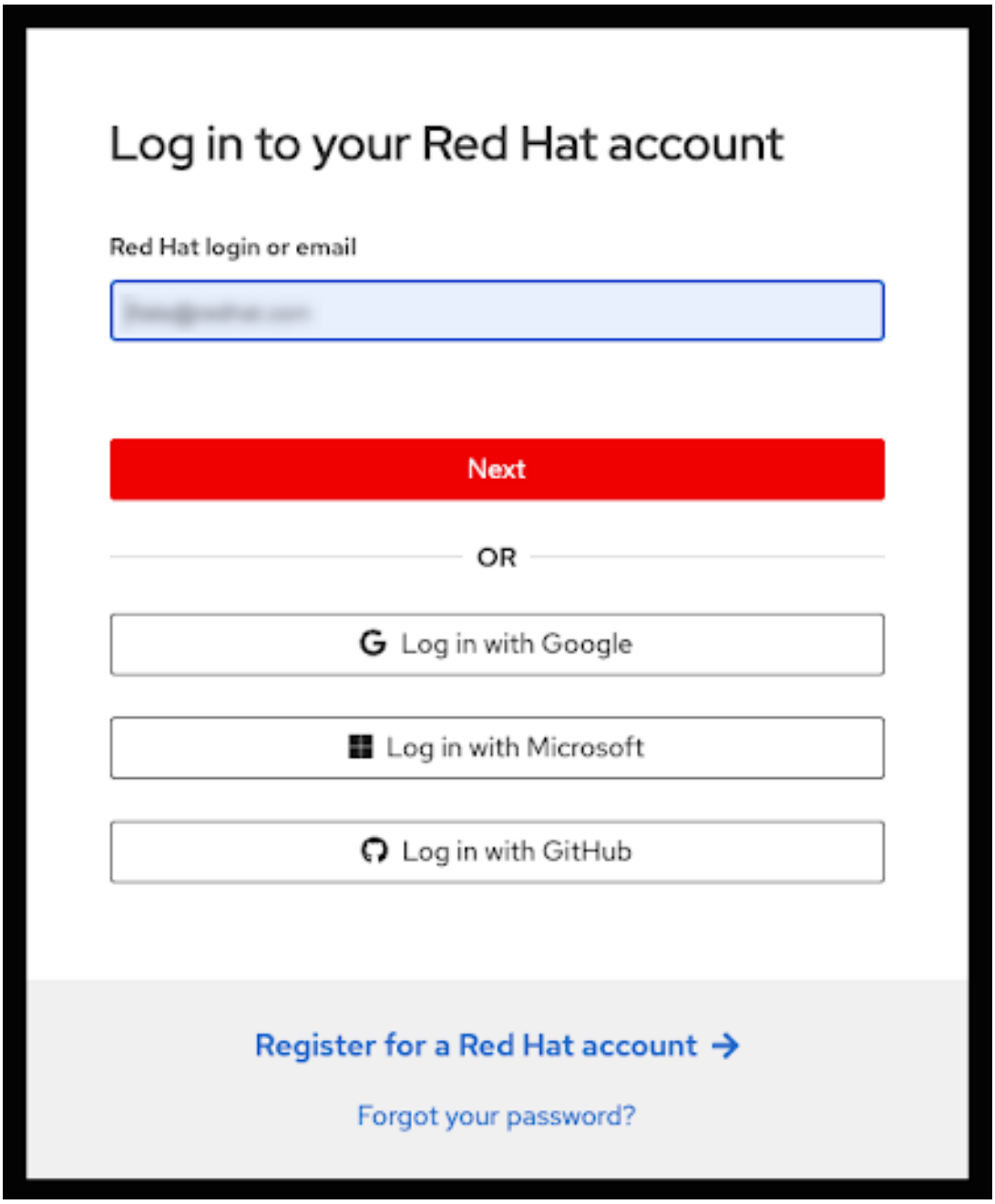
2. Dans le cas où vous n'êtes pas déjà connecté à votre compte Red Hat dans la session de votre navigateur actuel, il vous sera demandé de vous connecter à votre compte:



NOTE

Le compte AWS doit être lié à une seule organisation Red Hat.

Figure 2.7. Connectez-vous à votre compte Red Hat



- Il est également possible de vous inscrire à un nouveau compte Red Hat ou de réinitialiser votre mot de passe sur cette page.
- Assurez-vous de vous connecter au compte Red Hat que vous envisagez d'associer au compte AWS où vous avez activé ROSA avec HCP dans les étapes précédentes.
- Le compte AWS utilisé pour la facturation de service ne peut être associé qu'à un seul compte Red Hat. Généralement, un compte de payeur AWS est celui qui est utilisé pour s'abonner à ROSA et utilisé pour le lien de compte et la facturation.
- Les membres de l'équipe appartenant à la même organisation Red Hat peuvent utiliser le compte AWS lié pour la facturation de service tout en créant ROSA avec des clusters HCP.

3. Complétez le lien du compte Red Hat après avoir examiné les termes et conditions:

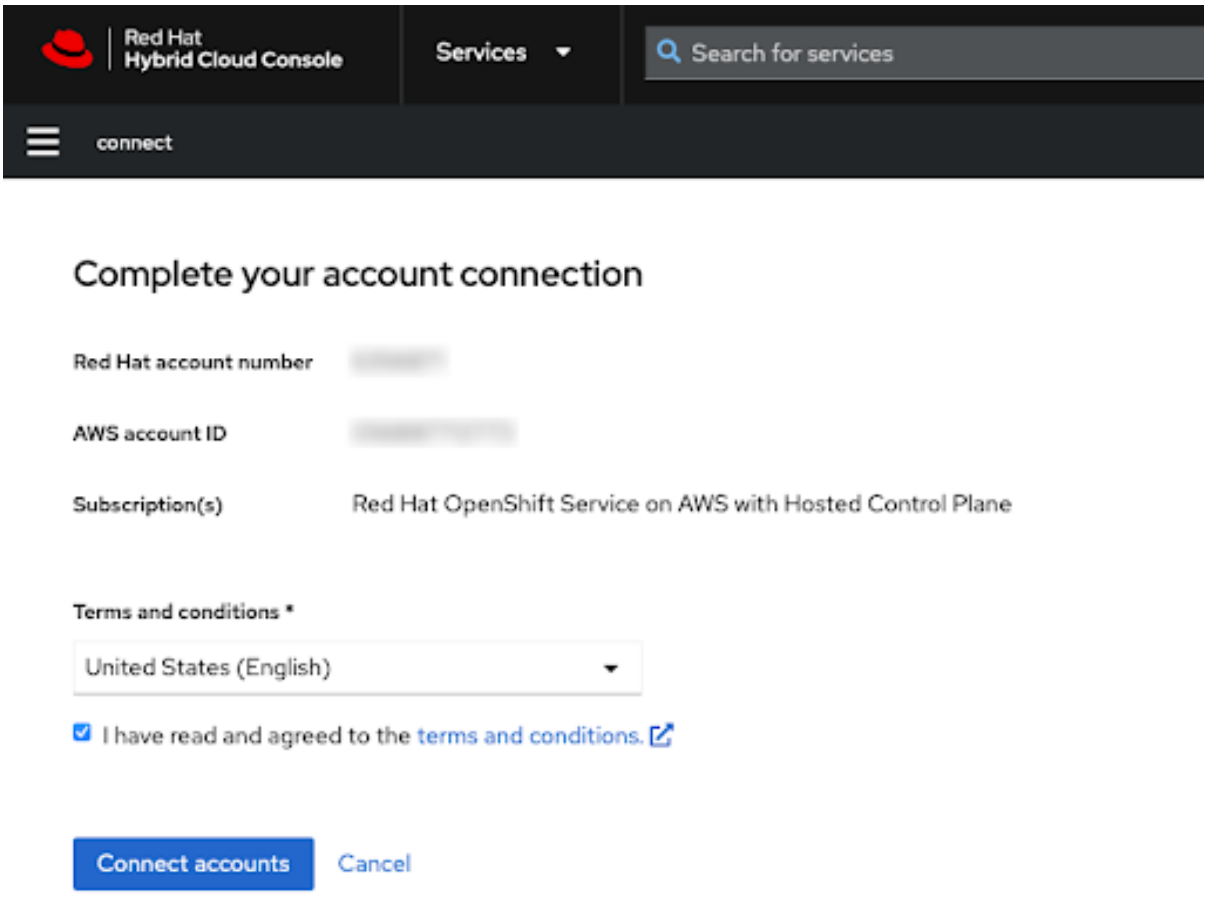
NOTE

Cette étape n'est disponible que si le compte AWS n'était pas lié à un compte Red Hat auparavant.

Cette étape est ignorée si le compte AWS est déjà lié au compte Red Hat connecté à l'utilisateur.

Lorsque le compte AWS est lié à un autre compte Red Hat, une erreur s'affiche. Consultez les informations sur le compte de facturation pour les clusters HCP pour le dépannage.

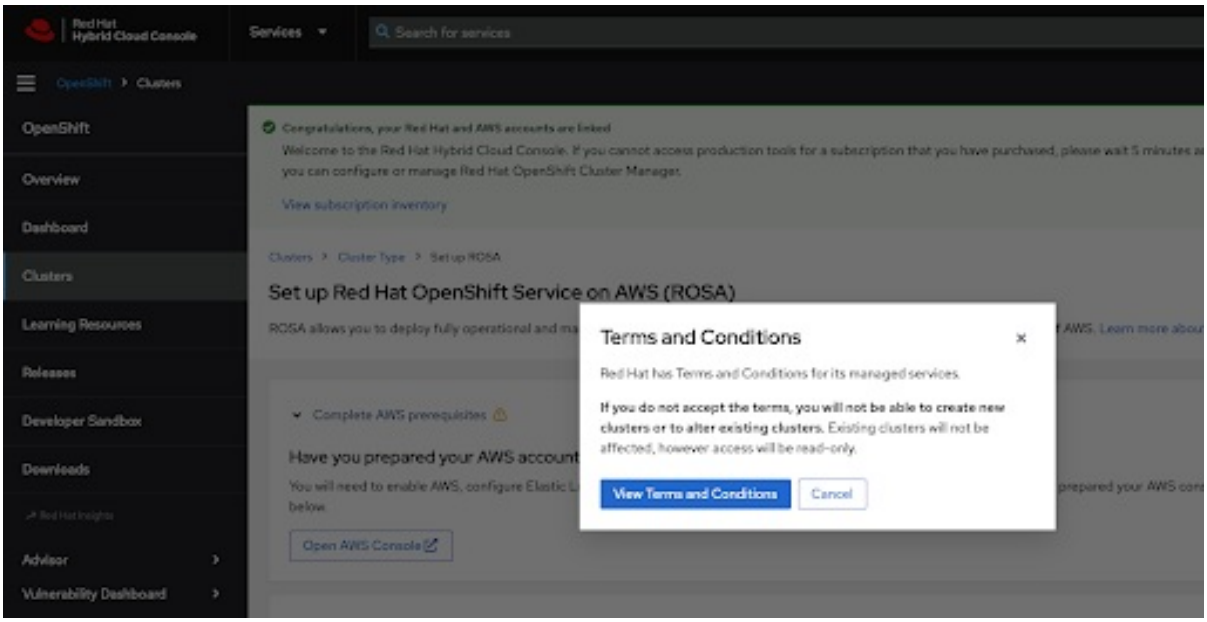
Figure 2.8. Complétez votre connexion de compte



Les numéros de compte Red Hat et AWS sont affichés sur cet écran.

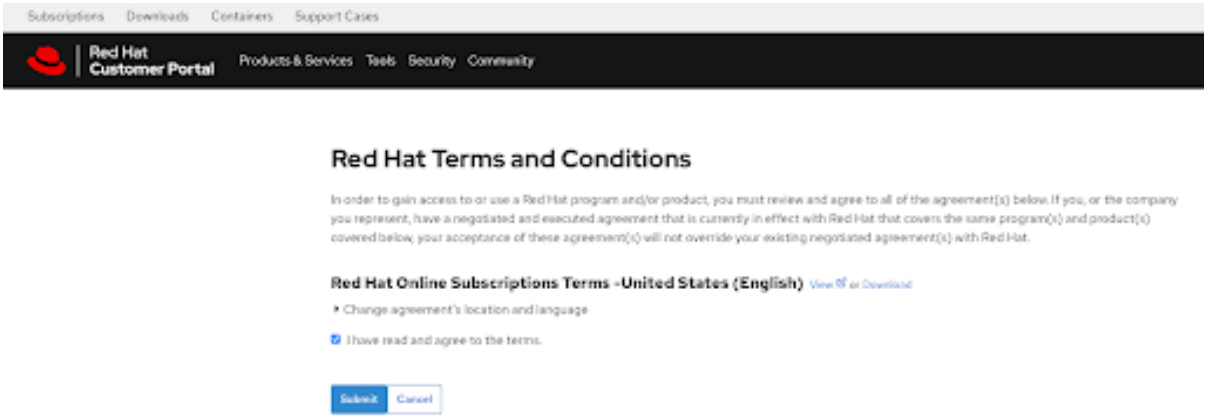
4. Cliquez sur le bouton Connecter les comptes si vous êtes d'accord avec les conditions de service.
   Lorsque c'est la première fois que vous utilisez la console Red Hat Hybrid Cloud, il vous sera demandé d'être d'accord avec les conditions générales des services gérés avant de pouvoir créer le premier cluster ROSA:

Figure 2.9. Conditions générales



D'autres termes qui doivent être examinés et acceptés sont affichés après avoir cliqué sur le bouton Afficher les conditions générales:

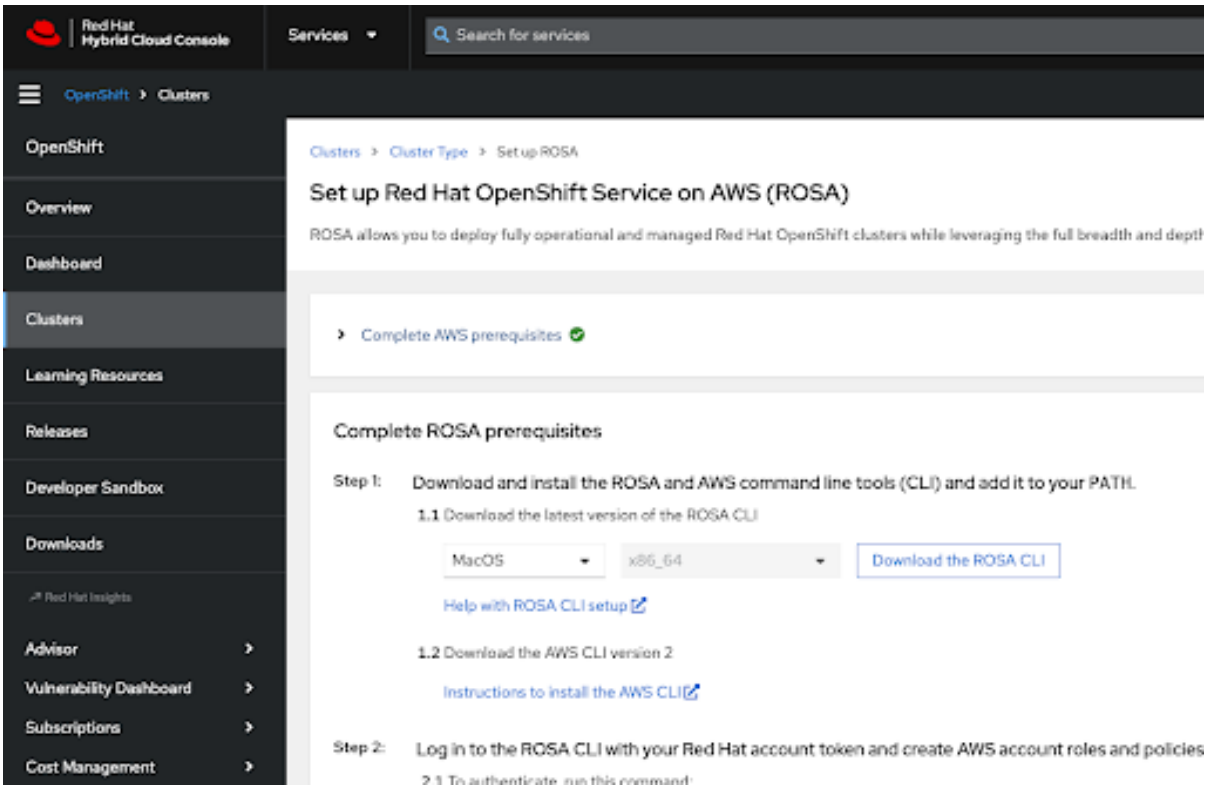
Figure 2.10. Conditions générales de Red Hat



Envoyez votre accord une fois que vous avez examiné des conditions supplémentaires lorsque vous l'invitez à ce moment-là.

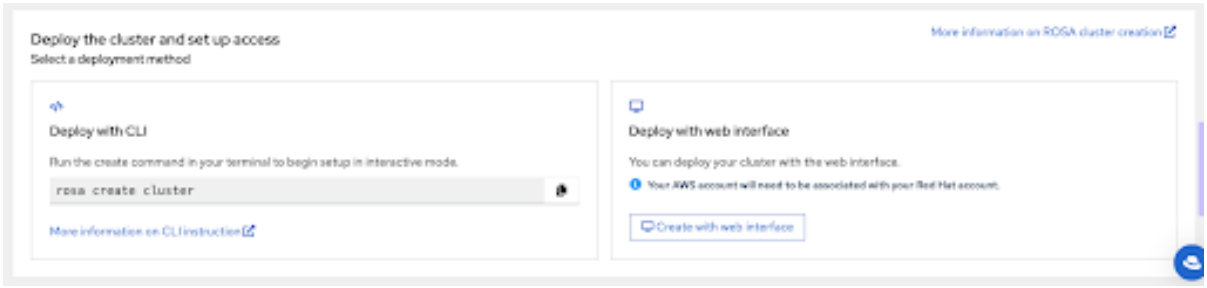
5. La console hybride Cloud fournit une confirmation que la configuration du compte AWS a été terminée et répertorie les conditions préalables au déploiement de clusters:

Figure 2.11. Conditions préalables complètes de ROSA



La dernière section de cette page montre les options de déploiement de clusters, soit en utilisant le rosa CLI ou via la console web:

Figure 2.12. Déployez le cluster et configurez l'accès



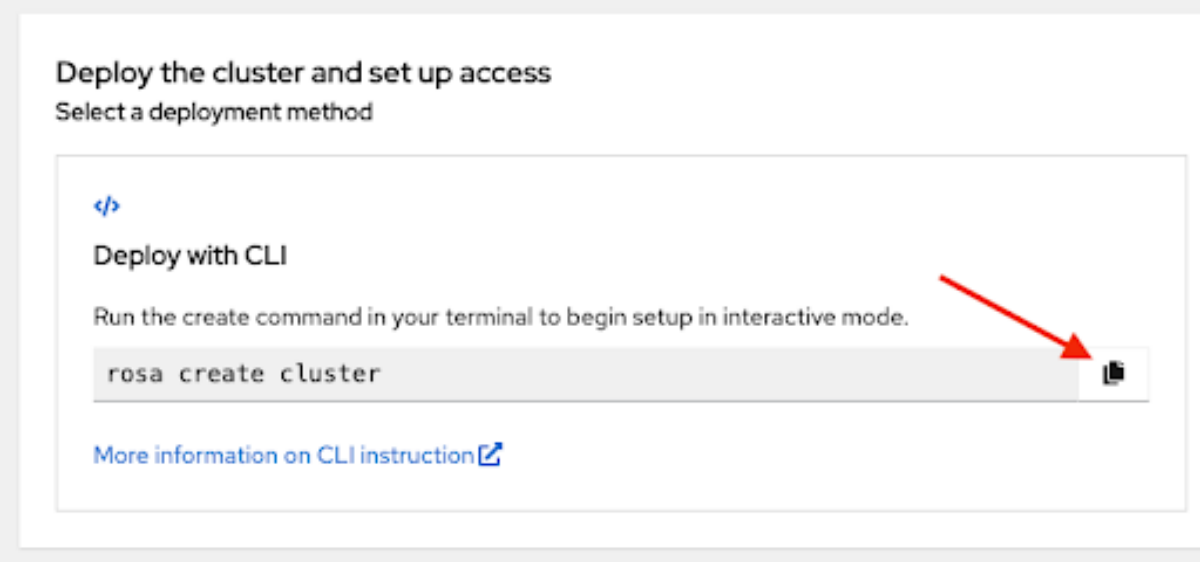
2.4. CHOISIR LE COMPTE DE FACTURATION AWS POUR ROSA AVEC HCP LORS DU DÉPLOIEMENT DU CLUSTER À L'AIDE DU CLI

IMPORTANT

Assurez-vous que vous avez installé l'interface de ligne de commande ROSA la plus récente (CLI) et AWS CLI et que vous ayez rempli les prérequis ROSA couverts dans la section précédente. Consultez Aide avec ROSA CLI configuration et Instructions pour installer l'AWS CLI pour plus d'informations.

1. Lancez le déploiement de clusters à l'aide de la commande rosa create cluster. Cliquez sur le bouton copier sur la page de la console de Red Hat OpenShift sur la page de la console AWS (ROSA) et collez la commande dans votre terminal. Cela lance le processus de création de clusters en mode interactif:

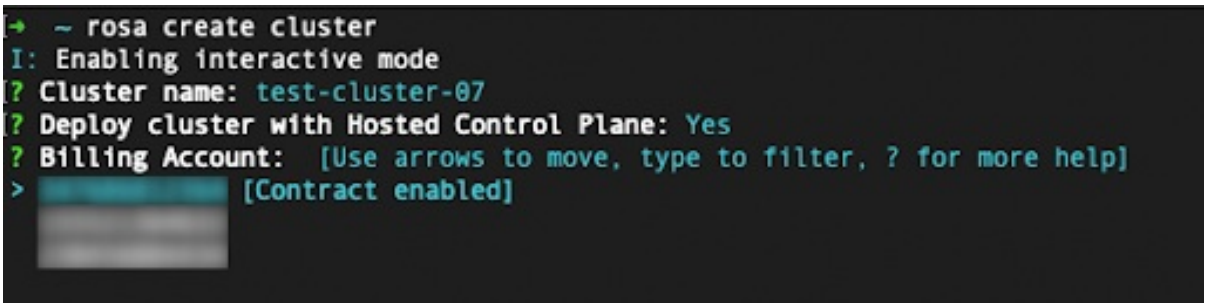
Figure 2.13. Déployez le cluster et configurez l'accès



2. Afin d'utiliser un profil AWS personnalisé, l'un des profils non par défaut spécifiés dans votre ~/.aws/credentials, vous pouvez ajouter le sélecteur -proffile <profile_name> à la commande rosa créer cluster afin que la commande ressemble à rosa créer cluster étape. Dans le cas où aucun profil AWS CLI n'est spécifié à l'aide de cette option, le profil AWS CLI par défaut déterminera le profil d'infrastructure AWS dans lequel le cluster est déployé. Le profil AWS de facturation est sélectionné dans l'une des étapes suivantes.

3. Lors du déploiement d'un ROSA avec le cluster HCP, le compte AWS de facturation doit être spécifié:

Figure 2.14. Indiquez le compte de facturation



- Les comptes AWS qui sont liés au compte Red Hat connecté à l'utilisateur sont affichés.
- Le compte AWS spécifié est facturé pour l'utilisation du service ROSA.
- L'indicateur indique si le contrat ROSA est activé ou non pour un compte de facturation AWS donné.
    - Lorsque vous sélectionnez un compte de facturation AWS qui affiche l'étiquette activée par le contrat, les taux de consommation à la demande ne sont facturés qu'après la consommation de la capacité de votre contrat prépayé.
    - Les comptes AWS sans l'étiquette activée par le contrat sont facturés les taux de consommation applicables à la demande.

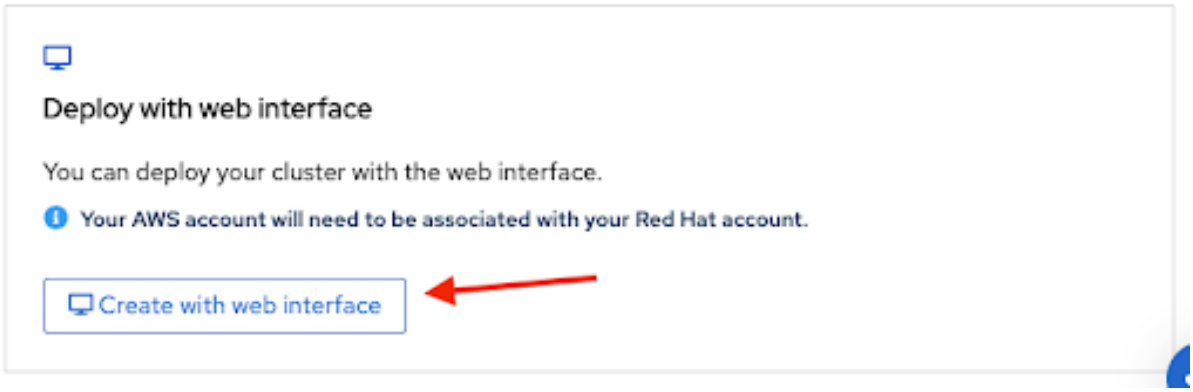
Ressources supplémentaires

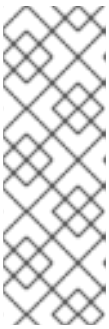
- Les étapes détaillées de déploiement de clusters dépassent le cadre de ce tutoriel. Consultez Création de ROSA avec des clusters HCP en utilisant les options par défaut pour plus de détails sur la façon de compléter le ROSA avec le déploiement de cluster HCP à l'aide du CLI.

2.5. CHOISIR LE COMPTE DE FACTURATION AWS POUR ROSA AVEC HCP LORS DU DÉPLOIEMENT DU CLUSTER À L'AIDE DE LA CONSOLE WEB

1. Le cluster peut être créé à l'aide de la console web en sélectionnant la deuxième option dans la section inférieure de la page Introduction de ROSA:

   Figure 2.15. Déploiement avec interface web





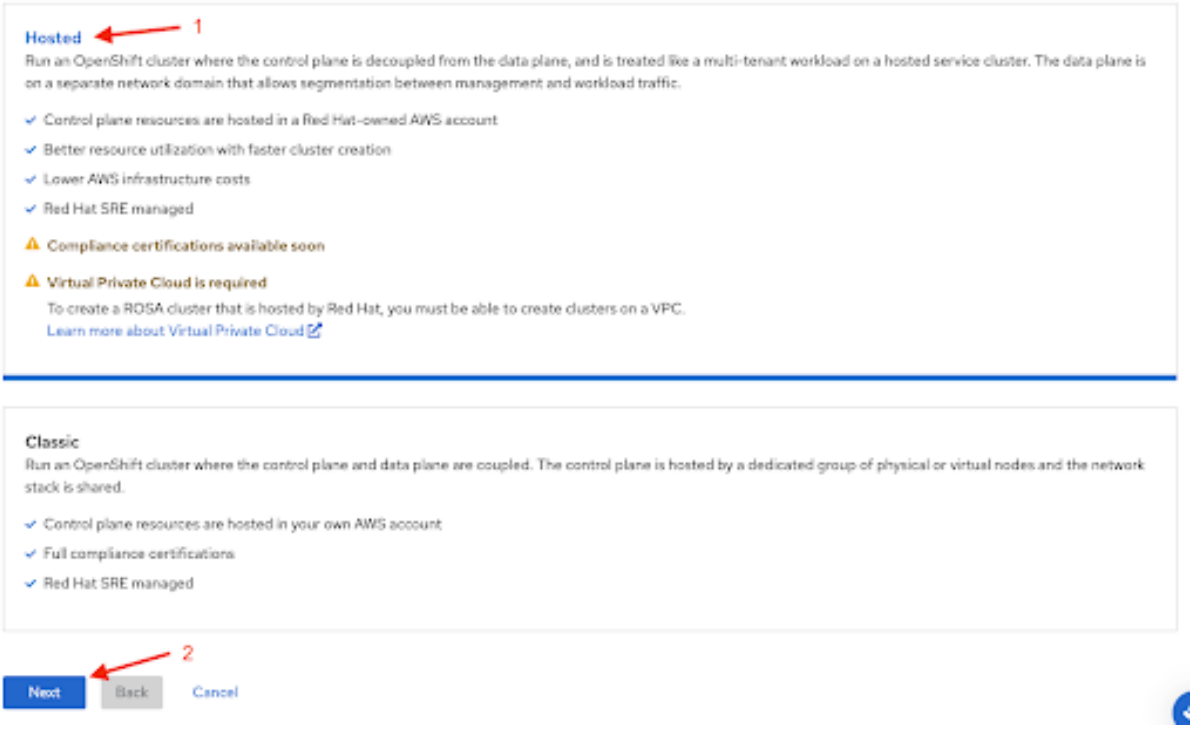
NOTE

Complétez les prérequis avant de démarrer le processus de déploiement de la console Web.

La rosa CLI est requise pour certaines tâches, telles que la création des rôles de compte. Lorsque vous déployez ROSA pour la première fois, suivez les étapes CLI jusqu'à l'exécution de la commande rosa whoami, avant de commencer les étapes de déploiement de la console Web.

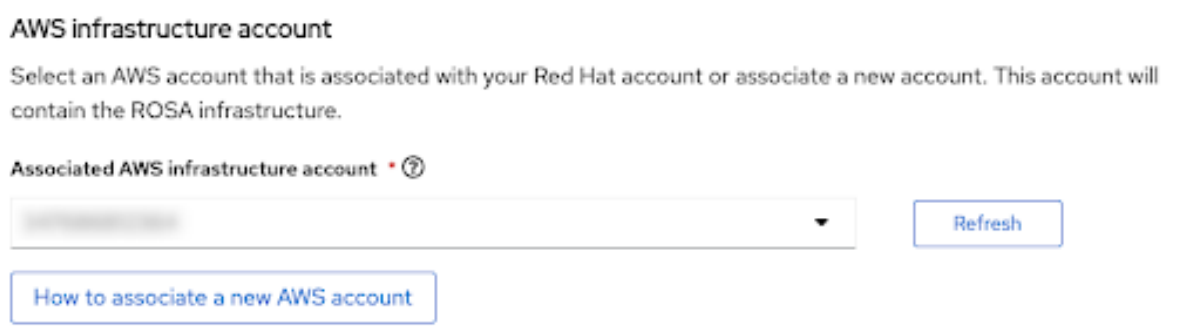
2. La première étape lors de la création d'un cluster ROSA à l'aide de la console Web est la sélection du plan de contrôle. Assurez-vous que l'option Hosted est sélectionnée avant de cliquer sur le bouton suivant:

Figure 2.16. Choisissez l'option hébergée



3. L'étape suivante Comptes et rôles vous permet de spécifier l'infrastructure du compte AWS, dans lequel le cluster ROSA est déployé et où les ressources sont consommées et gérées:

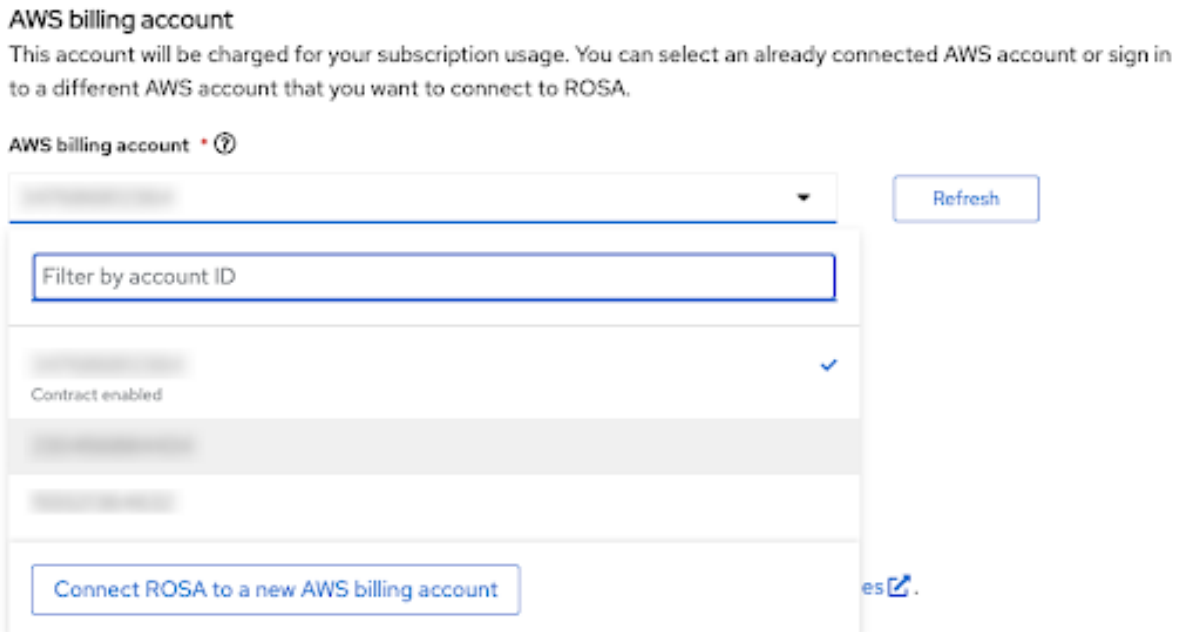
Figure 2.17. Compte infrastructure AWS



- Cliquez sur Comment associer un nouveau compte AWS, si vous ne voyez pas le compte dans lequel vous souhaitez déployer le cluster ROSA pour des informations détaillées sur la façon de créer ou de lier des rôles de compte pour cette association.
- La rosa CLI est utilisée pour cela.
- Lorsque vous utilisez plusieurs comptes AWS et que leurs profils sont configurés pour le CLI AWS, vous pouvez utiliser le sélecteur --profile pour spécifier le profil AWS lorsque vous travaillez avec les commandes rosa CLI.

4. Le compte AWS de facturation est sélectionné dans la section immédiatement suivante:

Figure 2.18. Compte de facturation AWS



- Les comptes AWS qui sont liés au compte Red Hat connecté à l'utilisateur sont affichés.
- Le compte AWS spécifié est facturé pour l'utilisation du service ROSA.
- L'indicateur indique si le contrat ROSA est activé ou non pour un compte de facturation AWS donné.
    - Lorsque vous sélectionnez un compte de facturation AWS qui affiche l'étiquette activée par le contrat, les taux de consommation à la demande ne sont facturés qu'après la consommation de la capacité de votre contrat prépayé.
    - Les comptes AWS sans l'étiquette activée par le contrat sont facturés les taux de consommation applicables à la demande.

Les étapes suivantes après la sélection de compte AWS de facturation dépassent le cadre de ce tutoriel.

Ressources supplémentaires

- Afin d'obtenir de l'information sur l'utilisation du CLI pour créer un cluster, consultez Créer un ROSA avec le cluster HCP à l'aide du CLI.
- Consultez ce chemin d'apprentissage pour plus de détails sur la façon de compléter le déploiement de cluster ROSA à l'aide de la console Web.

CHAPITRE 3. TUTORIEL: ROSA AVEC HCP OFFRE PRIVÉE ACCEPTATION ET PARTAGE

Ce guide décrit comment accepter une offre privée pour Red Hat OpenShift Service sur AWS (ROSA) avec des avions de contrôle hébergés (HCP) et comment s'assurer que tous les membres de l'équipe peuvent utiliser l'offre privée pour les clusters qu'ils fournissent.

Les coûts de ROSA avec HCP sont composés des coûts d'infrastructure AWS et du ROSA avec les coûts de service HCP. Les coûts d'infrastructure AWS, tels que les instances EC2 qui exécutent les charges de travail nécessaires, sont imputés sur le compte AWS où l'infrastructure est déployée. Les coûts de service ROSA sont facturés au compte AWS spécifié comme le « compte de facturation AWS » lors du déploiement d'un cluster.

Les composants de coûts peuvent être facturés à différents comptes AWS. La description détaillée de la façon dont le coût du service ROSA et les coûts d'infrastructure AWS sont calculés peut être trouvée sur la page de tarification AWS de Red Hat OpenShift Service.

3.1. ACCEPTER UNE OFFRE PRIVÉE

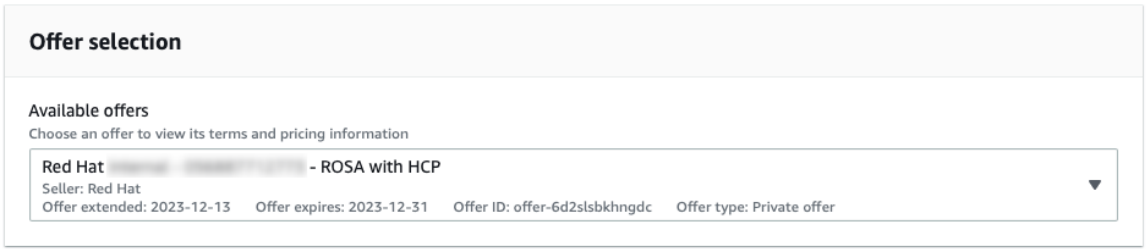
1. Lorsque vous obtenez une offre privée pour ROSA avec HCP, vous bénéficiez d'une URL unique accessible uniquement par un identifiant de compte AWS spécifique qui a été spécifié par le vendeur.

NOTE

Assurez-vous que vous êtes connecté à l'aide du compte AWS qui a été spécifié comme acheteur. Essayer d'accéder à l'offre à l'aide d'un autre compte AWS produit un message d'erreur « page non trouvée » comme le montre la figure 11 dans la section de dépannage ci-dessous.

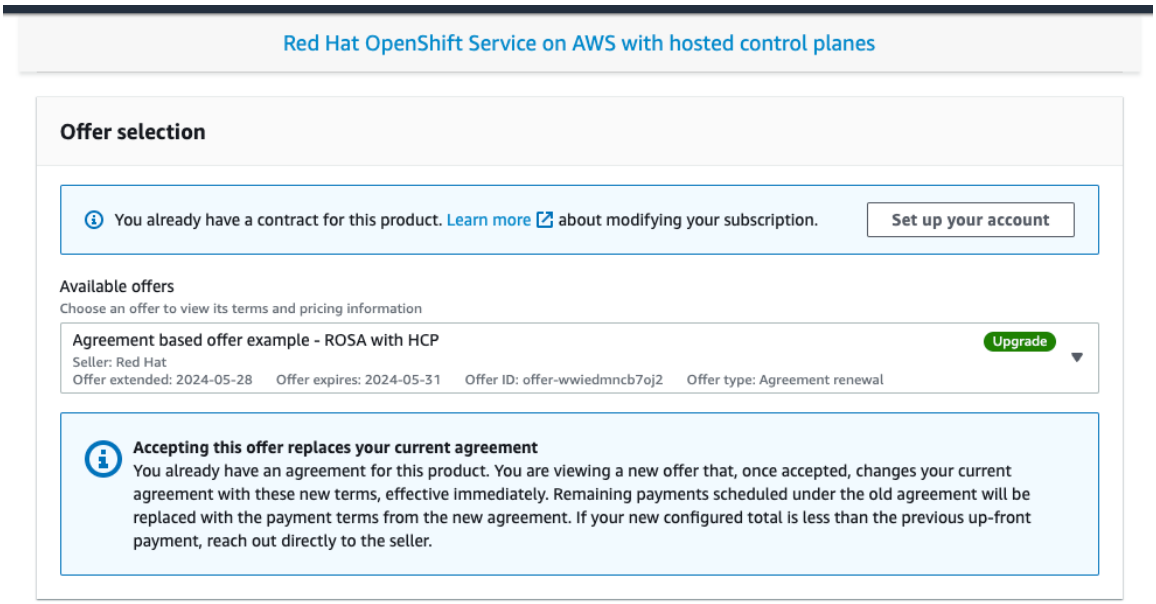
   a. Le menu déroulant de sélection d'offre avec une offre privée régulière présélectionnée dans la figure 1. Ce type d'offre ne peut être accepté que si la ROSA avec HCP n'a pas été activée avant d'utiliser l'offre publique ou une autre offre privée.

Figure 3.1. L'offre privée régulière



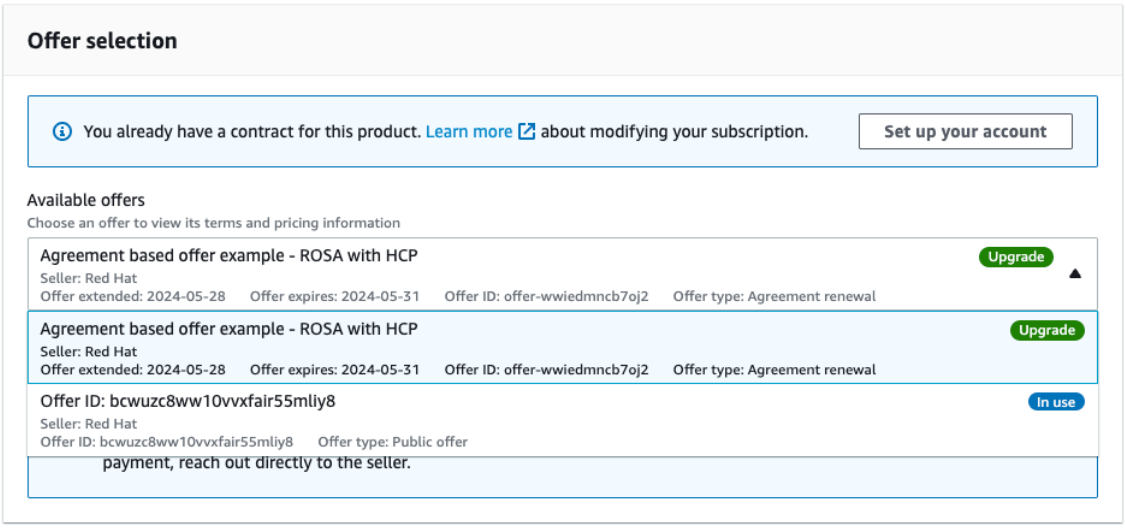
   b. Il est possible de voir une offre privée qui a été créée pour un compte AWS qui a précédemment activé ROSA avec HCP en utilisant l'offre publique, montrant le nom du produit et l'offre privée sélectionnée étiquetée comme « Mise à niveau », qui remplace le contrat actuellement en cours d'exécution pour ROSA par HCP dans la figure 2.

Figure 3.2. Écran de sélection d'offre privée

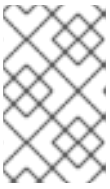


c. Le menu déroulant permet de sélectionner entre plusieurs offres, si elles sont disponibles. L'offre publique précédemment activée est présentée en même temps que l'offre nouvellement fournie basée sur l'accord qui est étiquetée comme « Mise à niveau » à la figure 3.

Figure 3.3. Liste déroulante de sélection d'offre privée



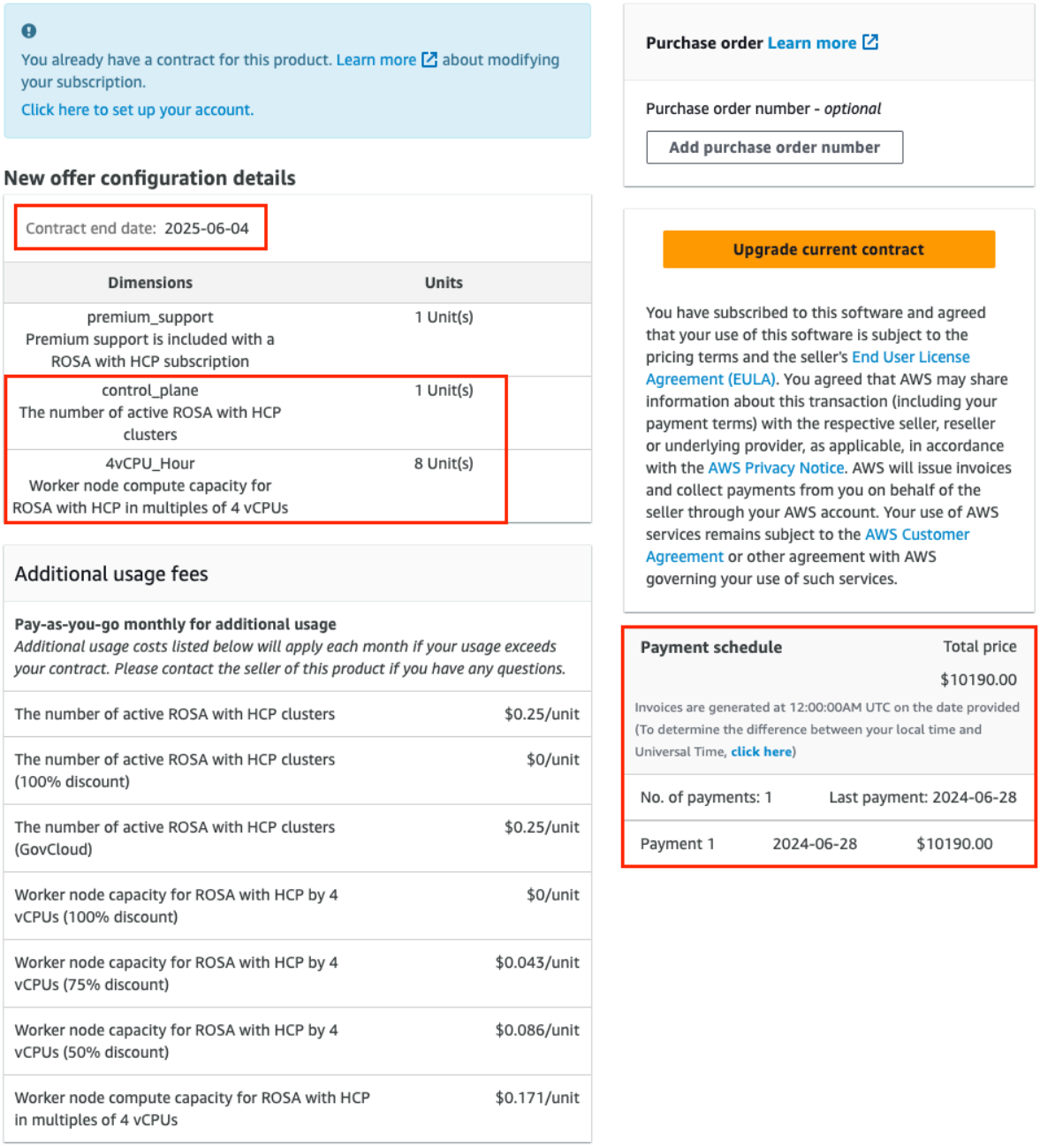
2. Assurez-vous que la configuration de votre offre est sélectionnée. La figure 4 montre la partie inférieure de la page de l'offre avec les détails de l'offre.



NOTE

La date de fin du contrat, le nombre d'unités incluses dans l'offre et le calendrier de paiement. Dans cet exemple, 1 cluster et jusqu'à 3 nœuds utilisant 4 vCPU sont inclus.

Figure 3.4. Détails de l'offre privée



3. Facultatif: vous pouvez ajouter votre propre numéro de bon de commande (PO) à l'abonnement qui est acheté, de sorte qu'il est inclus sur vos factures AWS ultérieures. En outre, vérifiez les « frais d'utilisation supplémentaires » qui sont facturés pour toute utilisation au-dessus du champ d'application de la « Nouvelle offre de détails de configuration ».

NOTE

Les offres privées ont plusieurs configurations disponibles.

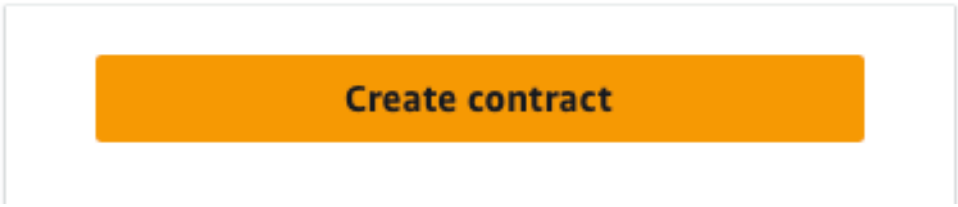
- Il est possible que l'offre privée que vous acceptez soit configurée avec une date de début future fixe.
- Lorsque vous n'avez pas une autre ROSA active avec abonnement HCP au moment de l'acceptation de l'offre privée, d'une offre publique ou d'une offre privée plus ancienne, acceptez l'offre privée elle-même et continuez avec les étapes de liaison de compte et de déploiement de clusters après la date de début du service spécifié.

Il faut avoir une ROSA active ayant le droit de remplir ces étapes. Les dates de début du service sont toujours indiquées dans le fuseau horaire UTC

4. Créez ou améliorez votre contrat.

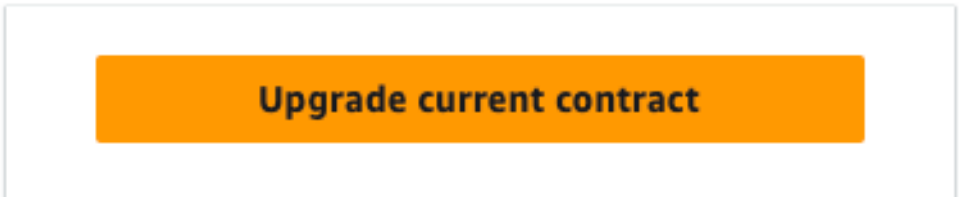
   a. Dans le cas des offres privées acceptées par un compte AWS qui n'a pas encore ROSA avec HCP activée et qui crée le premier contrat pour ce service, cliquez sur le bouton Créer un contrat.

   Figure 3.5. Créer un bouton de contrat



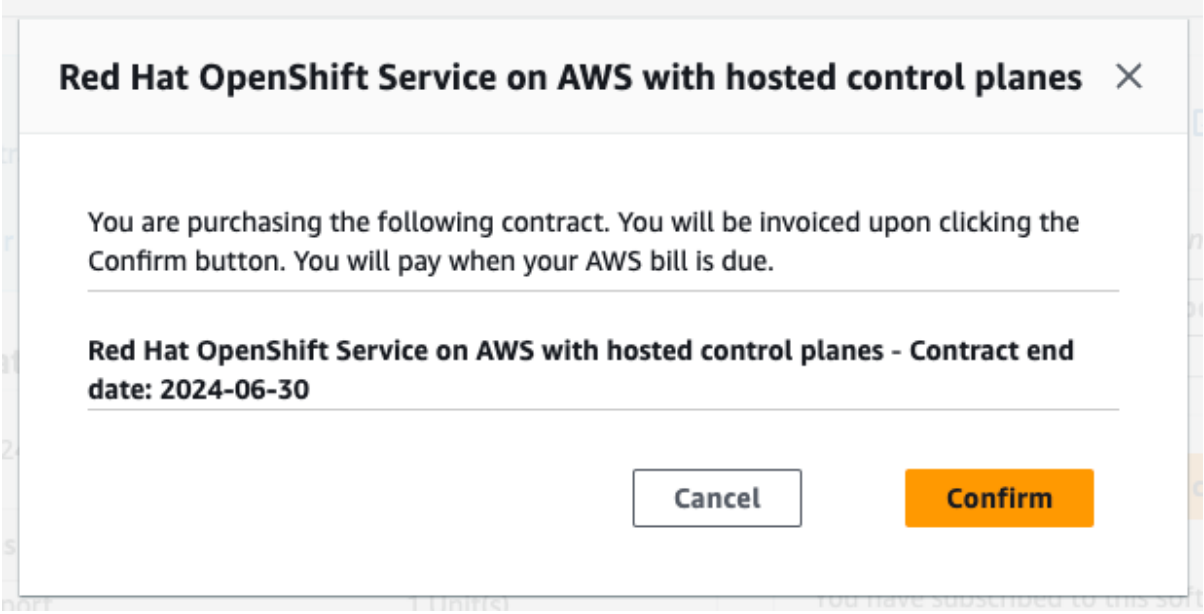
   b. Dans le cas des offres fondées sur l'accord, cliquez sur le bouton Mise à niveau du contrat actuel affiché aux figures 4 et 6.

   Figure 3.6. Bouton de mise à niveau du contrat



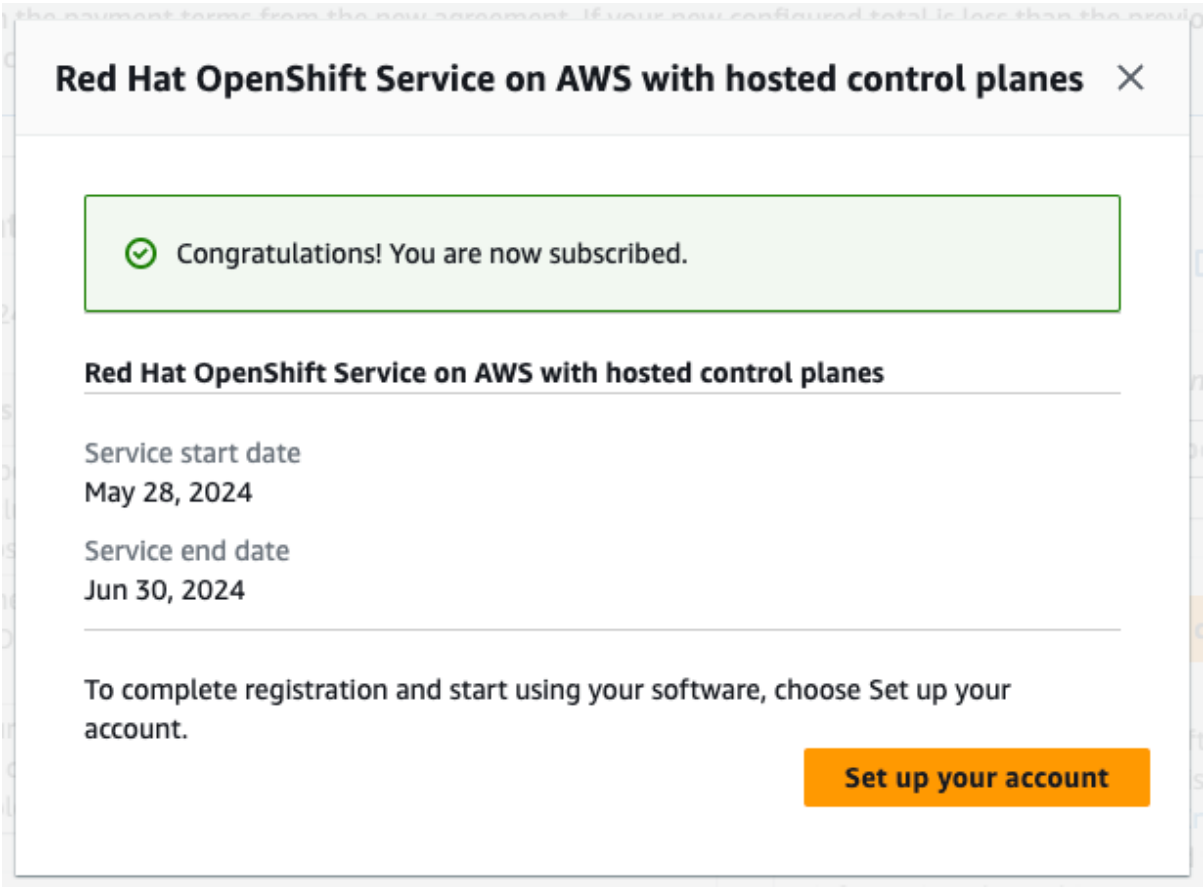
5. Cliquez sur Confirmer.

Figure 3.7. Fenêtre de confirmation d'acceptation de l'offre privée

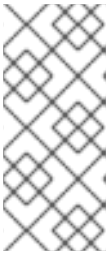


6. Lorsque la date de début du service d'offre privée acceptée est définie pour être immédiatement après l'acceptation de l'offre, cliquez sur le bouton Configurer votre compte dans la fenêtre modale de confirmation.

Figure 3.8. Confirmation de l'abonnement



7. Lorsque l'offre privée acceptée a une date de début ultérieure spécifiée, retournez à la page de l'offre privée après la date de début du service, puis cliquez sur le bouton Configuration de votre compte pour procéder à la liaison du compte Red Hat et AWS.



NOTE

En l'absence d'accord actif, le lien de compte décrit ci-dessous n'est pas déclenché, le processus de configuration du compte ne peut être effectué qu'après la date de début du service.

Ceux-ci sont toujours dans le fuseau horaire UTC.

3.2. LE PARTAGE D'UNE OFFRE PRIVÉE

1. En cliquant sur le bouton Configurer votre compte dans l'étape précédente, vous passez à l'étape de liaison du compte AWS et Red Hat. À ce moment-là, vous êtes déjà connecté avec le compte AWS qui a accepté l'offre. Lorsque vous n'êtes pas connecté avec un compte Red Hat, vous serez invité à le faire.
   Le ROSA avec droit HCP est partagé avec d'autres membres de l'équipe via votre compte d'organisation Red Hat. Les utilisateurs existants d'une même organisation Red Hat peuvent sélectionner le compte AWS de facturation qui a accepté l'offre privée en suivant les étapes décrites ci-dessus. Lorsque vous êtes connecté en tant qu'administrateur de l'organisation Red Hat, vous pouvez gérer les utilisateurs de votre organisation Red Hat et inviter ou créer de nouveaux utilisateurs.

NOTE

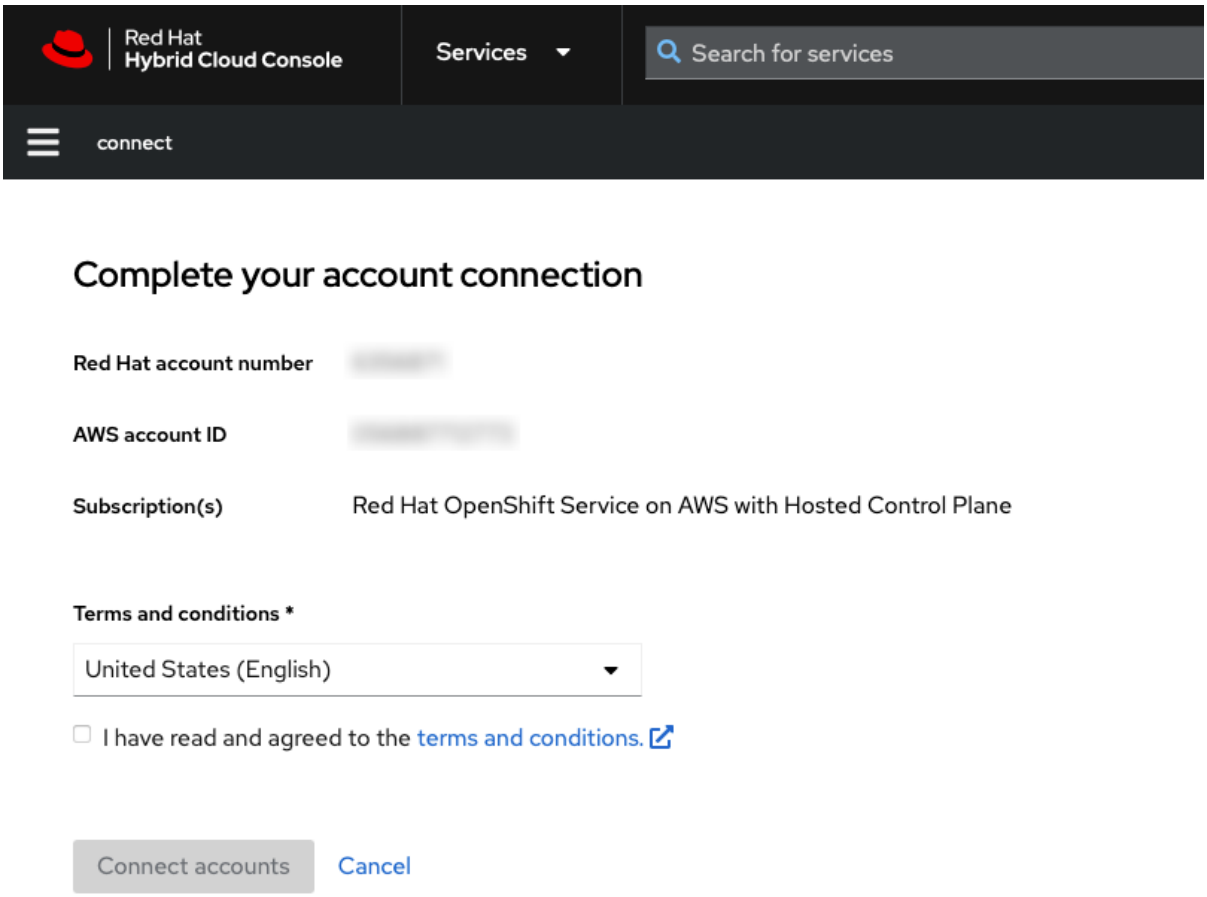
Le ROSA avec l'offre privée HCP ne peut pas être partagé avec des comptes liés AWS via AWS License Manager.

2. Ajoutez tous les utilisateurs que vous souhaitez déployer des clusters ROSA. Consultez cette FAQ de gestion de l'utilisateur pour plus de détails sur les tâches de gestion des utilisateurs du compte Red Hat.

3. Assurez-vous que le compte Red Hat déjà connecté inclut tous les utilisateurs qui sont censés être des déployeurs de cluster ROSA bénéficiant de l'offre privée acceptée.

4. Assurez-vous que le numéro de compte Red Hat et l'ID de compte AWS sont les comptes souhaités qui doivent être liés. Ce lien est unique et un compte Red Hat ne peut être connecté qu'à un seul compte AWS (facturation).

Figure 3.9. Connexion aux comptes AWS et Red Hat



5. Lorsque vous souhaitez lier le compte AWS à un autre compte Red Hat que ce qui est affiché sur cette page dans la Figure 9, déconnectez-vous de la console de cloud hybride Red Hat avant de connecter les comptes et répétez l'étape de configuration du compte en retournant à l'URL de l'offre privée qui est déjà acceptée.
   Le compte AWS ne peut être connecté qu'à un seul compte Red Hat. Lorsque les comptes Red Hat et AWS sont connectés, cela ne peut pas être modifié par l'utilisateur. En cas de changement, l'utilisateur doit créer un ticket d'assistance.

6. Acceptez les termes et conditions, puis cliquez sur Connecter les comptes.

3.3. LA SÉLECTION DU COMPTE DE FACTURATION AWS

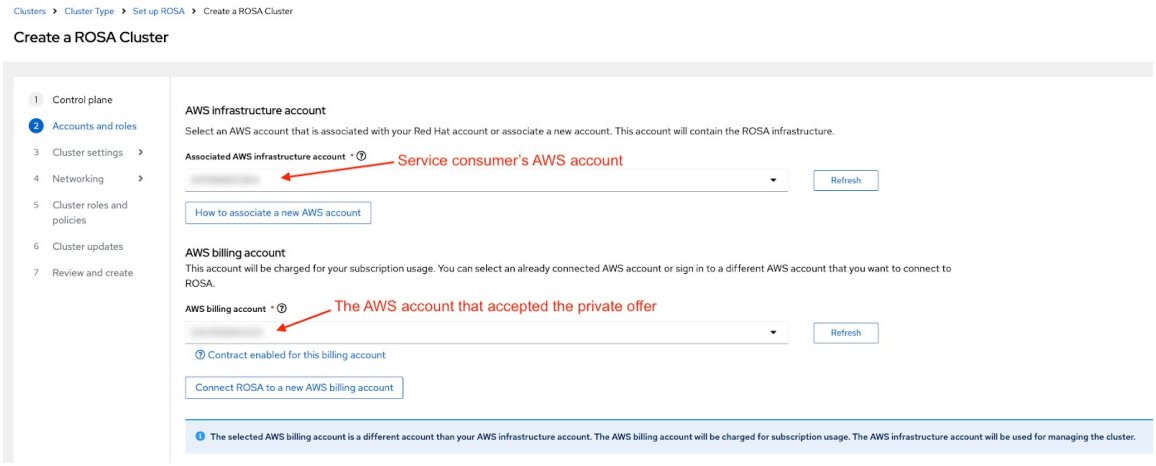
- Lors du déploiement de ROSA avec des clusters HCP, vérifiez que les utilisateurs finaux sélectionnent le compte de facturation AWS qui a accepté l'offre privée.

- Lors de l'utilisation de l'interface Web pour déployer ROSA avec HCP, le compte d'infrastructure AWS associé » est généralement défini sur l'ID de compte AWS utilisé par l'administrateur du cluster créé.

  - Cela peut être le même compte AWS que le compte AWS de facturation.

  - Les ressources AWS sont déployées dans ce compte et toutes les facturations associées à ces ressources sont traitées en conséquence.

Figure 3.10. Infrastructure et facturation de la sélection de compte AWS pendant ROSA avec le déploiement de cluster HCP



- La liste déroulante du compte de facturation AWS sur la capture d'écran ci-dessus doit être définie sur le compte AWS qui a accepté l'offre privée, à condition que le quota acheté soit utilisé par le cluster créé. Lorsque différents comptes AWS sont sélectionnés dans l'infrastructure et les « rôles » de facturation, la note d'information bleue visible à la figure 10 est affichée.

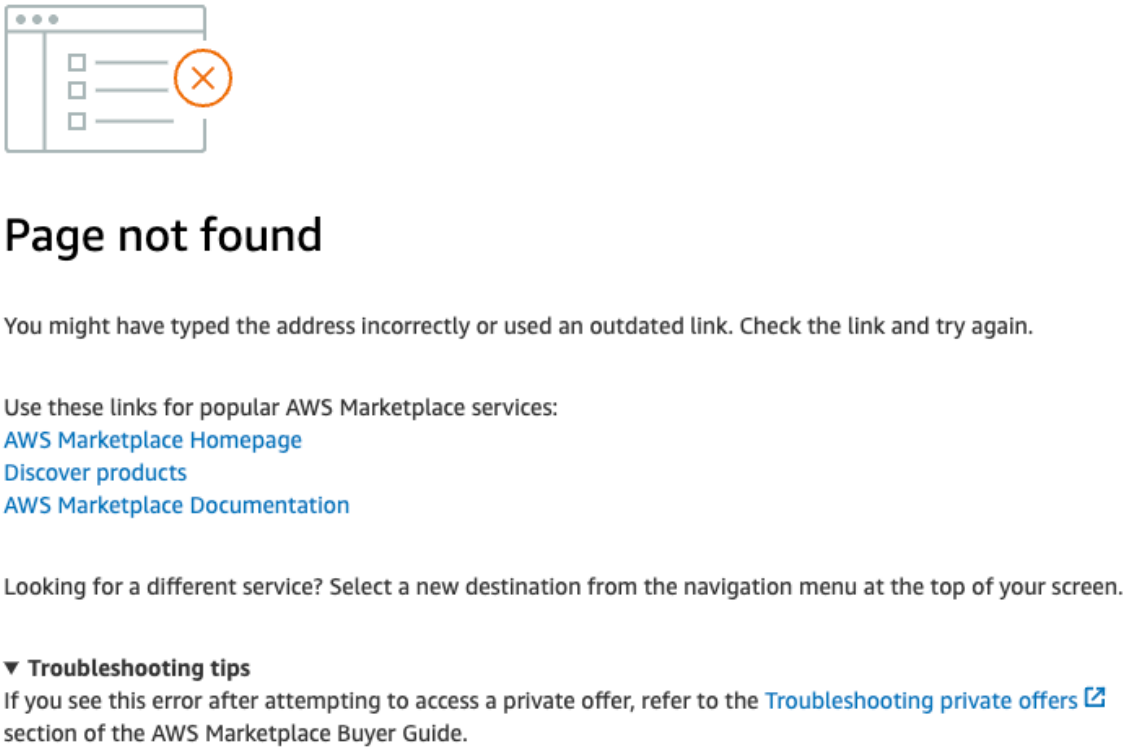
3.4. RÉSOLUTION DE PROBLÈMES

Les problèmes les plus fréquents associés à l'acceptation d'une offre privée et à la liaison de compte Red Hat.

3.4.1. Accéder à une offre privée à l'aide d'un autre compte AWS

- Lorsque vous essayez d'accéder à l'offre privée lorsque vous vous connectez sous l'ID de compte AWS qui n'est pas défini dans l'offre et que vous voyez le message affiché à la figure 11, vérifiez que vous êtes connecté comme compte de facturation AWS souhaité.

Figure 3.11. Erreur HTTP 404 lors de l'utilisation de l'URL de l'offre privée

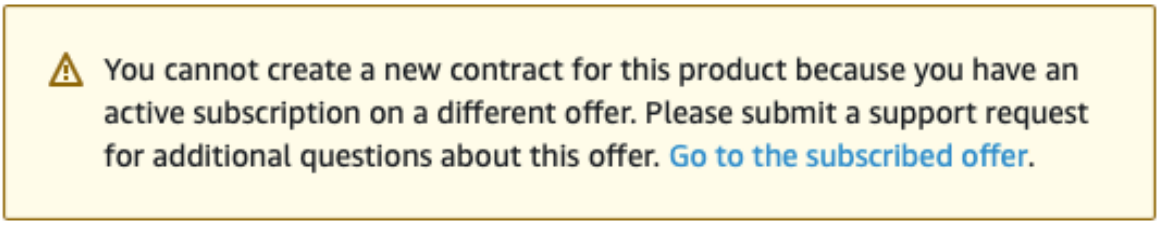


- Contactez le vendeur si vous avez besoin que l'offre privée soit étendue à un autre compte AWS.

3.4.2. L'offre privée ne peut pas être acceptée en raison d'un abonnement actif

- Lorsque vous essayez d'accéder à une offre privée créée pour la première fois ROSA avec activation HCP, alors que vous avez déjà ROSA avec HCP activé à l'aide d'une autre offre publique ou privée, et voir l'avis suivant, alors contactez le vendeur qui vous a fourni l'offre. Le vendeur peut vous fournir une nouvelle offre qui remplacera de manière transparente votre accord actuel, sans avoir besoin d'annuler votre abonnement précédent.

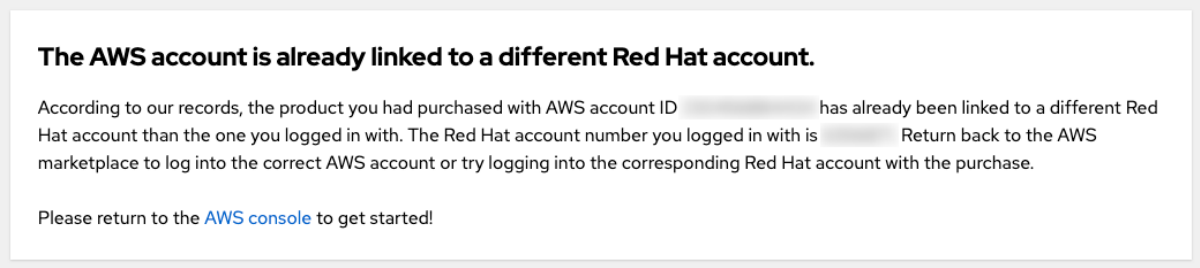
Figure 3.12. Abonnement existant empêchant l'acceptation de l'offre privée



3.4.3. Le compte AWS est déjà lié à un autre compte Red Hat

- Lorsque vous voyez le message d'erreur "AWS compte est déjà lié à un autre compte Red Hat" lorsque vous essayez de connecter le compte AWS qui a accepté l'offre privée avec un utilisateur Red Hat actuellement connecté, le compte AWS est déjà connecté à un autre utilisateur Red Hat.

Figure 3.13. Le compte AWS est déjà lié à un autre compte Red Hat



- Connectez-vous à l'aide d'un autre compte Red Hat ou d'un autre compte AWS.
    - Cependant, étant donné que ce guide concerne les offres privées, l'hypothèse est que vous êtes connecté au compte AWS qui a été spécifié comme acheteur et a déjà accepté l'offre privée, donc elle est destinée à être utilisée comme compte de facturation. La connexion comme un autre compte AWS n'est pas attendue après l'acceptation d'une offre privée.
- Il est toujours possible de vous connecter avec un autre utilisateur Red Hat qui est déjà connecté au compte AWS qui a accepté l'offre privée. D'autres utilisateurs de Red Hat appartenant à la même organisation Red Hat peuvent utiliser le compte AWS lié comme ROSA avec le compte de facturation AWS HCP lors de la création de clusters comme le montre la figure 10.
- Dans le cas où vous croyez que le lien de compte existant pourrait ne pas être correct, consultez la question « Les membres de mon équipe appartiennent à différentes organisations Red Hat » ci-dessous pour obtenir des conseils sur la façon dont vous pouvez procéder.

3.4.4. Les membres de mon équipe appartiennent à différentes organisations Red Hat

- Le compte AWS ne peut être connecté qu'à un seul compte Red Hat. Chaque utilisateur qui souhaite créer un cluster et bénéficier de l'offre privée accordée à ce compte AWS doit être dans le même compte Red Hat. Cela peut être réalisé en invitant l'utilisateur sur le même compte Red Hat et en créant un nouvel utilisateur Red Hat.

3.4.5. Compte de facturation AWS incorrect a été sélectionné lors de la création d'un cluster

- Lorsque l'utilisateur a sélectionné un compte de facturation AWS incorrect, le moyen le plus rapide de le corriger est de supprimer le cluster et d'en créer un nouveau, tout en sélectionnant le compte de facturation AWS correct.
- Dans le cas où il s'agit d'un cluster de production qui ne peut pas être facilement supprimé, veuillez contacter le support de Red Hat pour modifier le compte de facturation d'un cluster existant. Attendez-vous à un certain délai pour que cela soit résolu.

CHAPITRE 4. TUTORIEL : VÉRIFIER LES AUTORISATIONS POUR UN DÉPLOIEMENT ROSA STS

Afin de procéder au déploiement d'un cluster ROSA, un compte doit prendre en charge les rôles et autorisations requis. Les stratégies de contrôle des services AWS (SCP) ne peuvent pas bloquer les appels API effectués par les rôles d'installateur ou d'opérateur.

Les détails sur les ressources IAM nécessaires pour une installation STS de ROSA peuvent être trouvés ici: À propos des ressources de l'IAM Les détails sur les ressources IAM nécessaires pour une installation STS de ROSA peuvent être trouvés ici: À propos des ressources de l'IAM pour les clusters ROSA qui utilisent STS

Ce guide est validé pour ROSA v4.11.X.

4.1. CONDITIONS PRÉALABLES

- AWS CLI
- CLI ROSA v1.2.6
- JQ CLI
- Le rôle AWS avec les autorisations requises

4.2. LA VÉRIFICATION DES AUTORISATIONS ROSA

Afin de vérifier les autorisations requises pour ROSA, nous pouvons exécuter le script inclus dans la section suivante sans jamais créer de ressources AWS.

Le script utilise les commandes rosa, aws et jq CLI pour créer des fichiers dans le répertoire de travail qui seront utilisés pour vérifier les autorisations dans le compte connecté à la configuration AWS actuelle.

Le simulateur de stratégie AWS est utilisé pour vérifier les autorisations de chaque stratégie de rôle par rapport aux appels API extraits par jq; les résultats sont ensuite stockés dans un fichier texte joint avec .results.

Ce script est conçu pour vérifier les autorisations pour le compte courant et la région.

4.3. INSTRUCTIONS D'UTILISATION

1. Exécutez les commandes suivantes dans un terminal bash (l'option -p définit un préfixe pour les rôles):

```
$ mkdir scratch
$ cd scratch
$ cat << 'EOF' > verify-permissions.sh
#!/bin/bash
while getopts 'p:' OPTION; do
  case "$OPTION" in
    p)
      PREFIX="$OPTARG"
      ;;
    ?)
      echo "script usage: $(basename \$0) [-p PREFIX]" >&2
```

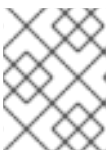
```
      exit 1
      ;;
  esac
done
shift "$(($OPTIND -1))"
rosa create account-roles --mode manual --prefix $PREFIX
INSTALLER_POLICY=$(cat sts_installer_permission_policy.json | jq )
CONTROL_PLANE_POLICY=$(cat sts_instance_controlplane_permission_policy.json | jq)
WORKER_POLICY=$(cat sts_instance_worker_permission_policy.json | jq)
SUPPORT_POLICY=$(cat sts_support_permission_policy.json | jq)
simulatePolicy () {
    outputFile="${2}.results"
    echo $2
    aws iam simulate-custom-policy --policy-input-list "$1" --action-names $(jq '.Statement |
map(select(.Effect == "Allow"))[].Action | if type == "string" then . else .[] end' "$2" -r) --output
text > $outputFile
}
simulatePolicy "$INSTALLER_POLICY" "sts_installer_permission_policy.json"
simulatePolicy "$CONTROL_PLANE_POLICY"
"sts_instance_controlplane_permission_policy.json"
simulatePolicy "$WORKER_POLICY" "sts_instance_worker_permission_policy.json"
simulatePolicy "$SUPPORT_POLICY" "sts_support_permission_policy.json"
EOF
$ chmod +x verify-permissions.sh
$ ./verify-permissions.sh -p SimPolTest
```

2. Après la fin du script, examinez chaque fichier de résultats pour s'assurer qu'aucun des appels API requis n'est bloqué:

```
$ for file in $(ls *.results); do echo $file; cat $file; done
```

La sortie ressemblera à ce qui suit:

```
sts_installer_permission_policy.json.results
EVALUATIONRESULTS       autoscaling:DescribeAutoScalingGroups   allowed *
MATCHEDSTATEMENTS       PolicyInputList.1       IAM Policy
ENDPOSITION     6       195
STARTPOSITION   17      3
EVALUATIONRESULTS       ec2:AllocateAddress     allowed *
MATCHEDSTATEMENTS       PolicyInputList.1       IAM Policy
ENDPOSITION     6       195
STARTPOSITION   17      3
EVALUATIONRESULTS       ec2:AssociateAddress    allowed *
MATCHEDSTATEMENTS       PolicyInputList.1       IAM Policy
...
```



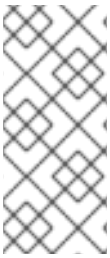
NOTE

En cas de blocage, examinez l'erreur fournie par AWS et consultez votre Administrateur pour déterminer si les SCP bloquent les appels API requis.

CHAPITRE 5. TUTORIEL: DÉPLOIEMENT DE ROSA AVEC UN RÉSOLVEUR DNS PERSONNALISÉ

Le jeu d'options DHCP personnalisé vous permet de personnaliser votre VPC avec votre propre serveur DNS, votre nom de domaine et plus encore. Les clusters Red Hat OpenShift Service sur AWS (ROSA) prennent en charge en utilisant des ensembles d'options DHCP personnalisés. Les clusters ROSA nécessitent par défaut de définir l'option "serveurs de noms de domaine" sur AmazonProvidedDNS pour assurer la création et l'exploitation réussies des clusters. Les clients qui souhaitent utiliser des serveurs DNS personnalisés pour la résolution DNS doivent faire une configuration supplémentaire pour assurer le succès de la création et du fonctionnement du cluster ROSA.

Dans ce tutoriel, nous configurerons notre serveur DNS pour transférer les recherches DNS pour des zones DNS spécifiques (détaillées plus loin ci-dessous) vers un Résolvateur entrant Amazon Route 53.



NOTE

Ce tutoriel utilise le serveur DNS BIND open-source (nommé) pour démontrer la configuration nécessaire pour transférer les recherches DNS vers un Résolvateur entrant Amazon Route 53 situé dans le VPC dans lequel vous prévoyez de déployer un cluster ROSA. Consultez la documentation de votre serveur DNS préféré pour savoir comment configurer le transfert de zone.

5.1. CONDITIONS PRÉALABLES

- CLI ROSA (rosa)
- AWS CLI (aws)
- Création manuelle d'AWS VPC
- Configuration d'une option DHCP configurée pour pointer vers un serveur DNS personnalisé et définie par défaut pour votre VPC

5.2. CONFIGURATION DE VOTRE ENVIRONNEMENT

1. Configurez les variables d'environnement suivantes:

```
$ export VPC_ID=<vpc_ID> 1
$ export REGION=<region> 2
$ export VPC_CIDR=<vpc_CIDR> 3
```

1. <vpc_ID> par l'ID du VPC dans lequel vous souhaitez installer votre cluster.
2. Remplacez <région> par la région AWS dans laquelle vous souhaitez installer votre cluster.
3. <vpc_CIDR> par la gamme CIDR de votre VPC.

2. Assurez-vous que tous les champs sortent correctement avant de passer à la section suivante:

```
$ echo "VPC ID: ${VPC_ID}, VPC CIDR Range: ${VPC_CIDR}, Region: ${REGION}"
```

5.3. CRÉER UNE SOLUTION AMAZON ROUTE 53 INBOUND RESOLVER

Dans le VPC, utilisez la procédure suivante pour déployer un Inbound Resolver Amazon Route 53 dans lequel nous prévoyons de déployer le cluster.

AVERTISSEMENT

Dans cet exemple, nous déployons Amazon Route 53 Inbound Resolver dans le même VPC que le cluster utilisera. Lorsque vous souhaitez le déployer dans un VPC séparé, vous devez associer manuellement la zone(s) hébergée(s) privée(s) décrite ci-dessous une fois la création de cluster démarrée. Il est impossible d'associer la zone avant le début du processus de création de clusters. Le défaut d'associer la zone privée hébergée pendant le processus de création de clusters entraînera des échecs de création de clusters.

1. Créer un groupe de sécurité et permettre l'accès aux ports 53/tcp et 53/udp à partir du VPC:

```
$ SG_ID=$(aws ec2 create-security-group --group-name rosa-inbound-resolver --description
"Security group for ROSA inbound resolver" --vpc-id ${VPC_ID} --region ${REGION} --
output text)
$ aws ec2 authorize-security-group-ingress --group-id ${SG_ID} --protocol tcp --port 53 --cidr
${VPC_CIDR} --region ${REGION}
$ aws ec2 authorize-security-group-ingress --group-id ${SG_ID} --protocol udp --port 53 --
cidr ${VPC_CIDR} --region ${REGION}
```

2. Créez une solution Amazon Route 53 Inbound Resolver dans votre VPC:

```
$ RESOLVER_ID=$(aws route53resolver create-resolver-endpoint \
  --name rosa-inbound-resolver \
  --creator-request-id rosa-$(date '+%Y-%m-%d') \
  --security-group-ids ${SG_ID} \
  --direction INBOUND \
  --ip-addresses $(aws ec2 describe-subnets --filter Name=vpc-id,Values=${VPC_ID} --
region ${REGION} | jq -jr '.Subnets | map("SubnetId=\(.SubnetId) ") | .[]') \
  --region ${REGION} \
  --output text \
  --query 'ResolverEndpoint.Id')
```

NOTE

La commande ci-dessus fixe les points de terminaison Amazon Route 53 Inbound Resolver à tous les sous-réseaux du VPC fourni à l'aide d'adresses IP allouées dynamiquement. Lorsque vous préférez spécifier manuellement les sous-réseaux et/ou les adresses IP, exécutez plutôt la commande suivante:

```
$ RESOLVER_ID=$(aws route53resolver create-resolver-endpoint \
  --name rosa-inbound-resolver \
  --creator-request-id rosa-$(date '+%Y-%m-%d') \
  --security-group-ids ${SG_ID} \
  --direction INBOUND \
  --ip-addresses SubnetId=<subnet_ID>,Ip=<endpoint_IP> SubnetId=
<subnet_ID>,Ip=<endpoint_IP> \ 1
  --region ${REGION} \
  --output text \
  --query 'ResolverEndpoint.Id')
```

1 <subnet_ID> par les ID de sous-réseau et <endpoint_IP> par les adresses IP statiques auxquelles vous souhaitez ajouter des points de terminaison de résolution entrant.

3. Accédez aux adresses IP de vos points de terminaison de résolution entrants à configurer dans la configuration de votre serveur DNS:

```
$ aws route53resolver list-resolver-endpoint-ip-addresses \
  --resolver-endpoint-id ${RESOLVER_ID} \
  --region=${REGION} \
  --query 'IpAddresses[*].Ip'
```

Exemple de sortie

```
[
   "10.0.45.253",
   "10.0.23.131",
   "10.0.148.159"
]
```

5.4. CONFIGUREZ VOTRE SERVEUR DNS

La procédure suivante permet de configurer votre serveur DNS afin de transférer les zones hébergées privées nécessaires à votre solution Amazon Route 53 Inbound Resolver.

Les clusters ROSA Classic vous obligent à configurer le transfert DNS pour une zone privée hébergée:

- **<domain-prefix>.<unique-ID>.p1.openshiftapps.com**

Cette Amazon Route 53 zones hébergées privées est créée lors de la création de clusters. Le préfixe de domaine est une valeur spécifiée par le client, mais l'ID unique est généré au hasard lors de la création de clusters et ne peut pas être présélectionné. En tant que tel, vous devez attendre que le processus de création de cluster commence avant de configurer le transfert pour la zone hébergée privée p1.openshiftapps.com.

1. Créez votre cluster.

2. Lorsque votre cluster a commencé le processus de création, localisez la zone hébergée privée nouvellement créée:

```
$ aws route53 list-hosted-zones-by-vpc \
  --vpc-id ${VPC_ID} \
  --vpc-region ${REGION} \
  --query 'HostedZoneSummaries[*].Name' \
  --output table
```

Exemple de sortie

```
----------------------------------------------
|          ListHostedZonesByVPC              |
+--------------------------------------------+
|  domain-prefix.agls.p3.openshiftapps.com.  |
+--------------------------------------------+
```

NOTE

Le processus de création de clusters peut prendre quelques minutes pour créer les zones privées hébergées sur la Route 53. Lorsque vous ne voyez pas un domaine p1.openshiftapps.com, attendez quelques minutes et exécutez à nouveau la commande.

3. Lorsque vous connaissez l'identifiant unique du domaine du cluster, configurez votre serveur DNS pour transférer toutes les requêtes DNS pour <domain-prefix>.<unique-ID>.p1.openshiftapps.com vers vos points de terminaison Amazon Route 53 Inbound Resolver. Dans le cas des serveurs DNS BIND, modifiez votre fichier /etc/named.conf dans votre éditeur de texte préféré et ajoutez une nouvelle zone à l'aide de l'exemple ci-dessous:

Exemple :

```
zone "<domain-prefix>.<unique-ID>.p1.openshiftapps.com" { 1
  type forward;
  forward only;
  forwarders { 2
    10.0.45.253;
    10.0.23.131;
    10.0.148.159;
  };
};
```

1 <domain-prefix> par votre préfixe de domaine cluster et <unique-ID> par votre identifiant unique recueilli ci-dessus.

2 Le remplacement par les adresses IP de vos endpoints de résolution entrants recueillis ci-dessus, en veillant à ce que, après chaque adresse IP, il y ait un ;.

CHAPITRE 6. TUTORIEL : UTILISATION D'AWS WAF ET D'AMAZON CLOUDFRONT POUR PROTÉGER LES CHARGES DE TRAVAIL ROSA

AWS WAF est un pare-feu d'application Web qui vous permet de surveiller les requêtes HTTP et HTTPS qui sont transmises à vos ressources d'application Web protégées.

Amazon CloudFront vous permet d'ajouter un pare-feu d'application Web (WAF) à vos charges de travail Red Hat OpenShift Service sur AWS (ROSA). L'utilisation d'une solution externe protège les ressources ROSA contre le déni de service dû à la gestion du WAF.

6.1. CONDITIONS PRÉALABLES

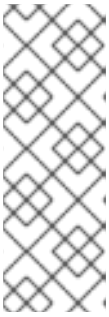
- A ROSA (HCP ou Classic) cluster.
- Accès à l'OpenShift CLI (oc).
- Accès à AWS CLI (aws).

6.1.1. Configuration de l'environnement

- Élaborer les variables de l'environnement:

```
$ export DOMAIN=apps.example.com 1
$ export AWS_PAGER=""
$ export CLUSTER_NAME=$(oc get infrastructure cluster -o=jsonpath="
{.status.infrastructureName}"  | sed 's/-[a-z0-9]\{5\}$//')
$ export REGION=$(oc get infrastructure cluster -o=jsonpath="
{.status.platformStatus.aws.region}")
$ export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
$ export SCRATCH="/tmp/${CLUSTER}/cloudfront-waf"
$ mkdir -p ${SCRATCH}
$ echo "Cluster: ${CLUSTER}, Region: ${REGION}, AWS Account ID:
${AWS_ACCOUNT_ID}"
```

1 Le remplacement par le domaine personnalisé que vous souhaitez utiliser pour IngressController.



NOTE

La sortie "Cluster" de la commande précédente peut être le nom de votre cluster, l'ID interne de votre cluster ou le préfixe de domaine du cluster. Lorsque vous préférez utiliser un autre identifiant, vous pouvez définir manuellement cette valeur en exécutant la commande suivante:

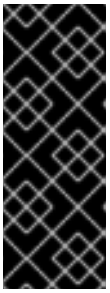
```
$ export CLUSTER=my-custom-value
```

6.2. CONFIGURATION DU CONTRÔLEUR D'ENTRÉE SECONDAIRE

Il est nécessaire de configurer un contrôleur d'entrée secondaire pour segmenter votre trafic externe protégé par WAF de votre contrôleur d'entrée standard (et par défaut).

Conditions préalables

- Certificat SAN ou wildcard de confiance pour votre domaine personnalisé, tel que CN=*.apps.example.com



IMPORTANT

Amazon CloudFront utilise HTTPS pour communiquer avec le contrôleur secondaire d'entrée de votre cluster. Comme expliqué dans la documentation Amazon CloudFront, vous ne pouvez pas utiliser un certificat autosigné pour la communication HTTPS entre CloudFront et votre cluster. Amazon CloudFront vérifie que le certificat a été délivré par une autorité de certification de confiance.

Procédure

1. Créez un nouveau secret TLS à partir d'une clé privée et d'un certificat public, où fullchain.pem est votre chaîne de certificats wildcard complète (y compris les intermédiaires) et privkey.pem est la clé privée de votre certificat wildcard.

   Exemple :

   ```
   $ oc -n openshift-ingress create secret tls waf-tls --cert=fullchain.pem --key=privkey.pem
   ```

2. Créer une nouvelle ressource IngressController:

   Exemple waf-ingress-controller.yaml

   ```
   apiVersion: operator.openshift.io/v1
   kind: IngressController
   metadata:
     name: cloudfront-waf
     namespace: openshift-ingress-operator
   spec:
     domain: apps.example.com 1
     defaultCertificate:
       name: waf-tls
     endpointPublishingStrategy:
       loadBalancer:
         dnsManagementPolicy: Unmanaged
         providerParameters:
           aws:
             type: NLB
           type: AWS
         scope: External
       type: LoadBalancerService
     routeSelector: 2
       matchLabels:
        route: waf
   ```

1 Le remplacement par le domaine personnalisé que vous souhaitez utiliser pour IngressController.

2

Filtre l'ensemble d'itinéraires desservis par le contrôleur Ingress. Dans ce tutoriel, nous utiliserons le sélecteur de route waf, mais si aucune valeur n'était fournie, aucun filtrage ne

3. Appliquer le logiciel IngressController:

Exemple :

```
$ oc apply -f waf-ingress-controller.yaml
```

4. Assurez-vous que votre IngressController a réussi à créer un équilibreur de charge externe:

```
$ oc -n openshift-ingress get service/router-cloudfront-waf
```

Exemple de sortie

```
NAME                 TYPE          CLUSTER-IP      EXTERNAL-IP
PORT(S)                   AGE
router-cloudfront-waf   LoadBalancer   172.30.16.141
a68a838a7f26440bf8647809b61c4bc8-4225395f488830bd.elb.us-east-1.amazonaws.com
80:30606/TCP,443:31065/TCP   2m19s
```

6.2.1. Configurer l'AWS WAF

Le service AWS WAF est un pare-feu d'application Web qui vous permet de surveiller, de protéger et de contrôler les requêtes HTTP et HTTPS qui sont transmises à vos ressources d'application Web protégées, comme ROSA.

1. Créer un fichier de règles AWS WAF pour s'appliquer à notre ACL web:

```
$ cat << EOF > ${SCRATCH}/waf-rules.json
[
  {
    "Name": "AWS-AWSManagedRulesCommonRuleSet",
    "Priority": 0,
    "Statement": {
      "ManagedRuleGroupStatement": {
        "VendorName": "AWS",
        "Name": "AWSManagedRulesCommonRuleSet"
      }
    },
    "OverrideAction": {
      "None": {}
    },
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "AWS-AWSManagedRulesCommonRuleSet"
    }
  },
  {
    "Name": "AWS-AWSManagedRulesSQLiRuleSet",
    "Priority": 1,
    "Statement": {
```

```
        "ManagedRuleGroupStatement": {
          "VendorName": "AWS",
          "Name": "AWSManagedRulesSQLiRuleSet"
        }
      },
      "OverrideAction": {
        "None": {}
      },
      "VisibilityConfig": {
        "SampledRequestsEnabled": true,
        "CloudWatchMetricsEnabled": true,
        "MetricName": "AWS-AWSManagedRulesSQLiRuleSet"
      }
    }
]
EOF
```

Cela activera les jeux de règles gérés Core (Common) et SQL AWS.

2. Créer un AWS WAF Web ACL en utilisant les règles que nous avons spécifiées ci-dessus:

```
$ WAF_WACL=$(aws wafv2 create-web-acl \
  --name cloudfront-waf \
  --region ${REGION} \
  --default-action Allow={} \
  --scope CLOUDFRONT \
  --visibility-config
SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=${CLUSTER}-
waf-metrics \
  --rules file://${SCRATCH}/waf-rules.json \
  --query 'Summary.Name' \
  --output text)
```

6.3. CONFIGURER AMAZON CLOUDFRONT

1. Le nom d'hôte NLB du contrôleur d'entrée personnalisé nouvellement créé:

```
$ NLB=$(oc -n openshift-ingress get service router-cloudfront-waf \
  -o jsonpath='{.status.loadBalancer.ingress[0].hostname}')
```

2. Importez votre certificat dans Amazon Certificate Manager, où cert.pem est votre certificat wildcard, fullchain.pem est la chaîne de votre certificat wildcard et privkey.pem est la clé privée de votre certificat wildcard.

NOTE

Indépendamment de la région dans laquelle votre cluster est déployé, vous devez importer ce certificat pour nous-Est-1 car Amazon CloudFront est un service AWS mondial.

Exemple :

```
$ aws acm import-certificate --certificate file://cert.pem \
```

```
--certificate-chain file://fullchain.pem \
--private-key file://privkey.pem \
--region us-east-1
```

3. Connectez-vous à la console AWS pour créer une distribution CloudFront.
4. Configurez la distribution CloudFront en utilisant les informations suivantes:

NOTE

Dans le cas où une option n'est pas spécifiée dans le tableau ci-dessous, laissez-les par défaut (qui peut être vide).

L'option	La valeur
Domaine d'origine	Sortie de la commande précédente [1]
Le nom	ingère Rosa-waf [2]
La politique du protocole d'affichage	Envoyer HTTP vers HTTPS
Les méthodes HTTP autorisées	GET, HEAD, OPTIONS, PUT, POSTER, PATCH, SUPPRIMER
Cache politique	CachingDisabled
Demande d'origine politique	AllViewer
Application Web Firewall (WAF)	Activer les protections de sécurité
Configuration WAF existante	C'est vrai
Choisissez un ACL web	**CloudFront-waf**
Autre nom de domaine (CNAME)	*.apps.example.com [3]
Certificat SSL personnalisé	Choisissez le certificat que vous avez importé à partir de l'étape ci-dessus [4]

1. Exécutez l'écho ${NLB} pour obtenir le domaine d'origine.
2. Lorsque vous avez plusieurs clusters, assurez-vous que le nom d'origine est unique.
3. Cela devrait correspondre au domaine wildcard que vous avez utilisé pour créer le contrôleur d'entrée personnalisé.
4. Cela devrait correspondre au nom de domaine alternatif entré ci-dessus.

5. Découvrez le point de terminaison Amazon CloudFront Distribution:

```
$ aws cloudfront list-distributions --query "DistributionList.Items[?Origins.Items[?
DomainName=='${NLB}']].DomainName" --output text
```

6. Actualisez le DNS de votre domaine wildcard personnalisé avec un CNAME vers le point de terminaison Amazon CloudFront Distribution à partir de l'étape ci-dessus.

   Exemple :

```
*.apps.example.com CNAME d1b2c3d4e5f6g7.cloudfront.net
```

6.4. DÉPLOYER UN EXEMPLE D'APPLICATION

1. Créez un nouveau projet pour votre exemple d'application en exécutant la commande suivante:

```
$ oc new-project hello-world
```

2. Déployez une application hello world:

```
$ oc -n hello-world new-app --image=docker.io/openshift/hello-openshift
```

3. Créer un itinéraire pour l'application spécifiant votre nom de domaine personnalisé:

   Exemple :

```
$ oc -n hello-world create route edge --service=hello-openshift hello-openshift-tls \
--hostname hello-openshift.${DOMAIN}
```

4. Étiqueter la route pour l'admettre à votre contrôleur d'entrée personnalisé:

```
$ oc -n hello-world label route.route.openshift.io/hello-openshift-tls route=waf
```

6.5. TESTER LE WAF

1. Essayez que l'application est accessible derrière Amazon CloudFront:

   Exemple :

```
$ curl "https://hello-openshift.${DOMAIN}"
```

   Exemple de sortie

```
Hello OpenShift!
```

2. Essayez que la WAF nie une mauvaise demande:

   Exemple :

```
$ curl -X POST "https://hello-openshift.${DOMAIN}" \
  -F "user='<script><alert>Hello></alert></script>'"
```

Exemple de sortie

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<HTML><HEAD><META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-
8859-1">
<TITLE>ERROR: The request could not be satisfied</TITLE>
</HEAD><BODY>
<H1>403 ERROR</H1>
<H2>The request could not be satisfied.</H2>
<HR noshade size="1px">
Request blocked.
We can't connect to the server for this app or website at this time. There might be too much
traffic or a configuration error. Try again later, or contact the app or website owner.
<BR clear="all">
If you provide content to customers through CloudFront, you can find steps to troubleshoot
and help prevent this error by reviewing the CloudFront documentation.
<BR clear="all">
<HR noshade size="1px">
<PRE>
Generated by cloudfront (CloudFront)
Request ID: nFk9q2yB8jddI6FZOTjdliexzx-FwZtr8xUQUNT75HThPlrALDxbag==
</PRE>
<ADDRESS>
</ADDRESS>
</BODY></HTML>
```

Le résultat attendu est un 403 ERROR, ce qui signifie que l'AWS WAF protège votre application.

6.6. RESSOURCES SUPPLÉMENTAIRES

- Ajout d'une sécurité supplémentaire avec AWS WAF, CloudFront et ROSA | Amazon Web Services sur YouTube

CHAPITRE 7. TUTORIEL : UTILISATION D'AWS WAF ET D'AWS ALB POUR PROTÉGER LES CHARGES DE TRAVAIL ROSA

AWS WAF est un pare-feu d'application Web qui vous permet de surveiller les requêtes HTTP et HTTPS qui sont transmises à vos ressources d'application Web protégées.

Il est possible d'utiliser un balanceur de charge d'application AWS (ALB) pour ajouter un pare-feu d'application Web (WAF) à votre Red Hat OpenShift Service sur AWS (ROSA). L'utilisation d'une solution externe protège les ressources ROSA contre le déni de service dû à la gestion du WAF.

IMPORTANT

Il est recommandé d'utiliser la méthode CloudFront plus flexible, sauf si vous devez absolument utiliser une solution basée sur ALB.

7.1. CONDITIONS PRÉALABLES

- Cluster de zones de disponibilité multiples (AZ) ROSA (HCP ou Classic).

NOTE

Les ALB AWS nécessitent au moins deux sous-réseaux publics à travers les AZ, selon la documentation AWS. C'est pour cette raison que seuls plusieurs clusters AZ ROSA peuvent être utilisés avec des ALB.

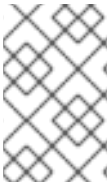
- Accès à l'OpenShift CLI (oc).
- Accès à AWS CLI (aws).

7.1.1. Configuration de l'environnement

- Élaborer les variables de l'environnement:

```
$ export AWS_PAGER=""
$ export CLUSTER=$(oc get infrastructure cluster -o=jsonpath="
{.status.infrastructureName}")
$ export REGION=$(oc get infrastructure cluster -o=jsonpath="
{.status.platformStatus.aws.region}")
$ export OIDC_ENDPOINT=$(oc get authentication.config.openshift.io cluster -o
jsonpath='{.spec.serviceAccountIssuer}' | sed  's|^https://||')
$ export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
$ export SCRATCH="/tmp/${CLUSTER}/alb-waf"
$ mkdir -p ${SCRATCH}
$ echo "Cluster: $(echo ${CLUSTER} | sed 's/-[a-z0-9]\{5\}$//'), Region: ${REGION}, OIDC
Endpoint: ${OIDC_ENDPOINT}, AWS Account ID: ${AWS_ACCOUNT_ID}"
```

7.1.2. AWS VPC et sous-réseaux



NOTE

Cette section ne s'applique qu'aux clusters qui ont été déployés dans des VPC existants. Dans le cas où vous n'avez pas déployé votre cluster dans un VPC existant, sautez cette section et passez à la section d'installation ci-dessous.

1. Définissez les variables ci-dessous aux valeurs appropriées pour votre déploiement ROSA:

```
$ export VPC_ID=<vpc-id> 1
$ export PUBLIC_SUBNET_IDS=(<space-separated-list-of-ids>) 2
$ export PRIVATE_SUBNET_IDS=(<space-separated-list-of-ids>) 3
```

1 Le remplacement par l'ID VPC du cluster, par exemple: exporte VPC_ID=vpc-04c429b7dbc4680ba.

2 Le remplacer par une liste séparée de l'espace des ID de sous-réseau privés du cluster, en veillant à préserver le (). Exportation PUBLIC_SUBNET_IDS=(subnet-056fd6861ad332ba2 sous-net-08ce3b4ec753fe74c sous-net-071aa2828664972f).

3 Le remplacer par une liste séparée de l'espace des ID de sous-réseau privés du cluster, en veillant à préserver le (). Exportation PRIVATE_SUBNET_IDS=(subnet-0b933d72a8d72c36a sousnet-0817eb72070f1d3c2 sousnet-0806e64159b66665a).

2. Ajoutez une balise au VPC de votre cluster avec l'identifiant du cluster:

```
$ aws ec2 create-tags --resources ${VPC_ID} \
  --tags Key=kubernetes.io/cluster/${CLUSTER},Value=shared --region ${REGION}
```

3. Ajoutez une étiquette à vos sous-réseaux publics:

```
$ aws ec2 create-tags \
  --resources ${PUBLIC_SUBNET_IDS} \
  --tags Key=kubernetes.io/role/elb,Value='1' \
       Key=kubernetes.io/cluster/${CLUSTER},Value=shared \
  --region ${REGION}
```

4. Ajouter une balise à vos sous-réseaux privés:

```
$ aws ec2 create-tags \
  --resources ${PRIVATE_SUBNET_IDS} \
  --tags Key=kubernetes.io/role/internal-elb,Value='1' \
       Key=kubernetes.io/cluster/${CLUSTER},Value=shared \
  --region ${REGION}
```

7.2. DÉPLOYER L'OPÉRATEUR AWS LOAD BALANCER

L'opérateur AWS Load Balancer est utilisé pour installer, gérer et configurer une instance de contrôleur aws-charge-balancer dans un cluster ROSA. « pour déployer des ALB dans ROSA, nous devons d'abord déployer l'opérateur AWS Load Balancer.

1. Créez un nouveau projet pour déployer l'opérateur AWS Load Balancer en exécutant la commande suivante:

```
$ oc new-project aws-load-balancer-operator
```

2. Créez une stratégie AWS IAM pour le contrôleur AWS Load Balancer si l'on n'existe pas déjà en exécutant la commande suivante:

NOTE

La politique provient de la politique AWS Load Balancer Controller en amont. Ceci est requis par l'opérateur pour fonctionner.

```
$ POLICY_ARN=$(aws iam list-policies --query \
    "Policies[?PolicyName=='aws-load-balancer-operator-policy'].{ARN:Arn}" \
    --output text)
```

```
$ if [[ -z "${POLICY_ARN}" ]]; then
   wget -O "${SCRATCH}/load-balancer-operator-policy.json" \
     https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/main/docs/install/iam_policy.json
    POLICY_ARN=$(aws --region "$REGION" --query Policy.Arn \
    --output text iam create-policy \
    --policy-name aws-load-balancer-operator-policy \
    --policy-document "file://${SCRATCH}/load-balancer-operator-policy.json")
fi
```

3. Créez une politique de confiance AWS IAM pour AWS Load Balancer Operator:

```
$ cat <<EOF > "${SCRATCH}/trust-policy.json"
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Condition": {
   "StringEquals" : {
    "${OIDC_ENDPOINT}:sub": ["system:serviceaccount:aws-load-balancer-operator:aws-
load-balancer-operator-controller-manager", "system:serviceaccount:aws-load-balancer-
operator:aws-load-balancer-controller-cluster"]
   }
 },
 "Principal": {
   "Federated": "arn:aws:iam::$AWS_ACCOUNT_ID:oidc-provider/${OIDC_ENDPOINT}"
 },
 "Action": "sts:AssumeRoleWithWebIdentity"
 }
 ]
}
EOF
```

4. Créer un rôle AWS IAM pour l'opérateur AWS Load Balancer:

```
$ ROLE_ARN=$(aws iam create-role --role-name "${CLUSTER}-alb-operator" \
   --assume-role-policy-document "file://${SCRATCH}/trust-policy.json" \
   --query Role.Arn --output text)
```

5. Joignez la politique AWS Load Balancer Operator au rôle IAM que nous avons créé précédemment en exécutant la commande suivante:

```
$ aws iam attach-role-policy --role-name "${CLUSTER}-alb-operator" \
    --policy-arn ${POLICY_ARN}
```

6. Créez un secret pour que l'opérateur AWS Load Balancer assume notre nouveau rôle AWS IAM:

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: aws-load-balancer-operator
  namespace: aws-load-balancer-operator
stringData:
  credentials: |
    [default]
    role_arn = ${ROLE_ARN}
    web_identity_token_file = /var/run/secrets/openshift/serviceaccount/token
EOF
```

7. Installez l'opérateur AWS Load Balancer:

```
$ cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: aws-load-balancer-operator
  namespace: aws-load-balancer-operator
spec:
  upgradeStrategy: Default
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: aws-load-balancer-operator
  namespace: aws-load-balancer-operator
spec:
  channel: stable-v1.0
  installPlanApproval: Automatic
  name: aws-load-balancer-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
  startingCSV: aws-load-balancer-operator.v1.0.0
EOF
```

8. Déployez une instance du contrôleur AWS Load Balancer à l'aide de l'opérateur:

NOTE

Lorsque vous obtenez une erreur ici, attendez une minute et essayez à nouveau, cela signifie que l'opérateur n'a pas encore terminé l'installation.

```
$ cat << EOF | oc apply -f -
```

```
apiVersion: networking.olm.openshift.io/v1
kind: AWSLoadBalancerController
metadata:
  name: cluster
spec:
  credentials:
    name: aws-load-balancer-operator
  enabledAddons:
    - AWSWAFv2
EOF
```

9. Assurez-vous que les pods de l'opérateur et du contrôleur sont en cours d'exécution:

```
$ oc -n aws-load-balancer-operator get pods
```

Il faut voir ce qui suit, sinon attendre un moment et réessayer:

```
NAME                                                READY   STATUS    RESTARTS   AGE
aws-load-balancer-controller-cluster-6ddf658785-pdp5d           1/1     Running   0          99s
aws-load-balancer-operator-controller-manager-577d9ffcb9-w6zqn   2/2     Running   0
2m4s
```

7.3. DÉPLOYER UN EXEMPLE D'APPLICATION

1. Créez un nouveau projet pour notre exemple d'application:

```
$ oc new-project hello-world
```

2. Déployez une application hello world:

```
$ oc new-app -n hello-world --image=docker.io/openshift/hello-openshift
```

3. Convertissez la ressource de service pré-créée en un type de service NodePort:

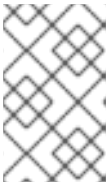
```
$ oc -n hello-world patch service hello-openshift -p '{"spec":{"type":"NodePort"}}'
```

4. Déployer un AWS ALB à l'aide de l'opérateur AWS Load Balancer:

```
$ cat << EOF | oc apply -f -
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: hello-openshift-alb
  namespace: hello-world
  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
spec:
  ingressClassName: alb
  rules:
    - http:
        paths:
          - path: /
            pathType: Exact
```

```
          backend:
            service:
              name: hello-openshift
              port:
                number: 8080
EOF
```

5. Curl le point de terminaison AWS ALB Ingress pour vérifier l'application hello world est accessible:



NOTE

AWS ALB provisioning prend quelques minutes. Lorsque vous recevez une erreur qui dit curl: (6) Ne peut pas résoudre l'hôte, s'il vous plaît attendre et essayer à nouveau.

```
$ INGRESS=$(oc -n hello-world get ingress hello-openshift-alb -o
jsonpath='{.status.loadBalancer.ingress[0].hostname}')
$ curl "http://${INGRESS}"
```

Exemple de sortie

```
Hello OpenShift!
```

7.3.1. Configurer l'AWS WAF

Le service AWS WAF est un pare-feu d'application Web qui vous permet de surveiller, de protéger et de contrôler les requêtes HTTP et HTTPS qui sont transmises à vos ressources d'application Web protégées, comme ROSA.

1. Créer un fichier de règles AWS WAF pour s'appliquer à notre ACL web:

```
$ cat << EOF > ${SCRATCH}/waf-rules.json
[
  {
    "Name": "AWS-AWSManagedRulesCommonRuleSet",
    "Priority": 0,
    "Statement": {
      "ManagedRuleGroupStatement": {
        "VendorName": "AWS",
        "Name": "AWSManagedRulesCommonRuleSet"
      }
    },
    "OverrideAction": {
      "None": {}
    },
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "AWS-AWSManagedRulesCommonRuleSet"
    }
  },
  {
```

```
    "Name": "AWS-AWSManagedRulesSQLiRuleSet",
    "Priority": 1,
    "Statement": {
      "ManagedRuleGroupStatement": {
        "VendorName": "AWS",
        "Name": "AWSManagedRulesSQLiRuleSet"
      }
    },
    "OverrideAction": {
      "None": {}
    },
    "VisibilityConfig": {
      "SampledRequestsEnabled": true,
      "CloudWatchMetricsEnabled": true,
      "MetricName": "AWS-AWSManagedRulesSQLiRuleSet"
    }
  }
]
EOF
```

Cela activera les jeux de règles gérés Core (Common) et SQL AWS.

2. Créer un AWS WAF Web ACL en utilisant les règles que nous avons spécifiées ci-dessus:

```
$ WAF_ARN=$(aws wafv2 create-web-acl \
  --name ${CLUSTER}-waf \
  --region ${REGION} \
  --default-action Allow={} \
  --scope REGIONAL \
  --visibility-config
SampledRequestsEnabled=true,CloudWatchMetricsEnabled=true,MetricName=${CLUSTER}-
waf-metrics \
  --rules file://${SCRATCH}/waf-rules.json \
  --query 'Summary.ARN' \
  --output text)
```

3. Annotez la ressource Ingress avec AWS WAF Web ACL ARN:

```
$ oc annotate -n hello-world ingress.networking.k8s.io/hello-openshift-alb \
  alb.ingress.kubernetes.io/wafv2-acl-arn=${WAF_ARN}
```

4. Attendez 10 secondes pour que les règles se propagent et testent que l'application fonctionne toujours:

```
$ curl "http://${INGRESS}"
```

Exemple de sortie

```
Hello OpenShift!
```

5. Essayez que la WAF nie une mauvaise demande:

```
$ curl -X POST "http://${INGRESS}" \
  -F "user='<script><alert>Hello></alert></script>'"
```

Exemple de sortie

```
<html>
<head><title>403 Forbidden</title></head>
<body>
<center><h1>403 Forbidden</h1></center>
</body>
</html
```

NOTE

L'activation de l'intégration AWS WAF peut parfois prendre plusieurs minutes. Dans le cas où vous ne recevez pas une erreur 403 interdite, veuillez attendre quelques secondes et essayer à nouveau.

Le résultat attendu est une erreur 403 interdite, ce qui signifie que le WAF AWS protège votre application.

7.4. RESSOURCES SUPPLÉMENTAIRES

- Ajout d'une sécurité supplémentaire avec AWS WAF, CloudFront et ROSA | Amazon Web Services sur YouTube

CHAPITRE 8. DÉPLOIEMENT DE L'API OPENSHIFT POUR LA PROTECTION DES DONNÉES SUR UN CLUSTER ROSA

IMPORTANT

Ce contenu est rédigé par des experts de Red Hat, mais n'a pas encore été testé sur toutes les configurations prises en charge.

Conditions préalables

- Cluster classique ROSA

Environnement

- Élaborer les variables de l'environnement:

NOTE

Changez le nom du cluster pour correspondre à votre cluster ROSA et assurez-vous d'être connecté au cluster en tant qu'administrateur. Assurez-vous que tous les champs sont affichés correctement avant de passer à autre chose.

```
$ export CLUSTER_NAME=$(oc get infrastructure cluster -o=jsonpath="
{.status.infrastructureName}"  | sed 's/-[a-z0-9]\{5\}$//')
$ export ROSA_CLUSTER_ID=$(rosa describe cluster -c ${CLUSTER_NAME} --output json
| jq -r .id)
$ export REGION=$(rosa describe cluster -c ${CLUSTER_NAME} --output json | jq -r
.region.id)
$ export OIDC_ENDPOINT=$(oc get authentication.config.openshift.io cluster -o
jsonpath='{.spec.serviceAccountIssuer}' | sed  's|^https://||')
$ export AWS_ACCOUNT_ID=`aws sts get-caller-identity --query Account --output text`
$ export CLUSTER_VERSION=`rosa describe cluster -c ${CLUSTER_NAME} -o json | jq -r
.version.raw_id | cut -f -2 -d '.'`
$ export ROLE_NAME="${CLUSTER_NAME}-openshift-oadp-aws-cloud-credentials"
$ export AWS_PAGER=""
$ export SCRATCH="/tmp/${CLUSTER_NAME}/oadp"
$ mkdir -p ${SCRATCH}
$ echo "Cluster ID: ${ROSA_CLUSTER_ID}, Region: ${REGION}, OIDC Endpoint:
${OIDC_ENDPOINT}, AWS Account ID: ${AWS_ACCOUNT_ID}"
```

8.1. CRÉER UN COMPTE AWS

1. Créer une politique IAM pour permettre l'accès S3:

```
$ POLICY_ARN=$(aws iam list-policies --query "Policies[?PolicyName=='RosaOadpVer1'].
{ARN:Arn}" --output text)
if [[ -z "${POLICY_ARN}" ]]; then
$ cat << EOF > ${SCRATCH}/policy.json
{
"Version": "2012-10-17",
"Statement": [
 {
```

```
    "Effect": "Allow",
    "Action": [
      "s3:CreateBucket",
      "s3:DeleteBucket",
      "s3:PutBucketTagging",
      "s3:GetBucketTagging",
      "s3:PutEncryptionConfiguration",
      "s3:GetEncryptionConfiguration",
      "s3:PutLifecycleConfiguration",
      "s3:GetLifecycleConfiguration",
      "s3:GetBucketLocation",
      "s3:ListBucket",
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject",
      "s3:ListBucketMultipartUploads",
      "s3:AbortMultipartUpload",
      "s3:ListMultipartUploadParts",
      "ec2:DescribeSnapshots",
      "ec2:DescribeVolumes",
      "ec2:DescribeVolumeAttribute",
      "ec2:DescribeVolumesModifications",
      "ec2:DescribeVolumeStatus",
      "ec2:CreateTags",
      "ec2:CreateVolume",
      "ec2:CreateSnapshot",
      "ec2:DeleteSnapshot"
    ],
    "Resource": "*"
  }
]}
EOF
$ POLICY_ARN=$(aws iam create-policy --policy-name "RosaOadpVer1" \
--policy-document file:///${SCRATCH}/policy.json --query Policy.Arn \
--tags Key=rosa_openshift_version,Value=${CLUSTER_VERSION}
Key=rosa_role_prefix,Value=ManagedOpenShift
Key=operator_namespace,Value=openshift-oadp Key=operator_name,Value=openshift-oadp
\
--output text)
fi
$ echo ${POLICY_ARN}
```

2. Créer une politique de confiance du rôle de l'IAM pour le cluster:

```
$ cat <<EOF > ${SCRATCH}/trust-policy.json
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "Federated": "arn:aws:iam::${AWS_ACCOUNT_ID}:oidc-provider/${OIDC_ENDPOINT}"
    },
    "Action": "sts:AssumeRoleWithWebIdentity",
    "Condition": {
      "StringEquals": {
        "${OIDC_ENDPOINT}:sub": [
```

```
            "system:serviceaccount:openshift-adp:openshift-adp-controller-manager",
            "system:serviceaccount:openshift-adp:velero"]
        }
      }
    }]
}
EOF
$ ROLE_ARN=$(aws iam create-role --role-name \
 "${ROLE_NAME}" \
  --assume-role-policy-document file://${SCRATCH}/trust-policy.json \
  --tags Key=rosa_cluster_id,Value=${ROSA_CLUSTER_ID}
Key=rosa_openshift_version,Value=${CLUSTER_VERSION}
Key=rosa_role_prefix,Value=ManagedOpenShift
Key=operator_namespace,Value=openshift-adp Key=operator_name,Value=openshift-oadp \
  --query Role.Arn --output text)

$ echo ${ROLE_ARN}
```

3. Joindre la politique de l'IAM au rôle de l'IAM:

```
$ aws iam attach-role-policy --role-name "${ROLE_NAME}" \
 --policy-arn ${POLICY_ARN}
```

8.2. DÉPLOYER OADP SUR LE CLUSTER

1. Créer un espace de noms pour OADP:

```
$ oc create namespace openshift-adp
```

2. Créer un secret d'identification:

```
$ cat <<EOF > ${SCRATCH}/credentials
[default]
role_arn = ${ROLE_ARN}
web_identity_token_file = /var/run/secrets/openshift/serviceaccount/token
region=<aws_region> (1)
EOF
$ oc -n openshift-adp create secret generic cloud-credentials \
 --from-file=${SCRATCH}/credentials
```

(1) <aws_region> par la région AWS à utiliser pour le point d'extrémité du service de jeton de sécurité (STS).

3. Déployer l'opérateur OADP:

NOTE

Il y a actuellement un problème avec la version 1.1 de l'opérateur avec des sauvegardes qui ont un statut PartiallyFailed. Cela ne semble pas affecter le processus de sauvegarde et de restauration, mais il convient de noter qu'il y a des problèmes avec lui.

```
$ cat << EOF | oc create -f -
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
 generateName: openshift-adp-
 namespace: openshift-adp
 name: oadp
spec:
 targetNamespaces:
 - openshift-adp
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
 name: redhat-oadp-operator
 namespace: openshift-adp
spec:
 channel: stable-1.2
 installPlanApproval: Automatic
 name: redhat-oadp-operator
 source: redhat-operators
 sourceNamespace: openshift-marketplace
EOF
```

4. Attendez que l'opérateur soit prêt:

```
$ watch oc -n openshift-adp get pods
```

Exemple de sortie

```
NAME                                READY   STATUS    RESTARTS   AGE
openshift-adp-controller-manager-546684844f-qqjhn   1/1     Running   0          22s
```

5. Créer le stockage en nuage:

```
$ cat << EOF | oc create -f -
apiVersion: oadp.openshift.io/v1alpha1
kind: CloudStorage
metadata:
 name: ${CLUSTER_NAME}-oadp
 namespace: openshift-adp
spec:
 creationSecret:
   key: credentials
   name: cloud-credentials
 enableSharedConfig: true
 name: ${CLUSTER_NAME}-oadp
 provider: aws
 region: $REGION
EOF
```

6. Consultez la classe de stockage par défaut de votre application:

```
$ oc get pvc -n <namespace> 1
```

```
-
```

1 Entrez l'espace de noms de votre application.

Exemple de sortie

```
NAME     STATUS   VOLUME                                     CAPACITY   ACCESS MODES
STORAGECLASS   AGE
applog   Bound    pvc-351791ae-b6ab-4e8b-88a4-30f73caf5ef8   1Gi        RWO            gp3-
csi        4d19h
mysql    Bound    pvc-16b8e009-a20a-4379-accc-bc81fedd0621   1Gi        RWO            gp3-
csi        4d19h
```

```
$ oc get storageclass
```

Exemple de sortie

```
NAME              PROVISIONER             RECLAIMPOLICY   VOLUMEBINDINGMODE
ALLOWVOLUMEEXPANSION   AGE
gp2               kubernetes.io/aws-ebs   Delete          WaitForFirstConsumer   true
4d21h
gp2-csi           ebs.csi.aws.com         Delete          WaitForFirstConsumer   true
4d21h
gp3               ebs.csi.aws.com         Delete          WaitForFirstConsumer   true
4d21h
gp3-csi (default)   ebs.csi.aws.com         Delete          WaitForFirstConsumer   true
4d21h
```

En utilisant gp3-csi, gp2-csi, gp3 ou gp2 fonctionneront. Dans le cas où les applications sauvegardées sont toutes utilisées par PV avec CSI, incluez le plugin CSI dans la configuration OADP DPA.

7. CSI seulement: Déployer une application de protection des données:

```
$ cat << EOF | oc create -f -
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
 name: ${CLUSTER_NAME}-dpa
 namespace: openshift-adp
spec:
 backupImages: true
 features:
   dataMover:
     enable: false
 backupLocations:
 - bucket:
     cloudStorageRef:
       name: ${CLUSTER_NAME}-oadp
     credential:
       key: credentials
       name: cloud-credentials
     prefix: velero
     default: true
```

```
    config:
      region: ${REGION}
 configuration:
   velero:
     defaultPlugins:
     - openshift
     - aws
     - csi
   restic:
     enable: false
EOF
```

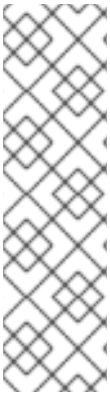
NOTE

Lorsque vous exécutez cette commande pour les volumes CSI, vous pouvez passer l'étape suivante.

8. Les volumes non CSI: Déployer une application de protection des données:

```
$ cat << EOF | oc create -f -
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
 name: ${CLUSTER_NAME}-dpa
 namespace: openshift-adp
spec:
 backupImages: true
 features:
   dataMover:
     enable: false
 backupLocations:
 - bucket:
     cloudStorageRef:
       name: ${CLUSTER_NAME}-oadp
     credential:
       key: credentials
       name: cloud-credentials
     prefix: velero
     default: true
     config:
       region: ${REGION}
 configuration:
   velero:
     defaultPlugins:
     - openshift
     - aws
   restic:
     enable: false
 snapshotLocations:
   - velero:
       config:
         credentialsFile: /tmp/credentials/openshift-adp/cloud-credentials-credentials
         enableSharedConfig: 'true'
         profile: default
```

```
      region: ${REGION}
    provider: aws
EOF
```



NOTE

- Dans les environnements OADP 1.1.x ROSA STS, la valeur de sauvegarde et de restauration d'image de conteneur (spec.backupImages) doit être définie sur false car elle n'est pas prise en charge.
- La fonction Restic (restic.enable=false) est désactivée et n'est pas prise en charge dans les environnements ROSA STS.
- La fonction DataMover (dataMover.enable=false) est désactivée et n'est pas prise en charge dans les environnements ROSA STS.

8.3. EFFECTUER UNE SAUVEGARDE

NOTE

L'échantillon suivant d'application hello-world n'a pas de volumes persistants attachés. La configuration DPA fonctionnera.

1. Créer une charge de travail pour sauvegarder:

```
$ oc create namespace hello-world
$ oc new-app -n hello-world --image=docker.io/openshift/hello-openshift
```

2. Exposer l'itinéraire:

```
$ oc expose service/hello-openshift -n hello-world
```

3. Assurez-vous que l'application fonctionne:

```
$ curl `oc get route/hello-openshift -n hello-world -o jsonpath='{.spec.host}'`
```

Exemple de sortie

```
Hello OpenShift!
```

4. Sauvegardez la charge de travail:

```
$ cat << EOF | oc create -f -
apiVersion: velero.io/v1
kind: Backup
metadata:
 name: hello-world
 namespace: openshift-adp
spec:
 includedNamespaces:
 - hello-world
```

```
 storageLocation: ${CLUSTER_NAME}-dpa-1
 ttl: 720h0m0s
EOF
```

5. Attendez que la sauvegarde soit faite:

```
$ watch "oc -n openshift-adp get backup hello-world -o json | jq .status"
```

Exemple de sortie

```
{
 "completionTimestamp": "2022-09-07T22:20:44Z",
 "expiration": "2022-10-07T22:20:22Z",
 "formatVersion": "1.1.0",
 "phase": "Completed",
 "progress": {
   "itemsBackedUp": 58,
   "totalItems": 58
 },
 "startTimestamp": "2022-09-07T22:20:22Z",
 "version": 1
}
```

6. B) Supprimer la charge de travail de démonstration:

```
$ oc delete ns hello-world
```

7. La restauration à partir de la sauvegarde:

```
$ cat << EOF | oc create -f -
apiVersion: velero.io/v1
kind: Restore
metadata:
 name: hello-world
 namespace: openshift-adp
spec:
 backupName: hello-world
EOF
```

8. Attendez que la restauration se termine:

```
$ watch "oc -n openshift-adp get restore hello-world -o json | jq .status"
```

Exemple de sortie

```
{
 "completionTimestamp": "2022-09-07T22:25:47Z",
 "phase": "Completed",
 "progress": {
   "itemsRestored": 38,
   "totalItems": 38
 },
```

```
 "startTimestamp": "2022-09-07T22:25:28Z",
 "warnings": 9
}
```

9. Assurez-vous que la charge de travail est restaurée:

```
$ oc -n hello-world get pods
```

Exemple de sortie

```
NAME                         READY   STATUS    RESTARTS   AGE
hello-openshift-9f885f7c6-kdjpj   1/1     Running   0          90s
```

```
$ curl `oc get route/hello-openshift -n hello-world -o jsonpath='{.spec.host}'`
```

Exemple de sortie

```
Hello OpenShift!
```

10. Conseils de dépannage s'il vous plaît consulter la documentation de dépannage de l'équipe OADP

11. D'autres exemples d'applications peuvent être trouvés dans le répertoire des exemples d'applications de l'équipe OADP

8.4. LE NETTOYAGE

1. B) Supprimer la charge de travail:

```
$ oc delete ns hello-world
```

2. Enlevez les ressources de sauvegarde et de restauration du cluster si elles ne sont plus nécessaires:

```
$ oc delete backups.velero.io hello-world
$ oc delete restores.velero.io hello-world
```

3. De supprimer la sauvegarde/restauration et les objets distants dans s3:

```
$ velero backup delete hello-world
$ velero restore delete hello-world
```

4. Effacer l'application de protection des données:

```
$ oc -n openshift-adp delete dpa ${CLUSTER_NAME}-dpa
```

5. Effacer le stockage en nuage:

```
$ oc -n openshift-adp delete cloudstorage ${CLUSTER_NAME}-oadp
```

AVERTISSEMENT

En cas de suspension de cette commande, vous devrez peut-être supprimer le finalisateur:

```
$ oc -n openshift-adp patch cloudstorage ${CLUSTER_NAME}-oadp -p
'{"metadata":{"finalizers":null}}' --type=merge
```

6. Enlevez l'opérateur s'il n'est plus nécessaire:

```
$ oc -n openshift-adp delete subscription oadp-operator
```

7. Enlever l'espace de noms de l'opérateur:

```
$ oc delete ns redhat-openshift-adp
```

8. Supprimez les Définitions de ressources personnalisées du cluster si vous ne souhaitez plus les avoir:

```
$ for CRD in `oc get crds | grep velero | awk '{print $1}'`; do oc delete crd $CRD; done
$ for CRD in `oc get crds | grep -i oadp | awk '{print $1}'`; do oc delete crd $CRD; done
```

9. Effacer le seau AWS S3:

```
$ aws s3 rm s3://${CLUSTER_NAME}-oadp --recursive
$ aws s3api delete-bucket --bucket ${CLUSTER_NAME}-oadp
```

10. Détachez la politique du rôle:

```
$ aws iam detach-role-policy --role-name "${ROLE_NAME}" \
 --policy-arn "${POLICY_ARN}"
```

11. Effacer le rôle:

```
$ aws iam delete-role --role-name "${ROLE_NAME}"
```

CHAPITRE 9. TUTORIEL: AWS LOAD BALANCER OPERATOR SUR ROSA

IMPORTANT

Ce contenu est rédigé par des experts de Red Hat, mais n'a pas encore été testé sur toutes les configurations prises en charge.

ASTUCE

Les balanceurs de charge créés par AWS Load Balancer Operator ne peuvent pas être utilisés pour les routes OpenShift, et ne doivent être utilisés que pour des services individuels ou des ressources d'entrée qui n'ont pas besoin des capacités de la couche 7 complète d'une route OpenShift.

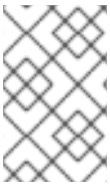
Le contrôleur AWS Load Balancer gère AWS Elastic Load Balancers pour un cluster Red Hat OpenShift Service sur AWS (ROSA). Le contrôleur prévoit AWS Application Load Balancers (ALB) lorsque vous créez des ressources Kubernetes Ingress et AWS Network Load Balancers (NLB) lorsque vous implémentez des ressources Kubernetes Service avec un type de LoadBalancer.

Comparé au fournisseur d'équilibrage de charge AWS par défaut, ce contrôleur est développé avec des annotations avancées pour les ALB et les NLB. Certains cas d'utilisation avancés sont:

- En utilisant des objets natives Kubernetes Ingress avec ALBs
- Intégrez les ALB au service AWS Web Application Firewall (WAF)
- Spécifiez les plages IP de source NLB personnalisées
- Indiquez les adresses IP internes NLB personnalisées

L'opérateur AWS Load Balancer est utilisé pour installer, gérer et configurer une instance de contrôleur aws-charge-balancer dans un cluster ROSA.

9.1. CONDITIONS PRÉALABLES



NOTE

Les ALB AWS nécessitent un cluster multi-AZ, ainsi que trois sous-réseaux publics répartis en trois AZ dans le même VPC que le cluster. Cela rend les ALB inadaptés à de nombreux clusters PrivateLink. Les NLB AWS n'ont pas cette restriction.

- Cluster classique ROSA multi-AZ
- BYO VPC cluster
- AWS CLI
- CLI D'OCC

9.1.1. Environnement

- Élaborer les variables de l'environnement:
  -

```
$ export AWS_PAGER=""
$ export ROSA_CLUSTER_NAME=$(oc get infrastructure cluster -o=jsonpath="
{.status.infrastructureName}"  | sed 's/-[a-z0-9]\{5\}$//')
$ export REGION=$(oc get infrastructure cluster -o=jsonpath="
{.status.platformStatus.aws.region}")
$ export OIDC_ENDPOINT=$(oc get authentication.config.openshift.io cluster -o
jsonpath='{.spec.serviceAccountIssuer}' | sed  's|^https://||')
$ export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
$ export SCRATCH="/tmp/${ROSA_CLUSTER_NAME}/alb-operator"
$ mkdir -p ${SCRATCH}
$ echo "Cluster: ${ROSA_CLUSTER_NAME}, Region: ${REGION}, OIDC Endpoint:
${OIDC_ENDPOINT}, AWS Account ID: ${AWS_ACCOUNT_ID}"
```

9.1.2. AWS VPC et sous-réseaux

NOTE

Cette section ne s'applique qu'aux clusters qui ont été déployés dans des VPC existants. Dans le cas où vous n'avez pas déployé votre cluster dans un VPC existant, sautez cette section et passez à la section d'installation ci-dessous.

1. Définissez les variables ci-dessous aux valeurs appropriées pour votre déploiement ROSA:

```
$ export VPC_ID=<vpc-id>
$ export PUBLIC_SUBNET_IDS=<public-subnets>
$ export PRIVATE_SUBNET_IDS=<private-subnets>
$ export CLUSTER_NAME=$(oc get infrastructure cluster -o=jsonpath="
{.status.infrastructureName}")
```

2. Ajoutez une balise au VPC de votre cluster avec le nom du cluster:

```
$ aws ec2 create-tags --resources ${VPC_ID} --tags
Key=kubernetes.io/cluster/${CLUSTER_NAME},Value=owned --region ${REGION}
```

3. Ajoutez une étiquette à vos sous-réseaux publics:

```
$ aws ec2 create-tags \
    --resources ${PUBLIC_SUBNET_IDS} \
    --tags Key=kubernetes.io/role/elb,Value='' \
    --region ${REGION}
```

4. Ajouter une balise à vos sous-réseaux privés:

```
$ aws ec2 create-tags \
    --resources "${PRIVATE_SUBNET_IDS}" \
    --tags Key=kubernetes.io/role/internal-elb,Value='' \
    --region ${REGION}
```

9.2. INSTALLATION

1. Créez une stratégie AWS IAM pour le contrôleur AWS Load Balancer:

NOTE

La stratégie provient de la stratégie AWS Load Balancer Controller en amont, plus l'autorisation de créer des balises sur les sous-réseaux. Ceci est requis par l'opérateur pour fonctionner.

```
$ oc new-project aws-load-balancer-operator
$ POLICY_ARN=$(aws iam list-policies --query \
    "Policies[?PolicyName=='aws-load-balancer-operator-policy'].{ARN:Arn}" \
    --output text)
$ if [[ -z "${POLICY_ARN}" ]]; then
   wget -O "${SCRATCH}/load-balancer-operator-policy.json" \
     https://raw.githubusercontent.com/rh-mobb/documentation/main/content/rosa/aws-load-balancer-operator/load-balancer-operator-policy.json
    POLICY_ARN=$(aws --region "$REGION" --query Policy.Arn \
    --output text iam create-policy \
    --policy-name aws-load-balancer-operator-policy \
    --policy-document "file://${SCRATCH}/load-balancer-operator-policy.json")
fi
$ echo $POLICY_ARN
```

2. Créez une politique de confiance AWS IAM pour AWS Load Balancer Operator:

```
$ cat <<EOF > "${SCRATCH}/trust-policy.json"
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Condition": {
   "StringEquals" : {
    "${OIDC_ENDPOINT}:sub": ["system:serviceaccount:aws-load-balancer-operator:aws-load-balancer-operator-controller-manager", "system:serviceaccount:aws-load-balancer-operator:aws-load-balancer-controller-cluster"]
   }
 },
 "Principal": {
   "Federated": "arn:aws:iam::$AWS_ACCOUNT_ID:oidc-provider/${OIDC_ENDPOINT}"
 },
 "Action": "sts:AssumeRoleWithWebIdentity"
 }
 ]
}
EOF
```

3. Créer un rôle AWS IAM pour l'opérateur AWS Load Balancer:

```
$ ROLE_ARN=$(aws iam create-role --role-name "${ROSA_CLUSTER_NAME}-alb-operator" \
   --assume-role-policy-document "file://${SCRATCH}/trust-policy.json" \
   --query Role.Arn --output text)
$ echo $ROLE_ARN

$ aws iam attach-role-policy --role-name "${ROSA_CLUSTER_NAME}-alb-operator" \
    --policy-arn $POLICY_ARN
```

4. Créez un secret pour que l'opérateur AWS Load Balancer assume notre nouveau rôle AWS IAM:

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: aws-load-balancer-operator
  namespace: aws-load-balancer-operator
stringData:
  credentials: |
    [default]
    role_arn = $ROLE_ARN
    web_identity_token_file = /var/run/secrets/openshift/serviceaccount/token
EOF
```

5. Installez l'opérateur AWS Load Balancer:

```
$ cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: aws-load-balancer-operator
  namespace: aws-load-balancer-operator
spec:
  upgradeStrategy: Default
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: aws-load-balancer-operator
  namespace: aws-load-balancer-operator
spec:
  channel: stable-v1.0
  installPlanApproval: Automatic
  name: aws-load-balancer-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
  startingCSV: aws-load-balancer-operator.v1.0.0
EOF
```

6. Déployez une instance du contrôleur AWS Load Balancer à l'aide de l'opérateur:

NOTE

Lorsque vous obtenez une erreur ici, attendez une minute et essayez à nouveau, cela signifie que l'opérateur n'a pas encore terminé l'installation.

```
$ cat << EOF | oc apply -f -
apiVersion: networking.olm.openshift.io/v1
kind: AWSLoadBalancerController
metadata:
  name: cluster
spec:
```

```
  credentials:
    name: aws-load-balancer-operator
EOF
```

7. Assurez-vous que les pods de l'opérateur et du contrôleur sont en cours d'exécution:

```
$ oc -n aws-load-balancer-operator get pods
```

Il faut voir ce qui suit, sinon attendre un moment et réessayer:

```
NAME                                                READY   STATUS    RESTARTS   AGE
aws-load-balancer-controller-cluster-6ddf658785-pdp5d           1/1     Running   0          99s
aws-load-balancer-operator-controller-manager-577d9ffcb9-w6zqn   2/2     Running   0
2m4s
```

9.3. LA VALIDATION DU DÉPLOIEMENT

1. Créer un nouveau projet:

```
$ oc new-project hello-world
```

2. Déployez une application hello world:

```
$ oc new-app -n hello-world --image=docker.io/openshift/hello-openshift
```

3. Configurez un service NodePort pour que AWS ALB se connecte à:

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: Service
metadata:
  name: hello-openshift-nodeport
  namespace: hello-world
spec:
  ports:
    - port: 80
      targetPort: 8080
      protocol: TCP
  type: NodePort
  selector:
    deployment: hello-openshift
EOF
```

4. Déployer un AWS ALB à l'aide de l'opérateur AWS Load Balancer:

```
$ cat << EOF | oc apply -f -
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: hello-openshift-alb
  namespace: hello-world
  annotations:
    alb.ingress.kubernetes.io/scheme: internet-facing
```

```
spec:
  ingressClassName: alb
  rules:
    - http:
        paths:
          - path: /
            pathType: Exact
            backend:
              service:
                name: hello-openshift-nodeport
                port:
                  number: 80
EOF
```

5. Curl le point de terminaison AWS ALB Ingress pour vérifier l'application hello world est accessible:

NOTE

AWS ALB provisioning prend quelques minutes. Lorsque vous recevez une erreur qui dit curl: (6) Ne peut pas résoudre l'hôte, s'il vous plaît attendre et essayer à nouveau.

```
$ INGRESS=$(oc -n hello-world get ingress hello-openshift-alb \
    -o jsonpath='{.status.loadBalancer.ingress[0].hostname}')
$ curl "http://${INGRESS}"
```

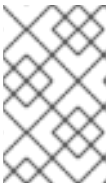
Exemple de sortie

```
Hello OpenShift!
```

6. Déployez une AWS NLB pour votre application hello world:

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: Service
metadata:
  name: hello-openshift-nlb
  namespace: hello-world
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: external
    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: instance
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  ports:
    - port: 80
      targetPort: 8080
      protocol: TCP
  type: LoadBalancer
  selector:
    deployment: hello-openshift
EOF
```

7. Essayez le point de terminaison AWS NLB:



NOTE

Le provisionnement de la NLB prend quelques minutes. Lorsque vous recevez une erreur qui dit curl: (6) Ne peut pas résoudre l'hôte, s'il vous plaît attendre et essayer à nouveau.

```
$ NLB=$(oc -n hello-world get service hello-openshift-nlb \
  -o jsonpath='{.status.loadBalancer.ingress[0].hostname}')
$ curl "http://${NLB}"
```

Exemple de sortie

```
Hello OpenShift!
```

9.4. LE NETTOYAGE

1. Effacer l'espace de noms de l'application hello world (et toutes les ressources dans l'espace de noms):

```
$ oc delete project hello-world
```

2. Effacer l'opérateur AWS Load Balancer et les rôles AWS IAM:

```
$ oc delete subscription aws-load-balancer-operator -n aws-load-balancer-operator
$ aws iam detach-role-policy \
  --role-name "${ROSA_CLUSTER_NAME}-alb-operator" \
  --policy-arn $POLICY_ARN
$ aws iam delete-role \
  --role-name "${ROSA_CLUSTER_NAME}-alb-operator"
```

3. Effacer la politique AWS IAM:

```
$ aws iam delete-policy --policy-arn $POLICY_ARN
```

CHAPITRE 10. CONFIGURATION DE MICROSOFT ENTRA ID (ANCIENNEMENT AZURE ACTIVE DIRECTORY) EN TANT QUE FOURNISSEUR D'IDENTITÉ

Il est possible de configurer Microsoft Entra ID (anciennement Azure Active Directory) en tant que fournisseur d'identité de cluster dans Red Hat OpenShift Service sur AWS (ROSA).

Ce tutoriel vous guide pour accomplir les tâches suivantes:

1. Enregistrez une nouvelle application dans Entra ID pour l'authentification.
2. Configurez l'enregistrement de l'application dans Entra ID pour inclure les revendications facultatives et de groupe dans les jetons.
3. Configurez le Red Hat OpenShift Service sur AWS cluster pour utiliser Entra ID comme fournisseur d'identité.
4. Accorder des autorisations supplémentaires à des groupes individuels.

10.1. CONDITIONS PRÉALABLES

- En suivant la documentation Microsoft, vous avez créé un ensemble de groupes de sécurité et d'utilisateurs assignés.

10.2. ENREGISTREMENT D'UNE NOUVELLE APPLICATION DANS ENTRA ID POUR L'AUTHENTIFICATION

Afin d'enregistrer votre application dans Entra ID, créez d'abord l'URL de rappel OAuth, puis enregistrez votre application.

Procédure

1. Créez l'URL de rappel OAuth du cluster en modifiant les variables spécifiées et en exécutant la commande suivante:

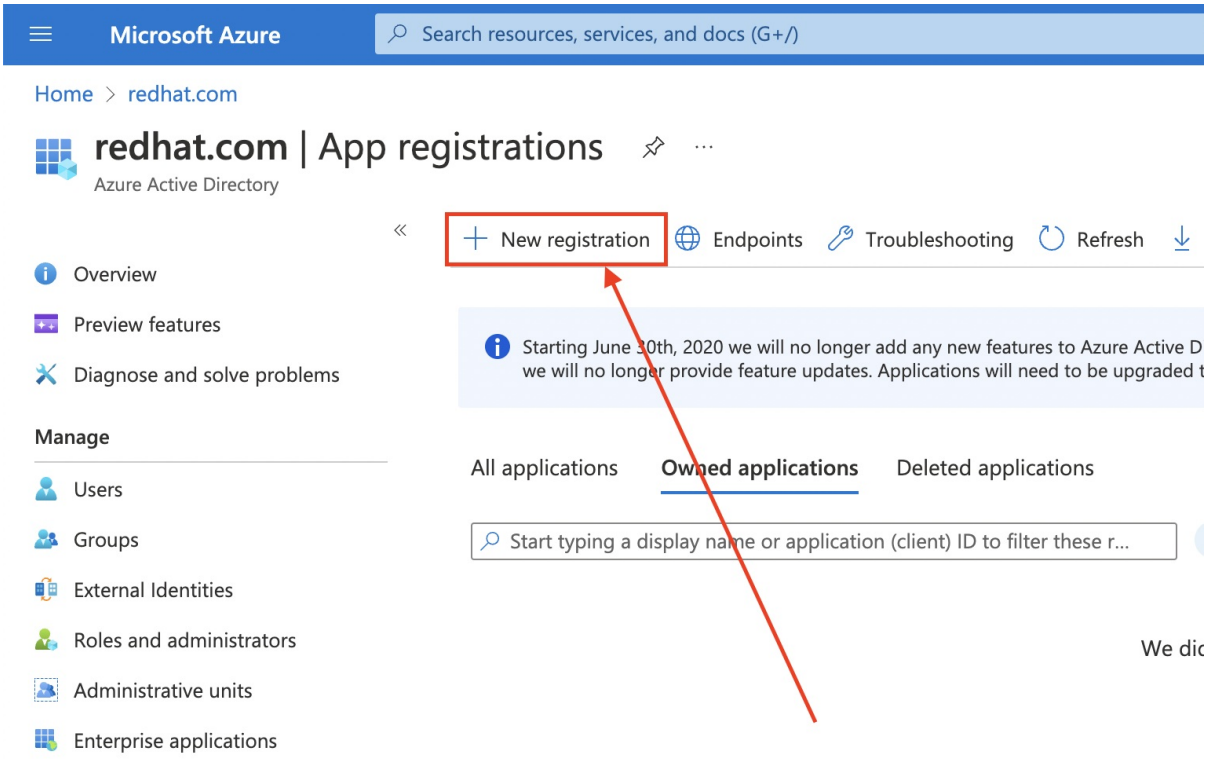
NOTE

Gardez à l'esprit d'enregistrer cette URL de rappel; elle sera nécessaire plus tard dans le processus.

```
$ domain=$(rosa describe cluster -c <cluster_name> | grep "DNS" | grep -oE
'\S+.openshiftapps.com')
echo "OAuth callback URL: https://oauth.${domain}/oauth2callback/AAD"
```

Le répertoire "AAD" à la fin de l'URL de rappel OAuth doit correspondre au nom du fournisseur d'identité OAuth que vous configurerez plus tard dans ce processus.

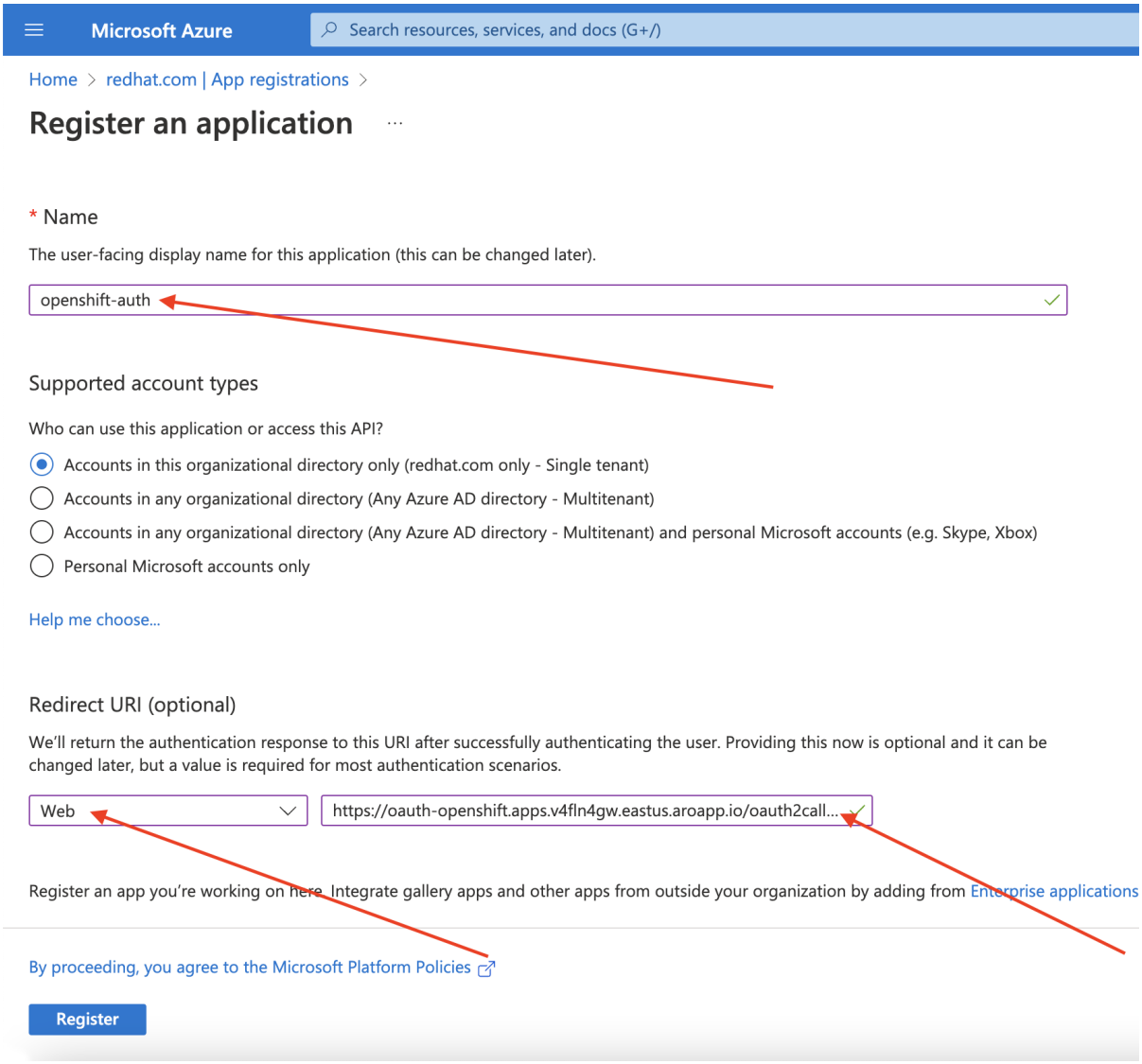
2. Créez l'application Entra ID en vous connectant au portail Azure et sélectionnez la lame d'enregistrement de l'application. Ensuite, sélectionnez Nouvelle inscription pour créer une nouvelle application.



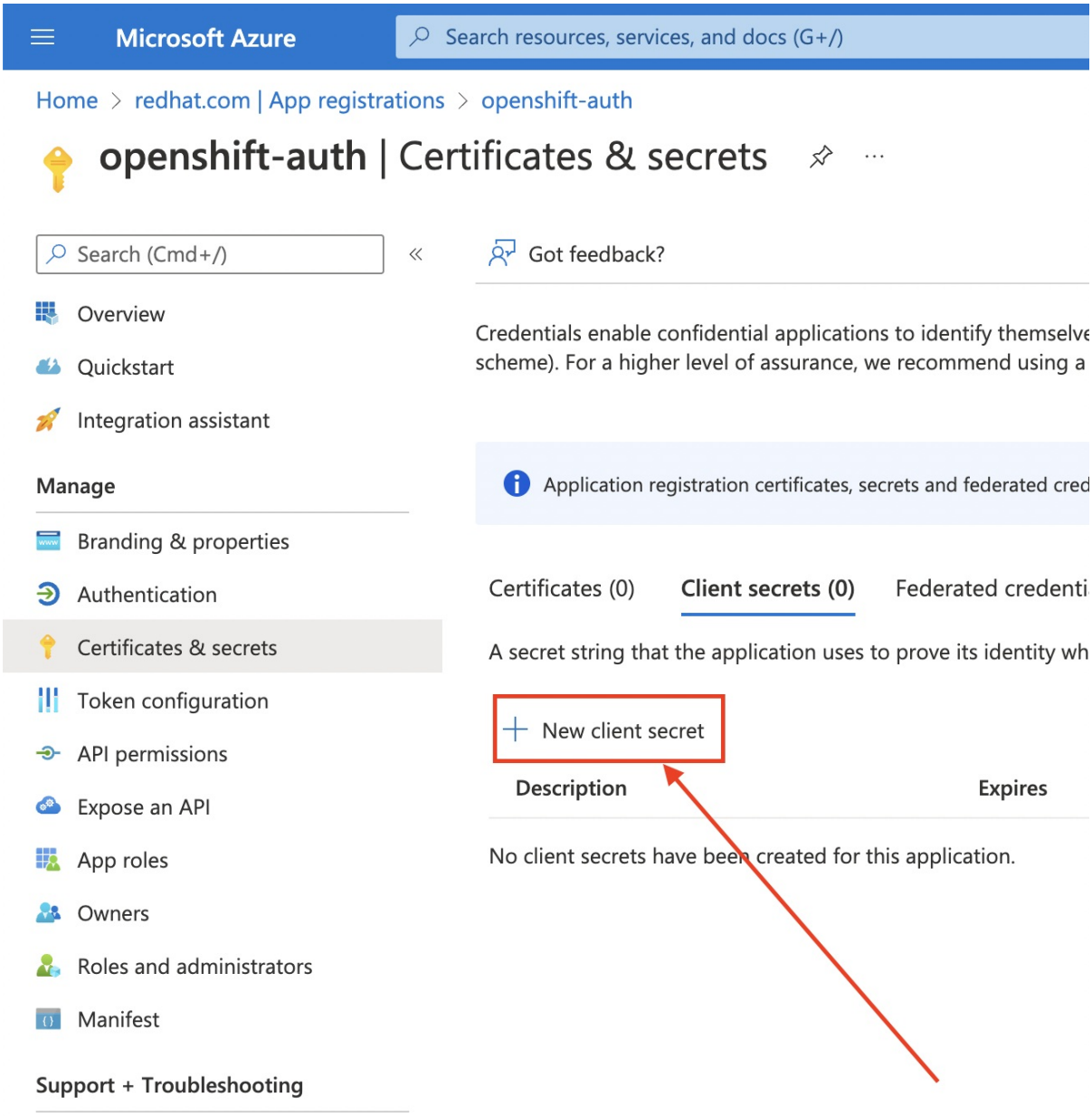
3. Indiquez l'application, par exemple openshift-auth.

4. Choisissez Web dans la liste déroulante Redirect URI et entrez la valeur de l'URL de rappel OAuth que vous avez récupérée dans l'étape précédente.

5. Après avoir fourni les informations requises, cliquez sur S'inscrire pour créer l'application.



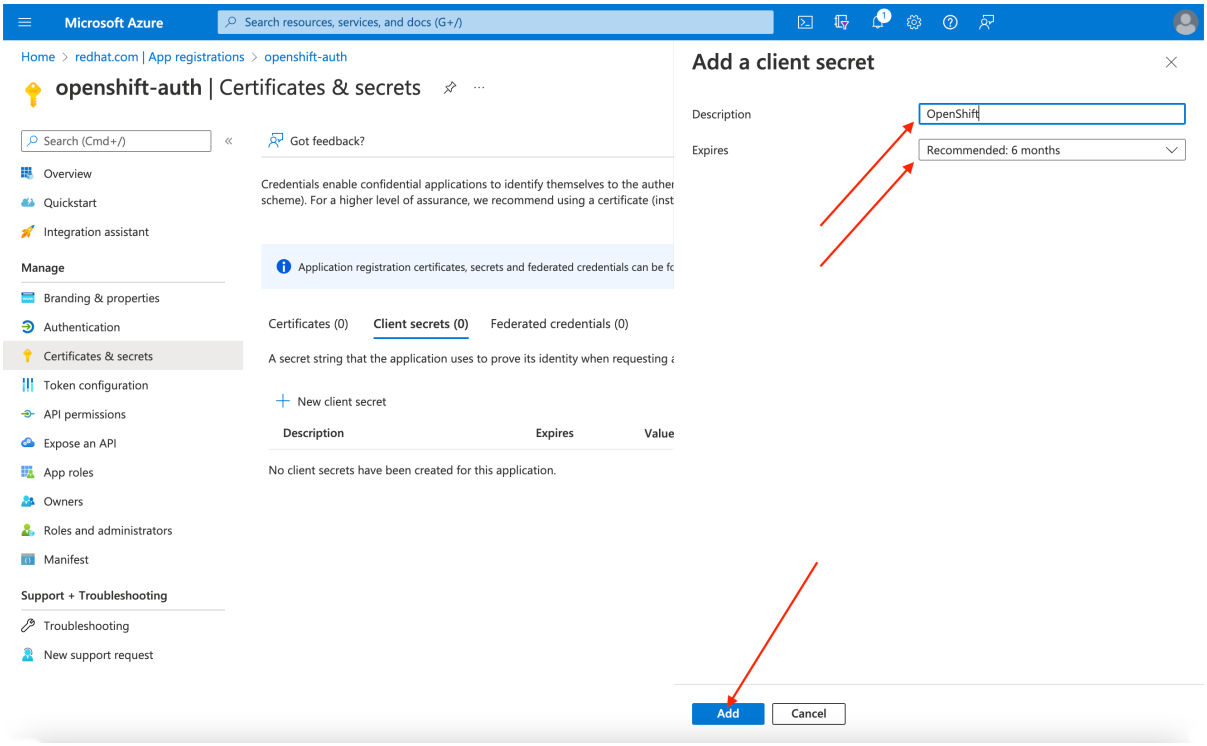
6. Choisissez la sous-lame Certificats & secrets et sélectionnez Nouveau secret client.



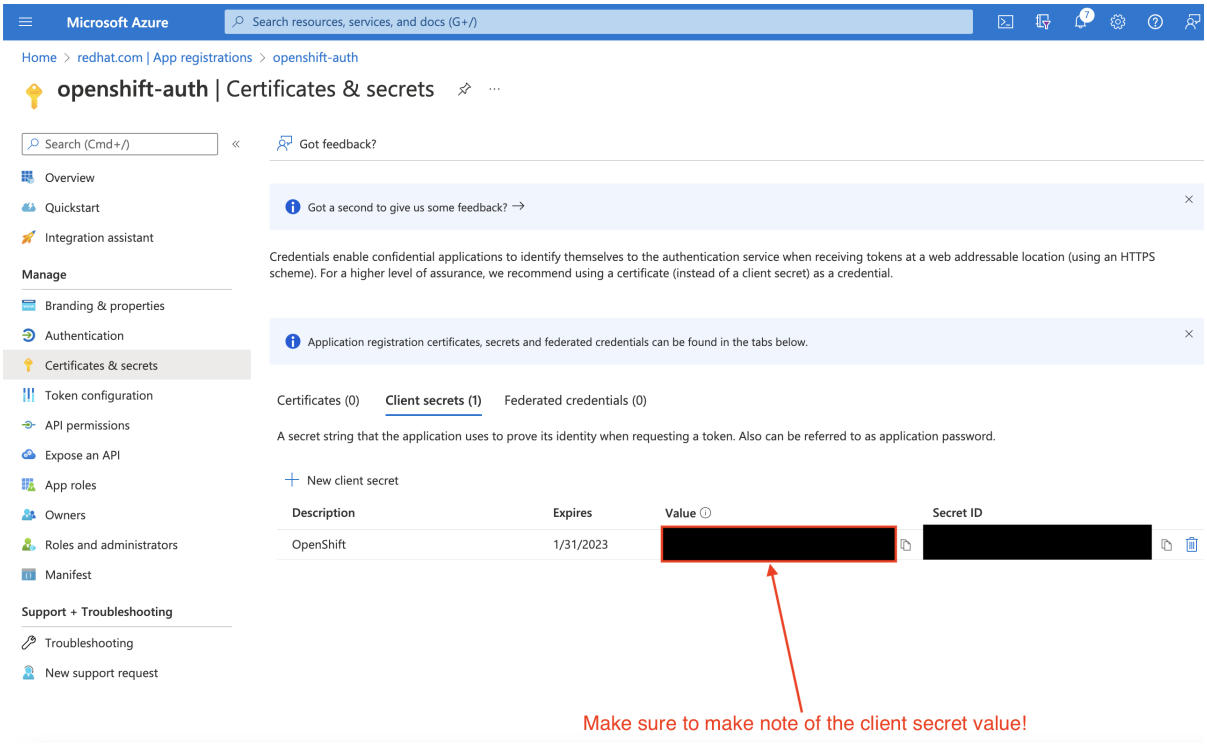
7. Complétez les détails demandés et stockez la valeur secrète du client générée. Ce secret est nécessaire plus tard dans ce processus.

IMPORTANT

Après la configuration initiale, vous ne pouvez pas voir le secret client. « si vous n'avez pas enregistré le secret client, vous devez en générer un nouveau.



8. Choisissez la sous-lame Aperçu et notez l'ID de l'Application (client) et l'ID d'annuaire (locataire). Il vous faudra ces valeurs dans une étape future.



10.3. CONFIGURATION DE L'ENREGISTREMENT DE L'APPLICATION DANS ENTRA ID POUR INCLURE LES REVENDICATIONS FACULTATIVES ET DE GROUPE

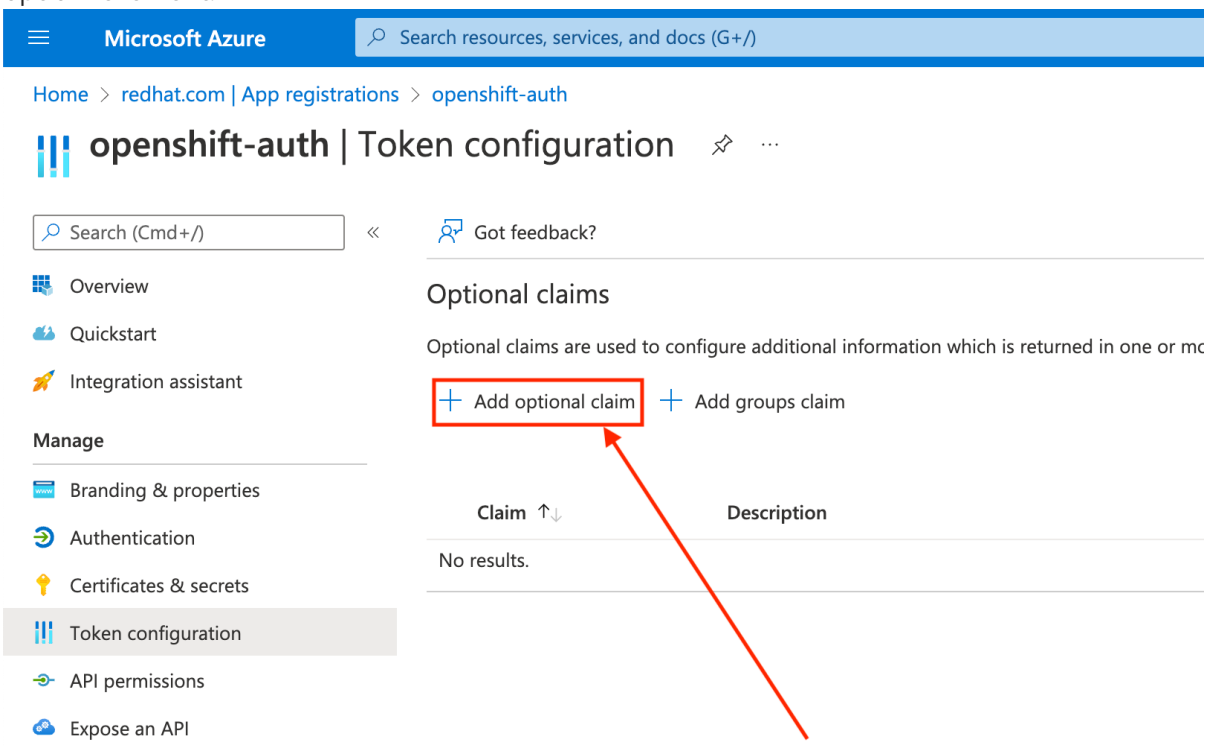
Afin que Red Hat OpenShift Service sur AWS dispose de suffisamment d'informations pour créer le compte de l'utilisateur, vous devez configurer Entra ID pour donner deux revendications facultatives: email et favorite_username. Consultez la documentation Microsoft pour plus d'informations sur les réclamations facultatives dans Entra ID.

En plus de l'authentification individuelle de l'utilisateur, Red Hat OpenShift Service sur AWS fournit des fonctionnalités de réclamation de groupe. Cette fonctionnalité permet à un fournisseur d'identité OpenID Connect (OIDC), tel qu'Entra ID, d'offrir l'adhésion à un groupe d'utilisateurs pour une utilisation dans Red Hat OpenShift Service sur AWS.

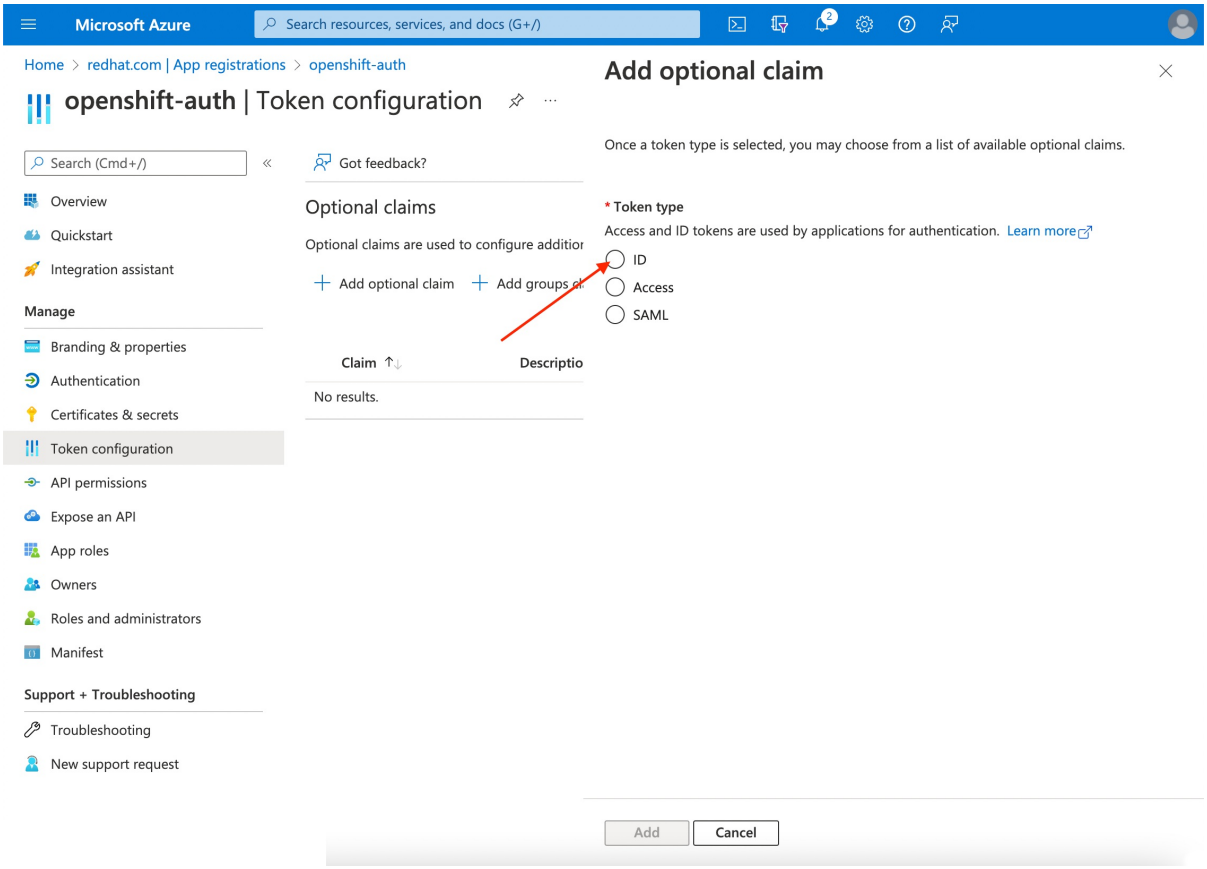
10.3.1. Configuration des revendications facultatives

Les réclamations optionnelles peuvent être configurées dans Entra ID.

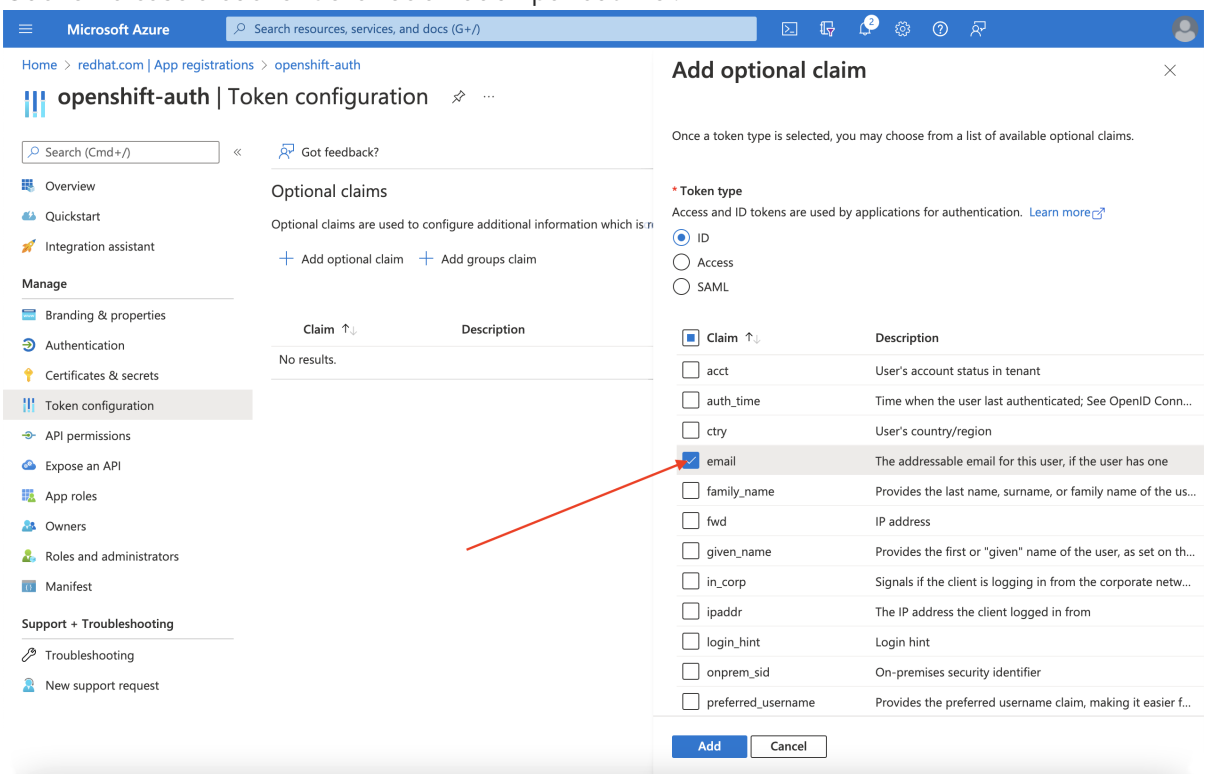
1. Cliquez sur le sous-lame de configuration du jeton et sélectionnez le bouton Ajouter optionnellement.



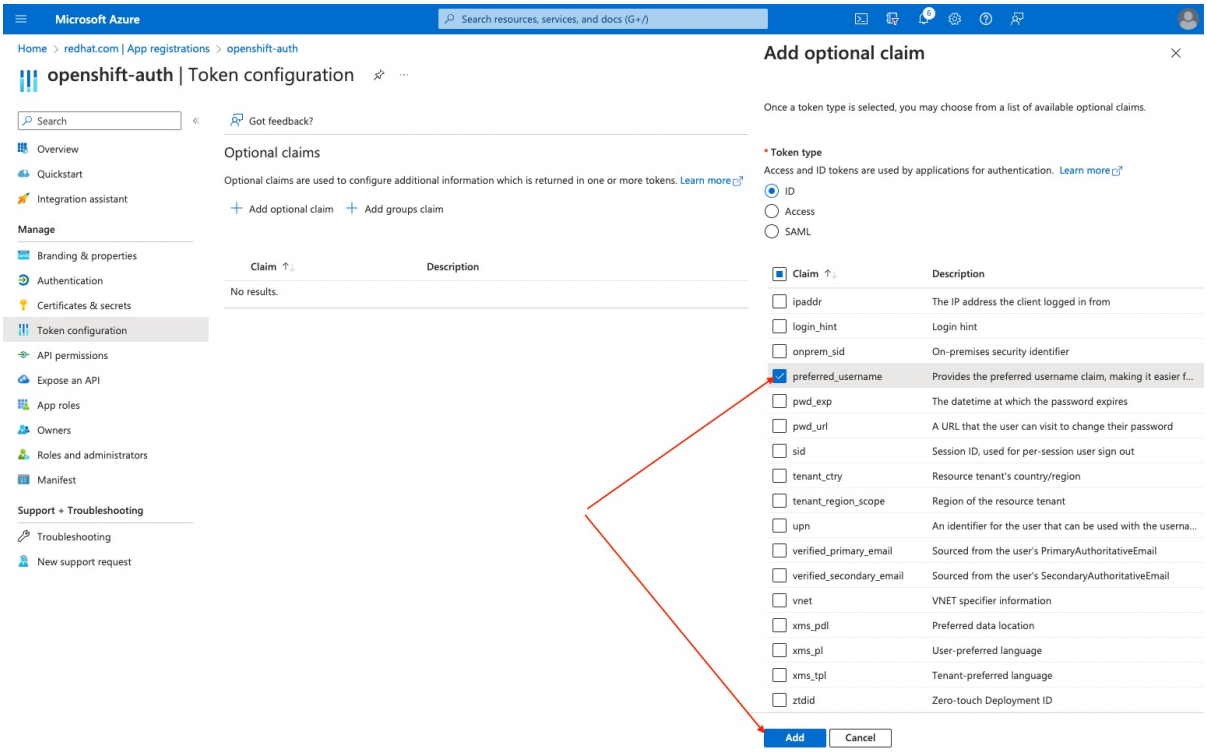
2. Appuyez sur le bouton ID radio.



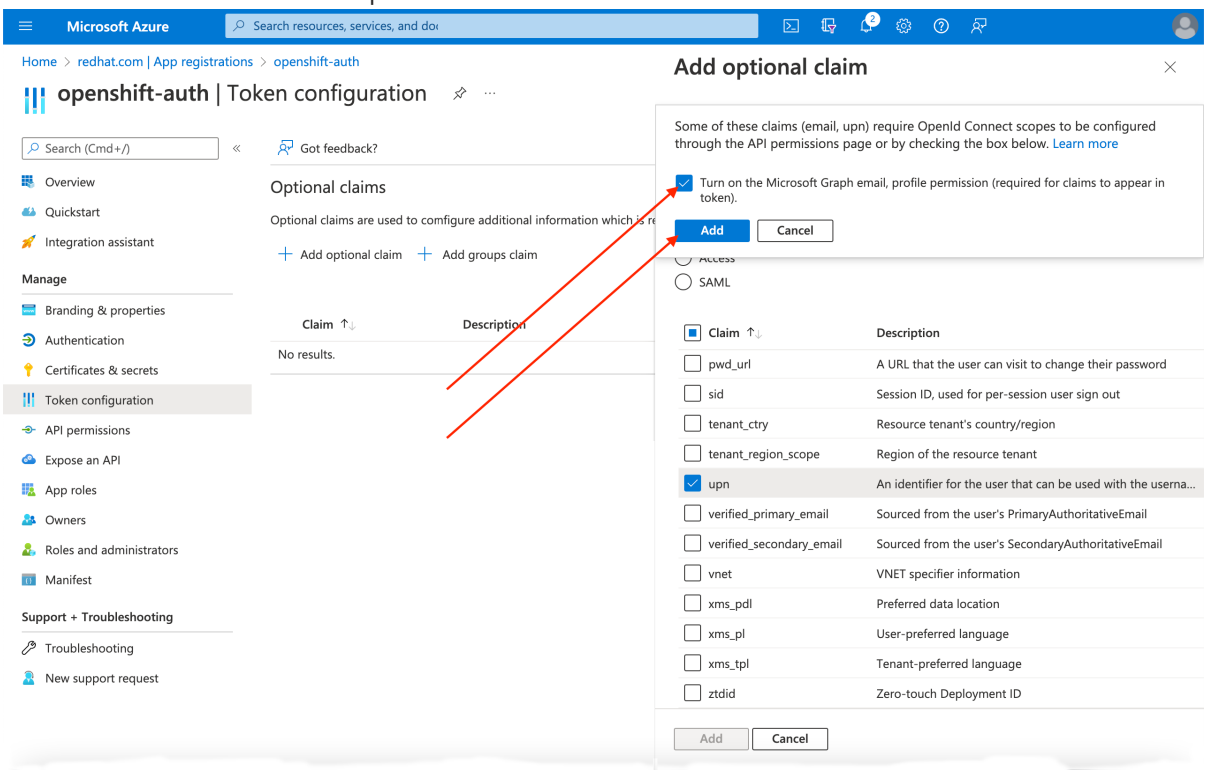
3. Cochez la case à cocher de la réclamation par courriel.



4. Cochez la case à cocher favorite_username. Ensuite, cliquez sur Ajouter pour configurer l'e-mail et favorite_username revendique votre application Entra ID.



5. La boîte de dialogue apparaît en haut de la page. Consultez l'invite pour activer les autorisations nécessaires de Microsoft Graph.

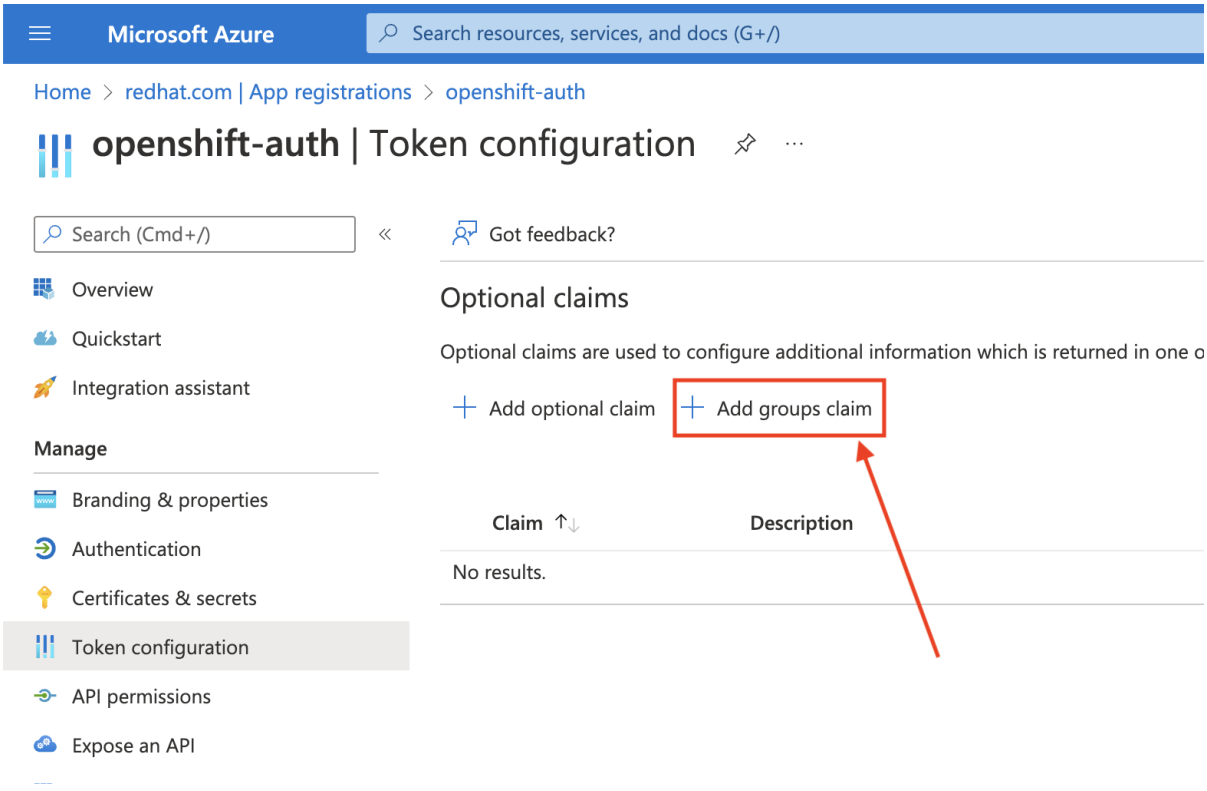


10.3.2. Configuration des revendications du groupe (facultatif)

Configurez l'ID d'Entra pour offrir une revendication de groupes.

Procédure

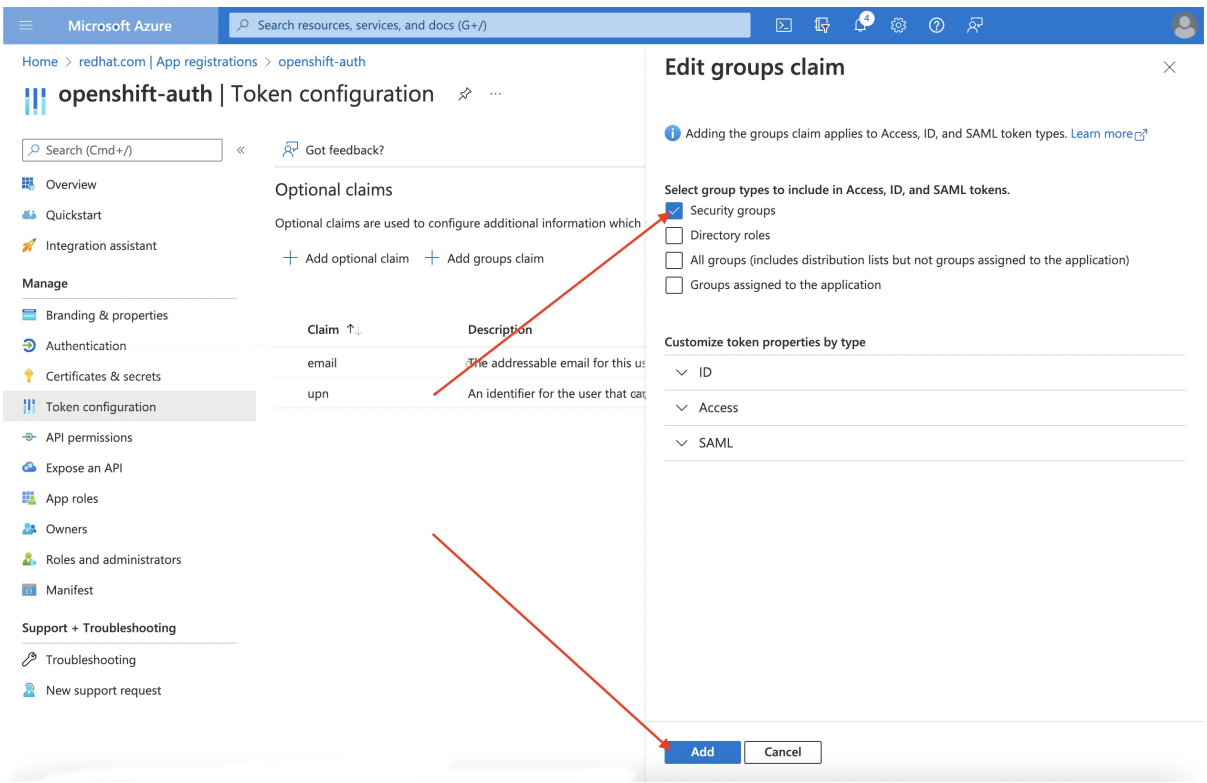
1. À partir de la sous-lame de configuration des jetons, cliquez sur Ajouter des groupes.



2. Afin de configurer les revendications de groupe pour votre application Entra ID, sélectionnez Groupes de sécurité, puis cliquez sur Ajouter.

NOTE

Dans cet exemple, la revendication de groupe inclut tous les groupes de sécurité dont un utilisateur est membre. Dans un environnement de production réel, assurez-vous que les groupes que le groupe revendique n'incluent que les groupes qui s'appliquent à Red Hat OpenShift Service sur AWS.



10.4. CONFIGURATION DU SERVICE RED HAT OPENSHIFT SUR LE CLUSTER AWS POUR UTILISER ENTRA ID COMME FOURNISSEUR D'IDENTITÉ

Il faut configurer Red Hat OpenShift Service sur AWS pour utiliser Entra ID comme fournisseur d'identité.

Bien que ROSA offre la possibilité de configurer des fournisseurs d'identité en utilisant OpenShift Cluster Manager, utilisez le ROSA CLI pour configurer le fournisseur OAuth du cluster pour utiliser Entra ID comme fournisseur d'identité. Avant de configurer le fournisseur d'identité, définissez les variables nécessaires à la configuration du fournisseur d'identité.

Procédure

1. Créez les variables en exécutant la commande suivante:

```
$ CLUSTER_NAME=example-cluster 1
$ IDP_NAME=AAD 2
$ APP_ID=yyyyyyyy-yyyy-yyyy-yyyy-yyyyyyyyyyyy 3
$ CLIENT_SECRET=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx 4
$ TENANT_ID=zzzzzzzz-zzzz-zzzz-zzzz-zzzzzzzzzzzz 5
```

1. Le remplacer par le nom de votre cluster ROSA.
2. Cette valeur remplace cette valeur par le nom que vous avez utilisé dans l'URL de rappel OAuth que vous avez générée plus tôt dans ce processus.
3. Le remplacer par l'ID de l'Application (client).
4. Le remplacer par le Client Secret.
5. Le remplacer par l'ID Directory (locataire).

2. Configurez le fournisseur OAuth du cluster en exécutant la commande suivante. Lorsque vous avez activé les revendications de groupe, assurez-vous d'utiliser l'argument des groupes --group-claims.

- Lorsque vous avez activé les revendications du groupe, exécutez la commande suivante:

```
$ rosa create idp \
--cluster ${CLUSTER_NAME} \
--type openid \
--name ${IDP_NAME} \
--client-id ${APP_ID} \
--client-secret ${CLIENT_SECRET} \
--issuer-url https://login.microsoftonline.com/${TENANT_ID}/v2.0 \
--email-claims email \
--name-claims name \
--username-claims preferred_username \
--extra-scopes email,profile \
--groups-claims groups
```

- Dans le cas où vous n'avez pas activé les revendications du groupe, exécutez la commande suivante:

```
$ rosa create idp \
--cluster ${CLUSTER_NAME} \
--type openid \
--name ${IDP_NAME} \
--client-id ${APP_ID} \
--client-secret ${CLIENT_SECRET} \
--issuer-url https://login.microsoftonline.com/${TENANT_ID}/v2.0 \
--email-claims email \
--name-claims name \
--username-claims preferred_username \
--extra-scopes email,profile
```

Après quelques minutes, l'opérateur d'authentification du cluster réconcilie vos modifications, et vous pouvez vous connecter au cluster en utilisant Entra ID.

10.5. ACCORDER DES AUTORISATIONS SUPPLÉMENTAIRES AUX UTILISATEURS ET AUX GROUPES INDIVIDUELS

Lors de votre première connexion, vous remarquerez peut-être que vous avez des autorisations très limitées. Le service Red Hat OpenShift sur AWS ne vous permet par défaut que de créer de nouveaux projets ou espaces de noms dans le cluster. D'autres projets sont restreints à la vue.

Ces capacités supplémentaires doivent être accordées à des utilisateurs et à des groupes individuels.

10.5.1. Accorder des autorisations supplémentaires aux utilisateurs individuels

Le service OpenShift Red Hat sur AWS inclut un nombre important de rôles préconfigurés, y compris le rôle cluster-admin qui accorde un accès et un contrôle complets sur le cluster.

Procédure

- Accordez à un utilisateur l'accès au rôle cluster-admin en exécutant la commande suivante:

```
$ rosa grant user cluster-admin \
  --user=<USERNAME> 1
  --cluster=${CLUSTER_NAME}
```

1 Fournissez le nom d'utilisateur Entra ID que vous souhaitez avoir des autorisations d'administration de cluster.

10.5.2. Accorder des autorisations supplémentaires à des groupes individuels

Lorsque vous avez choisi d'activer les revendications de groupe, le fournisseur de cluster OAuth crée ou met à jour automatiquement les adhésions de groupe de l'utilisateur à l'aide de l'ID de groupe. Le fournisseur de cluster OAuth ne crée pas automatiquement RoleBindings et ClusterRoleBindings pour les groupes qui sont créés; vous êtes responsable de la création de ces liaisons en utilisant vos propres processus.

Afin d'accorder un accès de groupe généré automatiquement au rôle de cluster-admin, vous devez créer un ClusterRoleBinding à l'ID du groupe.

Procédure

- Créez le ClusterRoleBinding en exécutant la commande suivante:

```
$ oc create clusterrolebinding cluster-admin-group \
--clusterrole=cluster-admin \
--group=<GROUP_ID> 1
```

1 Fournissez l'ID de groupe Entra ID que vous souhaitez avoir des autorisations d'administration de cluster.

Désormais, tout utilisateur du groupe spécifié reçoit automatiquement un accès cluster-admin.

10.6. RESSOURCES SUPPLÉMENTAIRES

En savoir plus sur la façon d'utiliser RBAC pour définir et appliquer des autorisations dans Red Hat OpenShift Service sur AWS, consultez le service Red Hat OpenShift sur la documentation AWS.

CHAPITRE 11. TUTORIEL: UTILISER AWS SECRETS MANAGER CSI SUR ROSA AVEC STS

AWS Secrets and Configuration Provider (ASCP) offre un moyen d'exposer AWS Secrets en tant que volumes de stockage Kubernetes. Avec l'ASCP, vous pouvez stocker et gérer vos secrets dans Secrets Manager, puis les récupérer via vos charges de travail fonctionnant sur Red Hat OpenShift Service sur AWS (ROSA).

11.1. CONDITIONS PRÉALABLES

Assurez-vous que vous disposez des ressources et des outils suivants avant de commencer ce processus:

- Cluster ROSA déployé avec STS
- Barre 3
- AWS CLI
- CLI d'OCC
- JQ CLI

Exigences supplémentaires en matière d'environnement

1. Connectez-vous à votre cluster ROSA en exécutant la commande suivante:

```
$ oc login --token=<your-token> --server=<your-server-url>
```

   Il est possible de trouver votre jeton de connexion en accédant à votre cluster en pull secret depuis Red Hat OpenShift Cluster Manager.

2. Validez que votre cluster a STS en exécutant la commande suivante:

```
$ oc get authentication.config.openshift.io cluster -o json \
  | jq .spec.serviceAccountIssuer
```

   Exemple de sortie

```
"https://xxxxx.cloudfront.net/xxxxx"
```

   Lorsque votre sortie est différente, ne procédez pas. Consultez la documentation Red Hat sur la création d'un cluster STS avant de poursuivre ce processus.

3. Définissez l'autorisation SecurityContextConstraints pour permettre au pilote CSI d'exécuter la commande suivante:

```
$ oc new-project csi-secrets-store
$ oc adm policy add-scc-to-user privileged \
    system:serviceaccount:csi-secrets-store:secrets-store-csi-driver
$ oc adm policy add-scc-to-user privileged \
    system:serviceaccount:csi-secrets-store:csi-secrets-store-provider-aws
```

4. Créez des variables d'environnement à utiliser plus tard dans ce processus en exécutant la commande suivante:

```
$ export REGION=$(oc get infrastructure cluster -o=jsonpath="
{.status.platformStatus.aws.region}")
$ export OIDC_ENDPOINT=$(oc get authentication.config.openshift.io cluster \
  -o jsonpath='{.spec.serviceAccountIssuer}' | sed  's|^https://||')
$ export AWS_ACCOUNT_ID=`aws sts get-caller-identity --query Account --output text`
$ export AWS_PAGER=""
```

11.2. DÉPLOIEMENT DU FOURNISSEUR AWS SECRETS ET CONFIGURATION

1. Enregistrez le pilote CSI du magasin secret en exécutant la commande suivante:

```
$ helm repo add secrets-store-csi-driver \
    https://kubernetes-sigs.github.io/secrets-store-csi-driver/charts
```

2. Actualisez vos dépôts Helm en exécutant la commande suivante:

```
$ helm repo update
```

3. Installez le pilote CSI store secret en exécutant la commande suivante:

```
$ helm upgrade --install -n csi-secrets-store \
    csi-secrets-store-driver secrets-store-csi-driver/secrets-store-csi-driver
```

4. Déployez le fournisseur AWS en exécutant la commande suivante:

```
$ oc -n csi-secrets-store apply -f \
    https://raw.githubusercontent.com/rh-mobb/documentation/main/content/misc/secrets-
store-csi/aws-provider-installer.yaml
```

5. Vérifiez que les deux Daemonsets s'exécutent en exécutant la commande suivante:

```
$ oc -n csi-secrets-store get ds \
    csi-secrets-store-provider-aws \
    csi-secrets-store-driver-secrets-store-csi-driver
```

6. Étiqueter le pilote CSI Secrets Store pour permettre l'utilisation avec le profil de sécurité de pod restreint en exécutant la commande suivante:

```
$ oc label csidriver.storage.k8s.io/secrets-store.csi.k8s.io security.openshift.io/csi-ephemeral-
volume-profile=restricted
```

11.3. CRÉER UN SECRET ET DES POLITIQUES D'ACCÈS IAM

1. Créez un secret dans Secrets Manager en exécutant la commande suivante:

```
$ SECRET_ARN=$(aws --region "$REGION" secretsmanager create-secret \
    --name MySecret --secret-string \
```

```
    '{"username":"shadowman", "password":"hunter2"}' \
    --query ARN --output text); echo $SECRET_ARN
```

2. Créez un document de stratégie d'accès IAM en exécutant la commande suivante:

```
$ cat << EOF > policy.json
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "secretsmanager:GetSecretValue",
      "secretsmanager:DescribeSecret"
    ],
    "Resource": ["$SECRET_ARN"]
    }]
}
EOF
```

3. Créez une stratégie d'accès IAM en exécutant la commande suivante:

```
$ POLICY_ARN=$(aws --region "$REGION" --query Policy.Arn \
--output text iam create-policy \
--policy-name openshift-access-to-mysecret-policy \
--policy-document file://policy.json); echo $POLICY_ARN
```

4. Créez un document de stratégie de confiance IAM Role en exécutant la commande suivante:

NOTE

La stratégie de confiance est verrouillée sur le compte de service par défaut d'un espace de noms que vous créez plus tard dans ce processus.

```
$ cat <<EOF > trust-policy.json
{
  "Version": "2012-10-17",
  "Statement": [
  {
  "Effect": "Allow",
  "Condition": {
    "StringEquals" : {
      "${OIDC_ENDPOINT}:sub": ["system:serviceaccount:my-application:default"]
    }
   },
   "Principal": {
      "Federated": "arn:aws:iam::$AWS_ACCOUNT_ID:oidc-provider/${OIDC_ENDPOINT}"
   },
   "Action": "sts:AssumeRoleWithWebIdentity"
   }
   ]
}
EOF
```

5. Créez un rôle IAM en exécutant la commande suivante:

```
$ ROLE_ARN=$(aws iam create-role --role-name openshift-access-to-mysecret \
--assume-role-policy-document file://trust-policy.json \
--query Role.Arn --output text); echo $ROLE_ARN
```

6. Attachez le rôle à la politique en exécutant la commande suivante:

```
$ aws iam attach-role-policy --role-name openshift-access-to-mysecret \
   --policy-arn $POLICY_ARN
```

11.4. CRÉER UNE APPLICATION POUR UTILISER CE SECRET

1. Créez un projet OpenShift en exécutant la commande suivante:

```
$ oc new-project my-application
```

2. Annotez le compte de service par défaut pour utiliser le rôle STS en exécutant la commande suivante:

```
$ oc annotate -n my-application serviceaccount default \
   eks.amazonaws.com/role-arn=$ROLE_ARN
```

3. Créez une classe de fournisseur secret pour accéder à notre secret en exécutant la commande suivante:

```
$ cat << EOF | oc apply -f -
apiVersion: secrets-store.csi.x-k8s.io/v1
kind: SecretProviderClass
metadata:
  name: my-application-aws-secrets
spec:
  provider: aws
  parameters:
    objects: |
      - objectName: "MySecret"
        objectType: "secretsmanager"
EOF
```

4. Créez un déploiement en utilisant notre secret dans la commande suivante:

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: Pod
metadata:
  name: my-application
  labels:
    app: my-application
spec:
  volumes:
  - name: secrets-store-inline
    csi:
      driver: secrets-store.csi.k8s.io
      readOnly: true
      volumeAttributes:
```

```
        secretProviderClass: "my-application-aws-secrets"
  containers:
  - name: my-application-deployment
    image: k8s.gcr.io/e2e-test-images/busybox:1.29
    command:
      - "/bin/sleep"
      - "10000"
    volumeMounts:
    - name: secrets-store-inline
      mountPath: "/mnt/secrets-store"
      readOnly: true
EOF
```

5. Assurez-vous que la gousse a le secret monté en exécutant la commande suivante:

```
$ oc exec -it my-application -- cat /mnt/secrets-store/MySecret
```

11.5. FAIRE LE MÉNAGE

1. Effacer l'application en exécutant la commande suivante:

```
$ oc delete project my-application
```

2. Effacer le pilote csi store secret en exécutant la commande suivante:

```
$ helm delete -n csi-secrets-store csi-secrets-store-driver
```

3. Effacer les contraintes du contexte de sécurité en exécutant la commande suivante:

```
$ oc adm policy remove-scc-from-user privileged \
    system:serviceaccount:csi-secrets-store:secrets-store-csi-driver; oc adm policy remove-
scc-from-user privileged \
    system:serviceaccount:csi-secrets-store:csi-secrets-store-provider-aws
```

4. Effacer le fournisseur AWS en exécutant la commande suivante:

```
$ oc -n csi-secrets-store delete -f \
https://raw.githubusercontent.com/rh-mobb/documentation/main/content/misc/secrets-store-
csi/aws-provider-installer.yaml
```

5. Effacer les rôles et les politiques AWS en exécutant la commande suivante:

```
$ aws iam detach-role-policy --role-name openshift-access-to-mysecret \
    --policy-arn $POLICY_ARN; aws iam delete-role --role-name openshift-access-to-
mysecret; aws iam delete-policy --policy-arn $POLICY_ARN
```

6. Effacer le secret Secrets Manager en exécutant la commande suivante:

```
$ aws secretsmanager --region $REGION delete-secret --secret-id $SECRET_ARN
```

CHAPITRE 12. TUTORIEL: UTILISATION DES CONTRÔLEURS AWS POUR KUBERNETES SUR ROSA

AWS Controllers for Kubernetes (ACK) vous permet de définir et d'utiliser les ressources de service AWS directement à partir du service Red Hat OpenShift sur AWS (ROSA). Avec ACK, vous pouvez profiter des services gérés par AWS pour vos applications sans avoir besoin de définir des ressources en dehors du cluster ou d'exécuter des services qui fournissent des capacités de support telles que des bases de données ou des files d'attente de messages dans le cluster.

Il est possible d'installer différents opérateurs ACK directement depuis OperatorHub. Cela facilite le démarrage et l'utilisation des opérateurs avec vos applications. Ce contrôleur est un composant du projet AWS Controller for Kubernetes, qui est actuellement en prévisualisation des développeurs.

Faites appel à ce tutoriel pour déployer l'opérateur ACK S3. Il est également possible de l'adapter à tout autre opérateur ACK dans OperatorHub de votre cluster.

12.1. CONDITIONS PRÉALABLES

- Le cluster ROSA
- Compte utilisateur avec des privilèges cluster-admin
- Le CLI OpenShift (oc)
- Amazon Web Services (AWS) CLI (aws)

12.2. CONFIGURATION DE VOTRE ENVIRONNEMENT

1. Configurez les variables d'environnement suivantes, en modifiant le nom du cluster en fonction de votre cluster:

```
$ export CLUSTER_NAME=$(oc get infrastructure cluster -o=jsonpath="
{.status.infrastructureName}"  | sed 's/-[a-z0-9]\{5\}$//')
$ export REGION=$(rosa describe cluster -c ${ROSA_CLUSTER_NAME} --output json | jq -r
.region.id)
$ export OIDC_ENDPOINT=$(oc get authentication.config.openshift.io cluster -o json | jq -r
.spec.serviceAccountIssuer | sed  's|^https://||')
$ export AWS_ACCOUNT_ID=`aws sts get-caller-identity --query Account --output text`
$ export ACK_SERVICE=s3
$ export ACK_SERVICE_ACCOUNT=ack-${ACK_SERVICE}-controller
$ export POLICY_ARN=arn:aws:iam::aws:policy/AmazonS3FullAccess
$ export AWS_PAGER=""
$ export SCRATCH="/tmp/${ROSA_CLUSTER_NAME}/ack"
$ mkdir -p ${SCRATCH}
```

2. Assurez-vous que tous les champs sortent correctement avant de passer à la section suivante:

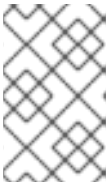
```
$ echo "Cluster: ${ROSA_CLUSTER_NAME}, Region: ${REGION}, OIDC Endpoint:
${OIDC_ENDPOINT}, AWS Account ID: ${AWS_ACCOUNT_ID}"
```

12.3. LA PRÉPARATION DE VOTRE COMPTE AWS

1. Créer une politique de confiance AWS Identity Access Management (IAM) pour l'opérateur ACK:

```
$ cat <<EOF > "${SCRATCH}/trust-policy.json"
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Condition": {
   "StringEquals" : {
     "${OIDC_ENDPOINT}:sub": "system:serviceaccount:ack-
system:${ACK_SERVICE_ACCOUNT}"
   }
 },
 "Principal": {
   "Federated": "arn:aws:iam::$AWS_ACCOUNT_ID:oidc-provider/${OIDC_ENDPOINT}"
 },
 "Action": "sts:AssumeRoleWithWebIdentity"
 }
 ]
}
EOF
```

2. Créer un rôle AWS IAM pour l'opérateur ACK à assumer avec la politique AmazonS3FullAccess jointe:



NOTE

La politique recommandée dans le référentiel GitHub de chaque projet, par exemple https://github.com/aws-controllers-k8s/s3-controller/blob/main/config/iam/recommended-policy-arn.

```
$ ROLE_ARN=$(aws iam create-role --role-name "ack-${ACK_SERVICE}-controller" \
   --assume-role-policy-document "file://${SCRATCH}/trust-policy.json" \
   --query Role.Arn --output text)
$ echo $ROLE_ARN

$ aws iam attach-role-policy --role-name "ack-${ACK_SERVICE}-controller" \
     --policy-arn ${POLICY_ARN}
```

12.4. INSTALLATION DU CONTRÔLEUR ACK S3

1. Créer un projet pour installer l'opérateur ACK S3 dans:

```
$ oc new-project ack-system
```

2. Créer un fichier avec la configuration ACK S3 Operator:

NOTE

ACK_WATCH_NAMESPACE est délibérément laissé vide afin que le contrôleur puisse correctement regarder tous les espaces de noms dans le cluster.

```
$ cat <<EOF > "${SCRATCH}/config.txt"
ACK_ENABLE_DEVELOPMENT_LOGGING=true
ACK_LOG_LEVEL=debug
ACK_WATCH_NAMESPACE=
AWS_REGION=${REGION}
AWS_ENDPOINT_URL=
ACK_RESOURCE_TAGS=${CLUSTER_NAME}
ENABLE_LEADER_ELECTION=true
LEADER_ELECTION_NAMESPACE=
EOF
```

3. À partir de l'étape précédente, utilisez le fichier pour créer un ConfigMap:

```
$ oc -n ack-system create configmap \
  --from-env-file=${SCRATCH}/config.txt ack-${ACK_SERVICE}-user-config
```

4. Installez l'opérateur ACK S3 de OperatorHub:

```
$ cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: ack-${ACK_SERVICE}-controller
  namespace: ack-system
spec:
  upgradeStrategy: Default
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: ack-${ACK_SERVICE}-controller
  namespace: ack-system
spec:
  channel: alpha
  installPlanApproval: Automatic
  name: ack-${ACK_SERVICE}-controller
  source: community-operators
  sourceNamespace: openshift-marketplace
EOF
```

5. Annoter le compte de service ACK S3 Operator avec le rôle AWS IAM pour assumer et redémarrer le déploiement:

```
$ oc -n ack-system annotate serviceaccount ${ACK_SERVICE_ACCOUNT} \
  eks.amazonaws.com/role-arn=${ROLE_ARN} && \
  oc -n ack-system rollout restart deployment ack-${ACK_SERVICE}-controller
```

6. Assurez-vous que l'opérateur ACK S3 est en cours d'exécution:

```
$ oc -n ack-system get pods
```

Exemple de sortie

```
NAME                                READY   STATUS    RESTARTS   AGE
ack-s3-controller-585f6775db-s4lfz  1/1     Running   0          51s
```

12.5. LA VALIDATION DU DÉPLOIEMENT

1. Déployez une ressource S3 bucket:

```
$ cat << EOF | oc apply -f -
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: ${CLUSTER-NAME}-bucket
  namespace: ack-system
spec:
  name: ${CLUSTER-NAME}-bucket
EOF
```

2. Assurez-vous que le seau S3 a été créé dans AWS:

```
$ aws s3 ls | grep ${CLUSTER_NAME}-bucket
```

Exemple de sortie

```
2023-10-04 14:51:45 mrmc-test-maz-bucket
```

12.6. LE NETTOYAGE

1. Effacer la ressource S3 bucket:

```
$ oc -n ack-system delete bucket.s3.services.k8s.aws/${CLUSTER-NAME}-bucket
```

2. Effacer l'opérateur ACK S3 et les rôles AWS IAM:

```
$ oc -n ack-system delete subscription ack-${ACK_SERVICE}-controller
$ aws iam detach-role-policy \
  --role-name "ack-${ACK_SERVICE}-controller" \
  --policy-arn ${POLICY_ARN}
$ aws iam delete-role \
  --role-name "ack-${ACK_SERVICE}-controller"
```

3. Effacer le projet ack-system:

```
$ oc delete project ack-system
```

CHAPITRE 13. TUTORIEL : DÉPLOIEMENT DE L'OPÉRATEUR DNS EXTERNE SUR ROSA

L'opérateur DNS externe déploie et gère ExternalDNS pour fournir la résolution de nom pour les services et les itinéraires du fournisseur DNS externe, comme Amazon Route 53, vers Red Hat OpenShift Service sur les clusters AWS (ROSA). Dans ce tutoriel, nous allons déployer et configurer l'opérateur DNS externe avec un contrôleur d'entrée secondaire pour gérer les enregistrements DNS dans Amazon Route 53.

IMPORTANT

L'opérateur DNS externe ne prend pas en charge STS en utilisant les rôles IAM pour les comptes de service (IRSA) et utilise plutôt des informations d'identification de longue durée de gestion de l'accès à l'identité (IAM). Ce tutoriel sera mis à jour lorsque l'opérateur prend en charge STS.

13.1. CONDITIONS PRÉALABLES

- A ROSA Classic cluster

NOTE

Le ROSA avec HCP n'est pas pris en charge pour le moment.

- Compte utilisateur avec des privilèges cluster-admin
- Le CLI OpenShift (oc)
- Amazon Web Services (AWS) CLI (aws)
- Domaine unique, tel que apps.example.com
- Amazon Route 53 zone hébergée par le public pour le domaine ci-dessus

13.2. CONFIGURATION DE VOTRE ENVIRONNEMENT

1. Configurez les variables d'environnement suivantes:

```
$ export DOMAIN=<apps.example.com> 1
$ export AWS_PAGER=""
$ export CLUSTER=$(oc get infrastructure cluster -o=jsonpath="{.status.infrastructureName}"
| sed 's/-[a-z0-9]\{5\}$//')
$ export REGION=$(oc get infrastructure cluster -o=jsonpath="
{.status.platformStatus.aws.region}")
$ export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
$ export SCRATCH="/tmp/${CLUSTER}/external-dns"
$ mkdir -p ${SCRATCH}
```

1 Le remplacement par le domaine personnalisé que vous souhaitez utiliser pour IngressController.

2. Assurez-vous que tous les champs sortent correctement avant de passer à la section suivante:

```
$ echo "Cluster: ${CLUSTER}, Region: ${REGION}, AWS Account ID:
${AWS_ACCOUNT_ID}"
```

NOTE

La sortie "Cluster" de la commande précédente peut être le nom de votre cluster, l'ID interne de votre cluster ou le préfixe de domaine du cluster. Lorsque vous préférez utiliser un autre identifiant, vous pouvez définir manuellement cette valeur en exécutant la commande suivante:

```
$ export CLUSTER=my-custom-value
```

13.3. CONFIGURATION DU CONTRÔLEUR D'ENTRÉE SECONDAIRE

Employez la procédure suivante pour déployer un contrôleur d'entrée secondaire à l'aide d'un domaine personnalisé.

Conditions préalables

- Domaine unique, tel que apps.example.com
- Certificat Wildcard ou SAN TLS configuré avec le domaine personnalisé sélectionné ci-dessus (CN=*.apps.example.com)

Procédure

1. Créez un nouveau secret TLS à partir d'une clé privée et d'un certificat public, où fullchain.pem est votre chaîne de certificats wildcard complète (y compris les intermédiaires) et privkey.pem est la clé privée de votre certificat wildcard:

```
$ oc -n openshift-ingress create secret tls external-dns-tls --cert=fullchain.pem --
key=privkey.pem
```

2. Créer une nouvelle ressource IngressController:

```
$ cat << EOF | oc apply -f -
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: external-dns-ingress
  namespace: openshift-ingress-operator
spec:
  domain: ${DOMAIN}
  defaultCertificate:
    name: external-dns-tls
  endpointPublishingStrategy:
    loadBalancer:
      dnsManagementPolicy: Unmanaged
      providerParameters:
        aws:
          type: NLB
        type: AWS
```

```
    scope: External
  type: LoadBalancerService
EOF
```

AVERTISSEMENT

Cet exemple IngressController créera un balanceur de charge réseau accessible à Internet (NLB) dans votre compte AWS. Afin de fournir une NLB interne, définissez le paramètre .spec.endpointPublishingStrategy.loadBalancer.scope sur Interne avant de créer la ressource IngressController.

3. Assurez-vous que votre domaine personnalisé IngressController a créé avec succès un équilibreur de charge externe:

```
$ oc -n openshift-ingress get service/router-external-dns-ingress
```

Exemple de sortie

```
NAME                          TYPE           CLUSTER-IP      EXTERNAL-IP
PORT(S)                       AGE
router-external-dns-ingress   LoadBalancer   172.30.71.250
a4838bb991c6748439134ab89f132a43-aeae124077b50c01.elb.us-east-1.amazonaws.com
80:32227/TCP,443:30310/TCP    43s
```

13.4. LA PRÉPARATION DE VOTRE COMPTE AWS

1. Découvrez l'ID de zone hébergée par Amazon Route 53:

```
$ export ZONE_ID=$(aws route53 list-hosted-zones-by-name --output json \
  --dns-name "${DOMAIN}." --query 'HostedZones[0]'.Id --out text | sed 's/\/hostedzone\///')
```

2. Élaborer un document avec les modifications DNS nécessaires pour activer la résolution DNS pour le domaine canonique du contrôleur Ingress:

```
$ NLB_HOST=$(oc -n openshift-ingress get service/router-external-dns-ingress -ojsonpath="
{.status.loadBalancer.ingress[0].hostname}")
$ cat << EOF > "${SCRATCH}/create-cname.json"
{
  "Comment":"Add CNAME to ingress controller canonical domain",
  "Changes":[{
      "Action":"CREATE",
      "ResourceRecordSet":{
        "Name": "router-external-dns-ingress.${DOMAIN}",
      "Type":"CNAME",
      "TTL":30,
      "ResourceRecords":[{
        "Value": "${NLB_HOST}"
```

```
      }]
    }
  }]
}
EOF
```

L'opérateur DNS externe utilise ce domaine canonique comme cible pour les enregistrements CNAME.

3. Envoyez vos modifications à Amazon Route 53 pour propagation:

```
aws route53 change-resource-record-sets \
  --hosted-zone-id ${ZONE_ID} \
  --change-batch file://${SCRATCH}/create-cname.json
```

4. Créez un document AWS IAM Policy qui permet à l'opérateur DNS externe de mettre à jour uniquement la zone hébergée par le domaine personnalisé:

```
$ cat << EOF > "${SCRATCH}/external-dns-policy.json"
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "route53:ChangeResourceRecordSets"
      ],
      "Resource": [
        "arn:aws:route53:::hostedzone/${ZONE_ID}"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "route53:ListHostedZones",
        "route53:ListResourceRecordSets"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
EOF
```

5. Créer un utilisateur AWS IAM:

```
$ aws iam create-user --user-name "${CLUSTER}-external-dns-operator"
```

6. Joindre la politique:

```
$ aws iam attach-user-policy --user-name "${CLUSTER}-external-dns-operator" --policy-arn
$POLICY_ARN
```

NOTE

Cela sera changé en STS en utilisant l'IRSA à l'avenir.

7. Créer des clés AWS pour l'utilisateur IAM:

```
$ SECRET_ACCESS_KEY=$(aws iam create-access-key --user-name "${CLUSTER}-
external-dns-operator")
```

8. Créer des informations d'identification statiques:

```
$ cat << EOF > "${SCRATCH}/credentials"
[default]
aws_access_key_id = $(echo $SECRET_ACCESS_KEY | jq -r '.AccessKey.AccessKeyId')
aws_secret_access_key = $(echo $SECRET_ACCESS_KEY | jq -r
'.AccessKey.SecretAccessKey')
EOF
```

13.5. INSTALLATION DE L'OPÉRATEUR DNS EXTERNE

1. Créer un nouveau projet:

```
$ oc new-project external-dns-operator
```

2. Installez l'opérateur DNS externe de OperatorHub:

```
$ cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: external-dns-group
  namespace: external-dns-operator
spec:
  targetNamespaces:
  - external-dns-operator
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: external-dns-operator
  namespace: external-dns-operator
spec:
  channel: stable-v1.1
  installPlanApproval: Automatic
  name: external-dns-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
EOF
```

3. Attendez que l'opérateur DNS externe soit en cours d'exécution:

```
$ oc rollout status deploy external-dns-operator --timeout=300s
```

4. Créez un secret à partir des identifiants d'utilisateur AWS IAM:

```
$ oc -n external-dns-operator create secret generic external-dns \
  --from-file "${SCRATCH}/credentials"
```

5. Déployez le contrôleur ExternalDNS:

```
$ cat << EOF | oc apply -f -
apiVersion: externaldns.olm.openshift.io/v1beta1
kind: ExternalDNS
metadata:
  name: ${DOMAIN}
spec:
  domains:
    - filterType: Include
      matchType: Exact
      name: ${DOMAIN}
  provider:
    aws:
      credentials:
        name: external-dns
    type: AWS
  source:
    openshiftRouteOptions:
      routerName: external-dns-ingress
    type: OpenShiftRoute
  zones:
    - ${ZONE_ID}
EOF
```

6. Attendez que le contrôleur soit en cours d'exécution:

```
$ oc rollout status deploy external-dns-${DOMAIN} --timeout=300s
```

13.6. DÉPLOIEMENT D'UNE APPLICATION D'ÉCHANTILLON

Depuis que le contrôleur ExternalDNS est en cours d'exécution, vous pouvez déployer un exemple d'application pour confirmer que le domaine personnalisé est configuré et fiable lorsque vous exposez un nouvel itinéraire.

1. Créez un nouveau projet pour votre exemple d'application:

```
$ oc new-project hello-world
```

2. Déployez une application hello world:

```
$ oc new-app -n hello-world --image=docker.io/openshift/hello-openshift
```

3. Créer un itinéraire pour l'application spécifiant votre nom de domaine personnalisé:

```
$ oc -n hello-world create route edge --service=hello-openshift hello-openshift-tls \
--hostname hello-openshift.${DOMAIN}
```

4. Vérifiez si l'enregistrement DNS a été créé automatiquement par ExternalDNS:

NOTE

Il peut prendre quelques minutes pour que l'enregistrement apparaisse sur Amazon Route 53.

```
$ aws route53 list-resource-record-sets --hosted-zone-id ${ZONE_ID} \
  --query "ResourceRecordSets[?Type == 'CNAME']" | grep hello-openshift
```

5. Facultatif: Vous pouvez également afficher les enregistrements TXT qui indiquent qu'ils ont été créés par ExternalDNS:

```
$ aws route53 list-resource-record-sets --hosted-zone-id ${ZONE_ID} \
  --query "ResourceRecordSets[?Type == 'TXT']" | grep ${DOMAIN}
```

6. Bouclez l'enregistrement DNS nouvellement créé à votre exemple d'application pour vérifier l'application hello world est accessible:

```
$ curl https://hello-openshift.${DOMAIN}
```

Exemple de sortie

```
Hello OpenShift!
```

CHAPITRE 14. DIDACTICIEL: ÉMETTRE DYNAMIQUEMENT DES CERTIFICATS EN UTILISANT L'OPÉRATEUR DE CERT-MANAGER SUR ROSA

Bien que les certificats wildcard offrent une simplicité en sécurisant tous les sous-domaines de premier niveau d'un domaine donné avec un seul certificat, d'autres cas d'utilisation peuvent nécessiter l'utilisation de certificats individuels par domaine.

Apprenez à utiliser l'opérateur gestionnaire de certificat pour Red Hat OpenShift et Let's Encrypt pour émettre dynamiquement des certificats pour les itinéraires créés à l'aide d'un domaine personnalisé.

14.1. CONDITIONS PRÉALABLES

- Cluster ROSA (HCP ou Classic)
- Compte utilisateur avec des privilèges cluster-admin
- Le CLI OpenShift (oc)
- Amazon Web Services (AWS) CLI (aws)
- Domaine unique, tel que *.apps.example.com
- Amazon Route 53 zone hébergée par le public pour le domaine ci-dessus

14.2. CONFIGURATION DE VOTRE ENVIRONNEMENT

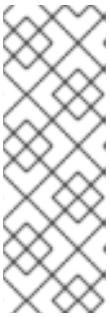
1. Configurez les variables d'environnement suivantes:

```
$ export DOMAIN=apps.example.com 1
$ export EMAIL=email@example.com 2
$ export AWS_PAGER=""
$ export CLUSTER=$(oc get infrastructure cluster -o=jsonpath="{.status.infrastructureName}"
| sed 's/-[a-z0-9]\{5\}$//')
$ export OIDC_ENDPOINT=$(oc get authentication.config.openshift.io cluster -o json | jq -r
.spec.serviceAccountIssuer | sed  's|^https://||')
$ export REGION=$(oc get infrastructure cluster -o=jsonpath="
{.status.platformStatus.aws.region}")
$ export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
$ export SCRATCH="/tmp/${CLUSTER}/dynamic-certs"
$ mkdir -p ${SCRATCH}
```

1. Le remplacement par le domaine personnalisé que vous souhaitez utiliser pour IngressController.
2. Il suffit de remplacer par l'e-mail que vous souhaitez crypter pour envoyer des notifications sur vos certificats.

2. Assurez-vous que tous les champs sortent correctement avant de passer à la section suivante:

```
$ echo "Cluster: ${CLUSTER}, Region: ${REGION}, OIDC Endpoint: ${OIDC_ENDPOINT},
AWS Account ID: ${AWS_ACCOUNT_ID}"
```



NOTE

La sortie "Cluster" de la commande précédente peut être le nom de votre cluster, l'ID interne de votre cluster ou le préfixe de domaine du cluster. Lorsque vous préférez utiliser un autre identifiant, vous pouvez définir manuellement cette valeur en exécutant la commande suivante:

```
$ export CLUSTER=my-custom-value
```

14.3. LA PRÉPARATION DE VOTRE COMPTE AWS

Lorsque le gestionnaire de certificat demande un certificat auprès de Let's Encrypt (ou d'un autre émetteur de certificat ACME), Let's Encrypt serveurs valide que vous contrôlez le nom de domaine dans ce certificat à l'aide de défis. Dans ce tutoriel, vous utilisez un défi DNS-01 qui prouve que vous contrôlez le DNS pour votre nom de domaine en plaçant une valeur spécifique dans un enregistrement TXT sous ce nom de domaine. Cela se fait automatiquement par cert-manager. Afin d'autoriser le gestionnaire de licence à modifier la zone hébergée par Amazon Route 53 pour votre domaine, vous devez créer un rôle de gestion d'accès à l'identité (IAM) avec des autorisations de stratégie spécifiques et une relation de confiance pour permettre l'accès au pod.

La zone hébergée publique utilisée dans ce tutoriel se trouve dans le même compte AWS que le cluster ROSA. Dans le cas où votre zone hébergée publique se trouve dans un autre compte, quelques étapes supplémentaires pour l'accès au compte croisé sont nécessaires.

1. Découvrez l'ID de zone hébergée par Amazon Route 53:

NOTE

Cette commande recherche une zone hébergée publique qui correspond au domaine personnalisé que vous avez spécifié plus tôt sous le nom de variable d'environnement DOMAIN. Il est possible de spécifier manuellement la zone hébergée publique Amazon Route 53 en exécutant l'exportation ZONE_ID=<zone_ID>, en remplaçant <zone_ID> par votre identifiant de zone hébergée publique Amazon Route 53 spécifique.

```
$ export ZONE_ID=$(aws route53 list-hosted-zones-by-name --output json \
  --dns-name "${DOMAIN}." --query 'HostedZones[0]'.Id --out text | sed 's/\/hostedzone\///')
```

2. Créez un document de stratégie AWS IAM pour l'opérateur de cert-manager qui offre la possibilité de mettre à jour uniquement la zone hébergée publique spécifiée:

```
$ cat <<EOF > "${SCRATCH}/cert-manager-policy.json"
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "route53:GetChange",
      "Resource": "arn:aws:route53:::change/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "route53:ChangeResourceRecordSets",
```

```
        "route53:ListResourceRecordSets"
      ],
      "Resource": "arn:aws:route53:::hostedzone/${ZONE_ID}"
    },
    {
      "Effect": "Allow",
      "Action": "route53:ListHostedZonesByName",
      "Resource": "*"
    }
  ]
}
EOF
```

3. Créez la stratégie IAM à l'aide du fichier que vous avez créé à l'étape précédente:

```
$ POLICY_ARN=$(aws iam create-policy --policy-name "${CLUSTER}-cert-manager-policy" \
  --policy-document file://${SCRATCH}/cert-manager-policy.json \
  --query 'Policy.Arn' --output text)
```

4. Créer une politique de confiance AWS IAM pour l'opérateur de cert-manager:

```
$ cat <<EOF > "${SCRATCH}/trust-policy.json"
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Condition": {
   "StringEquals" : {
     "${OIDC_ENDPOINT}:sub": "system:serviceaccount:cert-manager:cert-manager"
   }
 },
 "Principal": {
   "Federated": "arn:aws:iam::$AWS_ACCOUNT_ID:oidc-provider/${OIDC_ENDPOINT}"
 },
 "Action": "sts:AssumeRoleWithWebIdentity"
 }
 ]
}
EOF
```

5. Créer un rôle IAM pour l'opérateur de cert-manager en utilisant la politique de confiance que vous avez créée à l'étape précédente:

```
$ ROLE_ARN=$(aws iam create-role --role-name "${CLUSTER}-cert-manager-operator" \
   --assume-role-policy-document "file://${SCRATCH}/trust-policy.json" \
   --query Role.Arn --output text)
```

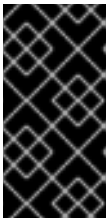
6. Joindre la politique de permissions au rôle:

```
$ aws iam attach-role-policy --role-name "${CLUSTER}-cert-manager-operator" \
  --policy-arn ${POLICY_ARN}
```

14.4. INSTALLATION DE L'OPÉRATEUR DE CERT-MANAGER

1. Créer un projet pour installer l'opérateur cert-manager dans:

```
$ oc new-project cert-manager-operator
```



IMPORTANT

Dans votre cluster, n'essayez pas d'utiliser plus d'un opérateur de cert-manager. Dans le cas où vous avez un opérateur de cert-manager de communauté installé dans votre cluster, vous devez le désinstaller avant d'installer l'opérateur cert-manager pour Red Hat OpenShift.

2. Installez l'opérateur de cert-manager pour Red Hat OpenShift:

```
$ cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: openshift-cert-manager-operator-group
  namespace: cert-manager-operator
spec:
  targetNamespaces:
  - cert-manager-operator
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: openshift-cert-manager-operator
  namespace: cert-manager-operator
spec:
  channel: stable-v1
  installPlanApproval: Automatic
  name: openshift-cert-manager-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
EOF
```

NOTE

Il faut quelques minutes pour que cet opérateur installe et complète sa configuration.

3. Assurez-vous que l'opérateur de cert-manager est en cours d'exécution:

```
$ oc -n cert-manager-operator get pods
```

Exemple de sortie

```
NAME                                         READY   STATUS    RESTARTS   AGE
cert-manager-operator-controller-manager-84b8799db5-gv8mx   2/2     Running   0          12s
```

4. Annotez le compte de service utilisé par les pods de cert-manager avec le rôle AWS IAM que vous avez créé précédemment:

```
$ oc -n cert-manager annotate serviceaccount cert-manager eks.amazonaws.com/role-arn=${ROLE_ARN}
```

5. Redémarrez la pod de contrôleur de cert-manager existant en exécutant la commande suivante:

```
$ oc -n cert-manager delete pods -l app.kubernetes.io/name=cert-manager
```

6. Corrigez la configuration de l'opérateur pour utiliser des noms externes pour éviter les problèmes de résolution des défis DNS-01:

```
$ oc patch certmanager.operator.openshift.io/cluster --type merge \
  -p '{"spec":{"controllerConfig":{"overrideArgs":["--dns01-recursive-nameservers-only","--dns01-recursive-nameservers=1.1.1.1:53"]}}}'
```

7. Créez une ressource ClusterIssuer pour utiliser Let's Encrypt en exécutant la commande suivante:

```
$ cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: letsencrypt-production
spec:
  acme:
    server: https://acme-v02.api.letsencrypt.org/directory
    email: ${EMAIL}
    # This key doesn't exist, cert-manager creates it
    privateKeySecretRef:
      name: prod-letsencrypt-issuer-account-key
    solvers:
    - dns01:
        route53:
          hostedZoneID: ${ZONE_ID}
          region: ${REGION}
          secretAccessKeySecretRef:
            name: ''
EOF
```

8. La ressource ClusterIssuer est prête:

```
$ oc get clusterissuer.cert-manager.io/letsencrypt-production
```

Exemple de sortie

```
NAME                  READY   AGE
letsencrypt-production   True    47s
```

14.5. CRÉATION D'UN DOMAINE PERSONNALISÉ INGRESS CONTROLLER

1. Créer et configurer une ressource de certificat pour fournir un certificat pour le domaine personnalisé Ingress Controller:

NOTE

L'exemple suivant utilise un certificat de domaine unique. Les certificats SAN et wildcard sont également pris en charge.

```
$ cat << EOF | oc apply -f -
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: custom-domain-ingress-cert
  namespace: openshift-ingress
spec:
  secretName: custom-domain-ingress-cert-tls
  issuerRef:
     name: letsencrypt-production
     kind: ClusterIssuer
  commonName: "${DOMAIN}"
  dnsNames:
  - "${DOMAIN}"
EOF
```

2. B) Vérifier que le certificat a été délivré:

NOTE

Il faut quelques minutes pour que ce certificat soit délivré par Let's Encrypt. Lorsque cela prend plus de 5 minutes, exécutez oc -n openshift-ingress décrit certificate.cert-manager.io/custom-domain-ingress-cert pour voir tous les problèmes signalés par cert-manager.

```
$ oc -n openshift-ingress get certificate.cert-manager.io/custom-domain-ingress-cert
```

Exemple de sortie

```
NAME                          READY   SECRET                          AGE
custom-domain-ingress-cert    True    custom-domain-ingress-cert-tls   9m53s
```

3. Créer une nouvelle ressource IngressController:

```
$ cat << EOF | oc apply -f -
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: custom-domain-ingress
  namespace: openshift-ingress-operator
spec:
```

```
  domain: ${DOMAIN}
  defaultCertificate:
    name: custom-domain-ingress-cert-tls
  endpointPublishingStrategy:
    loadBalancer:
      dnsManagementPolicy: Unmanaged
      providerParameters:
        aws:
          type: NLB
        type: AWS
      scope: External
    type: LoadBalancerService
EOF
```

AVERTISSEMENT

Cet exemple IngressController créera un balanceur de charge réseau accessible à Internet (NLB) dans votre compte AWS. Afin de fournir une NLB interne, définissez le paramètre .spec.endpointPublishingStrategy.loadBalancer.scope sur Interne avant de créer la ressource IngressController.

4. Assurez-vous que votre domaine personnalisé IngressController a créé avec succès un équilibreur de charge externe:

```
$ oc -n openshift-ingress get service/router-custom-domain-ingress
```

Exemple de sortie

```
NAME                         TYPE           CLUSTER-IP      EXTERNAL-IP
PORT(S)                      AGE
router-custom-domain-ingress   LoadBalancer   172.30.174.34
a309962c3bd6e42c08cadb9202eca683-1f5bbb64a1f1ec65.elb.us-east-1.amazonaws.com
80:31342/TCP,443:31821/TCP   7m28s
```

5. Créez un document avec les modifications DNS nécessaires pour activer la résolution DNS pour votre domaine personnalisé Ingress Controller:

```
$ INGRESS=$(oc -n openshift-ingress get service/router-custom-domain-ingress -
ojsonpath="{.status.loadBalancer.ingress[0].hostname}")
$ cat << EOF > "${SCRATCH}/create-cname.json"
{
  "Comment":"Add CNAME to custom domain endpoint",
  "Changes":[{
      "Action":"CREATE",
      "ResourceRecordSet":{
        "Name": "*.${DOMAIN}",
      "Type":"CNAME",
      "TTL":30,
```

```
      "ResourceRecords":[{
        "Value": "${INGRESS}"
      }]
    }
  }]
}
EOF
```

6. Envoyez vos modifications à Amazon Route 53 pour propagation:

```
$ aws route53 change-resource-record-sets \
  --hosted-zone-id ${ZONE_ID} \
  --change-batch file://${SCRATCH}/create-cname.json
```

NOTE

Bien que l'enregistrement wildcard CNAME évite la nécessité de créer un nouvel enregistrement pour chaque nouvelle application que vous déployez à l'aide du contrôleur d'Ingress de domaine personnalisé, le certificat que chacune de ces applications utilise n'est pas un certificat générique.

14.6. CONFIGURATION DES CERTIFICATS DYNAMIQUES POUR DES ITINÉRAIRES DE DOMAINE PERSONNALISÉS

Désormais, vous pouvez exposer des applications de cluster sur n'importe quel sous-domaine de premier niveau du domaine spécifié, mais la connexion ne sera pas sécurisée avec un certificat TLS correspondant au domaine de l'application. Afin de s'assurer que ces applications de cluster ont des certificats valides pour chaque nom de domaine, configurez cert-Manager pour délivrer dynamiquement un certificat à chaque nouvelle route créée sous ce domaine.

1. Créer les ressources OpenShift nécessaires pour gérer les certificats pour les routes OpenShift. Cette étape crée un nouveau déploiement (et donc un pod) qui surveille spécifiquement les routes annotées dans le cluster. Lorsque les annotations de type émetteur et de nom d'émetteur sont trouvées dans une nouvelle route, il demande à l'émetteur (ClusterIssuer dans ce cas) un nouveau certificat qui est unique à cette route et qui honorera le nom d'hôte spécifié lors de la création de l'itinéraire.

NOTE

Dans le cas où le cluster n'a pas accès à GitHub, vous pouvez enregistrer les contenus bruts localement et exécuter oc application -f localfilename.yaml -n cert-manager.

```
$ oc -n cert-manager apply -f https://github.com/cert-manager/openshift-
routes/releases/latest/download/cert-manager-openshift-routes.yaml
```

Les ressources supplémentaires d'OpenShift suivantes sont également créées dans cette étape:

- ClusterRole - accorde des autorisations pour regarder et mettre à jour les itinéraires à travers le cluster
- Compte ServiceAccount - utilise les autorisations pour exécuter le pod nouvellement créé

- ClusterRoleBinding - lie ces deux ressources

2. Assurez-vous que la nouvelle pod de cert-manager-openshift-routes fonctionne avec succès:

```
$ oc -n cert-manager get pods
```

Exemple de résultat

```
NAME                                     READY   STATUS    RESTARTS   AGE
cert-manager-866d8f788c-9kspc            1/1     Running   0          4h21m
cert-manager-cainjector-6885c585bd-znws8 1/1     Running   0          4h41m
cert-manager-openshift-routes-75b6bb44cd-f8kd5   1/1     Running   0          6s
cert-manager-webhook-8498785dd9-bvfdf    1/1     Running   0          4h41m
```

14.7. DÉPLOIEMENT D'UNE APPLICATION D'ÉCHANTILLON

Désormais que les certificats dynamiques sont configurés, vous pouvez déployer un exemple d'application pour confirmer que les certificats sont fournis et fiables lorsque vous exposez un nouvel itinéraire.

1. Créez un nouveau projet pour votre exemple d'application:

```
$ oc new-project hello-world
```

2. Déployez une application hello world:

```
$ oc -n hello-world new-app --image=docker.io/openshift/hello-openshift
```

3. Créer un itinéraire pour exposer l'application à partir de l'extérieur du cluster:

```
$ oc -n hello-world create route edge --service=hello-openshift hello-openshift-tls --hostname
hello.${DOMAIN}
```

4. La vérification du certificat de l'itinéraire n'est pas fiable:

```
$ curl -I https://hello.${DOMAIN}
```

Exemple de sortie

```
curl: (60) SSL: no alternative certificate subject name matches target host name
'hello.example.com'
More details here: https://curl.se/docs/sslcerts.html

curl failed to verify the legitimacy of the server and therefore could not
establish a secure connection to it. To learn more about this situation and
how to fix it, please visit the web page mentioned above.
```

5. Annoter l'itinéraire pour déclencher cert-manager pour fournir un certificat pour le domaine personnalisé:

```
$ oc -n hello-world annotate route hello-openshift-tls cert-manager.io/issuer-
kind=ClusterIssuer cert-manager.io/issuer-name=letsencrypt-production
```

NOTE

Il faut 2-3 minutes pour que le certificat soit créé. Le renouvellement du certificat sera automatiquement géré par l'opérateur responsable du certificat à l'approche de l'expiration.

6. La vérification du certificat de l'itinéraire est désormais fiable:

```
$ curl -I https://hello.${DOMAIN}
```

Exemple de sortie

```
HTTP/2 200
date: Thu, 05 Oct 2023 23:45:33 GMT
content-length: 17
content-type: text/plain; charset=utf-8
set-cookie: 52e4465485b6fb4f8a1b1bed128d0f3b=68676068bb32d24f0f558f094ed8e4d7;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

14.8. DÉPANNAGE DE CERTIFICAT DYNAMIQUE PROVISIONNEMENT

NOTE

Le processus de validation prend généralement 2-3 minutes à compléter tout en créant des certificats.

Lorsque l'annotation de votre itinéraire ne déclenche pas la création de certificat pendant l'étape de création du certificat, exécutez oc décrire par rapport à chacun du certificat, demande de certificat, commande et défiez les ressources pour voir les événements ou les raisons qui peuvent aider à identifier la cause du problème.

```
$ oc get certificate,certificaterequest,order,challenge
```

En cas de dépannage, vous pouvez vous référer à ce guide utile dans le débogage des certificats.

Il est également possible d'utiliser l'outil cmctl CLI pour diverses activités de gestion des certificats, telles que la vérification de l'état des certificats et les renouvellements de tests.

CHAPITRE 15. L'ATTRIBUTION D'UNE IP DE SORTIE COHÉRENTE POUR LE TRAFIC EXTERNE

Il est possible d'attribuer une adresse IP cohérente pour le trafic qui quitte votre cluster tel que les groupes de sécurité qui nécessitent une configuration basée sur l'IP pour répondre aux normes de sécurité.

Le service OpenShift de Red Hat sur AWS (ROSA) utilise l'interface réseau de conteneurs OVN-Kubernetes (CNI) pour attribuer des adresses IP aléatoires à partir d'un pool. Cela peut rendre la configuration des verrouillages de sécurité imprévisible ou ouverte.

Consultez Configuration d'une adresse IP sortante pour plus d'informations.

Les objectifs

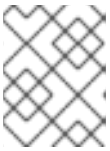
- Découvrez comment configurer un ensemble d'adresses IP prévisibles pour le trafic de clusters sortants.

Conditions préalables

- Cluster ROSA déployé avec OVN-Kubernetes
- Le CLI OpenShift (oc)
- Le ROSA CLI (rosa)
- JQ

15.1. CONFIGURATION DE VOS VARIABLES D'ENVIRONNEMENT

- Définissez vos variables d'environnement en exécutant la commande suivante:



NOTE

Il faut remplacer la valeur de la variable ROSA_MACHINE_POOL_NAME pour cibler un pool de machines différent.

```
$ export ROSA_CLUSTER_NAME=$(oc get infrastructure cluster -o=jsonpath="
{.status.infrastructureName}"  | sed 's/-[a-z0-9]\{5\}$//')
$ export ROSA_MACHINE_POOL_NAME=worker
```

15.2. ASSURER LA CAPACITÉ

Le nombre d'adresses IP attribuées à chaque nœud est limité pour chaque fournisseur de cloud public.

- Contrôlez une capacité suffisante en exécutant la commande suivante:

```
$ oc get node -o json | \
    jq '.items[] |
        {
            "name": .metadata.name,
            "ips": (.status.addresses | map(select(.type == "InternalIP") | .address)),
```

```
            "capacity": (.metadata.annotations."cloud.network.openshift.io/egress-ipconfig" |
fromjson[] | .capacity.ipv4)
        }'
```

Exemple de sortie

```
---
{
  "name": "ip-10-10-145-88.ec2.internal",
  "ips": [
    "10.10.145.88"
  ],
  "capacity": 14
}
{
  "name": "ip-10-10-154-175.ec2.internal",
  "ips": [
    "10.10.154.175"
  ],
  "capacity": 14
}
---
```

15.3. CRÉATION DES RÈGLES IP DE SORTIE

1. Avant de créer les règles IP de sortie, identifiez les IP de sortie que vous utiliserez.

NOTE

Les IP de sortie que vous sélectionnez doivent exister dans le cadre des sous-réseaux dans lesquels les nœuds de travail sont fournis.

2. Facultatif: Réservez les adresses IP de sortie que vous avez demandées pour éviter les conflits avec le service AWS Virtual Private Cloud (VPC) Dynamic Host Configuration Protocol (DHCP). Demandez des réservations IP explicites sur la documentation AWS pour la page de réservation CIDR.

15.4. ATTRIBUTION D'UNE SORTIE IP À UN ESPACE DE NOMS

1. Créez un nouveau projet en exécutant la commande suivante:

```
$ oc new-project demo-egress-ns
```

2. Créez la règle de sortie pour tous les pods dans l'espace de noms en exécutant la commande suivante:

```
$ cat <<EOF | oc apply -f -
apiVersion: k8s.ovn.org/v1
kind: EgressIP
metadata:
  name: demo-egress-ns
spec:
  # NOTE: these egress IPs are within the subnet range(s) in which my worker nodes
```

```
  #       are deployed.
  egressIPs:
    - 10.10.100.253
    - 10.10.150.253
    - 10.10.200.253
  namespaceSelector:
    matchLabels:
      kubernetes.io/metadata.name: demo-egress-ns
EOF
```

15.5. ATTRIBUTION D'UNE IP DE SORTIE À UN POD

1. Créez un nouveau projet en exécutant la commande suivante:

```
$ oc new-project demo-egress-pod
```

2. Créez la règle de sortie pour le pod en exécutant la commande suivante:

NOTE

le Spéc.namespaceSelector est un champ obligatoire.

```
$ cat <<EOF | oc apply -f -
apiVersion: k8s.ovn.org/v1
kind: EgressIP
metadata:
  name: demo-egress-pod
spec:
  # NOTE: these egress IPs are within the subnet range(s) in which my worker nodes
  #       are deployed.
  egressIPs:
    - 10.10.100.254
    - 10.10.150.254
    - 10.10.200.254
  namespaceSelector:
    matchLabels:
      kubernetes.io/metadata.name: demo-egress-pod
  podSelector:
    matchLabels:
      run: demo-egress-pod
EOF
```

15.5.1. Étiqueter les nœuds

1. Bénéficiez de vos affectations IP sortantes en cours en exécutant la commande suivante:

```
$ oc get egressips
```

Exemple de sortie

```
NAME              EGRESSIPS       ASSIGNED NODE   ASSIGNED EGRESSIPS
demo-egress-ns    10.10.100.253
demo-egress-pod   10.10.100.254
```

La règle IP de sortie que vous avez créée s'applique uniquement aux nœuds avec l'étiquette k8s.ovn.org/egress-assignable. Assurez-vous que l'étiquette est uniquement sur un pool de machines spécifique.

2. Attribuez l'étiquette à votre pool de machines en utilisant la commande suivante:

AVERTISSEMENT

Lorsque vous comptez sur des étiquettes de nœuds pour votre pool de machines, cette commande remplacera ces étiquettes. Assurez-vous d'entrer vos étiquettes souhaitées dans le champ --labels pour vous assurer que vos étiquettes de nœud restent.

```
$ rosa update machinepool ${ROSA_MACHINE_POOL_NAME} \
  --cluster="${ROSA_CLUSTER_NAME}" \
  --labels "k8s.ovn.org/egress-assignable="
```

15.5.2. Examen des IP de sortie

- Examinez les affectations IP sortantes en exécutant la commande suivante:

```
$ oc get egressips
```

Exemple de sortie

```
NAME              EGRESSIPS       ASSIGNED NODE                 ASSIGNED EGRESSIPS
demo-egress-ns    10.10.100.253   ip-10-10-156-122.ec2.internal   10.10.150.253
demo-egress-pod   10.10.100.254   ip-10-10-156-122.ec2.internal   10.10.150.254
```

15.6. LA VÉRIFICATION

15.6.1. Déploiement d'une application d'échantillon

Afin de tester la règle IP de sortie, créez un service limité aux adresses IP de sortie que nous avons spécifiées. Cela simule un service externe qui attend un petit sous-ensemble d'adresses IP.

1. Exécutez la commande echoserver pour reproduire une requête:

```
$ oc -n default run demo-service --image=gcr.io/google_containers/echoserver:1.4
```

2. Exposez le pod en tant que service et limitez l'entrée aux adresses IP de sortie que vous avez spécifiées en exécutant la commande suivante:

```
$ cat <<EOF | oc apply -f -
apiVersion: v1
kind: Service
metadata:
  name: demo-service
  namespace: default
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-scheme: "internal"
    service.beta.kubernetes.io/aws-load-balancer-internal: "true"
spec:
  selector:
    run: demo-service
  ports:
    - port: 80
      targetPort: 8080
  type: LoadBalancer
  externalTrafficPolicy: Local
  # NOTE: this limits the source IPs that are allowed to connect to our service.  It
  #       is being used as part of this demo, restricting connectivity to our egress
  #       IP addresses only.
  # NOTE: these egress IPs are within the subnet range(s) in which my worker nodes
  #       are deployed.
  loadBalancerSourceRanges:
    - 10.10.100.254/32
    - 10.10.150.254/32
    - 10.10.200.254/32
    - 10.10.100.253/32
    - 10.10.150.253/32
    - 10.10.200.253/32
EOF
```

3. Enregistrez le nom d'hôte de l'équilibreur de charge et enregistrez-le en tant que variable d'environnement en exécutant la commande suivante:

```
$ export LOAD_BALANCER_HOSTNAME=$(oc get svc -n default demo-service -o json | jq -
r '.status.loadBalancer.ingress[].hostname')
```

15.6.2. Tester la sortie de l'espace de noms

1. Démarrez un shell interactif pour tester la règle de sortie de l'espace de noms:

```
$ oc run \
  demo-egress-ns \
  -it \
  --namespace=demo-egress-ns \
  --env=LOAD_BALANCER_HOSTNAME=$LOAD_BALANCER_HOSTNAME \
  --image=registry.access.redhat.com/ubi9/ubi -- \
  bash
```

2. Envoyez une demande à l'équilibreur de charge et assurez-vous que vous pouvez vous connecter avec succès:

```
$ curl -s http://$LOAD_BALANCER_HOSTNAME
```

3. Consultez la sortie pour une connexion réussie:

NOTE

L'adresse client_address est l'adresse IP interne de l'équilibreur de charge et non votre adresse IP. Il est possible de vérifier que vous avez configuré correctement l'adresse client en vous connectant à votre service limité à .spec.loadBalancerSourceRanges.

Exemple de sortie

```
CLIENT VALUES:
client_address=10.10.207.247
command=GET
real path=/
query=nil
request_version=1.1
request_uri=http://internal-a3e61de18bfca4a53a94a208752b7263-148284314.us-east-
1.elb.amazonaws.com:8080/

SERVER VALUES:
server_version=nginx: 1.10.0 - lua: 10001

HEADERS RECEIVED:
accept=*/*
host=internal-a3e61de18bfca4a53a94a208752b7263-148284314.us-east-
1.elb.amazonaws.com
user-agent=curl/7.76.1
BODY:
-no body in request-
```

4. Sortez du pod en exécutant la commande suivante:

```
$ exit
```

15.6.3. Tester la sortie de la gousse

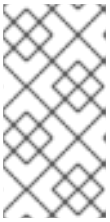
1. Démarrez un shell interactif pour tester la règle de sortie de pod:

```
$ oc run \
  demo-egress-pod \
  -it \
  --namespace=demo-egress-pod \
  --env=LOAD_BALANCER_HOSTNAME=$LOAD_BALANCER_HOSTNAME \
  --image=registry.access.redhat.com/ubi9/ubi -- \
  bash
```

2. Envoyez une demande à l'équilibreur de charge en exécutant la commande suivante:

```
$ curl -s http://$LOAD_BALANCER_HOSTNAME
```

3. Consultez la sortie pour une connexion réussie:



NOTE

L'adresse client_address est l'adresse IP interne de l'équilibreur de charge et non votre adresse IP. Il est possible de vérifier que vous avez configuré correctement l'adresse client en vous connectant à votre service limité à .spec.loadBalancerSourceRanges.

Exemple de sortie

```
CLIENT VALUES:
client_address=10.10.207.247
command=GET
real path=/
query=nil
request_version=1.1
request_uri=http://internal-a3e61de18bfca4a53a94a208752b7263-148284314.us-east-
1.elb.amazonaws.com:8080/

SERVER VALUES:
server_version=nginx: 1.10.0 - lua: 10001

HEADERS RECEIVED:
accept=*/*
host=internal-a3e61de18bfca4a53a94a208752b7263-148284314.us-east-
1.elb.amazonaws.com
user-agent=curl/7.76.1
BODY:
-no body in request-
```

4. Sortez du pod en exécutant la commande suivante:

```
$ exit
```

15.6.4. Facultatif: Essais bloqués sortie

1. Facultatif: Testez que le trafic est bloqué avec succès lorsque les règles de sortie ne s'appliquent pas en exécutant la commande suivante:

```
$ oc run \
  demo-egress-pod-fail \
  -it \
  --namespace=demo-egress-pod \
  --env=LOAD_BALANCER_HOSTNAME=$LOAD_BALANCER_HOSTNAME \
  --image=registry.access.redhat.com/ubi9/ubi -- \
  bash
```

2. Envoyez une demande à l'équilibreur de charge en exécutant la commande suivante:

```
$ curl -s http://$LOAD_BALANCER_HOSTNAME
```

3. En cas d'échec de la commande, egress est bloqué avec succès.

4. Sortez du pod en exécutant la commande suivante:

```
$ exit
```

15.7. LE NETTOYAGE DE VOTRE CLUSTER

1. Nettoyez votre cluster en exécutant les commandes suivantes:

```
$ oc delete svc demo-service -n default; \
$ oc delete pod demo-service -n default; \
$ oc delete project demo-egress-ns; \
$ oc delete project demo-egress-pod; \
$ oc delete egressip demo-egress-ns; \
$ oc delete egressip demo-egress-pod
```

2. Nettoyez les étiquettes de nœud assignées en exécutant la commande suivante:

AVERTISSEMENT

Lorsque vous comptez sur des étiquettes de nœuds pour votre pool de machines, cette commande remplace ces étiquettes. Entrez vos étiquettes souhaitées dans le champ --labels pour vous assurer que vos étiquettes de nœud restent.

```
$ rosa update machinepool ${ROSA_MACHINE_POOL_NAME} \
  --cluster="${ROSA_CLUSTER_NAME}" \
  --labels ""
```

CHAPITRE 16. TUTORIEL: MISE À JOUR DES ROUTES DE COMPOSANTS AVEC DES DOMAINES PERSONNALISÉS ET DES CERTIFICATS TLS

Ce guide montre comment modifier le nom d'hôte et le certificat TLS de la console Web, du serveur OAuth et des itinéraires de composants Téléchargements dans Red Hat OpenShift Service sur AWS (ROSA) version 4.14 et ci-dessus.[1]

Les changements que nous apportons aux routes des composants[2] ce guide est décrit plus en détail dans la personnalisation de l'URL interne du serveur OAuth, de l'itinéraire de la console et de la documentation OpenShift Container Platform.

16.1. CONDITIONS PRÉALABLES

- CLI ROSA (rosa) version 1.2.37 ou supérieure
- AWS CLI (aws)
- A ROSA Classic cluster version 4.14 ou supérieure

NOTE

Le ROSA avec HCP n'est pas pris en charge pour le moment.

- CLI OpenShift (oc)
- JQ CLI
- Accès au cluster en tant qu'utilisateur avec le rôle cluster-admin.
- L'OpenSSL (pour générer les certificats SSL/TLS de démonstration)

16.2. CONFIGURATION DE VOTRE ENVIRONNEMENT

1. Connectez-vous à votre cluster à l'aide d'un compte avec des privilèges cluster-admin.
2. Configurez une variable d'environnement pour le nom de votre cluster:

```
$ export CLUSTER_NAME=$(oc get infrastructure cluster -o=jsonpath="
{.status.infrastructureName}"  | sed 's/-[a-z0-9]\{5\}$//')
```

3. Assurez-vous que tous les champs sortent correctement avant de passer à la section suivante:

```
$ echo "Cluster: ${CLUSTER_NAME}"
```

Exemple de sortie

```
Cluster: my-rosa-cluster
```

16.3. DÉCOUVREZ LES ITINÉRAIRES ACTUELS

1. Assurez-vous que vous pouvez atteindre les routes des composants sur leurs noms d'hôte par défaut.
   Les noms d'hôte se trouvent en interrogeant les listes d'itinéraires dans les projets openshift-console et openshift-authentication.

```
$ oc get routes -n openshift-console
$ oc get routes -n openshift-authentication
```

Exemple de sortie

```
NAME        HOST/PORT                                                    PATH      SERVICES
PORT    TERMINATION          WILDCARD
console     console-openshift-console.apps.my-example-cluster-
aws.z9a9.p1.openshiftapps.com     ... 1 more  console     https    reencrypt/Redirect    None
downloads   downloads-openshift-console.apps.my-example-cluster-
aws.z9a9.p1.openshiftapps.com  ... 1 more  downloads  http    edge/Redirect        None
NAME            HOST/PORT                                                PATH      SERVICES
PORT    TERMINATION          WILDCARD
oauth-openshift   oauth-openshift.apps.my-example-cluster-aws.z9a9.p1.openshiftapps.com
... 1 more  oauth-openshift   6443    passthrough/Redirect    None
```

   À partir de cette sortie, vous pouvez voir que notre nom d'hôte de base est z9a9.p1.openshiftapps.com.

2. Accédez à l'ID de l'entrée par défaut en exécutant la commande suivante:

```
$ export INGRESS_ID=$(rosa list ingress -c ${CLUSTER_NAME} -o json | jq -r '.[] |
select(.default == true) | .id')
```

3. Assurez-vous que tous les champs sortent correctement avant de passer à la section suivante:

```
$ echo "Ingress ID: ${INGRESS_ID}"
```

Exemple de sortie

```
Ingress ID: r3l6
```

   En exécutant ces commandes, vous pouvez voir que les routes de composants par défaut pour notre cluster sont:

   - console-openshift-console.apps.my-example-cluster-aws.z9a9.p1.openshiftapps.com pour Console
   - Downloads-openshift-console.apps.my-example-cluster-aws.z9a9.p1.openshiftapps.com pour les téléchargements
   - accueil OAuth-openshift.apps.my-example-cluster-aws.z9a9.p1.openshiftapps.com pour OAuth

La commande rosa edit ingress permet de modifier le nom d'hôte de chaque service et d'ajouter un certificat TLS pour tous nos itinéraires de composants. Les paramètres pertinents sont affichés dans cet extrait de l'aide de ligne de commande pour la commande rosa edit ingress:

```
$ rosa edit ingress -h
Edit a cluster ingress for a cluster. Usage:
  rosa edit ingress ID [flags]
  [...]
  --component-routes string                Component routes settings. Available keys [oauth, console, downloads]. For each key a pair of hostname and tlsSecretRef is expected to be supplied. Format should be a comma separate list 'oauth: hostname=example-hostname;tlsSecretRef=example-secret-ref,downloads:...'
```

Dans cet exemple, nous utiliserons les itinéraires de composants personnalisés suivants:

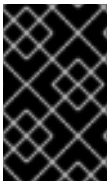
- console.my-new-domain.dev pour Console
- Downloads.my-new-domain.dev pour les téléchargements
- cliquez ici pour OAuth.my-new-domain.dev

16.4. CRÉER UN CERTIFICAT TLS VALIDE POUR CHAQUE ROUTE DE COMPOSANT

Dans cette section, nous créons trois paires de clés de certificat autosignées distinctes, puis nous leur faisons confiance pour vérifier que nous pouvons accéder à nos nouveaux itinéraires de composants à l'aide d'un navigateur Web réel.

AVERTISSEMENT

Ceci est uniquement à des fins de démonstration et n'est pas recommandé comme solution pour les charges de travail de production. Consultez votre autorité de certification pour comprendre comment créer des certificats avec des attributs similaires pour vos charges de travail de production.



IMPORTANT

Afin d'éviter les problèmes liés au coalescing de connexion HTTP/2, vous devez utiliser un certificat individuel distinct pour chaque point de terminaison. L'utilisation d'une wildcard ou d'un certificat SAN n'est pas prise en charge.

1. Générer un certificat pour chaque route de composant, en prenant soin de définir le sujet de notre certificat (-subj) sur le domaine personnalisé de l'itinéraire du composant que nous voulons utiliser:

   Exemple :

   ```
   $ openssl req -newkey rsa:2048 -new -nodes -x509 -days 365 -keyout key-console.pem -out cert-console.pem -subj "/CN=console.my-new-domain.dev"
   $ openssl req -newkey rsa:2048 -new -nodes -x509 -days 365 -keyout key-downloads.pem -out cert-downloads.pem -subj "/CN=downloads.my-new-domain.dev"
   $ openssl req -newkey rsa:2048 -new -nodes -x509 -days 365 -keyout key-oauth.pem -out cert-oauth.pem -subj "/CN=oauth.my-new-domain.dev"
   ```

Cela génère trois paires de fichiers .pem, key-<component>.pem et cert-<component>.pem.

16.5. AJOUTER LES CERTIFICATS AU CLUSTER EN TANT QUE SECRETS

1. Créez trois secrets TLS dans l'espace de noms openshift-config.
   Ceux-ci deviennent votre référence secrète lorsque vous mettez à jour les itinéraires des composants plus tard dans ce guide.

```
$ oc create secret tls console-tls --cert=cert-console.pem --key=key-console.pem -n openshift-config
$ oc create secret tls downloads-tls --cert=cert-downloads.pem --key=key-downloads.pem -n openshift-config
$ oc create secret tls oauth-tls --cert=cert-oauth.pem --key=key-oauth.pem -n openshift-config
```

16.6. CHERCHEZ LE NOM D'HÔTE DE L'ÉQUILIBREUR DE CHARGE DANS VOTRE CLUSTER

Lorsque vous créez un cluster, le service crée un équilibreur de charge et génère un nom d'hôte pour cet équilibreur de charge. Il faut connaître le nom d'hôte de l'équilibreur de charge afin de créer des enregistrements DNS pour notre cluster.

Il est possible de trouver le nom d'hôte en exécutant la commande oc get svc par rapport à l'espace de noms openshift-ingress. Le nom d'hôte de l'équilibreur de charge est l'EXTERNAL-IP associé au service routeur-défaut dans l'espace de noms openshift-ingress.

```
$ oc get svc -n openshift-ingress
NAME              TYPE           CLUSTER-IP     EXTERNAL-IP                                                              PORT(S)
AGE
router-default  LoadBalancer  172.30.237.88  a234gsr3242rsfsfs-1342r624.us-east-1.elb.amazonaws.com  80:31175/TCP,443:31554/TCP  76d
```

Dans notre cas, le nom d'hôte est a234gsr3242rsfs-1342r624.us-east-1.elb.amazonaws.com.

Enregistrez cette valeur pour plus tard, car nous en aurons besoin pour configurer des enregistrements DNS pour nos nouveaux noms d'hôte de route des composants.

16.7. AJOUTEZ DES ENREGISTREMENTS DNS D'ITINÉRAIRE DES COMPOSANTS À VOTRE FOURNISSEUR D'HÉBERGEMENT

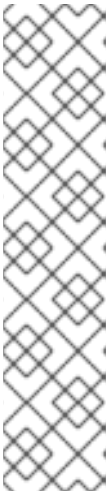
Dans votre fournisseur d'hébergement, ajoutez des enregistrements DNS qui mappent le CNAME de vos nouveaux noms d'hôte d'itinéraire de composants au nom d'hôte de balanceur de charge que nous avons trouvé à l'étape précédente.

16.8. ACTUALISEZ LES ROUTES DES COMPOSANTS ET LE SECRET TLS EN UTILISANT LE ROSA CLI

Lorsque vos enregistrements DNS ont été mis à jour, vous pouvez utiliser le ROSA CLI pour modifier les itinéraires des composants.

1. La commande rosa edit ingress permet de mettre à jour votre route d'entrée par défaut avec le nouveau domaine de base et la référence secrète qui y est associée, en prenant soin de mettre à jour les noms d'hôte pour chaque route de composant.

```
$ rosa edit ingress -c ${CLUSTER_NAME} ${INGRESS_ID} --component-routes 'console:
hostname=console.my-new-domain.dev;tlsSecretRef=console-tls,downloads:
hostname=downloads.my-new-domain.dev;tlsSecretRef=downloads-tls,oauth:
hostname=oauth.my-new-domain.dev;tlsSecretRef=oauth-tls'
```



NOTE

En outre, vous pouvez modifier uniquement un sous-ensemble des routes de composants en laissant les itinéraires de composants que vous ne voulez pas modifier sur une chaîne vide. À titre d'exemple, si vous souhaitez uniquement modifier les noms d'hôte de console et de serveur OAuth et les certificats TLS, vous exécuteriez la commande suivante:

```
$ rosa edit ingress -c ${CLUSTER_NAME} ${INGRESS_ID} --component-
routes 'console: hostname=console.my-new-
domain.dev;tlsSecretRef=console-tls,downloads:
hostname="";tlsSecretRef="", oauth: hostname=oauth.my-new-
domain.dev;tlsSecretRef=oauth-tls'
```

2. Exécutez la commande d'entrée de liste rosa pour vérifier que vos modifications ont été apportées avec succès:

```
$ rosa list ingress -c ${CLUSTER_NAME} -ojson | jq ".[] | select(.id == \"${INGRESS_ID}\") |
.component_routes"
```

Exemple de sortie

```
{
  "console": {
    "kind": "ComponentRoute",
    "hostname": "console.my-new-domain.dev",
    "tls_secret_ref": "console-tls"
  },
  "downloads": {
    "kind": "ComponentRoute",
    "hostname": "downloads.my-new-domain.dev",
    "tls_secret_ref": "downloads-tls"
  },
  "oauth": {
    "kind": "ComponentRoute",
    "hostname": "oauth.my-new-domain.dev",
    "tls_secret_ref": "oauth-tls"
  }
}
```

3. Ajoutez votre certificat au magasin de confiance de votre système local, puis confirmez que vous pouvez accéder à vos composants à leurs nouveaux itinéraires à l'aide de votre navigateur Web local.

16.9. LA RÉINITIALISATION DES ROUTES DES COMPOSANTS PAR DÉFAUT À L'AIDE DU ROSA CLI

Lorsque vous souhaitez réinitialiser les routes des composants vers la configuration par défaut, exécutez la commande rosa edit ingress suivante:

```
$ rosa edit ingress -c ${CLUSTER_NAME} ${INGRESS_ID} --component-routes 'console: hostname="";tlsSecretRef="",downloads: hostname="";tlsSecretRef="", oauth: hostname="";tlsSecretRef=""'
```

[1] La modification de ces itinéraires sur Red Hat OpenShift Service sur les versions AWS ROSA avant 4.14 n'est généralement pas prise en charge. Cependant, si vous avez un cluster à l'aide de la version 4.13, vous pouvez demander le support Red Hat pour activer la prise en charge de cette fonctionnalité sur votre cluster version 4.13 en ouvrant un cas de support.

[2] Le terme « route des composants » désigne les itinéraires OAuth, Console et Téléchargements qui sont fournis lors de l'installation de ROSA.

CHAPITRE 17. DÉBUTER AVEC ROSA

17.1. TUTORIEL: QU'EST-CE QUE ROSA

Le service OpenShift Red Hat sur AWS (ROSA) est une plateforme d'applications clé en main entièrement gérée qui vous permet de vous concentrer sur ce qui compte le plus, en fournissant de la valeur à vos clients en créant et en déployant des applications. Les experts de Red Hat et AWS SRE gèrent la plate-forme sous-jacente afin que vous n'ayez pas à vous soucier de la gestion de l'infrastructure. La ROSA fournit une intégration transparente avec une large gamme de services de calcul AWS, de bases de données, d'analyse, d'apprentissage automatique, de réseautage, de mobile et d'autres services afin d'accélérer la création et la fourniture d'expériences différenciées à vos clients.

Le ROSA utilise AWS Security Token Service (STS) pour obtenir des informations d'identification pour gérer l'infrastructure de votre compte AWS. AWS STS est un service Web mondial qui crée des informations d'identification temporaires pour les utilisateurs IAM ou les utilisateurs fédérés. La ROSA s'en sert pour attribuer des informations d'identification de sécurité à court terme et à privilèges limités. Ces informations d'identification sont associées aux rôles IAM qui sont spécifiques à chaque composant qui effectue des appels API AWS. Cette méthode s'harmonise avec les principes du moindre privilège et des pratiques sécurisées dans la gestion des ressources de service cloud. L'outil d'interface de ligne de commande ROSA (CLI) gère les informations d'identification STS qui sont assignées pour des tâches uniques et agit sur les ressources AWS dans le cadre de la fonctionnalité OpenShift.

17.1.1. Caractéristiques clés de ROSA

- Accès et utilisation de Red Hat OpenShift à la demande grâce à une expérience d'intégration en libre-service via la console de gestion AWS.
- Des prix flexibles et basés sur la consommation : Évoluez en fonction des besoins de votre entreprise et payez au fur et à mesure de la flexibilité des prix et d'un modèle de facturation horaire ou annuel à la demande.
- Facture unique pour l'utilisation de Red Hat OpenShift et AWS : les clients recevront une seule facture d'AWS pour la consommation de Red Hat OpenShift et AWS.
- Expérience de support entièrement intégrée : l'installation, la gestion, la maintenance et les mises à niveau sont effectuées par des ingénieurs de fiabilité du site de Red Hat (SRE) avec le support conjoint Red Hat et Amazon et un accord de niveau de service (SLA) de 99,95 %.
- Intégration de services AWS: AWS dispose d'un portefeuille robuste de services cloud, tels que le calcul, le stockage, la mise en réseau, la base de données, l'analyse et l'apprentissage automatique. L'ensemble de ces services est directement accessible via ROSA. Cela facilite la construction, l'exploitation et l'échelle mondiale et à la demande grâce à une interface de gestion familière.
- Disponibilité maximale : Déployez des clusters sur plusieurs zones de disponibilité dans les régions prises en charge pour maximiser la disponibilité et maintenir une disponibilité élevée pour vos applications et données critiques.
- Cluster node scaling: Ajoutez ou supprimez facilement des nœuds de calcul pour correspondre à la demande de ressources.
- Clusters optimisés: Choisissez parmi les types d'instance EC2 optimisés, optimisés par calcul ou à usage général avec des clusters dimensionnés pour répondre à vos besoins.

- Disponibilité mondiale : Se reporter à la page de disponibilité régionale du produit pour voir où ROSA est disponible à l'échelle mondiale.

17.1.2. La ROSA et les Kubernetes

Dans ROSA, tout ce dont vous avez besoin pour déployer et gérer des conteneurs est groupé, y compris la gestion des conteneurs, les opérateurs, le réseau, l'équilibrage de charge, le maillage de service, le CI/CD, le pare-feu, la surveillance, le registre, l'authentification et les capacités d'autorisation. Ces composants sont testés ensemble pour des opérations unifiées en tant que plate-forme complète. Les opérations de cluster automatisées, y compris les mises à niveau de plate-forme en direct, améliorent encore votre expérience Kubernetes.

17.1.3. Les responsabilités de base

En général, le déploiement et l'entretien des clusters relèvent de la responsabilité de Red Hat ou d'AWS, tandis que les applications, les utilisateurs et les données relèvent de la responsabilité du client. Afin d'obtenir une ventilation plus détaillée des responsabilités, voir la matrice des responsabilités.

17.1.4. Feuille de route et demandes de fonctionnalités

Consultez la feuille de route ROSA pour rester au courant de l'état des fonctionnalités en cours de développement. Lancez un nouveau problème si vous avez des suggestions pour l'équipe produit.

17.1.5. Disponibilité de la région AWS

Consultez la page de disponibilité régionale du produit pour obtenir une vue à jour de l'endroit où le ROSA est disponible.

17.1.6. Certifications de conformité

Actuellement, ROSA est conforme aux normes SOC-2 type 2, SOC 3, ISO-27001, ISO 27017, ISO 27018, HIPAA, GDPR et PCI-DSS. En outre, nous travaillons actuellement vers FedRAMP High.

17.1.7. Les nœuds

17.1.7.1. Nœuds de travail dans plusieurs régions AWS

Les nœuds d'un cluster ROSA doivent être situés dans la même région AWS. Dans le cas des clusters configurés pour plusieurs zones de disponibilité, les nœuds de plan de contrôle et les nœuds ouvriers seront répartis entre les zones de disponibilité.

17.1.7.2. Le nombre minimum de nœuds de travailleurs

Dans le cas d'un cluster ROSA, le minimum est de 2 nœuds ouvriers pour la zone de disponibilité unique et de 3 nœuds ouvriers pour plusieurs zones de disponibilité.

17.1.7.3. Le système d'exploitation des nœuds sous-jacents

Comme pour toutes les offres OpenShift v4.x, le plan de contrôle, l'infra et les nœuds de travail exécutent Red Hat Enterprise Linux CoreOS (RHCOS).

17.1.7.4. Hibernation ou arrêt des nœuds

À l'heure actuelle, ROSA n'a pas de fonction d'hibernation ou d'arrêt pour les nœuds. La fonction d'arrêt et d'hibernation est une fonctionnalité de plate-forme OpenShift qui n'est pas encore assez mature pour une utilisation généralisée des services cloud.

17.1.7.5. Instances prises en charge pour les nœuds ouvriers

Dans la liste complète des instances prises en charge pour les nœuds de travail, voir les types d'instances AWS. Les instances ponctuelles sont également prises en charge.

17.1.7.6. Autoscaling des nœuds

La mise à l'échelle automatique vous permet d'ajuster automatiquement la taille du cluster en fonction de la charge de travail actuelle. Consultez À propos des nœuds autoscaling sur un cluster pour plus de détails.

17.1.7.7. Le nombre maximal de nœuds de travailleurs

Le nombre maximal de nœuds de travail dans les versions 4.14.14 et ultérieures des clusters ROSA est de 249. Dans les versions précédentes, la limite est de 180 nœuds.

La documentation de l'ACR fournit une liste des rôles à l'échelle du compte et par groupe.

17.1.8. Administrateurs

L'administrateur d'un client ROSA peut gérer les utilisateurs et les quotas en plus d'accéder à tous les projets créés par l'utilisateur.

17.1.9. Les versions et mises à jour OpenShift

Le ROSA est un service géré basé sur OpenShift Container Platform. La version actuelle et les dates du cycle de vie sont affichées dans la documentation ROSA.

Les clients peuvent passer à la dernière version d'OpenShift et utiliser les fonctionnalités de cette version d'OpenShift. Consultez les dates du cycle de vie pour plus d'informations. Les fonctionnalités OpenShift ne sont pas toutes disponibles sur ROSA. Examinez la définition du service pour plus d'informations.

17.1.10. Le soutien

Il est possible d'ouvrir un billet directement à partir de l'OpenShift Cluster Manager. Consultez la documentation de support ROSA pour plus de détails sur l'obtention du soutien.

En outre, vous pouvez visiter le portail client Red Hat pour rechercher ou parcourir la base de connaissances Red Hat sur les articles et solutions relatifs aux produits Red Hat ou soumettre un dossier d'assistance à Red Hat Support.

17.1.10.1. Le soutien limité

Lorsqu'un cluster ROSA n'est pas mis à niveau avant la date de fin de vie, le cluster continue de fonctionner dans un statut de support limité. Le SLA pour ce cluster ne sera plus applicable, mais vous pouvez toujours obtenir un support pour ce cluster. Consultez la documentation d'état du support limité pour plus de détails.

Autres ressources d ' appui

- Appui au chapeau rouge
- Le support AWS
  Les clients d'assistance AWS doivent avoir un contrat de support AWS valide

17.1.11. Accord de niveau de service (SLA)

Consultez la page ROSA SLA pour plus de détails.

17.1.12. Les notifications et la communication

Le Red Hat fournira des notifications concernant les nouvelles fonctionnalités Red Hat et AWS, les mises à jour et la maintenance programmée par e-mail et le journal de service Hybrid Cloud Console.

17.1.13. Courtier Open Service pour AWS (OBSA)

Il est possible d'utiliser OSBA avec ROSA. Cependant, la méthode préférée est la plus récente AWS Controller pour Kubernetes. Consultez Open Service Broker pour AWS pour plus d'informations sur OSBA.

17.1.14. Hors-bord

Les clients peuvent cesser d'utiliser ROSA à tout moment et déplacer leurs applications sur site, dans un cloud privé ou dans d'autres fournisseurs de cloud. La politique standard des instances réservées (RI) s'applique aux RI inutilisés.

17.1.15. Authentification

La ROSA prend en charge les mécanismes d'authentification suivants : OpenID Connect (un profil d'OAuth2), Google OAuth, GitHub OAuth, GitLab et LDAP.

17.1.16. Accès au cluster SRE

L'accès au cluster SRE est sécurisé par MFA. Consultez SRE accès pour plus de détails.

17.1.17. Chiffrement

17.1.17.1. Clés de chiffrement

La ROSA utilise une clé stockée dans KMS pour chiffrer les volumes EBS. Les clients ont également la possibilité de fournir leurs propres clés KMS lors de la création de clusters.

17.1.17.2. Clés KMS

Lorsque vous spécifiez une clé KMS, le plan de contrôle, l'infrastructure et les volumes racine des nœuds de travail et les volumes persistants sont chiffrés avec la clé.

17.1.17.3. Chiffrement des données

Il y a par défaut un cryptage au repos. La plate-forme AWS Storage crypte automatiquement vos données avant de les persister et décrypte les données avant leur récupération. Consultez AWS EBS Encryption pour plus de détails.

Il est également possible de chiffrer etcd dans le cluster, en le combinant avec le chiffrement de stockage AWS. Cela se traduit par le double du chiffrement qui ajoute jusqu'à 20% de performance. Consultez la documentation de cryptage etcd pour plus de détails.

17.1.17.4. chiffrement etcd

le chiffrement etcd ne peut être activé que lors de la création de clusters.

NOTE

le chiffrement etcd entraîne des frais généraux supplémentaires avec une réduction négligeable des risques de sécurité.

17.1.17.5. configuration de chiffrement etcd

le chiffrement etcd est configuré comme dans OpenShift Container Platform. Le cypher aescbc est utilisé et le réglage est corrigé lors du déploiement du cluster. Consultez la documentation Kubernetes pour plus de détails.

17.1.17.6. Clés KMS multi-régions pour le cryptage EBS

Actuellement, le ROSA CLI n'accepte pas les clés KMS multi-régions pour le cryptage EBS. Cette fonctionnalité est dans notre backlog pour les mises à jour des produits. Le ROSA CLI accepte les clés KMS d'une seule région pour le chiffrement EBS s'il est défini lors de la création de clusters.

17.1.18. L'infrastructure

La ROSA utilise plusieurs services cloud différents tels que les machines virtuelles, le stockage et les équilibreurs de charge. Dans les prérequis AWS, vous pouvez voir une liste définie.

17.1.19. Les méthodes d'identification

Il existe deux méthodes d'identification pour accorder à Red Hat les autorisations nécessaires pour effectuer les actions requises dans votre compte AWS: AWS avec STS ou un utilisateur IAM avec des autorisations d'administration. AWS avec STS est la méthode préférée, et la méthode utilisateur IAM sera éventuellement dépréciée. AWS et STS s'alignent mieux avec les principes du moindre privilège et des pratiques sécurisées dans la gestion des ressources de service cloud.

17.1.20. Erreurs d'autorisation ou d'échec préalables

Consultez une nouvelle version de la ROSA CLI. Chaque version de la ROSA CLI est située dans deux endroits: Github et le Red Hat signé des versions binaires.

17.1.21. Le stockage

Consultez la section de stockage de la définition du service.

Le pilote OpenShift inclut le pilote CSI pour AWS EFS. Cliquez ici pour plus d'informations sur la configuration d'AWS EFS pour Red Hat OpenShift Service sur AWS.

17.1.22. À l'aide d'un VPC

Lors de l'installation, vous pouvez choisir de vous déployer sur un VPC existant ou d'apporter votre propre VPC. Ensuite, vous pouvez sélectionner les sous-réseaux requis et fournir une plage CIDR valide qui englobe les sous-réseaux du programme d'installation lors de l'utilisation de ces sous-réseaux.

La ROSA permet à plusieurs clusters de partager le même VPC. Le nombre de clusters sur un VPC est limité par le quota de ressources AWS restant et les plages CIDR qui ne peuvent pas se chevaucher. Consultez Définitions de gamme CIDR pour plus d'informations.

17.1.23. Plugin réseau

La ROSA utilise le fournisseur de réseau CNI OpenShift OVN-Kubernetes par défaut.

17.1.24. La mise en réseau cross-namespace

Les administrateurs de clusters peuvent personnaliser, et nier, cross-namespace sur une base de projet à l'aide d'objets NetworkPolicy. Consultez la section Configurer l'isolement multilocataire avec la politique de réseau pour plus d'informations.

17.1.25. En utilisant Prometheus et Grafana

Il est possible d'utiliser Prometheus et Grafana pour surveiller les conteneurs et gérer la capacité grâce à OpenShift User Workload Monitoring. Il s'agit d'une option de case à cocher dans OpenShift Cluster Manager.

17.1.26. Enregistre les résultats de l'audit à partir du plan de contrôle du cluster

Lorsque le module d'extension de l'opérateur de journalisation du cluster a été ajouté au cluster, les journaux d'audit sont disponibles via CloudWatch. Dans le cas contraire, une demande d'assistance vous permettrait de demander certains journaux d'audit. De petits journaux ciblés et encadrés dans le temps peuvent être demandés pour l'exportation et envoyés à un client. La sélection des journaux d'audit disponibles est à la discrétion de SRE dans la catégorie de la sécurité et de la conformité de la plate-forme. Les demandes d'exportation de l'ensemble des grumes d'un groupe seront rejetées.

17.1.27. Limites des autorisations AWS

Il est possible d'utiliser une limite d'autorisation AWS autour des stratégies de votre cluster.

17.1.28. AMI

Les nœuds de travail ROSA utilisent un AMI différent de OSD et OpenShift Container Platform. Les AMI de Control Plane et Infra sont communs à tous les produits dans la même version.

17.1.29. Les sauvegardes de clusters

Les clusters ROSA STS n'ont pas de sauvegardes. Les utilisateurs doivent avoir leurs propres stratégies de sauvegarde pour les applications et les données. Consultez notre politique de sauvegarde pour plus d'informations.

17.1.30. Domaine personnalisé

Il est possible de définir un domaine personnalisé pour vos applications. Consultez Configuration des domaines personnalisés pour les applications pour plus d'informations.

17.1.31. Certificats de domaine ROSA

L'infrastructure Red Hat (Hive) gère la rotation des certificats pour l'entrée d'application par défaut.

17.1.32. Environnements déconnectés

Le ROSA ne prend pas en charge un environnement déconnecté de l'air. Le cluster ROSA doit avoir accès à Internet pour accéder à notre registre, S3, et envoyer des métriques. Le service nécessite un certain nombre de points de terminaison de sortie. L'entrée peut être limitée à un PrivateLink pour Red Hat SREs et à un VPN pour l'accès des clients.

Ressources supplémentaires

- Les pages de produits ROSA:
  - La page de produit Red Hat
  - AWS page produit
  - Le portail client Red Hat
- Les ressources spécifiques de ROSA
  - AWS ROSA pour démarrer le guide
  - Documentation du ROSA
  - Définition du service ROSA
  - La matrice d'affectation des responsabilités ROSA
  - Comprendre le processus et la sécurité
  - À propos de Disponibilité
  - Les mises à jour du cycle de vie
  - Feuille de route ROSA
- En savoir plus sur OpenShift
- Gestionnaire de cluster OpenShift
- Appui au chapeau rouge

17.2. TUTORIEL : ROSA AVEC AWS STS EXPLIQUÉ

Ce tutoriel décrit les deux options permettant à Red Hat OpenShift Service sur AWS (ROSA) d'interagir avec les ressources dans le compte Amazon Web Service (AWS) d'un utilisateur. Il détaille les composants et les processus utilisés par ROSA avec Security Token Service (STS) pour obtenir les informations d'identification nécessaires. Il examine également pourquoi ROSA avec STS est la méthode la plus sûre et préférée.

NOTE

Ce contenu couvre actuellement ROSA Classic avec AWS STS. Dans le cas de ROSA avec des avions de contrôle hébergés (HCP) avec AWS STS, voir AWS STS et ROSA avec HCP expliqué.

Ce tutoriel sera:

- Énumérez deux des options de déploiement:
  - La ROSA avec les utilisateurs IAM
  - La ROSA avec STS
- Expliquer les différences entre les deux options
- Expliquez pourquoi ROSA avec STS est plus sûr et l'option préférée
- Expliquez comment fonctionne ROSA avec STS

17.2.1. Différentes méthodes d'identification pour déployer ROSA

Dans le cadre de ROSA, Red Hat gère les ressources d'infrastructure de votre compte AWS et doit recevoir les autorisations nécessaires. Il existe actuellement deux méthodes prises en charge pour accorder ces autorisations:

- En utilisant des identifiants d'utilisateur IAM statiques avec une stratégie AdministratorAccess
  Ceci est appelé "ROSA avec les utilisateurs IAM" dans ce tutoriel. Ce n'est pas la méthode d'identification préférée.
- AWS STS avec des jetons dynamiques et de courte durée
  Ceci est appelé "ROSA avec STS" dans ce tutoriel. C'est la méthode d'identification préférée.

17.2.1.1. La Rosa avec les utilisateurs de l'IAM

Lorsque ROSA a été publié pour la première fois, la seule méthode d'identification était ROSA avec IAM Users. Cette méthode accorde aux utilisateurs de l'IAM un accès complet à la stratégie AdministratorAccess pour créer les ressources nécessaires dans le compte AWS qui utilise ROSA. Le cluster peut ensuite créer et étendre ses informations d'identification au besoin.

17.2.1.2. La ROSA avec STS

Avec STS, ROSA accorde aux utilisateurs un accès limité et à court terme aux ressources de votre compte AWS. La méthode STS utilise des rôles et des politiques prédéfinis pour accorder des autorisations temporaires et moins privilégiées aux utilisateurs IAM ou aux utilisateurs fédérés authentifiés. Les informations d'identification expirent généralement une heure après avoir été demandées. Lorsqu'ils ont expiré, ils ne sont plus reconnus par AWS et n'ont plus accès au compte à partir des demandes d'API faites avec eux. Consultez la documentation AWS pour plus d'informations. Alors que les deux ROSA avec les utilisateurs IAM et ROSA avec STS sont actuellement activés, ROSA avec STS est l'option préférée et recommandée.

17.2.2. La ROSA avec la sécurité STS

De nombreux composants essentiels rendent ROSA avec STS plus sécurisé que ROSA avec IAM Users:

- Ensemble explicite et limité de rôles et de politiques que l'utilisateur crée à l'avance. L'utilisateur connaît toutes les autorisations demandées et chaque rôle utilisé.
- Le service ne peut rien faire en dehors de ces autorisations.
- Chaque fois que le service doit effectuer une action, il obtient des informations d'identification qui expirent en une heure ou moins. Cela signifie qu'il n'est pas nécessaire de faire pivoter ou de révoquer les informations d'identification. De plus, l'expiration des informations d'identification réduit les risques de fuite et de réutilisation des informations d'identification.

17.2.3. AWS STS expliqué

La ROSA utilise AWS STS pour accorder des autorisations les moins privilégiées avec des informations d'identification de sécurité à court terme pour des rôles IAM spécifiques et séparés. Les informations d'identification sont associées à des rôles IAM spécifiques à chaque composant et cluster qui font des appels API AWS. Cette méthode s'harmonise avec les principes des pratiques moins privilégiées et sécurisées dans la gestion des ressources de service cloud. L'outil d'interface de ligne de commande ROSA (CLI) gère les rôles et les stratégies STS qui sont assignés pour des tâches uniques et agit sur les ressources AWS dans le cadre de la fonctionnalité OpenShift.

Les rôles et les politiques STS doivent être créés pour chaque cluster ROSA. Afin de faciliter cette tâche, les outils d'installation fournissent toutes les commandes et fichiers nécessaires pour créer les rôles en tant que stratégies et une option permettant à la CLI de créer automatiquement les rôles et les stratégies. Consultez Créer un cluster ROSA avec STS en utilisant des personnalisations pour plus d'informations sur les différentes options --mode.

17.2.4. Composants spécifiques à ROSA avec STS

- Infrastructure AWS - Cela fournit l'infrastructure requise pour le cluster. Il contient les instances, le stockage et les composants de réseau EC2. Consultez les types de calcul AWS pour voir les types d'instance pris en charge pour les nœuds de calcul et l'infrastructure AWS provisionnée pour la configuration des avions de contrôle et des nœuds d'infrastructure.
- AWS STS - Voir la section de méthode d'identification ci-dessus.
- Connexion OpenID (OIDC) - Cela fournit un mécanisme permettant aux opérateurs de clusters d'authentifier avec AWS, d'assumer les rôles de cluster grâce à une politique de confiance et d'obtenir des informations d'identification temporaires de STS pour effectuer les appels API requis.
- Les rôles et les politiques - Les rôles et les politiques sont l'une des principales différences entre ROSA avec STS et ROSA avec les utilisateurs IAM. Dans le cas de ROSA avec STS, les rôles et les politiques utilisés par ROSA sont divisés en rôles et politiques à l'échelle du compte et rôles et politiques des opérateurs.
  Les politiques déterminent les actions autorisées pour chacun des rôles. Consultez les ressources de l'IAM pour plus de détails sur les rôles et les politiques individuels.
  - Les rôles suivants à l'échelle du compte sont requis:
    - GérerOpenShift-Installer-Role
    - GéréOpenShift-ControlPlane-Role
    - GéréOpenShift-Worker-Role
    - GéréOpenShift-Support-Role

- Les politiques suivantes à l'échelle du compte sont requises:
  - GéréOpenShift-Installer-Role-Policy
  - GéréOpenShift-ControlPlane-Role-Policy
  - GéréOpenShift-Worker-Role-Policy
  - GéréOpenShift-Soutien-Rôle-Politique
  - GérerOpenShift-openshift-ingress-operator-cloud-credentials [1]
  - ManagedOpenShift-openshift-cluster-csi-drivers-ebs-cloud-credent [1]
  - ManagedOpenShift-openshift-cloud-network-config-controller-cloud [1]
  - GérerOpenShift-openshift-machine-api-aws-cloud-credentials [1]
  - GérerOpenShift-openshift-cloud-credential-operator-cloud-crede [1]
  - ManagedOpenShift-openshift-image-registry-installer-cloud-creden [1]

    1. Cette politique est utilisée par les rôles d'opérateur de cluster, énumérés ci-dessous. Les rôles d'opérateur sont créés dans une deuxième étape parce qu'ils dépendent d'un nom de cluster existant et ne peuvent pas être créés en même temps que les rôles à l'échelle du compte.

- Les rôles d'opérateur sont:
  - <cluster-name\>-xxxx-openshift-cluster-csi-drivers-ebs-cloud-credent
  - <cluster-name\>-xxxx-openshift-cloud-network-config-controller-cloud
  - <cluster-name\>-xxxx-openshift-machine-api-aws-cloud-credentials
  - <cluster-name\>-xxxx-openshift-cloud-credential-operator-cloud-crede
  - <cluster-name\>-xxxx-openshift-image-registry-installer-cloud-creden
  - <cluster-name\>-xxxx-openshift-ingress-operator-cloud-credentials
- Des politiques de confiance sont créées pour chaque rôle de l'ensemble du compte et de l'opérateur.

17.2.5. Déploiement d'un cluster ROSA STS

Il n'est pas prévu de créer les ressources énumérées dans les étapes ci-dessous à partir de zéro. Le ROSA CLI crée les fichiers JSON requis pour vous et affiche les commandes dont vous avez besoin. Le ROSA CLI peut également aller plus loin et exécuter les commandes pour vous, si vous le souhaitez.

Étapes pour déployer un ROSA avec le cluster STS

1. Créer les rôles et les politiques à l'échelle du compte.
2. Attribuez la politique d'autorisations au rôle correspondant à l'échelle du compte.
3. Créez le cluster.

4. Créez les rôles et les politiques de l'opérateur.
5. Attribuez la politique d'autorisation au rôle correspondant de l'opérateur.
6. Créez le fournisseur OIDC.

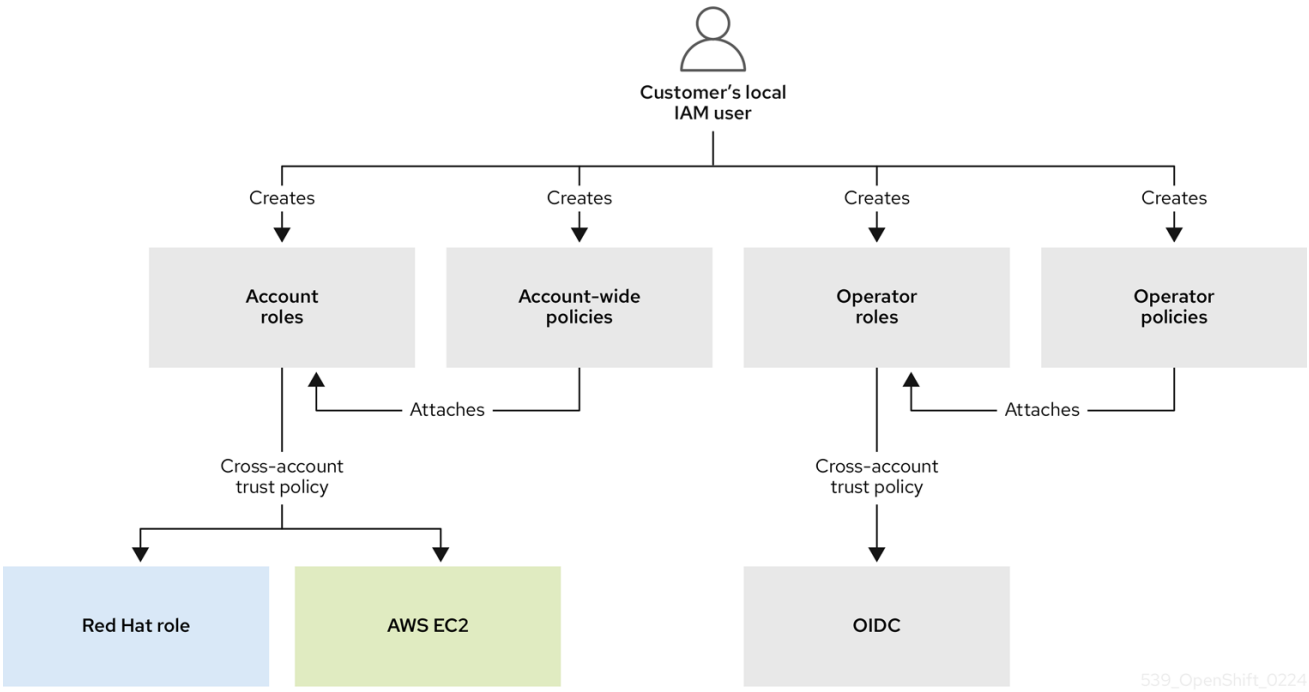
Les rôles et les stratégies peuvent être créés automatiquement par le ROSA CLI, ou ils peuvent être créés manuellement en utilisant les drapeaux automatiques --mode ou --mode dans le ROSA CLI. Cliquez ici pour plus de détails sur le déploiement, voir Créer un cluster avec des personnalisations ou le tutoriel Déploiement du cluster.

17.2.6. Flux de travail ROSA avec STS

L'utilisateur crée les rôles requis à l'échelle du compte et les politiques à l'échelle du compte. Consultez la section des composants de ce tutoriel pour plus d'informations. Au cours de la création de rôles, une politique de confiance, connue sous le nom de politique de confiance intercompte, est créée, ce qui permet à un rôle appartenant à Red Hat d'assumer les rôles. Des politiques de confiance sont également créées pour le service EC2, ce qui permet aux charges de travail des instances EC2 d'assumer des rôles et d'obtenir des pouvoirs. L'utilisateur peut alors attribuer une stratégie d'autorisation correspondante à chaque rôle.

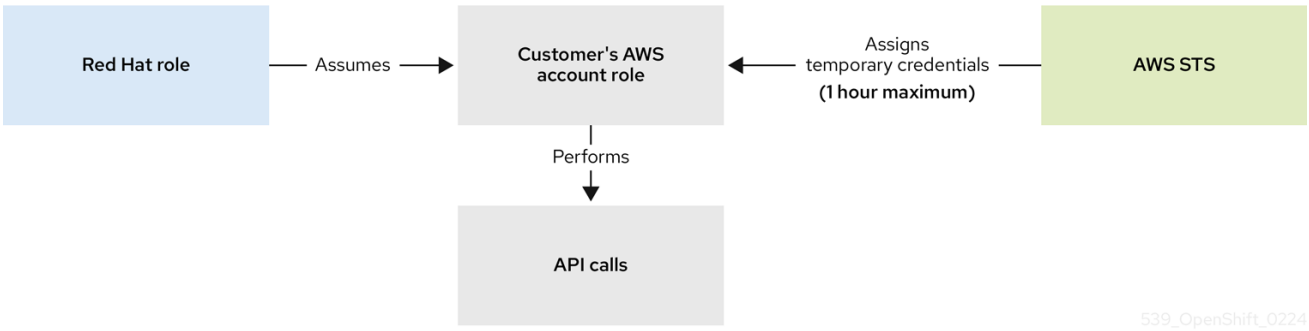
Après la création des rôles et des politiques à l'échelle du compte, l'utilisateur peut créer un cluster. Lorsque la création de cluster est lancée, les rôles d'opérateur sont créés afin que les opérateurs de cluster puissent faire des appels API AWS. Ces rôles sont ensuite attribués aux politiques d'autorisation correspondantes qui ont été créées plus tôt et à une politique de confiance avec un fournisseur OIDC. Les rôles d'opérateur diffèrent des rôles à l'échelle du compte en ce qu'ils représentent en fin de compte les pods qui ont besoin d'accéder aux ressources AWS. Comme un utilisateur ne peut pas joindre les rôles IAM aux pods, il doit créer une politique de confiance avec un fournisseur OIDC afin que l'opérateur, et donc les pods, puissent accéder aux rôles dont ils ont besoin.

Lorsque l'utilisateur attribue les rôles aux autorisations de stratégie correspondantes, l'étape finale est de créer le fournisseur OIDC.

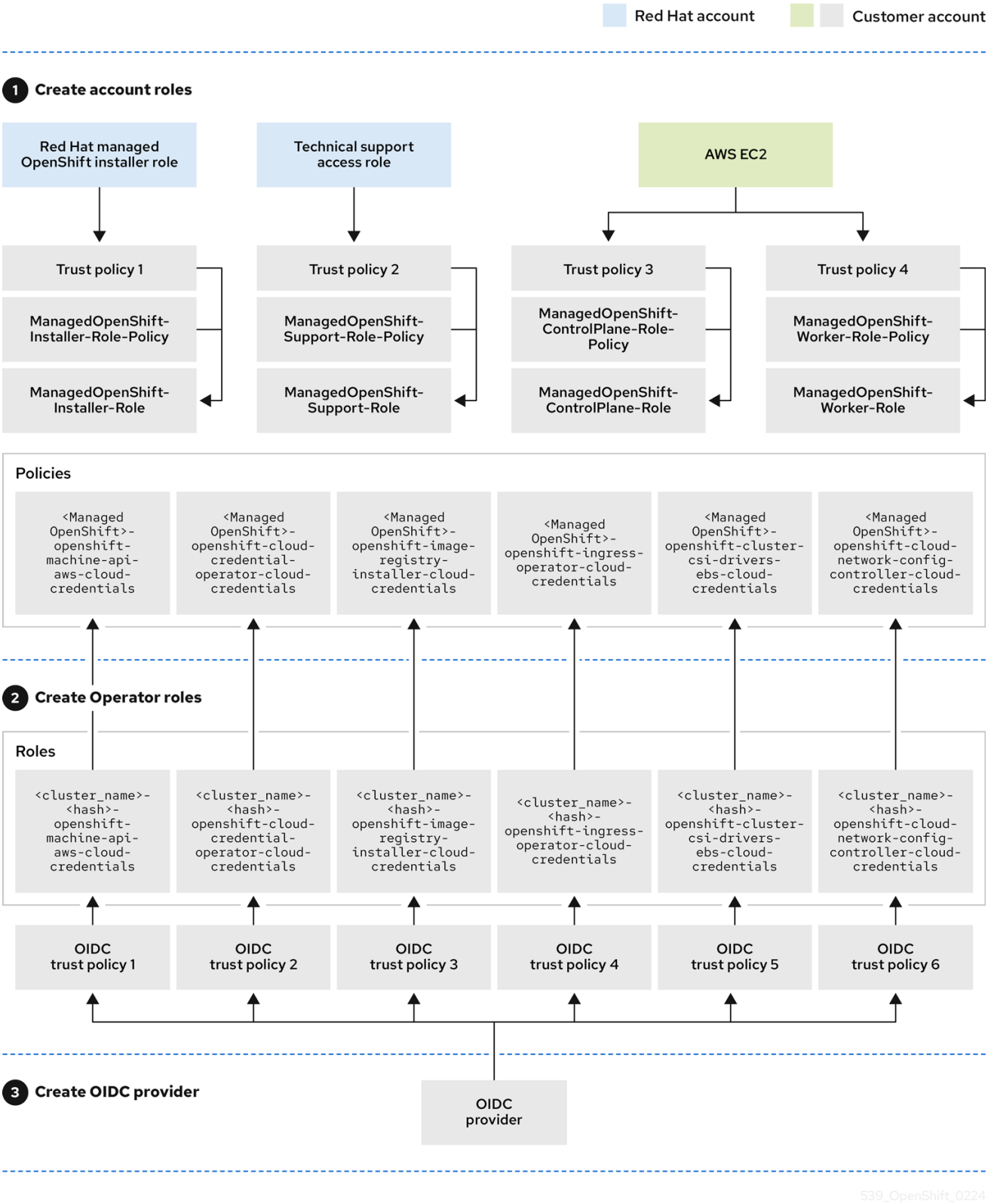


Lorsqu'un nouveau rôle est nécessaire, la charge de travail actuellement utilisée dans le rôle Red Hat

assumera le rôle dans le compte AWS, obtiendra des informations d'identification temporaires d'AWS STS et commencera à effectuer les actions à l'aide d'appels API dans le compte AWS du client, comme le permet la politique de permissions du rôle assumé. Les informations d'identification sont temporaires et ont une durée maximale d'une heure.

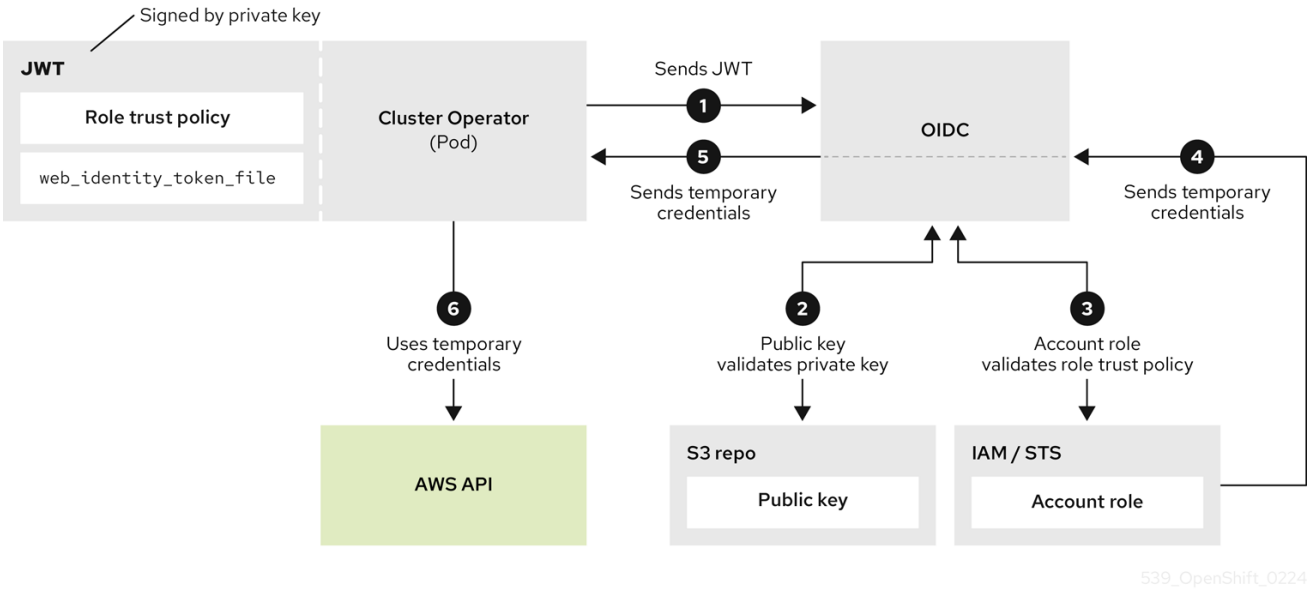


L'ensemble du flux de travail est représenté dans le graphique suivant:



Les opérateurs utilisent le processus suivant pour obtenir les informations d'identification requises pour effectuer leurs tâches. Chaque opérateur se voit attribuer un rôle d'opérateur, une politique d'autorisations et une politique de confiance avec un fournisseur OIDC. L'opérateur assumera le rôle en passant un jeton Web JSON qui contient le rôle et un fichier jeton (web_identity_token_file) au fournisseur OIDC, qui authentifie ensuite la clé signée avec une clé publique. La clé publique est créée lors de la création de clusters et stockée dans un seau S3. L'opérateur confirme ensuite que le sujet dans le fichier de jeton signé correspond au rôle dans la politique de confiance des rôles, ce qui garantit

que le fournisseur OIDC ne peut obtenir que le rôle autorisé. Le fournisseur OIDC retourne ensuite les informations d'identification temporaires à l'opérateur afin que l'opérateur puisse effectuer des appels API AWS. Dans le cas d'une représentation visuelle, voir ci-dessous:



17.2.7. Cas d'utilisation de ROSA avec STS

Créer des nœuds à l'installation du cluster

Le programme d'installation Red Hat utilise le rôle RH-Managed-OpenShift-Installer et une politique de confiance pour assumer le rôle Managed-OpenShift-Installer-Role dans le compte du client. Ce processus renvoie les informations d'identification temporaires d'AWS STS. Le programme d'installation commence à faire les appels API requis avec les informations d'identification temporaires reçues de STS. Le programme d'installation crée l'infrastructure requise dans AWS. Les informations d'identification expirent dans une heure et le programme d'installation n'a plus accès au compte du client.

Le même processus s'applique également aux cas de soutien. Dans les cas de soutien, un ingénieur de fiabilité du site Red Hat (SRE) remplace le programme d'installation.

Dimensionnement du cluster

L'opérateur machine-api utilise AssumeRoleWithWebIdentity pour assumer le rôle machine-api-aws-cloud-credentials. Cela lance la séquence pour que les opérateurs de cluster reçoivent les informations d'identification. Le rôle d'opérateur de machine-api peut maintenant faire les appels API pertinents pour ajouter plus d'instances EC2 au cluster.

17.3. DIDACTICIEL: CONCEPTS OPENSHIFT

17.3.1. La source à l'image (S2I)

La source à image (S2I) est une boîte à outils et un flux de travail pour créer des images de conteneurs reproductibles à partir du code source. Le S2I produit des images prêtes à l'emploi en insérant du code source dans une image de conteneur et en laissant le conteneur préparer le code source. En créant des images de constructeur auto-assemblant, vous pouvez versionner et contrôler vos environnements de construction exactement comme vous utilisez des images conteneur pour la version de vos environnements d'exécution.

Ressources supplémentaires

- Le projet source à image (S2I) en amont

17.3.1.1. Comment ça marche

Dans un langage dynamique tel que Ruby, les environnements de temps de construction et de temps d'exécution sont généralement les mêmes. En supposant que Ruby, Bundler, Rake, Apache, GCC et tous les autres paquets nécessaires pour configurer et exécuter une application Ruby sont déjà installés, une image de constructeur effectue les étapes suivantes:

1. L'image du constructeur démarre un conteneur avec la source de l'application injectée dans un répertoire connu.

2. Le processus de conteneur transforme ce code source en configuration exécutable appropriée. Il installe par exemple des dépendances avec Bundler et déplace le code source dans un répertoire où Apache a été préconfiguré pour rechercher le fichier de configuration Ruby.

3. Il engage ensuite le nouveau conteneur et définit le point d'entrée de l'image pour être un script qui va démarrer Apache pour héberger l'application Ruby.

Dans le cas des langages compilés tels que C, C++, Go ou Java, les dépendances nécessaires à la compilation peuvent être supérieures à la taille des artefacts d'exécution. Afin de garder les images d'exécution petites, S2I active un processus de construction en plusieurs étapes, où un artefact binaire tel qu'un fichier exécutable est créé dans la première image du constructeur, extrait et injecté dans une deuxième image d'exécution qui place simplement le programme exécutable à l'emplacement correct.

À titre d'exemple, créer un pipeline de construction reproductible pour Tomcat et Maven:

1. Créez une image constructeur contenant OpenJDK et Tomcat qui s'attend à ce qu'un fichier WAR soit injecté.

2. Créez une deuxième image qui couche au-dessus de la première image Maven et toute autre dépendance standard, et s'attend à ce qu'un projet Maven soit injecté.

3. Démarrez S2I à l'aide de la source de l'application Java et de l'image Maven pour créer la WAR de l'application souhaitée.

4. Démarrez S2I une deuxième fois en utilisant le fichier WAR de l'étape précédente et l'image Tomcat initiale pour créer l'image d'exécution.

En plaçant la logique de construction à l'intérieur des images et en combinant les images en plusieurs étapes, l'environnement d'exécution est proche de l'environnement de construction sans nécessiter le déploiement d'outils de construction à la production.

17.3.1.2. Avantages S2I

La reproductibilité

Permettre aux environnements de construction d'être rigoureusement versionnés en les encapsulant dans une image conteneur et en définissant une interface simple de code source injecté pour les appelants. Les constructions reproductibles sont une exigence clé pour permettre des mises à jour de sécurité et une intégration continue dans l'infrastructure conteneurisée, et les images du constructeur aident à assurer la répétabilité et la possibilité d'échanger des temps d'exécution.

Flexibilité

Chaque système de construction existant qui peut fonctionner sur Linux peut fonctionner à l'intérieur

d'un conteneur, et chaque constructeur individuel peut également faire partie d'un pipeline plus grand. Les scripts qui traitent le code source de l'application peuvent être injectés dans l'image du constructeur, permettant aux auteurs d'adapter les images existantes pour permettre la gestion de la source.

La vitesse

Au lieu de construire plusieurs couches dans un seul Dockerfile, S2I encourage les auteurs à représenter une application dans une seule couche d'image. Cela permet d'économiser du temps pendant la création et le déploiement et permet un meilleur contrôle sur la sortie de l'image finale.

La sécurité

Dockerfiles sont exécutés sans la plupart des contrôles opérationnels normaux des conteneurs. Ils s'exécutent généralement comme racine et ont accès au réseau de conteneurs. Le S2I peut contrôler les autorisations et privilèges disponibles pour l'image du constructeur depuis le lancement de la construction dans un seul conteneur. De concert avec des plateformes comme OpenShift, S2I permet aux administrateurs de contrôler les privilèges dont disposent les développeurs au moment de la construction.

17.3.2. Itinéraires

L'itinéraire OpenShift expose un service à un nom d'hôte afin que les clients externes puissent l'atteindre par leur nom. Lorsqu'un objet Route est créé sur OpenShift, il est récupéré par l'équilibreur de charge HAProxy intégré pour exposer le service demandé et le rendre disponible en externe avec la configuration donnée.

À l'instar de l'objet de Kubernetes Ingress, Red Hat a créé le concept de route pour combler un besoin et a ensuite contribué aux principes de conception derrière elle à la communauté, ce qui a fortement influencé le design de l'Ingress. L'itinéraire comporte des caractéristiques supplémentaires, comme on peut le voir dans le tableau suivant:

Caractéristique	Entrée sur OpenShift	Itinéraire sur OpenShift
Kubernetes objet standard	**A) X**	
Accès externe aux services	**A) X**	**A) X**
Des sessions persistantes (collantes)	**A) X**	**A) X**
Les stratégies d'équilibrage de la charge (par exemple, le vol à rondes)	**A) X**	**A) X**
Limite de vitesse et d'étroitesse	**A) X**	**A) X**
Liste blanche IP	**A) X**	**A) X**
Terminaison de bord TLS pour une sécurité améliorée	**A) X**	**A) X**
Le recryptage TLS pour une sécurité améliorée		**A) X**

Caractéristique	Entrée sur OpenShift	Itinéraire sur OpenShift
Le passhtrough TLS pour une sécurité améliorée		A) X
Backends pondérés multiples (trafic partagé)		A) X
Des noms d'hôte basés sur des modèles générés		A) X
Domaines wildcard		A) X

NOTE

La résolution DNS pour un nom d'hôte est gérée séparément du routage. Il se peut que votre administrateur ait configuré un domaine cloud qui se résoudra toujours correctement sur le routeur ou modifie indépendamment vos enregistrements DNS nom d'hôte indépendants pour le résoudre au routeur.

L'itinéraire individuel peut remplacer certaines valeurs par défaut en fournissant des configurations spécifiques dans ses annotations.

Ressources supplémentaires

- Annotations spécifiques à la route

17.3.3. Flux d'images

Le flux d'images stocke une cartographie des balises en images, des métadonnées qui sont appliquées lorsque les images sont étiquetées dans un flux, et une référence facultative à un référentiel d'images Docker sur un registre.

17.3.3.1. Avantages du flux d'images

En utilisant un flux d'images, il est plus facile de modifier une balise pour une image de conteneur. Dans le cas contraire, pour changer manuellement une balise, vous devez télécharger l'image, la modifier localement, puis tout repousser. La promotion des applications en modifiant manuellement une balise, puis la mise à jour de l'objet de déploiement implique de nombreuses étapes.

Avec les flux d'images, vous téléchargez une fois une image conteneur, puis vous gérez ses balises virtuelles en interne dans OpenShift. Dans un projet, vous pouvez utiliser la balise de développement et ne changer qu'une référence à elle en interne, tandis que dans la production, vous pouvez utiliser une balise de production et la gérer en interne. Il n'est pas nécessaire de traiter le registre.

Il est également possible d'utiliser des flux d'images en conjonction avec les configurations de déploiement pour définir un déclencheur qui démarrera un déploiement dès qu'une nouvelle image apparaît ou qu'une balise change de référence.

Ressources supplémentaires

- Blog Red Hat: Comment simplifier la gestion de l'image de conteneur dans Kubernetes avec OpenShift Image Streams

17.3.4. Constructions

La construction est le processus de transformation des paramètres d'entrée en un objet résultant. Le plus souvent, le processus est utilisé pour transformer les paramètres d'entrée ou le code source en une image exécutable. L'objet BuildConfig est la définition de l'ensemble du processus de construction.

La plate-forme de conteneurs OpenShift exploite Kubernetes en créant des conteneurs au format Docker à partir de la création d'images et en les poussant vers un registre d'images de conteneur.

Construire des objets partagent des caractéristiques communes:

- Entrées pour une construction
- Exigences pour compléter un processus de construction
- Journalisation du processus de construction
- La publication de ressources issues de constructions réussies
- La publication du statut final de la construction

Les builds profitent des restrictions de ressources, spécifiant des limites sur les ressources telles que l'utilisation du CPU, l'utilisation de la mémoire et le temps d'exécution de la construction ou du pod.

Ressources supplémentaires

- Comprendre les constructions d'images

17.4. DÉPLOIEMENT D'UN CLUSTER

17.4.1. Tutoriel : Choisir une méthode de déploiement

Ce tutoriel décrit les différentes façons de déployer un cluster. Choisissez la méthode de déploiement qui correspond le mieux à vos préférences et besoins.

17.4.1.1. Les options de déploiement

Et si vous voulez:

- Il n'y a que les commandes CLI nécessaires - Guide CLI simple
- Interface utilisateur - Guide simple de l'interface utilisateur
- Les commandes CLI avec des détails - Guide CLI détaillé
- Interface utilisateur avec détails - Guide d'interface utilisateur détaillée

Les options de déploiement ci-dessus fonctionnent bien pour ce tutoriel. Lorsque vous faites ce tutoriel pour la première fois, le guide CLI simple est la méthode la plus simple et recommandée.

17.4.2. Didacticiel: Guide CLI simple

Cette page décrit la liste minimale de commandes pour déployer un Red Hat OpenShift Service sur AWS (ROSA) cluster à l'aide de l'interface de ligne de commande (CLI).

NOTE

Bien que ce déploiement simple fonctionne bien pour un paramètre de tutoriel, les clusters utilisés dans la production devraient être déployés avec une méthode plus détaillée.

17.4.2.1. Conditions préalables

- Dans le tutoriel de configuration, vous avez rempli les conditions préalables.

17.4.2.2. Création de rôles de compte

Exécutez la commande suivante une fois pour chaque compte AWS et la version OpenShift y-stream:

```
rosa create account-roles --mode auto --yes
```

17.4.2.3. Déploiement du cluster

1. Créez le cluster avec la configuration par défaut en exécutant la commande suivante en remplaçant votre propre nom de cluster:

   ```
   rosa create cluster --cluster-name <cluster-name> --sts --mode auto --yes
   ```

2. Consultez l'état de votre cluster en exécutant la commande suivante:

   ```
   rosa list clusters
   ```

17.4.3. Guide détaillé de CLI

Ce tutoriel décrit les étapes détaillées pour déployer un cluster ROSA à l'aide du ROSA CLI.

17.4.3.1. Les modes de déploiement CLI

Il existe deux modes pour déployer un cluster ROSA. La première est automatique, qui est plus rapide et effectue le travail manuel pour vous. L'autre est manuel, vous oblige à exécuter des commandes supplémentaires et vous permet d'inspecter les rôles et les politiques créés. Ce tutoriel documente les deux options.

Lorsque vous souhaitez créer un cluster rapidement, utilisez l'option automatique. Lorsque vous préférez explorer les rôles et les politiques créés, utilisez l'option manuelle.

Choisissez le mode de déploiement en utilisant l'indicateur --mode dans les commandes pertinentes.

Les options valides pour --mode sont:

- le rôle et les stratégies sont créés et enregistrés dans le répertoire actuel. Il faut exécuter manuellement les commandes fournies comme étape suivante. Cette option vous permet de revoir la politique et les rôles avant de les créer.

- auto : Les rôles et les stratégies sont créés et appliqués automatiquement à l'aide du compte AWS actuel.

ASTUCE

Il est possible d'utiliser l'une ou l'autre méthode de déploiement pour ce tutoriel. Le mode automatique est plus rapide et a moins d'étapes.

17.4.3.2. Flux de travail de déploiement

Le flux de travail global de déploiement suit ces étapes:

1. créer des rôles de compte - Ceci n'est exécuté qu'une seule fois pour chaque compte. Lorsqu'ils ont été créés, les rôles de compte n'ont pas besoin d'être créés à nouveau pour plus de clusters de la même version y-stream.
2. **créer le cluster Rosa**
3. créer des rôles d'opérateur - Pour le mode manuel seulement.
4. créer oidc-fourr - Pour le mode manuel seulement.

Chaque cluster supplémentaire dans le même compte pour la même version y-stream, seule l'étape 2 est nécessaire pour le mode automatique. Les étapes 2 à 4 sont nécessaires pour le mode manuel.

17.4.3.3. Le mode automatique

Cette méthode est utilisée si vous souhaitez que le ROSA CLI automatise la création des rôles et des politiques pour créer rapidement votre cluster.

17.4.3.3.1. Création de rôles de compte

Lorsque c'est la première fois que vous déployez ROSA dans ce compte et que vous n'avez pas encore créé les rôles de compte, alors créez les rôles et les politiques à l'échelle du compte, y compris les politiques d'opérateur.

Exécutez la commande suivante pour créer les rôles à l'échelle du compte:

```
rosa create account-roles --mode auto --yes
```

Exemple de sortie

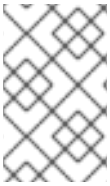
```
I: Creating roles using 'arn:aws:iam::000000000000:user/rosa-user'
I: Created role 'ManagedOpenShift-ControlPlane-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-ControlPlane-Role'
I: Created role 'ManagedOpenShift-Worker-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-Worker-Role'
I: Created role 'ManagedOpenShift-Support-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-Support-Role'
I: Created role 'ManagedOpenShift-Installer-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-Installer-Role'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-
machine-api-aws-cloud-credentials'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-cloud-
credential-operator-cloud-crede'
```

```
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-image-registry-installer-cloud-creden'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-ingress-operator-cloud-credentials'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-cluster-csi-drivers-ebs-cloud-credent'
I: To create a cluster with these roles, run the following command:
    rosa create cluster --sts
```

17.4.3.3.2. Créer un cluster

Exécutez la commande suivante pour créer un cluster avec toutes les options par défaut:

```
rosa create cluster --cluster-name <cluster-name> --sts --mode auto --yes
```



NOTE

Cela créera également les rôles d'opérateur requis et le fournisseur OIDC. Lorsque vous souhaitez voir toutes les options disponibles pour votre cluster, utilisez le drapeau --help ou --interactive pour le mode interactif.

Exemple d'entrée

```
$ rosa create cluster --cluster-name my-rosa-cluster --sts --mode auto --yes
```

Exemple de sortie

```
I: Creating cluster 'my-rosa-cluster'
I: To view a list of clusters and their status, run 'rosa list clusters'
I: Cluster 'my-rosa-cluster' has been created.
I: Once the cluster is installed you will need to add an Identity Provider before you can login into the cluster. See 'rosa create idp --help' for more information.
I: To determine when your cluster is Ready, run 'rosa describe cluster -c my-rosa-cluster'.
I: To watch your cluster installation logs, run 'rosa logs install -c my-rosa-cluster --watch'.
Name:                       my-rosa-cluster
ID:                         1mlhulb3bo0l54ojd0ji000000000000
External ID:
OpenShift Version:
Channel Group:              stable
DNS:                        my-rosa-cluster.ibhp.p1.openshiftapps.com
AWS Account:                000000000000
API URL:
Console URL:
Region:                     us-west-2
Multi-AZ:                   false
Nodes:
- Master:                   3
- Infra:                    2
- Compute:                  2
Network:
- Service CIDR:             172.30.0.0/16
- Machine CIDR:             10.0.0.0/16
- Pod CIDR:                 10.128.0.0/14
```

```
- Host Prefix:            /23
STS Role ARN:            arn:aws:iam::000000000000:role/ManagedOpenShift-Installer-Role
Support Role ARN:        arn:aws:iam::000000000000:role/ManagedOpenShift-Support-Role
Instance IAM Roles:
- Master:                arn:aws:iam::000000000000:role/ManagedOpenShift-ControlPlane-Role
- Worker:                arn:aws:iam::000000000000:role/ManagedOpenShift-Worker-Role
Operator IAM Roles:
- arn:aws:iam::000000000000:role/my-rosa-cluster-openshift-image-registry-installer-cloud-
credentials
- arn:aws:iam::000000000000:role/my-rosa-cluster-openshift-ingress-operator-cloud-credentials
- arn:aws:iam::000000000000:role/my-rosa-cluster-openshift-cluster-csi-drivers-ebs-cloud-credentials
- arn:aws:iam::000000000000:role/my-rosa-cluster-openshift-machine-api-aws-cloud-credentials
- arn:aws:iam::000000000000:role/my-rosa-cluster-openshift-cloud-credential-operator-cloud-
credential-oper
State:                   waiting (Waiting for OIDC configuration)
Private:                 No
Created:                 Oct 28 2021 20:28:09 UTC
Details Page:
https://console.redhat.com/openshift/details/s/1wupmiQy45xr1nN000000000000
OIDC Endpoint URL:       https://rh-oidc.s3.us-east-
1.amazonaws.com/1mlhulb3bo0l54ojd0ji000000000000
```

17.4.3.3.2.1. Configuration par défaut

Les paramètres par défaut sont les suivants:

- Les nœuds:
  - 3 nœuds de plan de contrôle
  - 2 nœuds d'infrastructure
  - 2 nœuds ouvriers
  - Aucune mise à l'échelle automatique
  - Consultez la documentation sur les instances ec2 pour plus de détails.
- Comme configuré pour l'aws CLI
- Gammes IP réseau:
  - CIDR machine: 10.0.0.0/16
  - CIDR de service: 172.30.0.0/16
  - Dose CIDR: 10.128.0.0/14
- Le nouveau VPC
- Clé AWS KMS par défaut pour le chiffrement
- La version la plus récente d'OpenShift disponible sur rosa
- D'une seule zone de disponibilité
- Groupe de travail public

17.4.3.3.3. Contrôle de l'état de l'installation

1. Exécutez l'une des commandes suivantes pour vérifier l'état de votre cluster:
   - Afin d'obtenir une vue détaillée de l'état, exécutez:

     ```
     rosa describe cluster --cluster <cluster-name>
     ```

   - Dans le cas d'une vue abrégée de l'état, courez:

     ```
     rosa list clusters
     ```

2. L'état du cluster passera de "attente" à "installation" à "prêt". Cela prendra environ 40 minutes.
3. Lorsque l'état change pour "prêt" votre cluster est installé.

17.4.3.4. Le mode manuel

Lorsque vous souhaitez revoir les rôles et les politiques avant de les appliquer à un cluster, utilisez la méthode manuelle. Cette méthode nécessite d'exécuter quelques commandes supplémentaires pour créer les rôles et les politiques.

Cette section utilise le mode --interactive. Consultez la documentation sur le mode interactif pour une description des champs de cette section.

17.4.3.4.1. Création de rôles de compte

1. Lorsque c'est la première fois que vous déployez ROSA dans ce compte et que vous n'avez pas encore créé les rôles de compte, créez les rôles et les politiques à l'échelle du compte, y compris les politiques d'opérateur. La commande crée les fichiers JSON nécessaires pour les rôles et stratégies requis pour votre compte dans le répertoire courant. Il affiche également les commandes aws CLI que vous devez exécuter pour créer ces objets.
   Exécutez la commande suivante pour créer les fichiers nécessaires et afficher les commandes supplémentaires:

   ```
   rosa create account-roles --mode manual
   ```

   Exemple de sortie

   ```
   I: All policy files saved to the current directory
   I: Run the following commands to create the account roles and policies:
   aws iam create-role \
   --role-name ManagedOpenShift-Worker-Role \
   --assume-role-policy-document file://sts_instance_worker_trust_policy.json \
   --tags Key=rosa_openshift_version,Value=4.8
   Key=rosa_role_prefix,Value=ManagedOpenShift
   Key=rosa_role_type,Value=instance_worker
   aws iam put-role-policy \
   --role-name ManagedOpenShift-Worker-Role \
   --policy-name ManagedOpenShift-Worker-Role-Policy \
   --policy-document file://sts_instance_worker_permission_policy.json
   ```

2. Consultez le contenu de votre répertoire actuel pour voir les nouveaux fichiers. L'Aws CLI permet de créer chacun de ces objets.

Exemple de sortie

```
$ ls
openshift_cloud_credential_operator_cloud_credential_operator_iam_ro_creds_policy.json
sts_instance_controlplane_permission_policy.json
openshift_cluster_csi_drivers_ebs_cloud_credentials_policy.json
sts_instance_controlplane_trust_policy.json
openshift_image_registry_installer_cloud_credentials_policy.json
sts_instance_worker_permission_policy.json
openshift_ingress_operator_cloud_credentials_policy.json
sts_instance_worker_trust_policy.json
openshift_machine_api_aws_cloud_credentials_policy.json
sts_support_permission_policy.json
sts_installer_permission_policy.json                    sts_support_trust_policy.json
sts_installer_trust_policy.json
```

3. Facultatif: Ouvrez les fichiers pour examiner ce que vous allez créer. À titre d'exemple, l'ouverture de sts_installer_permission_policy.json montre:

Exemple de sortie

```
$ cat sts_installer_permission_policy.json
        {
        "Version": "2012-10-17",
        "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:DescribeAutoScalingGroups",
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AssociateDhcpOptions",
                "ec2:AssociateRouteTable",
                "ec2:AttachInternetGateway",
                "ec2:AttachNetworkInterface",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                [...]
```

En outre, vous pouvez voir le contenu de la documentation des clusters ROSA sur les ressources de l'IAM.

4. Exécutez les commandes aws listées à l'étape 1. Il est possible de copier et coller si vous êtes dans le même répertoire que les fichiers JSON que vous avez créés.

17.4.3.4.2. Créer un cluster

1. Après que les commandes aws soient exécutées avec succès, exécutez la commande suivante pour commencer la création de cluster ROSA en mode interactif:

```
rosa create cluster --interactive --sts
```

Consultez la documentation ROSA pour une description des champs.

2. Aux fins de ce tutoriel, copiez puis entrez les valeurs suivantes:

```
Cluster name: my-rosa-cluster
OpenShift version: <choose version>
External ID (optional): <leave blank>
Operator roles prefix: <accept default>
Multiple availability zones: No
AWS region: <choose region>
PrivateLink cluster: No
Install into an existing VPC: No
Enable Customer Managed key: No
Compute nodes instance type: m5.xlarge
Enable autoscaling: No
Compute nodes: 2
Machine CIDR: <accept default>
Service CIDR: <accept default>
Pod CIDR: <accept default>
Host prefix: <accept default>
Encrypt etcd data (optional): No
Disable Workload monitoring: No
```

Exemple de sortie

```
I: Creating cluster 'my-rosa-cluster'
I: To create this cluster again in the future, you can run:
rosa create cluster --cluster-name my-rosa-cluster --role-arn
arn:aws:iam::000000000000:role/ManagedOpenShift-Installer-Role --support-role-arn
arn:aws:iam::000000000000:role/ManagedOpenShift-Support-Role --master-iam-role
arn:aws:iam::000000000000:role/ManagedOpenShift-ControlPlane-Role --worker-iam-role
arn:aws:iam::000000000000:role/ManagedOpenShift-Worker-Role --operator-roles-prefix
my-rosa-cluster --region us-west-2 --version 4.8.13 --compute-nodes 2 --machine-cidr
10.0.0.0/16 --service-cidr 172.30.0.0/16 --pod-cidr 10.128.0.0/14 --host-prefix 23
I: To view a list of clusters and their status, run 'rosa list clusters'
I: Cluster 'my-rosa-cluster' has been created.
I: Once the cluster is installed you will need to add an Identity Provider before you can login
into the cluster. See 'rosa create idp --help' for more information.
Name:                       my-rosa-cluster
ID:                         1t6i760dbum4mqltqh6o000000000000
External ID:
OpenShift Version:
Channel Group:              stable
DNS:                        my-rosa-cluster.abcd.p1.openshiftapps.com
AWS Account:                000000000000
API URL:
Console URL:
Region:                     us-west-2
Multi-AZ:                   false
Nodes:
- Control plane:            3
- Infra:                    2
- Compute:                  2
Network:
- Service CIDR:             172.30.0.0/16
- Machine CIDR:             10.0.0.0/16
- Pod CIDR:                 10.128.0.0/14
- Host Prefix:              /23
STS Role ARN:               arn:aws:iam::000000000000:role/ManagedOpenShift-Installer-Role
```

```
Support Role ARN:         arn:aws:iam::000000000000:role/ManagedOpenShift-Support-Role
Instance IAM Roles:
- Control plane:          arn:aws:iam::000000000000:role/ManagedOpenShift-ControlPlane-
Role
- Worker:                 arn:aws:iam::000000000000:role/ManagedOpenShift-Worker-Role
Operator IAM Roles:
- arn:aws:iam::000000000000:role/my-rosa-cluster-w7i6-openshift-ingress-operator-cloud-
credentials
- arn:aws:iam::000000000000:role/my-rosa-cluster-w7i6-openshift-cluster-csi-drivers-ebs-
cloud-credentials
- arn:aws:iam::000000000000:role/my-rosa-cluster-w7i6-openshift-cloud-network-config-
controller-cloud-cre
- arn:aws:iam::000000000000:role/my-rosa-cluster-openshift-machine-api-aws-cloud-
credentials
- arn:aws:iam::000000000000:role/my-rosa-cluster-openshift-cloud-credential-operator-cloud-
credentia
- arn:aws:iam::000000000000:role/my-rosa-cluster-openshift-image-registry-installer-cloud-
credential
State:                   waiting (Waiting for OIDC configuration)
Private:                 No
Created:                 Jul  1 2022 22:13:50 UTC
Details Page:
https://console.redhat.com/openshift/details/s/2BMQm8xz8Hq5yEN000000000000
OIDC Endpoint URL:       https://rh-oidc.s3.us-east-
1.amazonaws.com/1t6i760dbum4mqltqh6o000000000000
I: Run the following commands to continue the cluster creation:
rosa create operator-roles --cluster my-rosa-cluster
rosa create oidc-provider --cluster my-rosa-cluster
I: To determine when your cluster is Ready, run 'rosa describe cluster -c my-rosa-cluster'.
I: To watch your cluster installation logs, run 'rosa logs install -c my-rosa-cluster --watch'.
```

NOTE

L'état du cluster restera « en attente » jusqu'à ce que les deux prochaines étapes soient terminées.

17.4.3.4.3. Créer des rôles d'opérateur

1. L'étape ci-dessus produit les commandes suivantes à exécuter. Ces rôles doivent être créés une fois pour chaque cluster. Afin de créer les rôles exécutez la commande suivante:

```
rosa create operator-roles --mode manual --cluster <cluster-name>
```

Exemple de sortie

```
I: Run the following commands to create the operator roles:
   aws iam create-role \
      --role-name my-rosa-cluster-openshift-image-registry-installer-cloud-credentials \
      --assume-role-policy-document
file://operator_image_registry_installer_cloud_credentials_policy.json \
      --tags Key=rosa_cluster_id,Value=1mkesci269png3tck000000000000000
Key=rosa_openshift_version,Value=4.8 Key=rosa_role_prefix,Value=
Key=operator_namespace,Value=openshift-image-registry
Key=operator_name,Value=installer-cloud-credentials
```

```
    aws iam attach-role-policy \
        --role-name my-rosa-cluster-openshift-image-registry-installer-cloud-credentials \
        --policy-arn arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-image-
registry-installer-cloud-creden
    [...]
```

2. Exécutez chacune des commandes aws.

17.4.3.4.4. Création du fournisseur OIDC

1. Exécutez la commande suivante pour créer le fournisseur OIDC:

```
rosa create oidc-provider --mode manual --cluster <cluster-name>
```

2. Cela affiche les commandes aws que vous devez exécuter.

Exemple de sortie

```
I: Run the following commands to create the OIDC provider:
$ aws iam create-open-id-connect-provider \
--url https://rh-oidc.s3.us-east-1.amazonaws.com/1mkesci269png3tckknhh0rfs2da5fj9 \
--client-id-list openshift sts.amazonaws.com \
--thumbprint-list a9d53002e97e00e043244f3d170d000000000000

$ aws iam create-open-id-connect-provider \
--url https://rh-oidc.s3.us-east-1.amazonaws.com/1mkesci269png3tckknhh0rfs2da5fj9 \
--client-id-list openshift sts.amazonaws.com \
--thumbprint-list a9d53002e97e00e043244f3d170d000000000000
```

3. Le cluster va maintenant poursuivre le processus d'installation.

17.4.3.4.5. Contrôle de l'état de l'installation

1. Exécutez l'une des commandes suivantes pour vérifier l'état de votre cluster:
   - Afin d'obtenir une vue détaillée de l'état, exécutez:

     ```
     rosa describe cluster --cluster <cluster-name>
     ```

   - Dans le cas d'une vue abrégée de l'état, courez:

     ```
     rosa list clusters
     ```

2. L'état du cluster passera de "attente" à "installation" à "prêt". Cela prendra environ 40 minutes.
3. Lorsque l'état change pour "prêt" votre cluster est installé.

17.4.3.5. L'obtention de l'URL Red Hat Hybrid Cloud Console

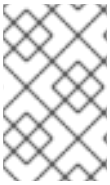
- Afin d'obtenir l'URL Hybrid Cloud Console, exécutez la commande suivante:

```
rosa describe cluster -c <cluster-name> | grep Console
```

Le cluster a maintenant été déployé avec succès. Le tutoriel suivant montre comment créer un utilisateur admin pour pouvoir utiliser le cluster immédiatement.

17.4.4. Guide de l'interface utilisateur simple

Cette page décrit la liste minimale de commandes pour déployer un cluster ROSA à l'aide de l'interface utilisateur (UI).



NOTE

Bien que ce déploiement simple fonctionne bien pour un paramètre de tutoriel, les clusters utilisés dans la production devraient être déployés avec une méthode plus détaillée.

17.4.4.1. Conditions préalables

- Dans le tutoriel de configuration, vous avez rempli les conditions préalables.

17.4.4.2. Création de rôles de compte

Exécutez la commande suivante une fois pour chaque compte AWS et la version OpenShift y-stream:

```
rosa create account-roles --mode auto --yes
```

17.4.4.3. Création de rôles Red Hat OpenShift Cluster Manager

1. Créez un rôle OpenShift Cluster Manager pour chaque compte AWS en exécutant la commande suivante:

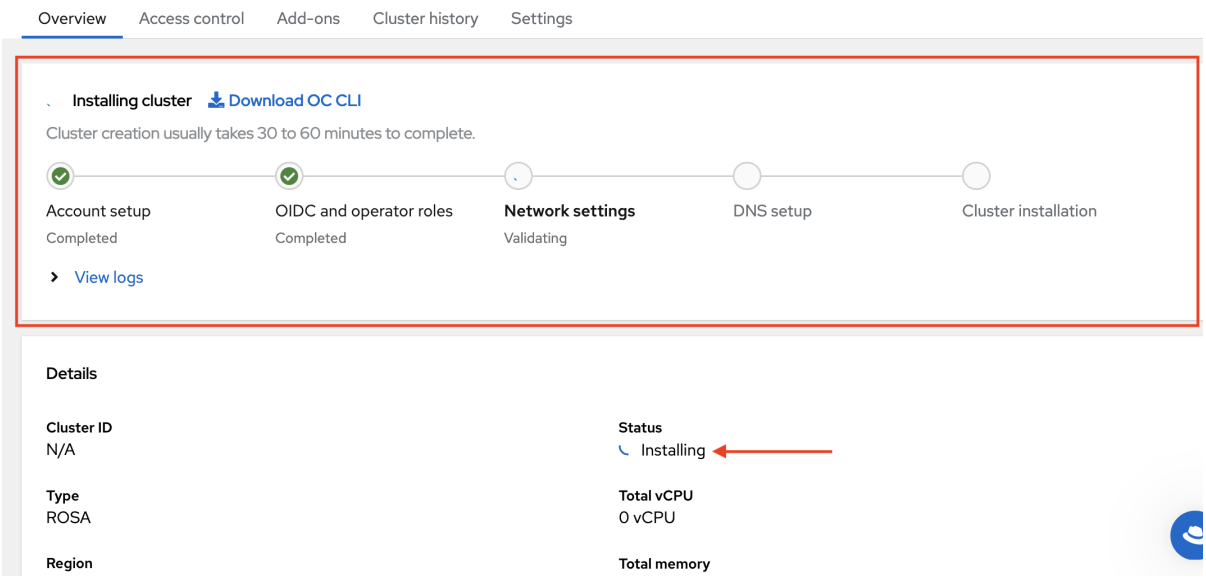
   ```
   rosa create ocm-role --mode auto --admin --yes
   ```

2. Créez un rôle utilisateur OpenShift Cluster Manager pour chaque compte AWS en exécutant la commande suivante:

   ```
   rosa create user-role --mode auto --yes
   ```

3. Le gestionnaire de cluster OpenShift vous permet de sélectionner votre compte AWS, les options de cluster et de commencer le déploiement.

4. L'interface utilisateur OpenShift Cluster Manager affiche l'état du cluster.



17.4.5. Guide détaillé de l'interface utilisateur

Ce tutoriel décrit les étapes détaillées pour déployer un Red Hat OpenShift Service sur AWS (ROSA) en utilisant l'interface utilisateur Red Hat OpenShift Cluster Manager (UI).

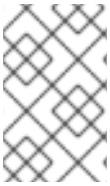
17.4.5.1. Flux de travail de déploiement

Le flux de travail global de déploiement suit ces étapes:

1. Créez les rôles et les politiques du compte.
2. Associez votre compte AWS à votre compte Red Hat.
    a. Créez et liez le rôle de Red Hat OpenShift Cluster Manager.
    b. Créer et lier le rôle de l'utilisateur.
3. Créez le cluster.

L'étape 1 ne doit être effectuée que la première fois que vous déployez un compte AWS. L'étape 2 ne doit être effectuée que la première fois que vous utilisez l'interface utilisateur. Il suffit de créer le cluster pour les clusters successifs d'une même version Y-stream.

17.4.5.2. Création de rôles larges de compte



NOTE

Lorsque vous avez déjà des rôles de compte lors d'un déploiement antérieur, sautez cette étape. L'interface utilisateur détectera vos rôles existants après avoir sélectionné un compte AWS associé.

Lorsque c'est la première fois que vous déployez ROSA dans ce compte et que vous n'avez pas encore créé les rôles de compte, créez les rôles et les politiques à l'échelle du compte, y compris les politiques d'opérateur.

- Dans votre terminal, exécutez la commande suivante pour créer les rôles à l'échelle du compte:

```
$ rosa create account-roles --mode auto --yes
```

Exemple de sortie

```
I: Creating roles using 'arn:aws:iam::000000000000:user/rosa-user'
I: Created role 'ManagedOpenShift-ControlPlane-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-ControlPlane-Role'
I: Created role 'ManagedOpenShift-Worker-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-Worker-Role'
I: Created role 'ManagedOpenShift-Support-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-Support-Role'
I: Created role 'ManagedOpenShift-Installer-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-Installer-Role'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-
machine-api-aws-cloud-credentials'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-
cloud-credential-operator-cloud-crede'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-
image-registry-installer-cloud-creden'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-
ingress-operator-cloud-credentials'
I: Created policy with ARN 'arn:aws:iam::000000000000:policy/ManagedOpenShift-openshift-
cluster-csi-drivers-ebs-cloud-credent'
I: To create a cluster with these roles, run the following command:
rosa create cluster --sts
```

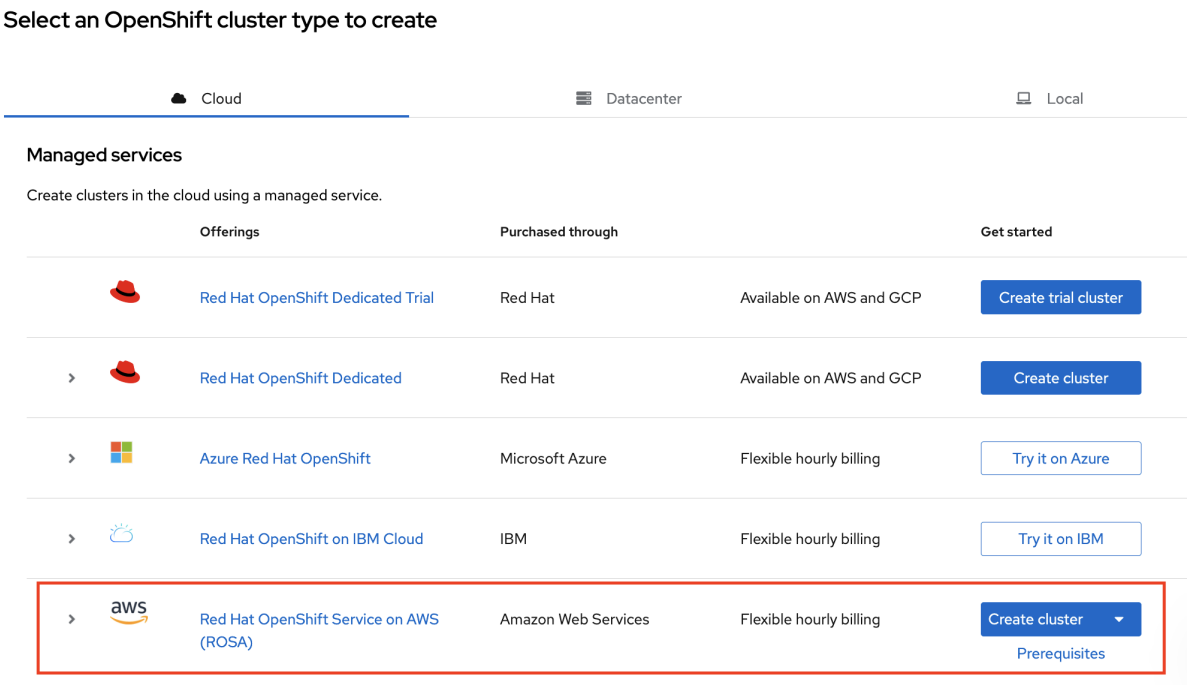
17.4.5.3. Associer votre compte AWS à votre compte Red Hat

Cette étape indique au gestionnaire de cluster OpenShift quel compte AWS vous souhaitez utiliser lors du déploiement de ROSA.

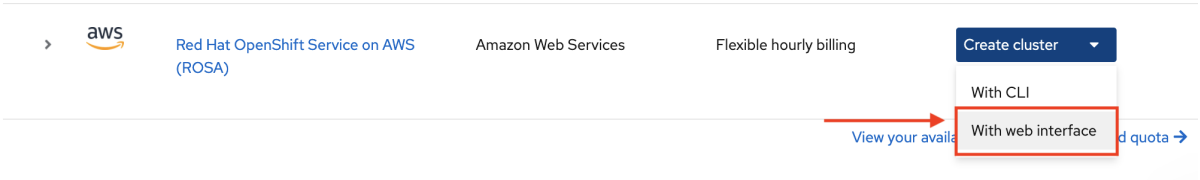
NOTE

Lorsque vous avez déjà associé vos comptes AWS, passez cette étape.

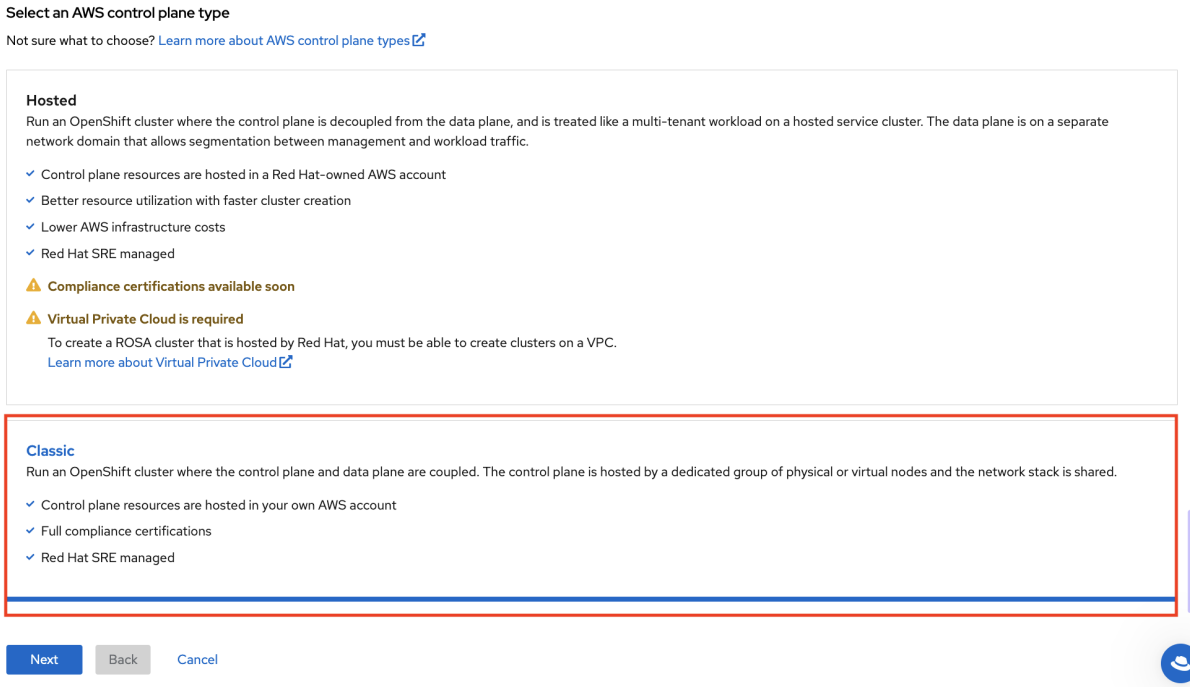
1. Ouvrez la console de cloud hybride Red Hat en visitant le gestionnaire de cluster OpenShift et en vous connectant à votre compte Red Hat.
2. Cliquez sur Créer un cluster.
3. Faites défiler vers le bas jusqu'à la ligne Red Hat OpenShift Service sur AWS (ROSA) et cliquez sur Créer un cluster.



4. Le menu déroulant s'affiche. Cliquez avec l'interface Web.

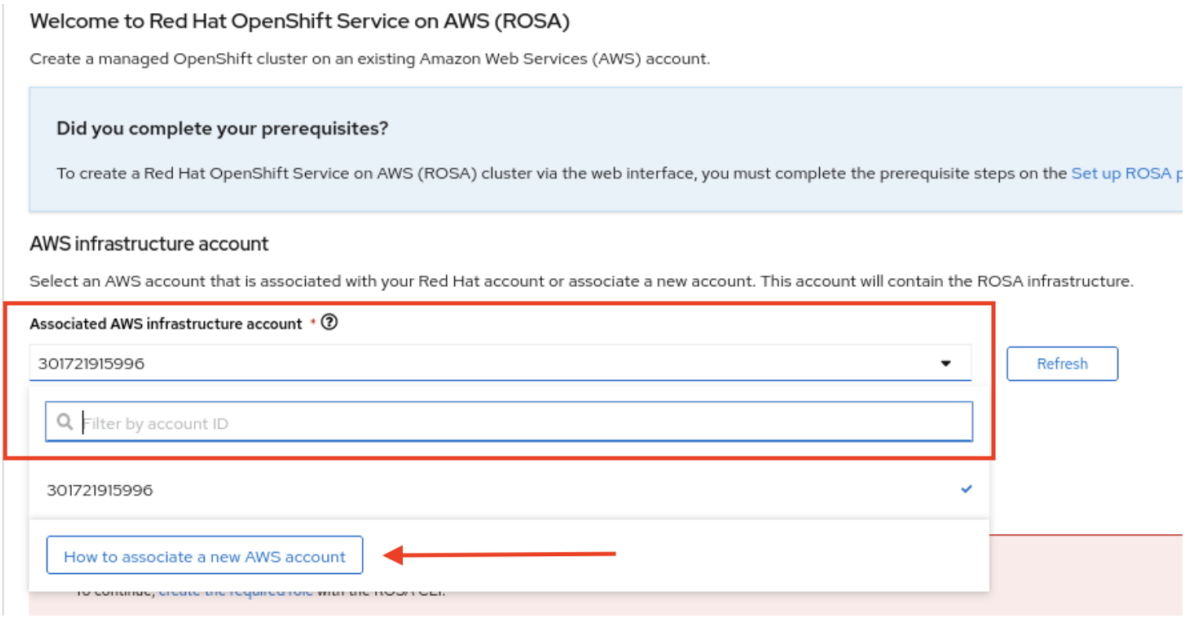


5. Dans « Sélectionnez un type de plan de contrôle AWS », choisissez Classic. Cliquez ensuite sur Suivant.

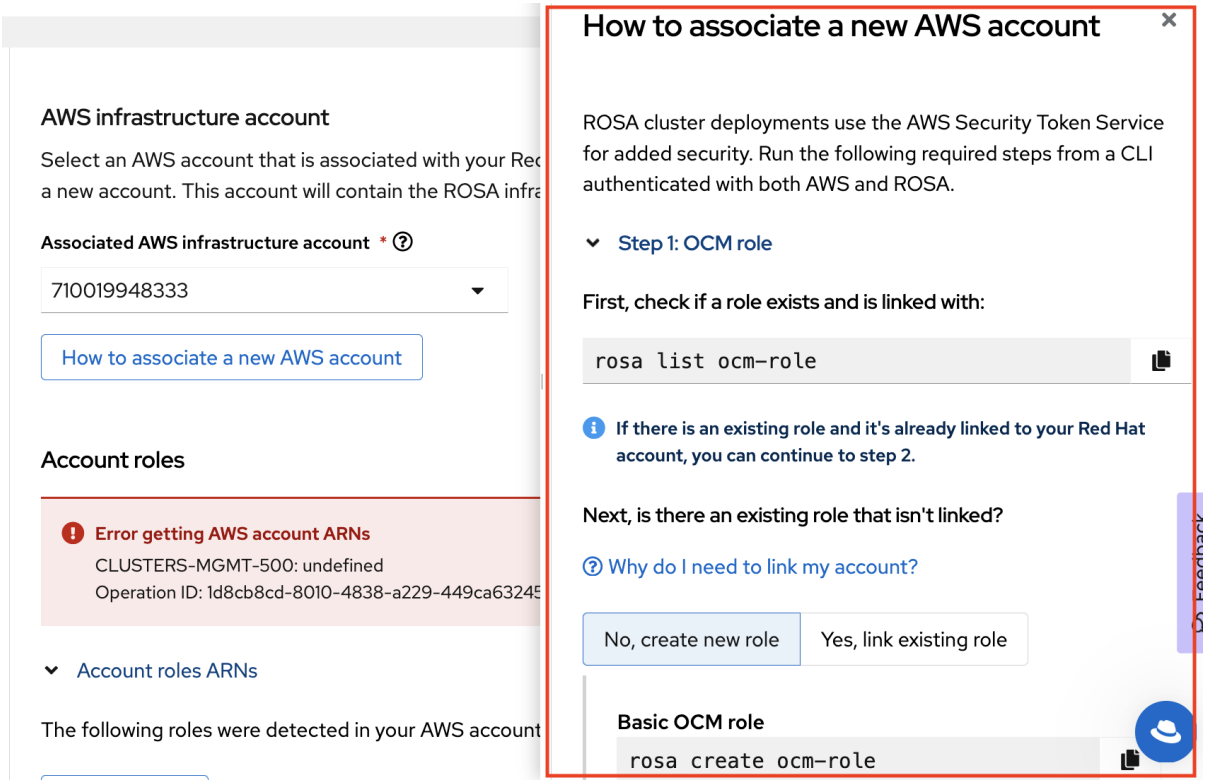


6. Cliquez sur la boîte de dépôt sous compte d'infrastructure AWS associé. Dans le cas où vous n'avez pas encore associé de comptes AWS, la dropbox peut être vide.

7. Cliquez sur Comment associer un nouveau compte AWS.



8. La barre latérale apparaît avec des instructions pour associer un nouveau compte AWS.



17.4.5.4. Créer et associer un rôle de gestionnaire de cluster OpenShift

1. Exécutez la commande suivante pour voir si un rôle OpenShift Cluster Manager existe:

```
$ rosa list ocm-role
```

2. L'interface utilisateur affiche les commandes pour créer un rôle OpenShift Cluster Manager avec deux niveaux différents d'autorisations:

   - Base OpenShift Cluster Manager : Permet au gestionnaire de cluster OpenShift d'avoir accès en lecture seule au compte pour vérifier si les rôles et les politiques requis par ROSA sont présents avant de créer un cluster. Il vous faudra créer manuellement les rôles, les stratégies et le fournisseur OIDC requis à l'aide du CLI.

- Admin OpenShift Cluster Manager: octroie à OpenShift Cluster Manager des autorisations supplémentaires pour créer les rôles, les politiques et le fournisseur OIDC requis pour ROSA. Cela rend le déploiement d'un cluster ROSA plus rapide puisque le gestionnaire de cluster OpenShift sera en mesure de créer les ressources nécessaires pour vous.
  En savoir plus sur ces rôles, consultez la section des rôles et autorisations OpenShift Cluster Manager de la documentation.

  Aux fins de ce tutoriel, utilisez le rôle d'administrateur OpenShift Cluster Manager pour l'approche la plus simple et la plus rapide.

3. Copiez la commande pour créer le rôle Admin OpenShift Cluster Manager à partir de la barre latérale ou basculez sur votre terminal et entrez la commande suivante:

```
$ rosa create ocm-role --mode auto --admin --yes
```

Cette commande crée le rôle OpenShift Cluster Manager et l'associe à votre compte Red Hat.

Exemple de sortie

```
I: Creating ocm role
I: Creating role using 'arn:aws:iam::000000000000:user/rosa-user'
I: Created role 'ManagedOpenShift-OCM-Role-12561000' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-OCM-Role-12561000'
I: Linking OCM role
I: Successfully linked role-arn 'arn:aws:iam::000000000000:role/ManagedOpenShift-OCM-
Role-12561000' with organization account '1MpZfntsZeUdjWHg7XRgP000000'
```

4. Cliquez sur l'étape 2: rôle de l'utilisateur.

17.4.5.4.1. Autres options de création de rôles OpenShift Cluster Manager

- Le mode manuel : Si vous préférez exécuter vous-même les commandes AWS CLI, vous pouvez définir le mode comme étant manuel plutôt que automatique. Le CLI affichera les commandes AWS et les fichiers JSON pertinents sont créés dans le répertoire actuel.
  En mode manuel, utilisez la commande suivante pour créer le rôle OpenShift Cluster Manager:

```
$ rosa create ocm-role --mode manual --admin --yes
```

- Basic OpenShift Cluster Manager: Si vous préférez que le gestionnaire de cluster OpenShift ait lu uniquement l'accès au compte, créez un rôle de gestionnaire de cluster OpenShift de base. Ensuite, vous devrez créer manuellement les rôles, les stratégies et le fournisseur OIDC requis à l'aide du CLI.
  Créez la commande suivante pour créer un rôle de gestionnaire de cluster OpenShift Basic:

```
$ rosa create ocm-role --mode auto --yes
```

17.4.5.5. Création d'un rôle utilisateur OpenShift Cluster Manager

Comme défini dans la documentation du rôle utilisateur, le rôle d'utilisateur doit être créé afin que ROSA puisse vérifier votre identité AWS. Ce rôle n'a pas d'autorisations, et il n'est utilisé que pour créer une relation de confiance entre le compte du programme d'installation et vos ressources de rôles OpenShift Cluster Manager.

1. Vérifiez si un rôle d'utilisateur existe déjà en exécutant la commande suivante:

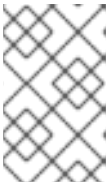
```
$ rosa list user-role
```

2. Exécutez la commande suivante pour créer le rôle de l'utilisateur et le lier à votre compte Red Hat:

```
$ rosa create user-role --mode auto --yes
```

Exemple de sortie

```
I: Creating User role
I: Creating ocm user role using 'arn:aws:iam::000000000000:user/rosa-user'
I: Created role 'ManagedOpenShift-User-rosa-user-Role' with ARN
'arn:aws:iam::000000000000:role/ManagedOpenShift-User-rosa-user-Role'
I: Linking User role
I: Successfully linked role ARN 'arn:aws:iam::000000000000:role/ManagedOpenShift-User-
rosa-user-Role' with account '1rbOQez0z5j1YoIInhcXY000000'
```



NOTE

Comme avant, vous pouvez définir --mode manuel si vous préférez exécuter vous-même les commandes AWS CLI. Les commandes AWS et les fichiers JSON pertinents sont créés dans le répertoire actuel. Assurez-vous de lier le rôle.

3. Cliquez sur l'étape 3: Rôles de compte.

17.4.5.6. Création de rôles de compte

1. Créez les rôles de votre compte en exécutant la commande suivante:

```
$ rosa create account-roles --mode auto
```

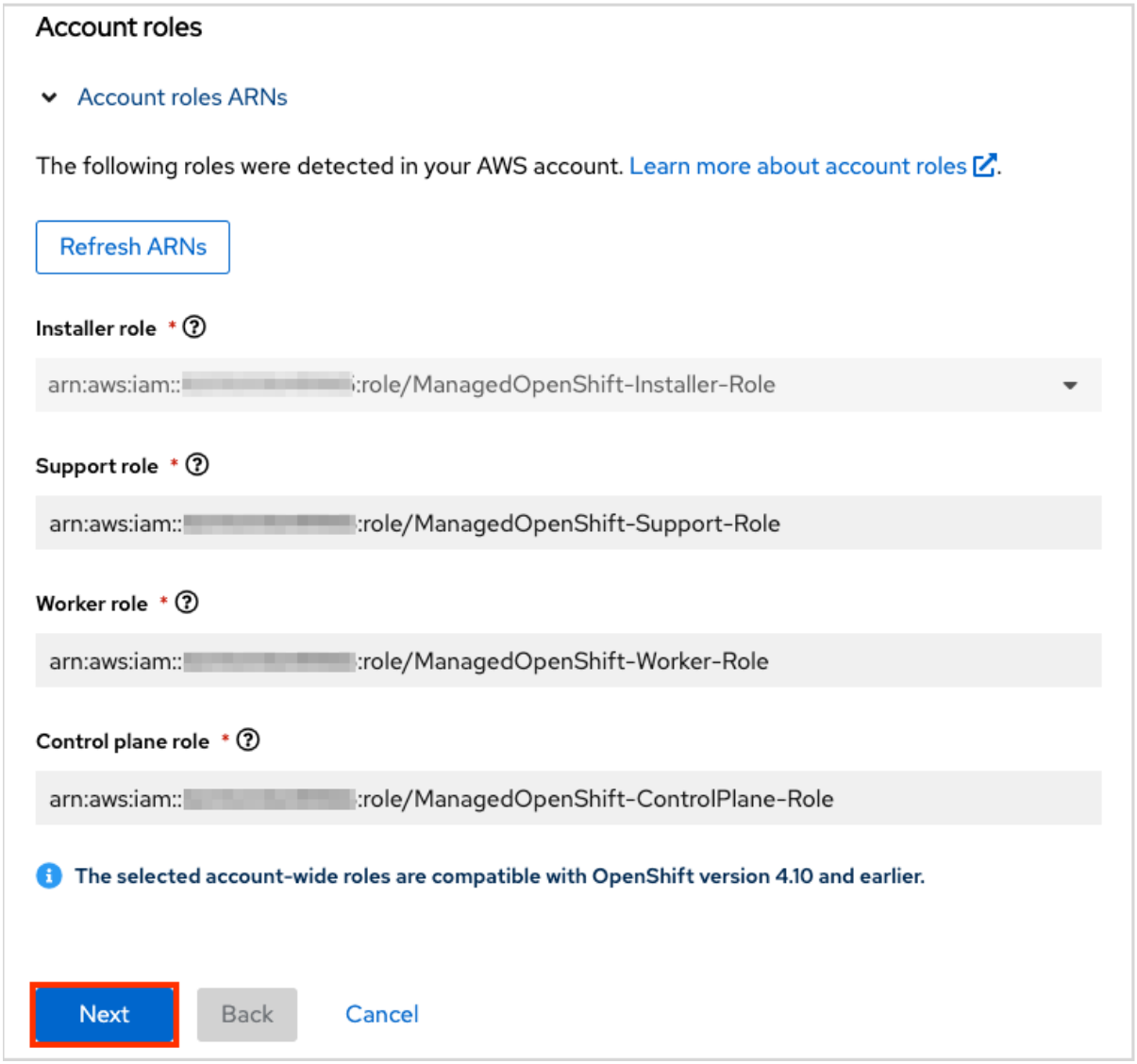
2. Cliquez sur OK pour fermer la barre latérale.

17.4.5.7. Confirmation de la réussite de l'association de comptes

1. Consultez maintenant votre compte AWS dans le menu déroulant du compte d'infrastructure AWS associé. Lorsque vous voyez votre compte, l'association de compte a été couronnée de succès.

2. Choisissez le compte.

3. Le rôle du compte ARN est peuplé ci-dessous.



4. Cliquez sur **Next**.

17.4.5.8. Créer le cluster

1. Aux fins de ce tutoriel, faites les sélections suivantes:

   Configurations du cluster

   - Cluster name: <pick a name\>
   - Dernière version: <sélectionner la dernière version\>
   - La région: <sélectionner la région\>
   - Disponibilité: Zone unique
   - Activer la surveillance de la charge de travail de l'utilisateur: laisser vérifié
   - Activer le chiffrement etcd supplémentaire: laisser non coché
   - Chiffrer les volumes persistants avec les clés client: laisser non coché

2. Cliquez sur **Next**.

3. Laissez les paramètres par défaut pour le pool de machines:

   Paramètres de pool de machines par défaut

   - Calculer le type d'instance du nœud: m5.xlarge - 4 vCPU 16 GiB RAM
   - Activer l'autoscaling: non vérifié
   - Calcul du nombre de nœuds: 2
   - Laisser les étiquettes de nœud vierges

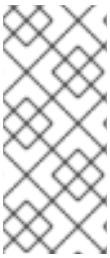
4. Cliquez sur **Next**.

17.4.5.8.1. Le réseautage

1. Laissez toutes les valeurs par défaut pour la configuration.
2. Cliquez sur **Next**.
3. Laissez toutes les valeurs par défaut pour les plages CIDR.
4. Cliquez sur **Next**.

17.4.5.8.2. Les rôles et les politiques des clusters

Dans ce tutoriel, laissez Auto sélectionné. Cela rendra le processus de déploiement du cluster plus simple et plus rapide.



NOTE

Lorsque vous avez sélectionné le rôle Basic OpenShift Cluster Manager plus tôt, vous pouvez uniquement utiliser le mode manuel. Il faut créer manuellement les rôles d'opérateur et le fournisseur OIDC. Consultez la section « Basic OpenShift Cluster Manager » ci-dessous après avoir terminé la section « Mises à jour du cluster » et commencé la création de cluster.

17.4.5.8.3. Les mises à jour des clusters

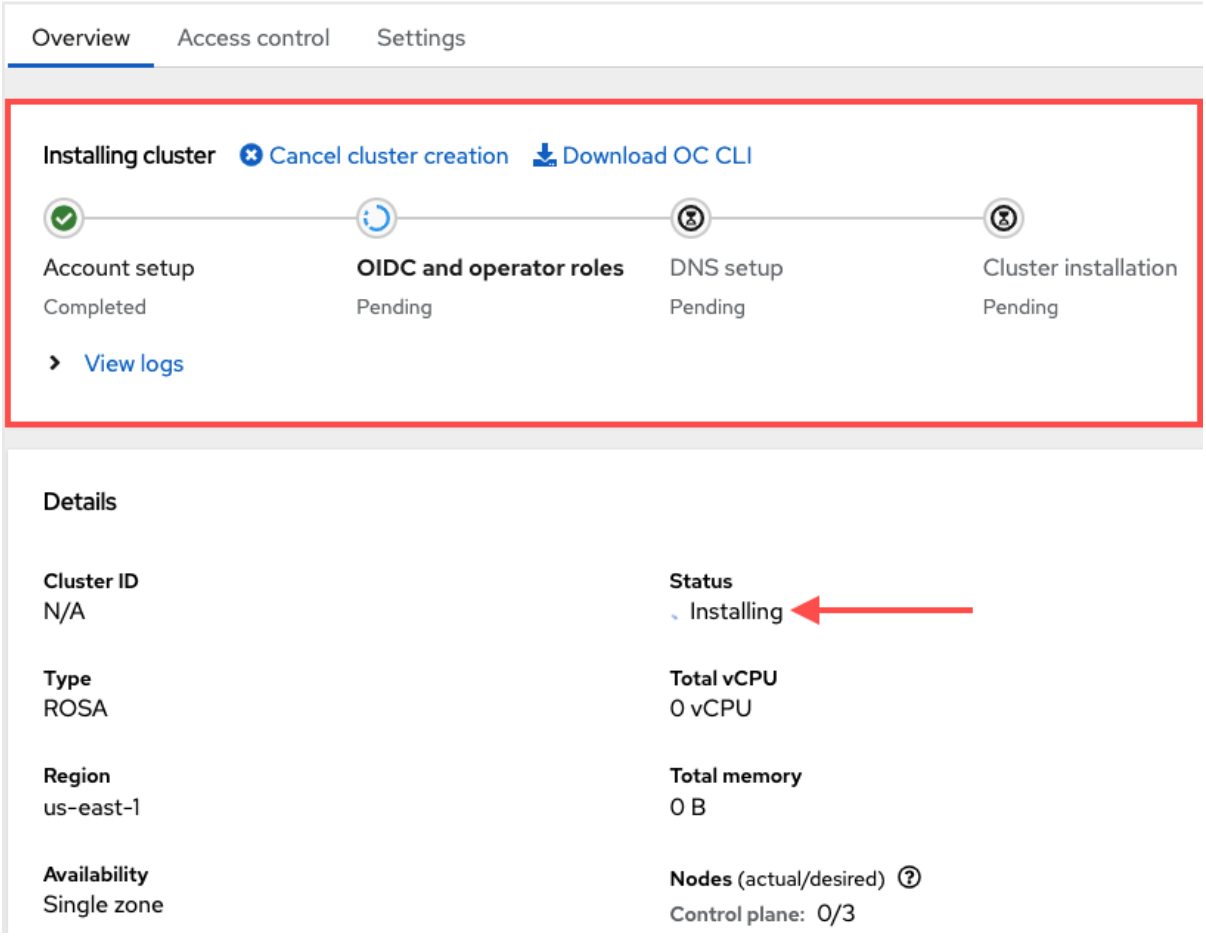
- Laissez toutes les options par défaut dans cette section.

17.4.5.8.4. Examiner et créer votre cluster

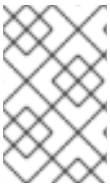
1. Examinez le contenu de la configuration du cluster.
2. Cliquez sur Créer un cluster.

17.4.5.8.5. Contrôle de l'état d'avancement de l'installation

- Demeurez sur la page actuelle pour suivre l'avancement de l'installation. Cela devrait prendre environ 40 minutes.



17.4.5.9. Base OpenShift Cluster Manager Rôle



NOTE

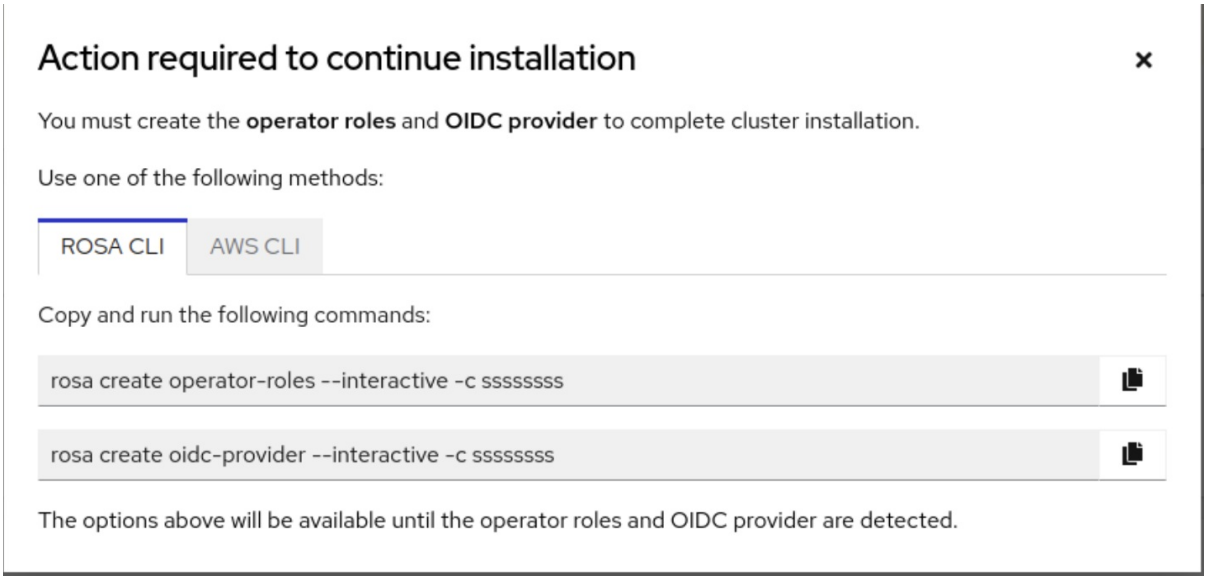
Lorsque vous avez créé un rôle d'administrateur OpenShift Cluster Manager comme indiqué ci-dessus, ignorez toute cette section. Le gestionnaire de cluster OpenShift créera les ressources pour vous.

Lorsque vous avez créé un rôle de base OpenShift Cluster Manager plus tôt, vous devrez créer manuellement deux autres éléments avant que l'installation du cluster puisse continuer:

- Les rôles d'opérateur
- Fournisseur OIDC

17.4.5.9.1. Créer des rôles d'opérateur

1. La fenêtre pop up vous montrera les commandes à exécuter.



2. Exécutez les commandes depuis la fenêtre de votre terminal pour lancer le mode interactif. Exécutez la commande suivante pour créer les rôles d'opérateur:

```
$ rosa create operator-roles --mode auto --cluster <cluster-name> --yes
```

Exemple de sortie

```
I: Creating roles using 'arn:aws:iam::000000000000:user/rosauser'
I: Created role 'rosacluster-b736-openshift-ingress-operator-cloud-credentials' with ARN
'arn:aws:iam::000000000000:role/rosacluster-b736-openshift-ingress-operator-cloud-
credentials'
I: Created role 'rosacluster-b736-openshift-cluster-csi-drivers-ebs-cloud-credent' with ARN
'arn:aws:iam::000000000000:role/rosacluster-b736-openshift-cluster-csi-drivers-ebs-cloud-
credent'
I: Created role 'rosacluster-b736-openshift-cloud-network-config-controller-cloud' with ARN
'arn:aws:iam::000000000000:role/rosacluster-b736-openshift-cloud-network-config-controller-
cloud'
I: Created role 'rosacluster-b736-openshift-machine-api-aws-cloud-credentials' with ARN
'arn:aws:iam::000000000000:role/rosacluster-b736-openshift-machine-api-aws-cloud-
credentials'
I: Created role 'rosacluster-b736-openshift-cloud-credential-operator-cloud-crede' with ARN
'arn:aws:iam::000000000000:role/rosacluster-b736-openshift-cloud-credential-operator-
cloud-crede'
I: Created role 'rosacluster-b736-openshift-image-registry-installer-cloud-creden' with ARN
'arn:aws:iam::000000000000:role/rosacluster-b736-openshift-image-registry-installer-cloud-
creden'
```

17.4.5.9.2. Création du fournisseur OIDC

- Dans votre terminal, exécutez la commande suivante pour créer le fournisseur OIDC:

```
$ rosa create oidc-provider --mode auto --cluster <cluster-name> --yes
```

Exemple de sortie

```
I: Creating OIDC provider using 'arn:aws:iam::000000000000:user/rosauser'
I: Created OIDC provider with ARN 'arn:aws:iam::000000000000:oidc-provider/rh-oidc.s3.us-east-1.amazonaws.com/1tt4kvrr2kha2rgs8gjfvf0000000000'
```

17.4.6. Guide du plan de contrôle hébergé (HCP)

Découvrez cet atelier pour déployer un échantillon de Red Hat OpenShift Service sur AWS (ROSA) avec un cluster de plans de contrôle hébergés (HCP). Ensuite, vous pouvez utiliser votre cluster dans les tutoriels suivants.

Les objectifs du tutoriel

- Apprenez à créer vos prérequis de cluster:
  - Créer un exemple de cloud privé virtuel (VPC)
  - Créer un échantillon de ressources OpenID Connect (OIDC)
- Créer des variables d'environnement d'échantillon
- Déployer un échantillon de cluster ROSA

Conditions préalables

- La version 1.2.31 de ROSA ou ultérieure
- Interface de ligne de commande Amazon Web Service (AWS) (CLI)
- CLI ROSA (rosa)

17.4.6.1. Créer vos prérequis de cluster

Avant de déployer un ROSA avec le cluster HCP, vous devez disposer à la fois d'un VPC et d'une ressource OIDC. D'abord, nous allons créer ces ressources. La ROSA utilise votre propre modèle VPC (BYO-VPC).

17.4.6.1.1. Créer un VPC

1. Assurez-vous que votre AWS CLI (aws) est configuré pour utiliser une région où ROSA est disponible. Consultez les régions prises en charge par AWS CLI en exécutant la commande suivante:

```
$ rosa list regions --hosted-cp
```

2. Créez le VPC. Dans ce tutoriel, le script suivant crée le VPC et ses composants requis. Il utilise la région configurée dans votre CLI aws.

```
#!/bin/bash

set -e
##########
# This script will create the network requirements for a ROSA cluster. This will be
# a public cluster. This creates:
# - VPC
```

```
# - Public and private subnets
# - Internet Gateway
# - Relevant route tables
# - NAT Gateway
#
# This will automatically use the region configured for the aws cli
#
##########

VPC_CIDR=10.0.0.0/16
PUBLIC_CIDR_SUBNET=10.0.1.0/24
PRIVATE_CIDR_SUBNET=10.0.0.0/24

# Create VPC
echo -n "Creating VPC..."
VPC_ID=$(aws ec2 create-vpc --cidr-block $VPC_CIDR --query Vpc.VpcId --output text)

# Create tag name
aws ec2 create-tags --resources $VPC_ID --tags Key=Name,Value=$CLUSTER_NAME

# Enable dns hostname
aws ec2 modify-vpc-attribute --vpc-id $VPC_ID --enable-dns-hostnames
echo "done."

# Create Public Subnet
echo -n "Creating public subnet..."
PUBLIC_SUBNET_ID=$(aws ec2 create-subnet --vpc-id $VPC_ID --cidr-block
$PUBLIC_CIDR_SUBNET --query Subnet.SubnetId --output text)

aws ec2 create-tags --resources $PUBLIC_SUBNET_ID --tags
Key=Name,Value=$CLUSTER_NAME-public
echo "done."

# Create private subnet
echo -n "Creating private subnet..."
PRIVATE_SUBNET_ID=$(aws ec2 create-subnet --vpc-id $VPC_ID --cidr-block
$PRIVATE_CIDR_SUBNET --query Subnet.SubnetId --output text)

aws ec2 create-tags --resources $PRIVATE_SUBNET_ID --tags
Key=Name,Value=$CLUSTER_NAME-private
echo "done."

# Create an internet gateway for outbound traffic and attach it to the VPC.
echo -n "Creating internet gateway..."
IGW_ID=$(aws ec2 create-internet-gateway --query InternetGateway.InternetGatewayId --
output text)
echo "done."

aws ec2 create-tags --resources $IGW_ID --tags Key=Name,Value=$CLUSTER_NAME

aws ec2 attach-internet-gateway --vpc-id $VPC_ID --internet-gateway-id $IGW_ID >
/dev/null 2>&1
echo "Attached IGW to VPC."

# Create a route table for outbound traffic and associate it to the public subnet.
echo -n "Creating route table for public subnet..."
```

```
PUBLIC_ROUTE_TABLE_ID=$(aws ec2 create-route-table --vpc-id $VPC_ID --query
RouteTable.RouteTableId --output text)

aws ec2 create-tags --resources $PUBLIC_ROUTE_TABLE_ID --tags
Key=Name,Value=$CLUSTER_NAME
echo "done."

aws ec2 create-route --route-table-id $PUBLIC_ROUTE_TABLE_ID --destination-cidr-block
0.0.0.0/0 --gateway-id $IGW_ID > /dev/null 2>&1
echo "Created default public route."

aws ec2 associate-route-table --subnet-id $PUBLIC_SUBNET_ID --route-table-id
$PUBLIC_ROUTE_TABLE_ID > /dev/null 2>&1
echo "Public route table associated"

# Create a NAT gateway in the public subnet for outgoing traffic from the private network.
echo -n "Creating NAT Gateway..."
NAT_IP_ADDRESS=$(aws ec2 allocate-address --domain vpc --query AllocationId --output
text)

NAT_GATEWAY_ID=$(aws ec2 create-nat-gateway --subnet-id $PUBLIC_SUBNET_ID --
allocation-id $NAT_IP_ADDRESS --query NatGateway.NatGatewayId --output text)

aws ec2 create-tags --resources $NAT_IP_ADDRESS --resources $NAT_GATEWAY_ID --
tags Key=Name,Value=$CLUSTER_NAME
sleep 10
echo "done."

# Create a route table for the private subnet to the NAT gateway.
echo -n "Creating a route table for the private subnet to the NAT gateway..."
PRIVATE_ROUTE_TABLE_ID=$(aws ec2 create-route-table --vpc-id $VPC_ID --query
RouteTable.RouteTableId --output text)

aws ec2 create-tags --resources $PRIVATE_ROUTE_TABLE_ID $NAT_IP_ADDRESS --
tags Key=Name,Value=$CLUSTER_NAME-private

aws ec2 create-route --route-table-id $PRIVATE_ROUTE_TABLE_ID --destination-cidr-block
0.0.0.0/0 --gateway-id $NAT_GATEWAY_ID > /dev/null 2>&1

aws ec2 associate-route-table --subnet-id $PRIVATE_SUBNET_ID --route-table-id
$PRIVATE_ROUTE_TABLE_ID > /dev/null 2>&1

echo "done."

# echo "***********VARIABLE VALUES*********"
# echo "VPC_ID="$VPC_ID
# echo "PUBLIC_SUBNET_ID="$PUBLIC_SUBNET_ID
# echo "PRIVATE_SUBNET_ID="$PRIVATE_SUBNET_ID
# echo "PUBLIC_ROUTE_TABLE_ID="$PUBLIC_ROUTE_TABLE_ID
# echo "PRIVATE_ROUTE_TABLE_ID="$PRIVATE_ROUTE_TABLE_ID
# echo "NAT_GATEWAY_ID="$NAT_GATEWAY_ID
# echo "IGW_ID="$IGW_ID
# echo "NAT_IP_ADDRESS="$NAT_IP_ADDRESS

echo "Setup complete."
echo ""
```

```
echo "To make the cluster create commands easier, please run the following commands to
set the environment variables:"
echo "export PUBLIC_SUBNET_ID=$PUBLIC_SUBNET_ID"
echo "export PRIVATE_SUBNET_ID=$PRIVATE_SUBNET_ID"
```

Ressources supplémentaires

- En savoir plus sur les exigences de VPC, consultez la documentation VPC.

3. Le script sort les commandes. Définissez les commandes comme variables d'environnement pour stocker les ID de sous-réseau pour une utilisation ultérieure. Copiez et exécutez les commandes:

```
$ export PUBLIC_SUBNET_ID=$PUBLIC_SUBNET_ID
$ export PRIVATE_SUBNET_ID=$PRIVATE_SUBNET_ID
```

4. Confirmez vos variables d'environnement en exécutant la commande suivante:

```
$ echo "Public Subnet: $PUBLIC_SUBNET_ID"; echo "Private Subnet:
$PRIVATE_SUBNET_ID"
```

Exemple de sortie

```
Public Subnet: subnet-0faeeeb0000000000
Private Subnet: subnet-011fe340000000000
```

17.4.6.1.2. Créer votre configuration OIDC

Dans ce tutoriel, nous utiliserons le mode automatique lors de la création de la configuration OIDC. En outre, nous stockerons l'IDC ID en tant que variable d'environnement pour une utilisation ultérieure. La commande utilise le ROSA CLI pour créer la configuration OIDC unique de votre cluster.

- Créez la configuration OIDC en exécutant la commande suivante:

```
$ export OIDC_ID=$(rosa create oidc-config --mode auto --managed --yes -o json | jq -r '.id')
```

17.4.6.2. Création de variables d'environnement supplémentaires

- Exécutez la commande suivante pour configurer des variables d'environnement. Ces variables facilitent l'exécution de la commande pour créer un cluster ROSA:

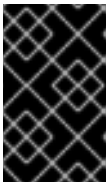
```
$ export CLUSTER_NAME=<cluster_name>
$ export REGION=<VPC_region>
```

ASTUCE

Exécutez rosa whoami pour trouver la région VPC.

17.4.6.3. Créer un cluster

1. Facultatif: exécutez la commande suivante pour créer les rôles et les politiques à l'échelle du compte, y compris les politiques d'opérateur et les rôles et politiques AWS IAM:



IMPORTANT

Complétez cette étape uniquement si c'est la première fois que vous déployez ROSA dans ce compte et que vous n'avez pas encore créé vos rôles et stratégies de compte.

```
$ rosa create account-roles --mode auto --yes
```

2. Exécutez la commande suivante pour créer le cluster:

```
$ rosa create cluster --cluster-name $CLUSTER_NAME \
--subnet-ids ${PUBLIC_SUBNET_ID},${PRIVATE_SUBNET_ID} \
--hosted-cp \
--region $REGION \
--oidc-config-id $OIDC_ID \
--sts --mode auto --yes
```

Le cluster est prêt après environ 10 minutes. Le cluster aura un plan de contrôle sur trois zones de disponibilité AWS dans votre région sélectionnée et créera deux nœuds de travail dans votre compte AWS.

17.4.6.4. Contrôle de l'état de l'installation

1. Exécutez l'une des commandes suivantes pour vérifier l'état du cluster:
   - Afin d'obtenir une vue détaillée de l'état du cluster, exécutez:

     ```
     $ rosa describe cluster --cluster $CLUSTER_NAME
     ```

   - Dans le cas d'une vue abrégée de l'état du cluster, exécutez:

     ```
     $ rosa list clusters
     ```

   - Afin de regarder le journal au fur et à mesure qu'il progresse, exécutez:

     ```
     $ rosa logs install --cluster $CLUSTER_NAME --watch
     ```

2. Lorsque l'état change pour "prêt" votre cluster est installé. Cela pourrait prendre quelques minutes de plus pour que les nœuds ouvriers viennent en ligne.

17.5. DIDACTICIEL: CRÉATION D'UN UTILISATEUR ADMIN

La création d'un utilisateur d'administration (admin) vous permet d'accéder rapidement à votre cluster. Suivez ces étapes pour créer un utilisateur admin.

NOTE

L'utilisateur admin fonctionne bien dans ce paramètre de tutoriel. Dans le cas d'un déploiement réel, utilisez un fournisseur d'identité formel pour accéder au cluster et accorder à l'utilisateur des privilèges d'administration.

1. Exécutez la commande suivante pour créer l'utilisateur admin:

```
rosa create admin --cluster=<cluster-name>
```

Exemple de sortie

```
W: It is recommended to add an identity provider to login to this cluster. See 'rosa create idp --help' for more information.
I: Admin account has been added to cluster 'my-rosa-cluster'. It may take up to a minute for the account to become active.
I: To login, run the following command:
oc login https://api.my-rosa-cluster.abcd.p1.openshiftapps.com:6443 \
--username cluster-admin \
--password FWGYL-2mkJI-00000-00000
```

2. Copiez la commande de connexion qui vous est retournée à l'étape précédente et collez-le dans votre terminal. Cela vous permettra de vous connecter au cluster en utilisant le CLI afin que vous puissiez commencer à utiliser le cluster.

```
$ oc login https://api.my-rosa-cluster.abcd.p1.openshiftapps.com:6443 \
>    --username cluster-admin \
>    --password FWGYL-2mkJI-00000-00000
```

Exemple de sortie

```
Login successful.

You have access to 79 projects, the list has been suppressed. You can list all projects with ' projects'

Using project "default".
```

3. Afin de vérifier que vous êtes connecté en tant qu'utilisateur administrateur, exécutez l'une des commandes suivantes:

   - L'option 1:

     ```
     $ oc whoami
     ```

     Exemple de sortie

     ```
     cluster-admin
     ```

   - L'option 2:

     ```
     oc get all -n openshift-apiserver
     ```

     Il n'y a qu'un utilisateur admin qui peut exécuter cette commande sans erreurs.

4. Désormais, vous pouvez utiliser le cluster en tant qu'utilisateur admin, ce qui suffira pour ce tutoriel. En cas de déploiement réel, il est fortement recommandé de mettre en place un fournisseur d'identité, ce qui est expliqué dans le prochain tutoriel.

17.6. TUTORIEL : CONFIGURATION D'UN FOURNISSEUR D'IDENTITÉ

Afin de vous connecter à votre cluster, configurez un fournisseur d'identité (IDP). Ce tutoriel utilise GitHub comme exemple de PDI. Consultez la liste complète des PDI prises en charge par ROSA.

- Afin d'afficher toutes les options IDP, exécutez la commande suivante:

```
rosa create idp --help
```

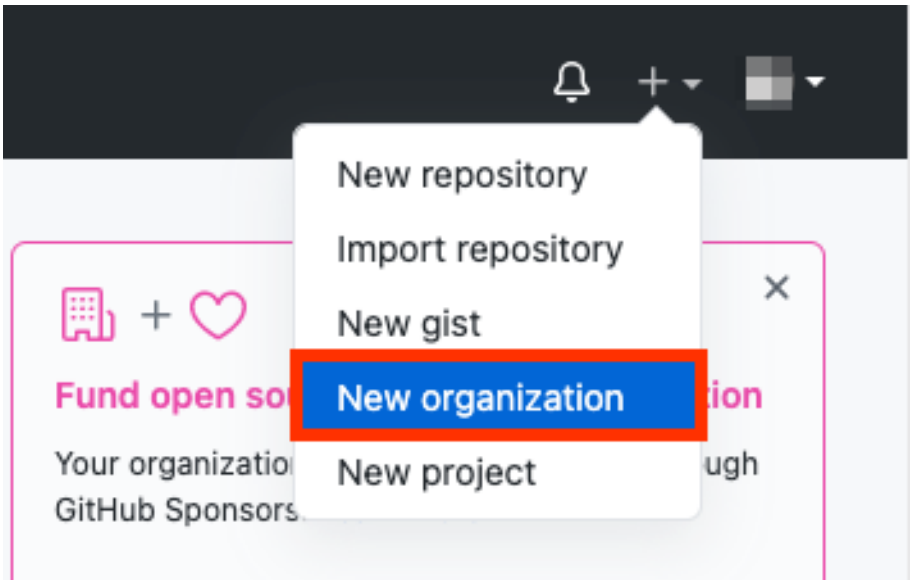
17.6.1. Configuration d'un PDI avec GitHub

1. Connectez-vous à votre compte GitHub.
2. Créez une nouvelle organisation GitHub où vous êtes administrateur.

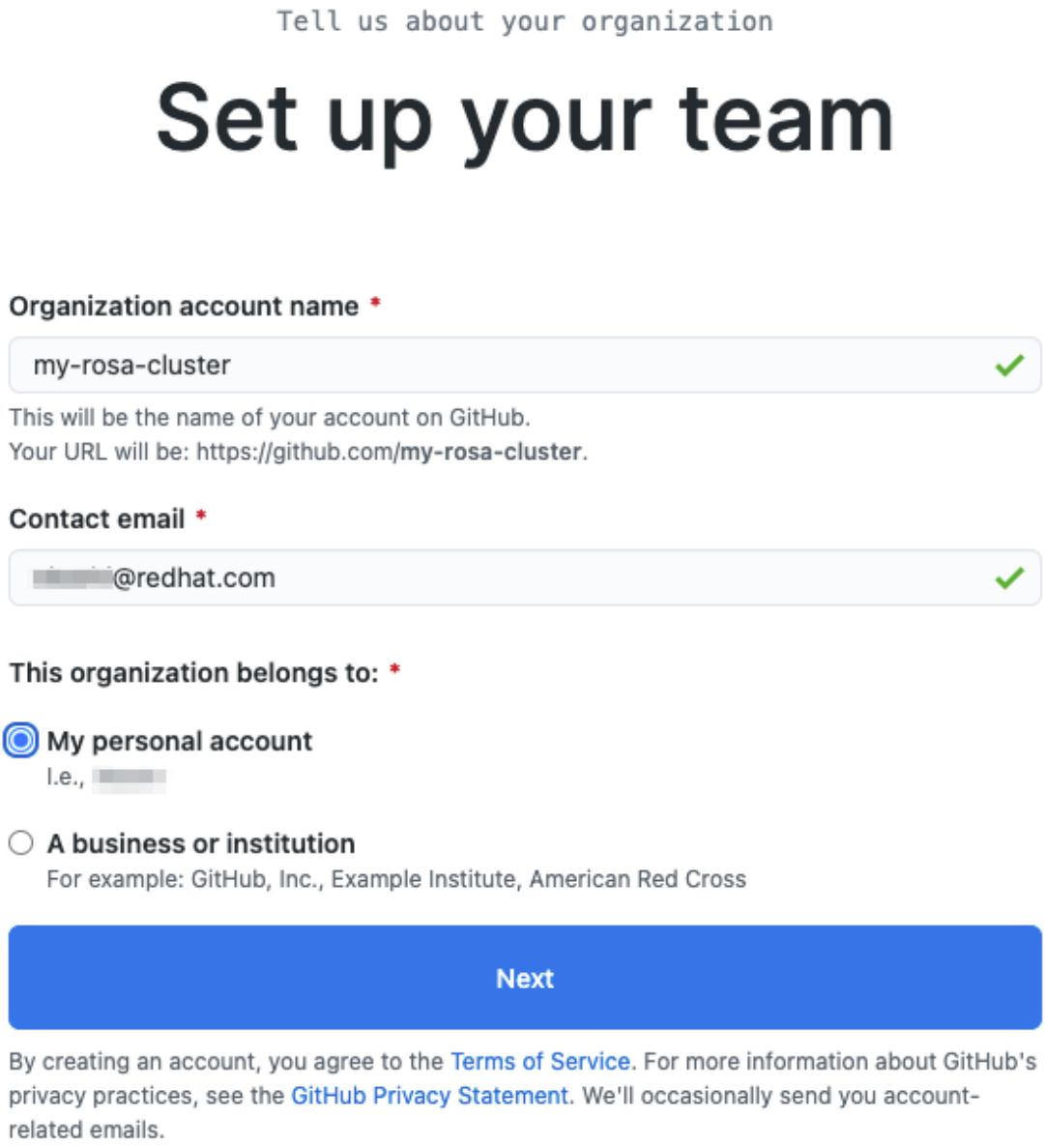
   ASTUCE

   Lorsque vous êtes déjà administrateur d'une organisation existante et que vous souhaitez utiliser cette organisation, passez à l'étape 9.

   Cliquez sur l'icône +, puis cliquez sur Nouvelle organisation.



3. Choisissez le plan le plus applicable pour votre situation ou cliquez sur Rejoindre gratuitement.
4. Entrez le nom d'un compte d'organisation, un courriel et s'il s'agit d'un compte personnel ou professionnel. Ensuite, cliquez sur Suivant.



5. Facultatif : Ajoutez les identifiants GitHub des autres utilisateurs pour accorder un accès supplémentaire à votre cluster ROSA. Il est également possible de les ajouter plus tard.
6. Cliquez sur Configuration complète.
7. Facultatif: Entrez les informations demandées sur la page suivante.
8. Cliquez sur Soumettre.
9. Accédez au terminal et entrez la commande suivante pour configurer le GitHub IDP:

```
rosa create idp --cluster=<cluster name> --interactive
```

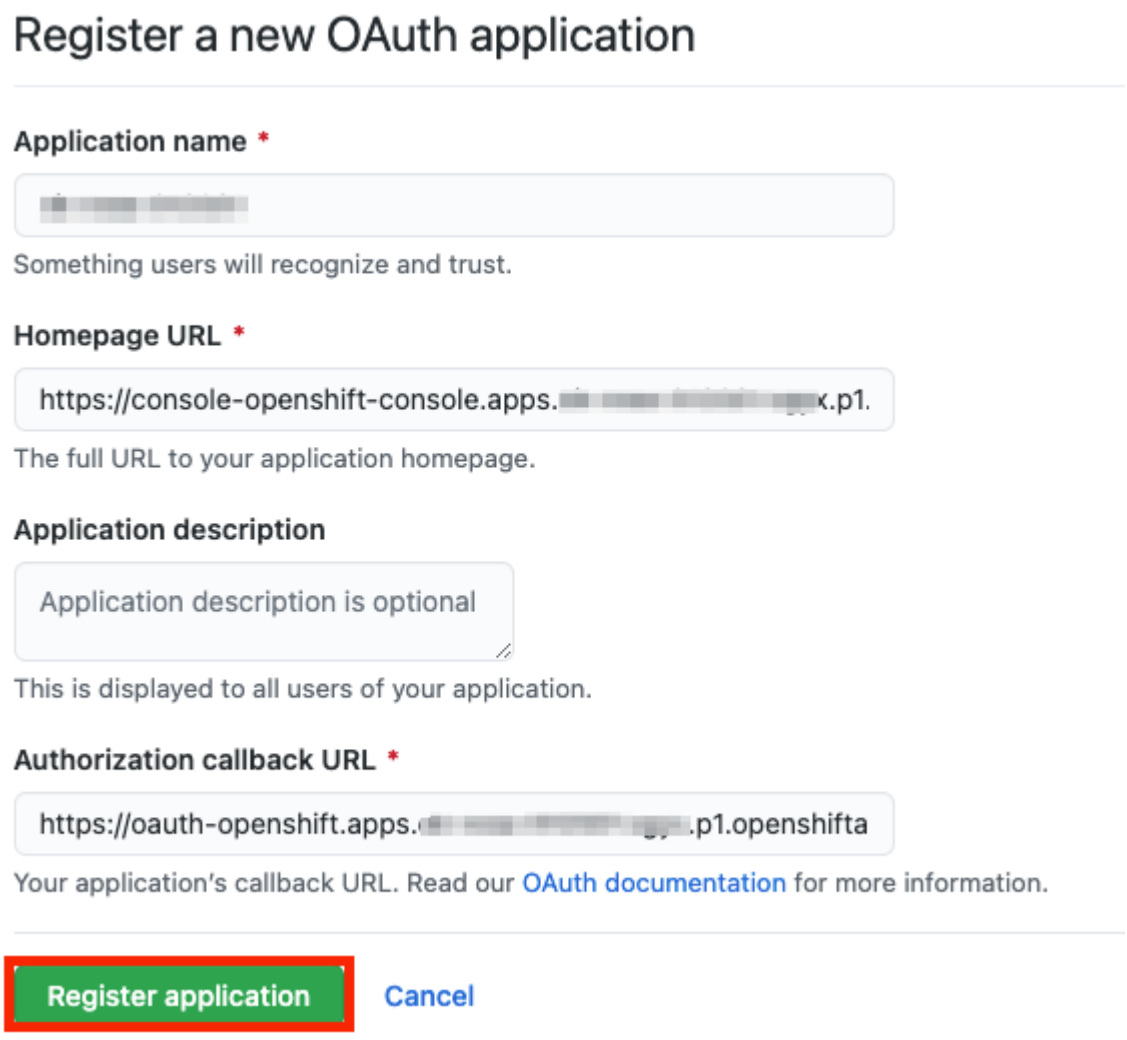
10. Entrez les valeurs suivantes:

```
Type of identity provider: github
Identity Provider Name: <IDP-name>
Restrict to members of: organizations
GitHub organizations: <organization-account-name>
```

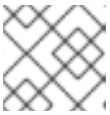
11. Le CLI vous fournira un lien. Copiez et collez le lien dans un navigateur et appuyez sur Entrée. Cela remplira les informations requises pour enregistrer cette demande pour OAuth. Il n'est pas nécessaire de modifier les informations.

```
I: Interactive mode enabled.
Any optional fields can be left empty and a default will be selected.
? Type of identity provider: github
[? Identity provider name: rosa-github
? Restrict to members of: organizations
[? GitHub organizations: my-rosa-cluster
? To use GitHub as an identity provider, you must first register the application:
  - Open the following URL:
    https://github.com/organizations/my-rosa-cluster/settings/applications/new?oauth_application%5Bcallback_url%5D=h
uth2callback%2Frosa-github&oauth_application%5Bname%5D=                &oauth_application%5Burl%5D=https%3A%2F%2Fconso
  - Click on 'Register application'
? Client ID: [? for help]
```

12. Cliquez sur Inscrire l'application.



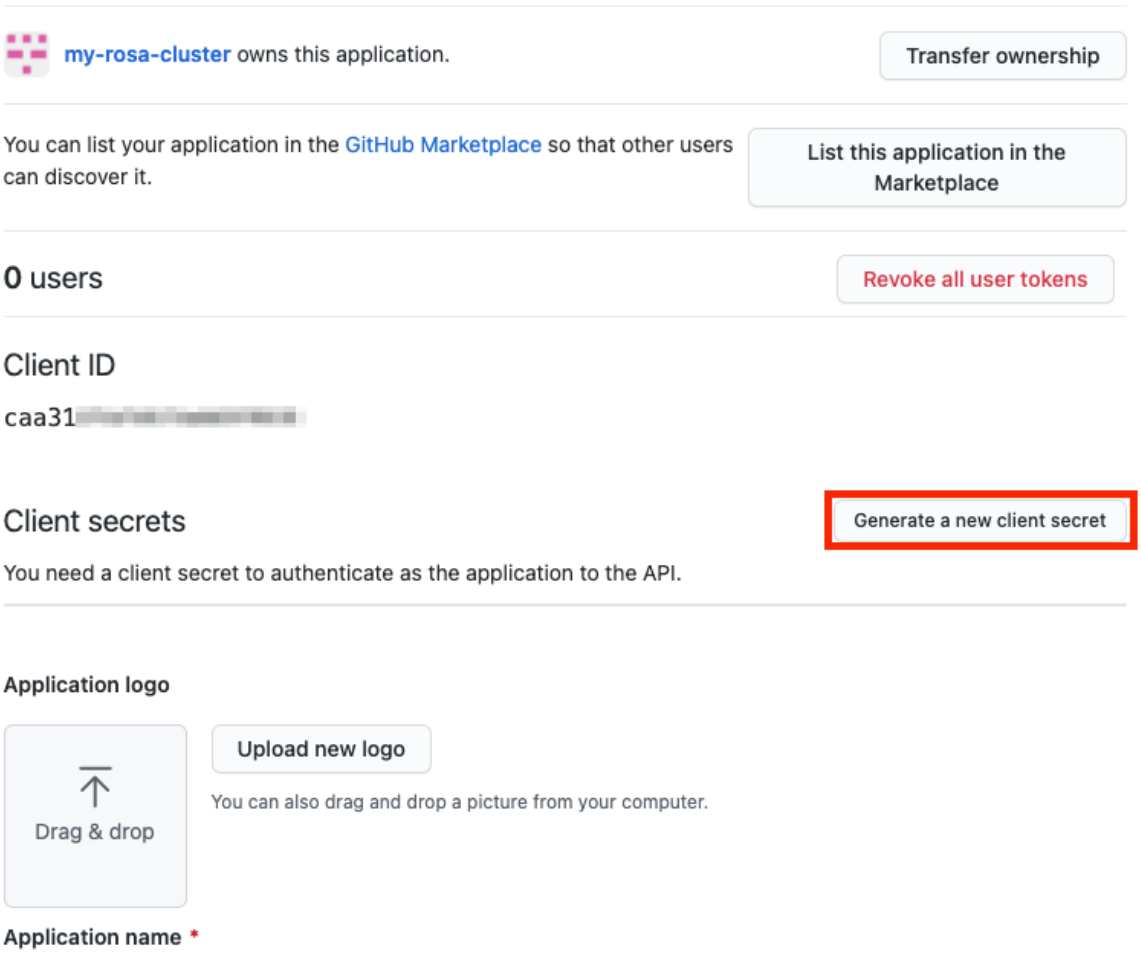
13. La page suivante affiche un identifiant client. Copiez l'ID et collez-le dans le terminal où il demande l'ID Client.



NOTE

Il ne faut pas fermer l'onglet.

14. Le CLI demandera un secret client. Entrez dans votre navigateur et cliquez sur Générer un nouveau secret client.



15. « un secret est généré pour vous. » Copiez votre secret parce qu'il ne sera plus jamais visible.

16. Collez votre secret dans le terminal et appuyez sur Entrée.

17. Laissez GitHub Enterprise Hostname vide.

18. Choisissez la réclamation.

19. Attendez environ 1 minute pour que le PDI soit créé et la configuration pour atterrir sur votre cluster.

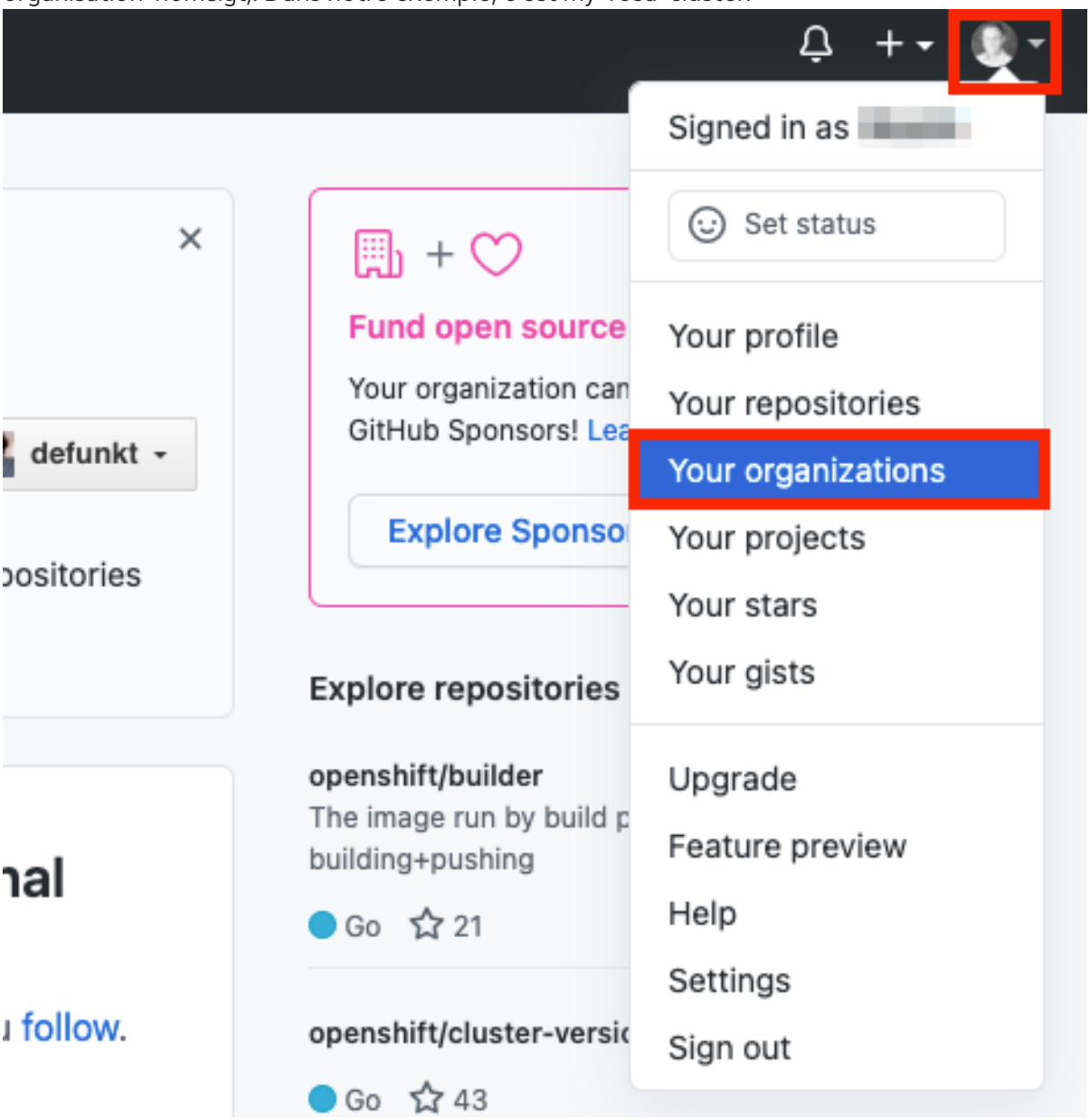
```
I: Interactive mode enabled.
Any optional fields can be left empty and a default will be selected.
? Type of identity provider: github
? Identity provider name: rosa-github
? Restrict to members of: organizations
? GitHub organizations: my-rosa-cluster
? To use GitHub as an identity provider, you must first register the application:
  - Open the following URL:
    https://github.com/organizations/my-rosa-cluster/settings/applications/new?oauth_application%5Bcallback_url%5D=
https%3A%2F%2Foauth-openshift.apps.                    .p1.openshiftapps.com%2Foauth2callback%2Frosa-github&oauth_ap
plication%5Bname%5D=                    L&oauth_application%5Burl%5D=https%3A%2F%2Fconsole-openshift-console.apps.
                    .p1.openshiftapps.com
  - Click on 'Register application'
? Client ID: caa31
? Client Secret: [? for help] ****************************************
? GitHub Enterprise Hostname (optional):
? Mapping method: claim
I: Configuring IDP for cluster '                    '
I: Identity Provider 'rosa-github' has been created.
   It will take up to 1 minute for this configuration to be enabled.
   To add cluster administrators, see 'rosa create user --help'.
   To login into the console, open https://console-openshift-console.apps.                    .p1.openshiftapps.com
and click on rosa-github.
```

20. Copiez le lien retourné et collez-le dans votre navigateur. Le nouveau PDI devrait être disponible sous votre nom choisi. Cliquez sur votre PDI et utilisez vos identifiants GitHub pour accéder au cluster.

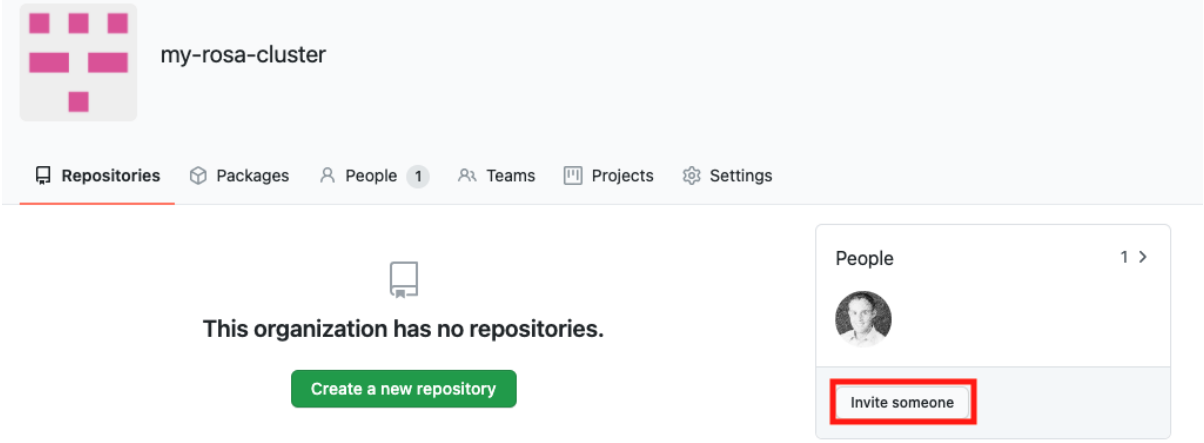
17.6.2. Accorder à d'autres utilisateurs l'accès au cluster

Afin d'accorder l'accès à un autre utilisateur de cluster, vous devrez ajouter leur identifiant d'utilisateur GitHub à l'organisation GitHub utilisée pour ce cluster.

1. Dans GitHub, allez à la page Vos organisations.
2. Cliquez sur l'icône de votre profil, puis sur Vos organisations. Cliquez ensuite sur <votre-organisation-nom>. Dans notre exemple, c'est my-rosa-cluster.



3. Cliquez sur Inviter quelqu'un.



4. Entrez l'ID GitHub du nouvel utilisateur, sélectionnez l'utilisateur correct et cliquez sur Inviter.

5. Dès que le nouvel utilisateur aura accepté l'invitation, il pourra se connecter au cluster ROSA à l'aide du lien Hybrid Cloud Console et de ses identifiants GitHub.

17.7. TUTORIEL: OCTROI DES PRIVILÈGES D'ADMINISTRATEUR

Les privilèges d'administration (admin) ne sont pas automatiquement accordés aux utilisateurs que vous ajoutez à votre cluster. Lorsque vous souhaitez accorder des privilèges de niveau administrateur à certains utilisateurs, vous devrez les accorder manuellement à chaque utilisateur. Il est possible d'accorder des privilèges d'administration à partir de l'interface de ligne de commande ROSA (CLI) ou de l'interface utilisateur Web Red Hat OpenShift Cluster Manager (UI).

Le Red Hat offre deux types de privilèges d'administration:

- cluster-admin: les privilèges cluster-admin donnent à l'utilisateur admin des privilèges complets dans le cluster.

- admin dédié : les privilèges d'administration dédié permettent à l'utilisateur admin de remplir la plupart des tâches administratives avec certaines limitations pour éviter les dommages au cluster. Il est préférable d'utiliser l'administration dédiée lorsque des privilèges élevés sont nécessaires.

Afin d'obtenir de plus amples renseignements sur les privilèges d'administration, consultez la documentation sur l'administration d'un cluster.

17.7.1. En utilisant le ROSA CLI

1. En supposant que vous êtes l'utilisateur qui a créé le cluster, exécutez l'une des commandes suivantes pour accorder des privilèges d'administrateur:

   - A) Pour l'administration en grappes:

     ```
     $ rosa grant user cluster-admin --user <idp_user_name> --cluster=<cluster-name>
     ```

   - En ce qui concerne l'administration dédiée:

     ```
     $ rosa grant user dedicated-admin --user <idp_user_name> --cluster=<cluster-name>
     ```

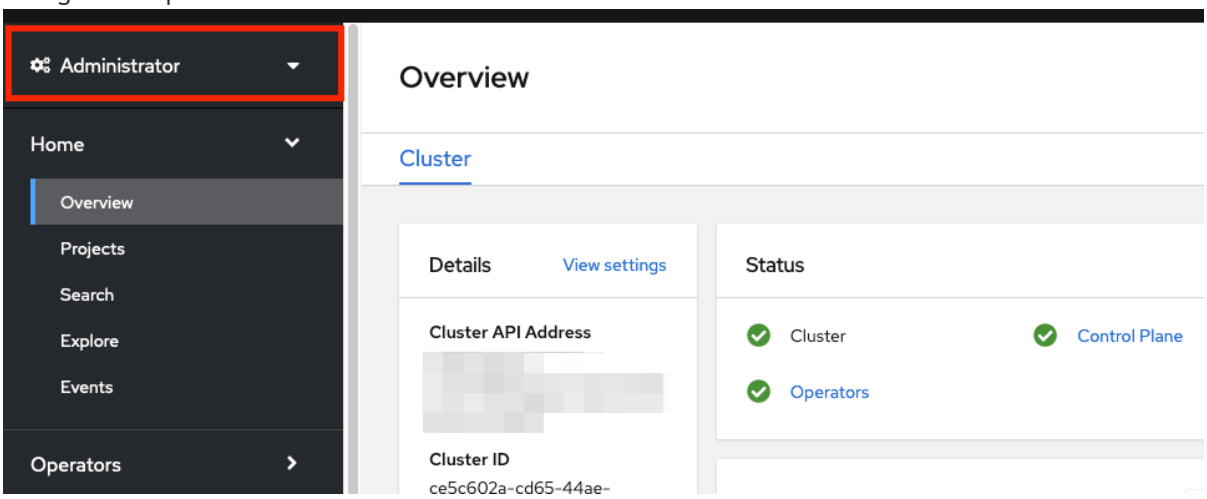
2. Assurez-vous que les privilèges d'administrateur ont été ajoutés en exécutant la commande suivante:

```
$ rosa list users --cluster=<cluster-name>
```

Exemple de sortie

```
$ rosa list users --cluster=my-rosa-cluster
ID                GROUPS
<idp_user_name>   cluster-admins
```

3. Lorsque vous êtes actuellement connecté à la console Red Hat Hybrid Cloud Console, déconnectez-vous de la console et connectez-vous au cluster pour voir une nouvelle perspective avec le "Panneau d'administrateur". Il se peut que vous ayez besoin d'une fenêtre incognito ou privée.

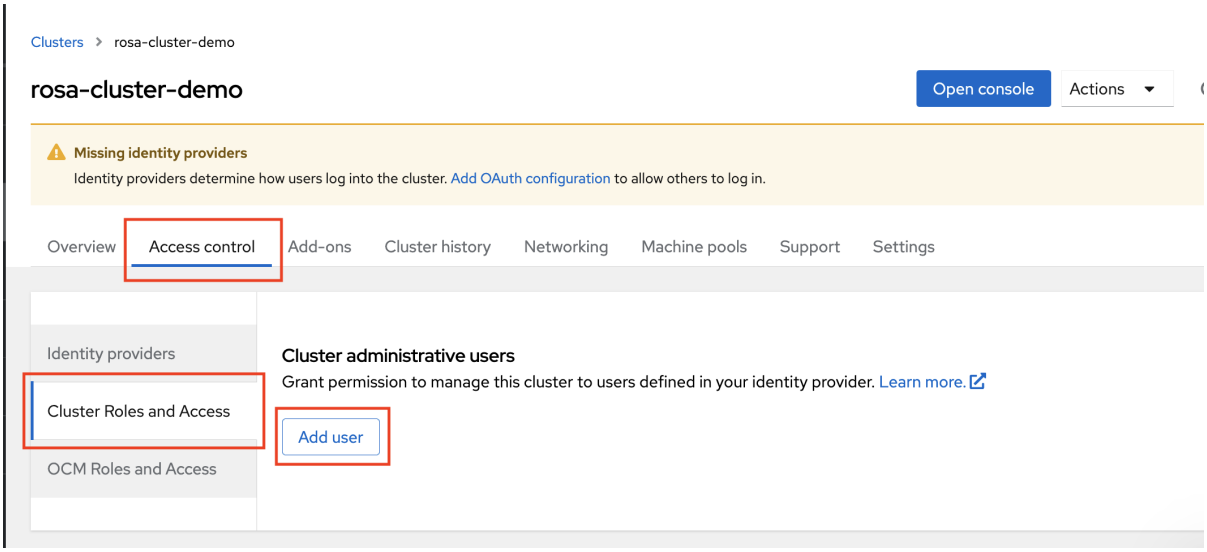


4. Il est également possible de tester que les privilèges d'administrateur ont été ajoutés à votre compte en exécutant la commande suivante. Il n'y a qu'un cluster-admin qui peut exécuter cette commande sans erreurs.

```
$ oc get all -n openshift-apiserver
```

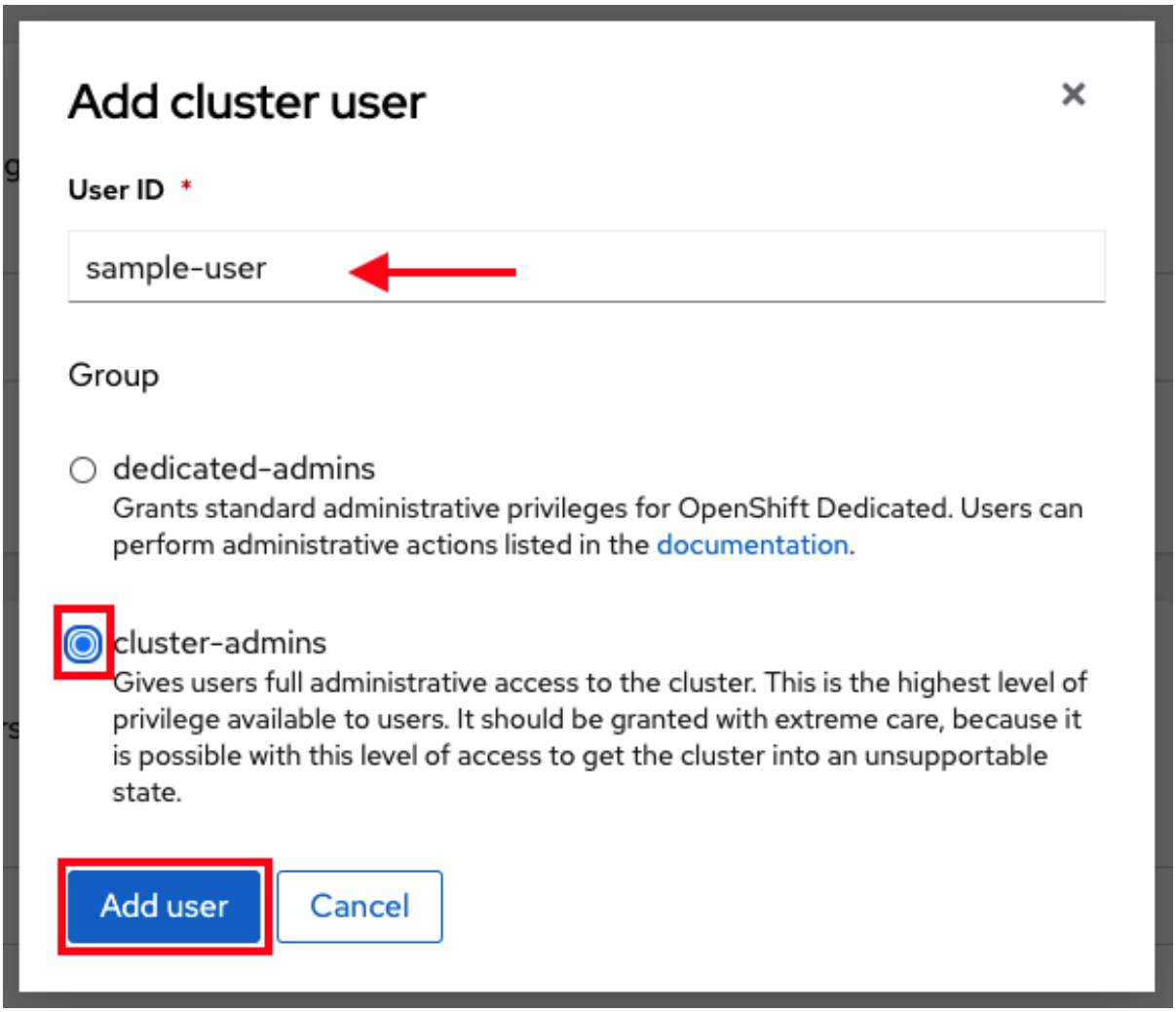
17.7.2. En utilisant le Red Hat OpenShift Cluster Manager UI

1. Connectez-vous à OpenShift Cluster Manager.
2. Choisissez votre cluster.
3. Cliquez sur l'onglet Contrôle d'accès.
4. Cliquez sur les rôles de cluster et l'onglet Accès dans la barre latérale.
5. Cliquez sur Ajouter un utilisateur.



6. Dans l'écran contextuel, entrez l'identifiant de l'utilisateur.

7. Choisissez si vous souhaitez accorder des privilèges de cluster utilisateur-admins ou d'administrations dédiées.



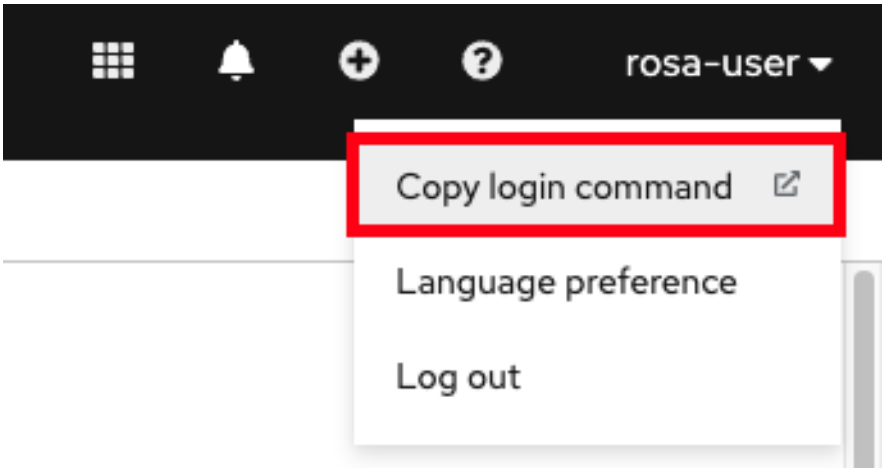
17.8. DIDACTICIEL: ACCÈS À VOTRE CLUSTER

Connectez-vous à votre cluster à l'aide de l'interface de ligne de commande (CLI) ou de l'interface utilisateur Red Hat Hybrid Cloud Console (UI).

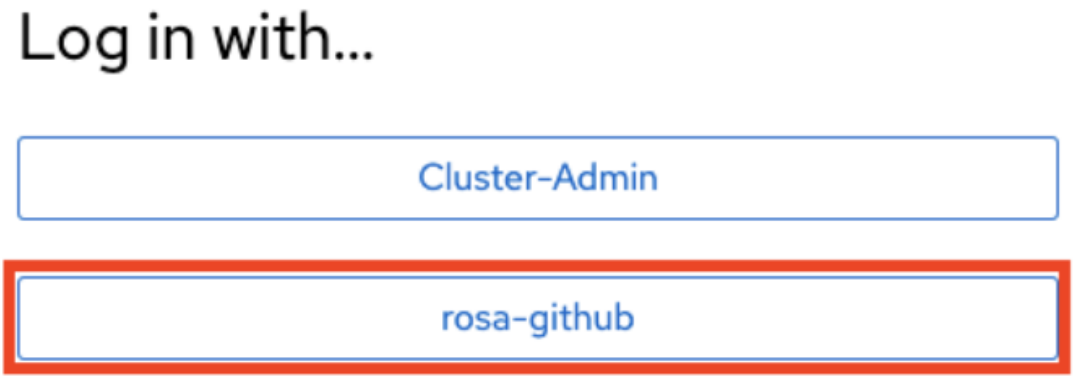
17.8.1. Accéder à votre cluster en utilisant le CLI

Afin d'accéder au cluster à l'aide du CLI, vous devez installer le CLI oc. En suivant les tutoriels, vous avez déjà installé le CLI oc.

1. Connectez-vous à OpenShift Cluster Manager.
2. Cliquez sur votre nom d'utilisateur dans le coin supérieur droit.
3. Cliquez sur Copier la commande de connexion.



4. Cela ouvre un nouvel onglet avec un choix de fournisseurs d'identité (IDP). Cliquez sur le PDI que vous souhaitez utiliser. A titre d'exemple, "rosa-github".



5. Le nouvel onglet s'ouvre. Cliquez sur Affichage du jeton.
6. Exécutez la commande suivante dans votre terminal:

```
$ oc login --token=sha256~GBAfS4JQ0t1UTKYHbWAK6OUWGUkdMGz000000000000 --
server=https://api.my-rosa-cluster.abcd.p1.openshiftapps.com:6443
```

Exemple de sortie

```
Logged into "https://api.my-rosa-cluster.abcd.p1.openshiftapps.com:6443" as "rosa-user"
using the token provided.

You have access to 79 projects, the list has been suppressed. You can list all projects with '
```

```
projects'

Using project "default".
```

7. Confirmez que vous êtes connecté en exécutant la commande suivante:

```
$ oc whoami
```

Exemple de sortie

```
rosa-user
```

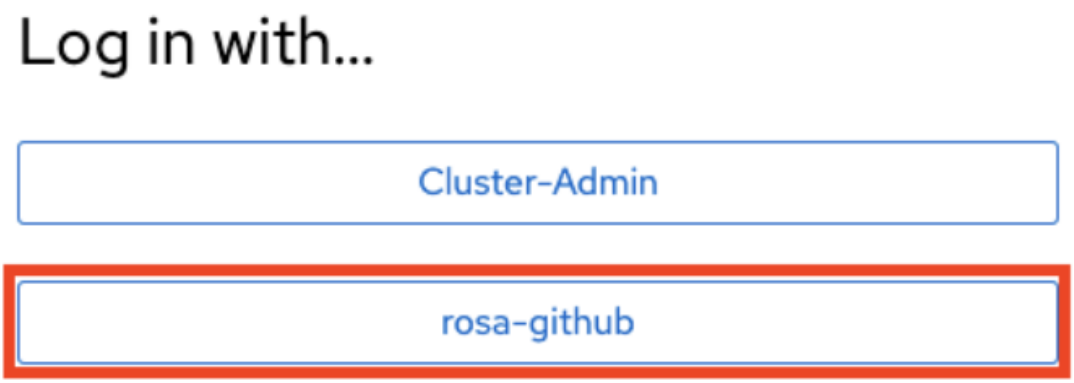
8. Désormais, vous pouvez accéder à votre cluster.

17.8.2. Accéder au cluster via la console hybride Cloud

1. Connectez-vous à OpenShift Cluster Manager.
   a. Afin de récupérer l'URL Hybrid Cloud Console:

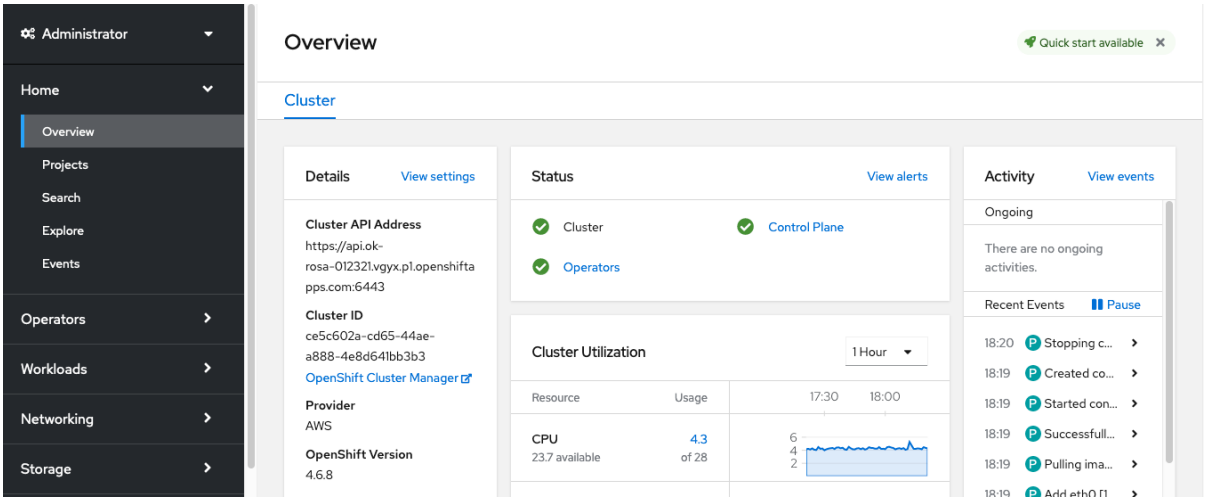
```
rosa describe cluster -c <cluster-name> | grep Console
```

2. Cliquez sur votre PDI. A titre d'exemple, "rosa-github".



3. Entrez vos identifiants d'utilisateur.

4. Il faut vous connecter. Lorsque vous suivez les tutoriels, vous serez un cluster-admin et vous devriez voir la page Web de la console hybride Cloud avec le panneau Administrateur visible.



17.9. DIDACTICIEL: GÉRER LES NŒUDS DE TRAVAILLEURS

Dans Red Hat OpenShift Service sur AWS (ROSA), des aspects changeants de vos nœuds de travail sont effectués grâce à l'utilisation de pools de machines. Le pool de machines permet aux utilisateurs de gérer de nombreuses machines en tant qu'entité unique. Chaque cluster ROSA a un pool de machines par défaut qui est créé lorsque le cluster est créé. Consultez la documentation du pool machine pour plus d'informations.

17.9.1. Création d'un pool de machines

Il est possible de créer un pool de machines avec l'interface de ligne de commande (CLI) ou l'interface utilisateur (UI).

17.9.1.1. Créer un pool de machines avec le CLI

1. Exécutez la commande suivante:

```
rosa create machinepool --cluster=<cluster-name> --name=<machinepool-name> --replicas=<number-nodes>
```

Exemple d'entrée

```
$ rosa create machinepool --cluster=my-rosa-cluster --name=new-mp
--replicas=2
```

Exemple de sortie

```
I: Machine pool 'new-mp' created successfully on cluster 'my-rosa-cluster'
I: To view all machine pools, run 'rosa list machinepools -c my-rosa-cluster'
```

2. Facultatif: Ajoutez des étiquettes de nœuds ou des taints à des nœuds spécifiques dans un nouveau pool de machines en exécutant la commande suivante:

```
rosa create machinepool --cluster=<cluster-name> --name=<machinepool-name> --replicas=<number-nodes> --labels=`<key=pair>`
```

Exemple d'entrée

```
$ rosa create machinepool --cluster=my-rosa-cluster --name=db-nodes-mp --replicas=2 --
labels='app=db','tier=backend'
```

Exemple de sortie

```
I: Machine pool 'db-nodes-mp' created successfully on cluster 'my-rosa-cluster'
```

Cela crée 2 nœuds supplémentaires qui peuvent être gérés en tant qu'unité et leur attribue également les étiquettes affichées.

3. Exécutez la commande suivante pour confirmer la création de pool de machines et les étiquettes assignées:

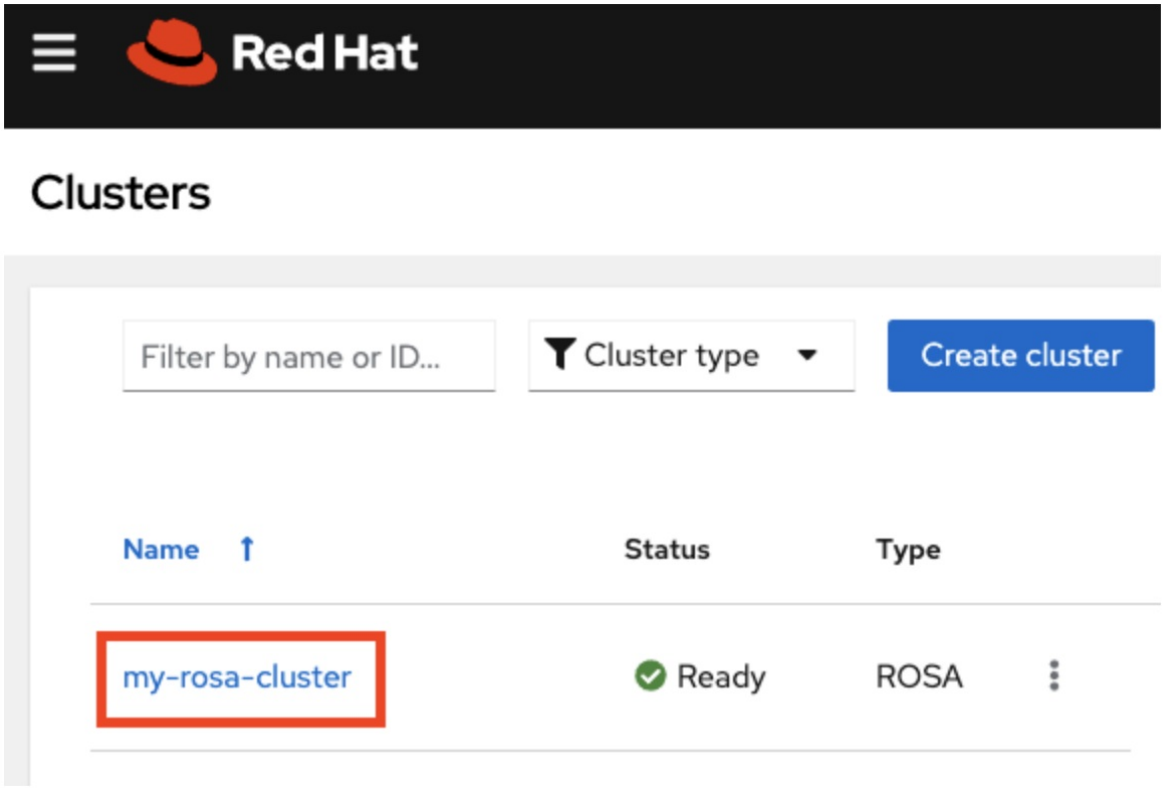
```
rosa list machinepools --cluster=<cluster-name>
```

Exemple de sortie

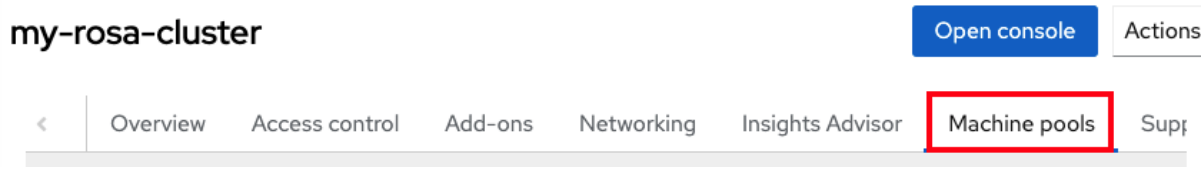
```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS          TAINTS
AVAILABILITY ZONES
Default     No           2         m5.xlarge                      us-east-1a
```

17.9.1.2. Créer un pool de machines avec l'interface utilisateur

1. Connectez-vous à OpenShift Cluster Manager et cliquez sur votre cluster.



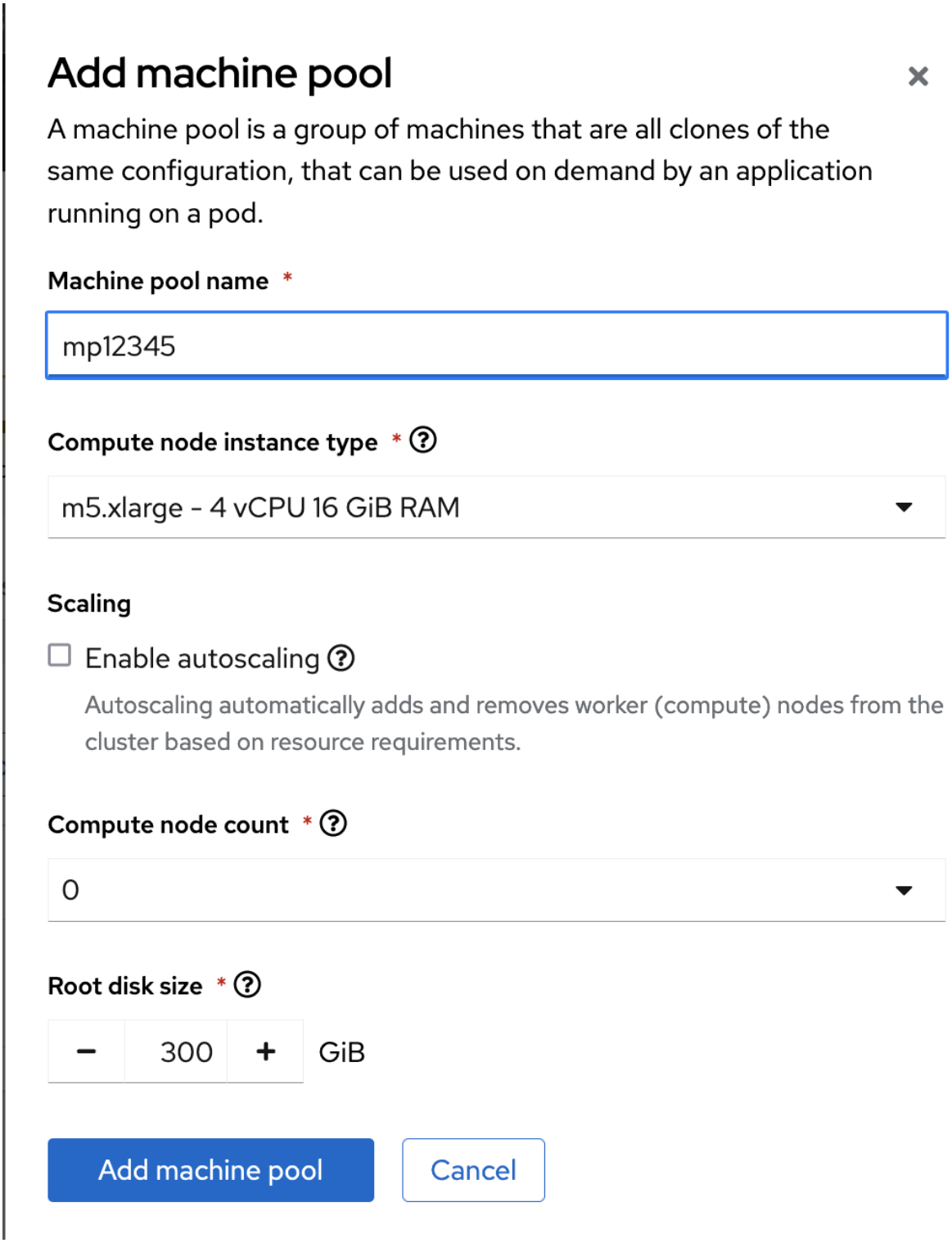
2. Cliquez sur Machine pools.



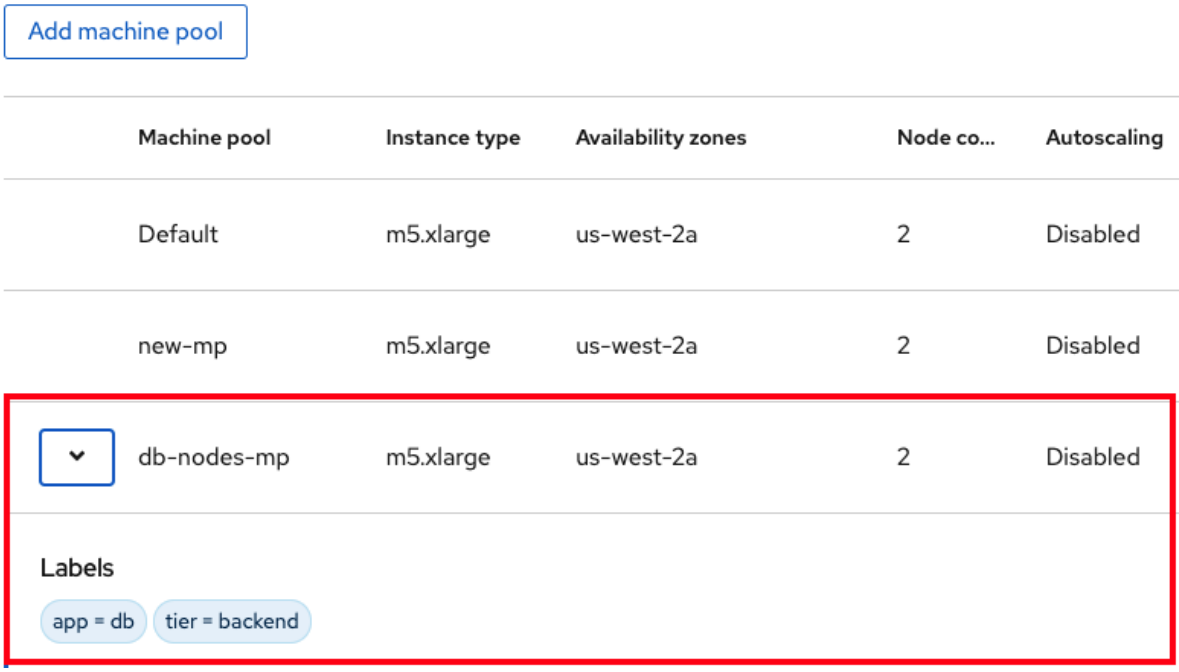
3. Cliquez sur Ajouter un pool de machine.
4. Entrez la configuration souhaitée.

ASTUCE

Il est également possible d'étendre la section Éditer les étiquettes et les taintes des nœuds pour ajouter des étiquettes et des taches de nœuds aux nœuds du pool de machines.



5. Le nouveau pool de machines que vous avez créé.



17.9.2. Les nœuds de travail à l'échelle

Éditez un pool de machines pour mettre à l'échelle le nombre de nœuds ouvriers dans ce pool de machines spécifique. Il est possible d'utiliser le CLI ou l'interface utilisateur pour mettre à l'échelle les nœuds des travailleurs.

17.9.2.1. Les nœuds de travail à l'échelle utilisant le CLI

1. Exécutez la commande suivante pour voir le pool de machines par défaut qui est créé avec chaque cluster:

```
rosa list machinepools --cluster=<cluster-name>
```

Exemple de sortie

```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS          TAINTS
AVAILABILITY ZONES
Default     No           2         m5.xlarge                      us-east-1a
```

2. Afin de mettre à l'échelle le pool de machines par défaut à un nombre différent de nœuds, exécutez la commande suivante:

```
rosa edit machinepool --cluster=<cluster-name> --replicas=<number-nodes> <machinepool-name>
```

Exemple d'entrée

```
rosa edit machinepool --cluster=my-rosa-cluster --replicas 3 Default
```

3. Exécutez la commande suivante pour confirmer que le pool de machines a mis à l'échelle:

```
rosa describe cluster --cluster=<cluster-name> | grep Compute
```

Exemple d'entrée

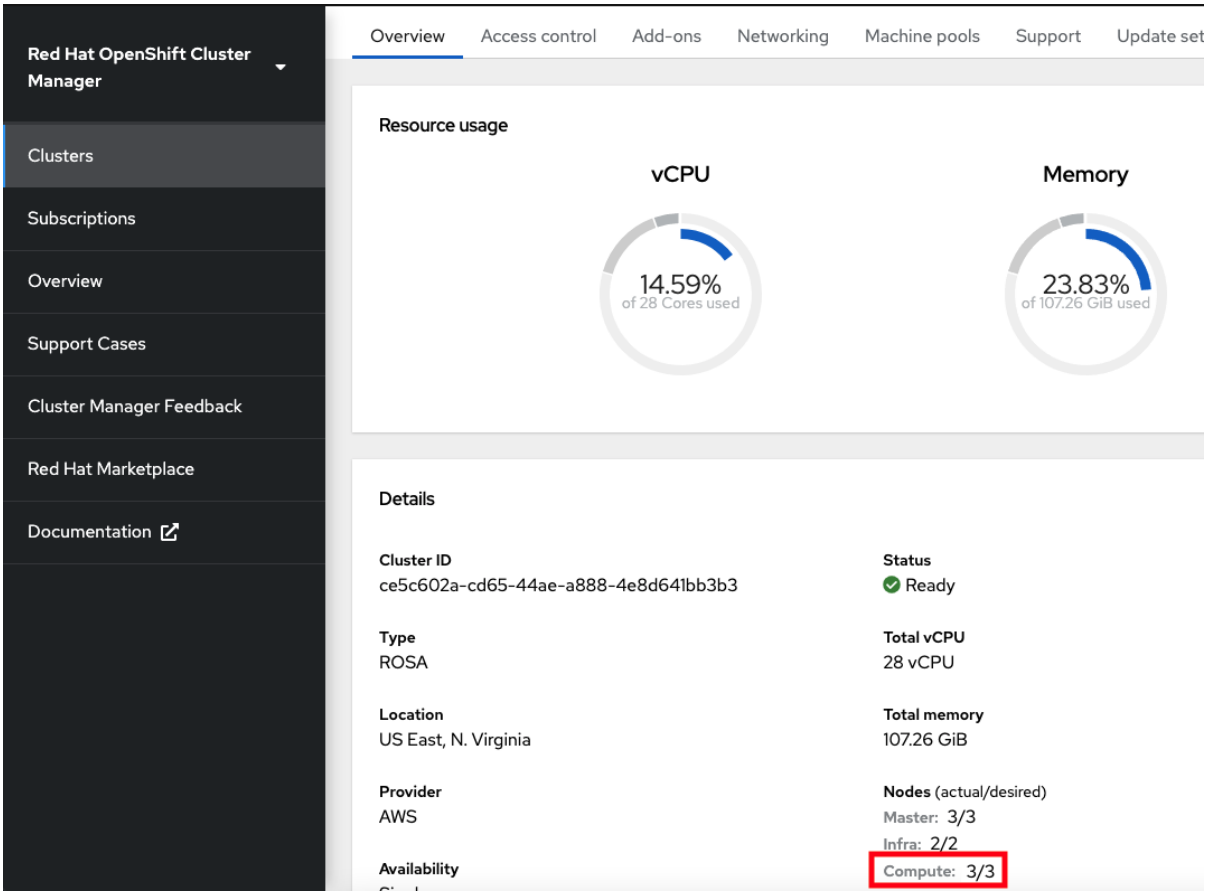
```
$ rosa describe cluster --cluster=my-rosa-cluster | grep Compute
```

Exemple de sortie

```
- Compute:                3 (m5.xlarge)
```

17.9.2.2. Les nœuds de travail à l'échelle utilisant l'interface utilisateur

1. Cliquez sur les trois points à droite du pool de machines que vous souhaitez modifier.
2. Cliquez sur Modifier.
3. Entrez le nombre de nœuds souhaité, puis cliquez sur Enregistrer.
4. Confirmez que le cluster s'est mis à l'échelle en sélectionnant le cluster, en cliquant sur l'onglet Aperçu et en faisant défiler jusqu'à la liste de calcul. La liste des calculs devrait être égale aux nœuds à l'échelle. À titre d'exemple, 3/3.



17.9.2.3. Ajout d'étiquettes de nœuds

1. À l'aide de la commande suivante pour ajouter des étiquettes de nœuds:

```
rosa edit machinepool --cluster=<cluster-name> --replicas=<number-nodes> --
labels='key=value' <machinepool-name>
```

Exemple d'entrée

```
rosa edit machinepool --cluster=my-rosa-cluster --replicas=2 --labels 'foo=bar','baz=one'
new-mp
```

Cela ajoute 2 étiquettes à la nouvelle piscine de machines.

IMPORTANT

Cette commande remplace toutes les configurations de pool de machines par la configuration nouvellement définie. Lorsque vous souhaitez ajouter une autre étiquette et conserver l'ancienne étiquette, vous devez indiquer à la fois la nouvelle et la préexistante de l'étiquette. Dans le cas contraire, la commande remplacera toutes les étiquettes préexistantes par celle que vous souhaitez ajouter. De même, si vous souhaitez supprimer une étiquette, exécutez la commande et indiquez celles que vous voulez, à l'exclusion de celle que vous souhaitez supprimer.

17.9.3. Le mélange des types de nœuds

Il est également possible de mélanger différents types de machines de nœuds ouvriers dans le même cluster en utilisant de nouveaux pools de machines. Il est impossible de modifier le type de nœud d'un pool de machines une fois qu'il est créé, mais vous pouvez créer un nouveau pool de machines avec différents nœuds en ajoutant le drapeau --instance-type.

1. À titre d'exemple, pour changer les nœuds de base de données en un type de nœud différent, exécutez la commande suivante:

```
rosa create machinepool --cluster=<cluster-name> --name=<mp-name> --replicas=<number-
nodes> --labels='<key=pair>' --instance-type=<type>
```

Exemple d'entrée

```
rosa create machinepool --cluster=my-rosa-cluster --name=db-nodes-large-mp --replicas=2 -
-labels='app=db','tier=backend' --instance-type=m5.2xlarge
```

2. Afin de voir tous les types d'instances disponibles, exécutez la commande suivante:

```
rosa list instance-types
```

3. Afin d'apporter des modifications étape par étape, utilisez l'indicateur --interactif:

```
rosa create machinepool -c <cluster-name> --interactive
```

```
[? Machine pool name: large-nodes-pool
[? Enable autoscaling (optional): No
[? Replicas: 3
 ? Instance type:  [Use arrows to move, type to filter, ? for more help]
 > m5.xlarge
   r5.xlarge
   r5.2xlarge
   m5.2xlarge
   c5.2xlarge
   r5.4xlarge
   m5.4xlarge
```

4. Exécutez la commande suivante pour répertorier les pools de machines et voir le nouveau type d'instance plus grand:

```
rosa list machinepools -c <cluster-name>
```

```
ID                  AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS                  TAINTS    AVAILABILITY ZONES
default             No           3         m5.xlarge                                        us-east-1a
db-nodes-large--mp  No           2         m5.2xlarge     app=db, tier=backend              us-east-1a
db-nodes-mp         No_          2         m5.xlarge      app=db, tier=backend              us-east-1a
```

17.10. TUTORIEL: AUTOSCALING

Le cluster autoscaler ajoute ou supprime les nœuds de travail d'un cluster basé sur les ressources de pod.

Le cluster autoscaler augmente la taille du cluster lorsque:

- Les pods n'arrivent pas à planifier les nœuds actuels en raison de ressources insuffisantes.
- Il faut un autre nœud pour répondre aux besoins de déploiement.

Le cluster autoscaler n'augmente pas les ressources de cluster au-delà des limites que vous spécifiez.

Le cluster autoscaler diminue la taille du cluster lorsque:

- Certains nœuds ne sont pas toujours nécessaires pendant une période significative. À titre d'exemple, lorsqu'un nœud a une faible utilisation des ressources et que tous ses gousses importants peuvent s'adapter à d'autres nœuds.

17.10.1. Activer la mise à l'échelle automatique pour un pool de machines existant à l'aide du CLI

NOTE

L'autoscaling de cluster peut être activé lors de la création de clusters et lors de la création d'un nouveau pool de machines en utilisant l'option --enable-autoscaling.

1. La mise à l'échelle automatique est définie en fonction de la disponibilité de la piscine de machine. Afin de savoir quels pools de machines sont disponibles pour l'autoscaling, exécutez la commande suivante:

```
$ rosa list machinepools -c <cluster-name>
```

Exemple de sortie

```
ID        AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS     TAINTS
AVAILABILITY ZONES
Default   No           2         m5.xlarge                 us-east-1a
```

2. Exécutez la commande suivante pour ajouter l'autoscaling à un pool de machines disponible:

```
$ rosa edit machinepool -c <cluster-name> --enable-autoscaling <machinepool-name> --min-replicas=<num> --max-replicas=<num>
```

Exemple d'entrée

```
$ rosa edit machinepool -c my-rosa-cluster --enable-autoscaling Default --min-replicas=2 --
max-replicas=4
```

La commande ci-dessus crée un autoscaler pour les nœuds ouvriers qui se situe entre 2 et 4 nœuds en fonction des ressources.

17.10.2. Activer la mise à l'échelle automatique pour un pool de machines existant à l'aide de l'interface utilisateur

NOTE

L'autoscaling de cluster peut être activé lors de la création de clusters en vérifiant la case à cocher Activer l'autoscaling lors de la création de pools de machines.

1. Allez dans l'onglet pools de machines et cliquez sur les trois points à droite..
2. Cliquez sur Échelle, puis activez l'autoscaling.
3. Exécutez la commande suivante pour confirmer que l'autoscaling a été ajouté:

```
$ rosa list machinepools -c <cluster-name>
```

Exemple de sortie

```
ID        AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS     TAINTS
AVAILABILITY ZONES
Default   Yes          2-4       m5.xlarge                 us-east-1a
```

17.11. TUTORIEL: MISE À NIVEAU DE VOTRE CLUSTER

Le service OpenShift Red Hat sur AWS (ROSA) exécute toutes les mises à niveau de cluster dans le cadre du service géré. Il n'est pas nécessaire d'exécuter des commandes ou d'apporter des modifications au cluster. Les mises à niveau peuvent être programmées à un moment opportun.

Les façons de planifier une mise à niveau de cluster comprennent:

- En utilisant manuellement l'interface de ligne de commande (CLI): Démarrez une mise à niveau immédiate unique ou planifiez une mise à niveau unique pour une date et une heure ultérieures.
- À l'aide de l'interface utilisateur Red Hat OpenShift Cluster Manager (UI): Démarrez une mise à jour immédiate unique ou planifiez une mise à niveau unique pour une date et une heure ultérieures.
- Configuration d'une fenêtre de mise à niveau pour les mises à niveau récurrentes de Y-stream chaque fois qu'une nouvelle version est disponible sans avoir besoin de la programmer manuellement. Les versions mineures doivent être planifiées manuellement.

Pour plus de détails sur les mises à niveau de cluster, exécutez la commande suivante:

```
$ rosa upgrade cluster --help
```

17.11.1. Améliorer manuellement votre cluster à l'aide du CLI

1. Vérifiez s'il y a une mise à niveau disponible en exécutant la commande suivante:

```
$ rosa list upgrade -c <cluster-name>
```

Exemple de sortie

```
$ rosa list upgrade -c <cluster-name>
VERSION  NOTES
4.14.7   recommended
4.14.6
...
```

Dans l'exemple ci-dessus, les versions 4.14.7 et 4.14.6 sont toutes deux disponibles.

2. Planifiez la mise à niveau du cluster dans l'heure en exécutant la commande suivante:

```
$ rosa upgrade cluster -c <cluster-name> --version <desired-version>
```

3. Facultatif : Planifiez le cluster pour qu'il soit mis à niveau à une date et à une heure ultérieures en exécutant la commande suivante:

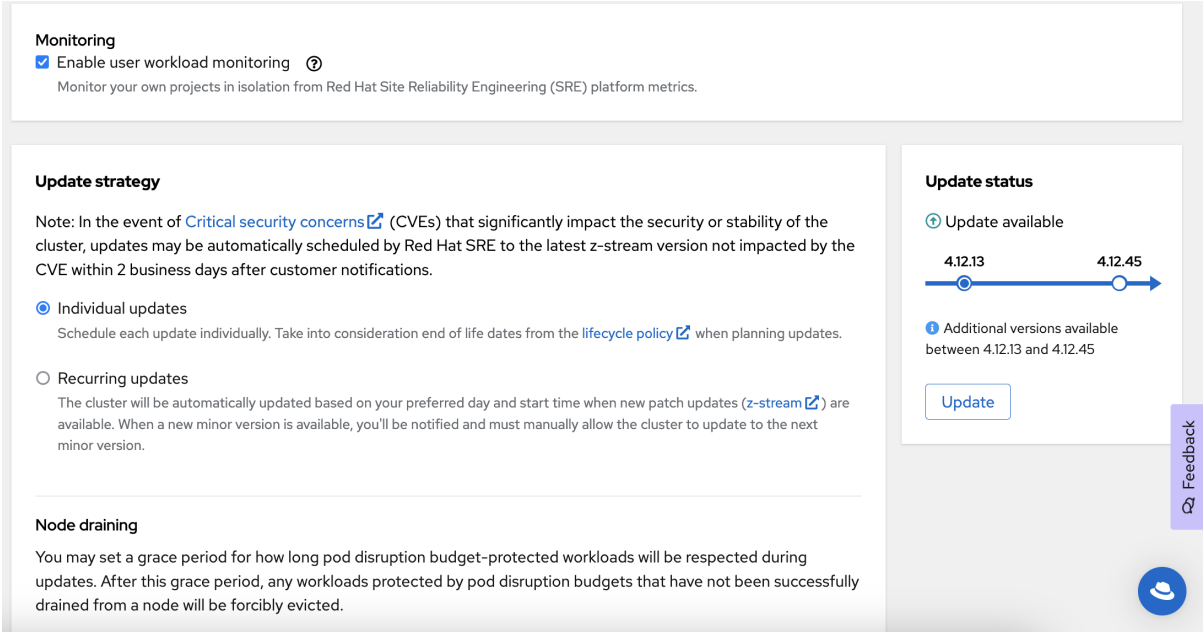
```
$ rosa upgrade cluster -c <cluster-name> --version <desired-version> --schedule-date
<future-date-for-update> --schedule-time <future-time-for-update>
```

17.11.2. Améliorer manuellement votre cluster à l'aide de l'interface utilisateur

1. Connectez-vous à OpenShift Cluster Manager et sélectionnez le cluster que vous souhaitez mettre à niveau.

2. Cliquez sur Paramètres.

3. En cas de mise à jour, cliquez sur Mettre à jour.



4. Choisissez la version vers laquelle vous souhaitez mettre à niveau dans la nouvelle fenêtre.

5. Planifiez un temps pour la mise à niveau ou commencez immédiatement.

17.11.3. Configuration de mises à jour automatiques récurrentes

1. Connectez-vous à OpenShift Cluster Manager et sélectionnez le cluster que vous souhaitez mettre à niveau.
2. Cliquez sur Paramètres.
    1. Dans Stratégie de mise à jour, cliquez sur Récurrence des mises à jour.
3. Définissez le jour et l'heure de la mise à niveau.
4. Dans le cadre de l'évacuation du nœud, sélectionnez un délai de grâce pour permettre aux nœuds de s'écouler avant l'expulsion de la gousse.
5. Cliquez sur **Save**.

17.12. TUTORIEL: SUPPRIMER VOTRE CLUSTER

Il est possible de supprimer votre Red Hat OpenShift Service sur AWS (ROSA) à l'aide de l'interface de ligne de commande (CLI) ou de l'interface utilisateur (UI).

17.12.1. La suppression d'un cluster ROSA à l'aide du CLI

1. Facultatif: Indiquez vos clusters pour vous assurer que vous supprimez le correct en exécutant la commande suivante:

```
$ rosa list clusters
```

2. Effacer un cluster en exécutant la commande suivante:

```
$ rosa delete cluster --cluster <cluster-name>
```

AVERTISSEMENT

Cette commande n'est pas récupérable.

3. Le CLI vous invite à confirmer que vous souhaitez supprimer le cluster. Appuyez sur y puis entrez. Le cluster et toutes ses infrastructures associées seront supprimés.

NOTE

Les rôles et stratégies AWS STS et IAM resteront et devront être supprimés manuellement une fois que la suppression du cluster est terminée en suivant les étapes ci-dessous.

4. Le CLI produit les commandes pour supprimer le fournisseur OpenID Connect (OIDC) et les

ressources de rôles IAM de l'opérateur qui ont été créées. Attendez que le cluster termine la suppression avant de supprimer ces ressources. Effectuez une vérification rapide de l'état en exécutant la commande suivante:

```
$ rosa list clusters
```

5. Lorsque le cluster est supprimé, supprimez le fournisseur OIDC en exécutant la commande suivante:

```
$ rosa delete oidc-provider -c <clusterID> --mode auto --yes
```

6. Effacer les rôles de l'opérateur IAM en exécutant la commande suivante:

```
$ rosa delete operator-roles -c <clusterID> --mode auto --yes
```

NOTE

Cette commande nécessite l'ID de cluster et non le nom du cluster.

7. Supprimez uniquement les rôles de compte restants s'ils ne sont plus nécessaires par d'autres clusters du même compte. Lorsque vous souhaitez créer d'autres clusters ROSA dans ce compte, n'exécutez pas cette étape.
Afin de supprimer les rôles de compte, vous devez connaître le préfixe utilisé lors de leur création. La valeur par défaut est "ManagedOpenShift" sauf indication contraire.

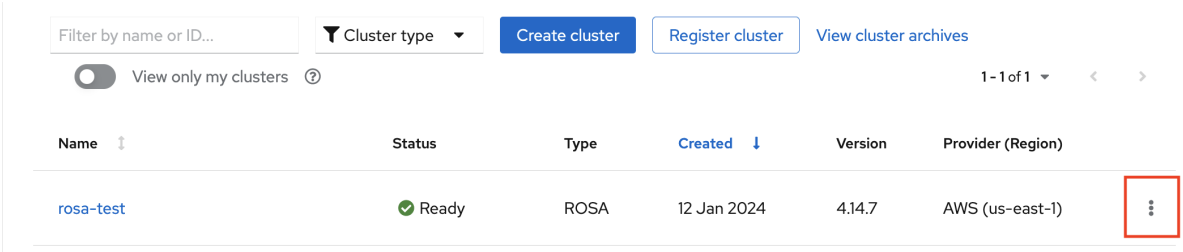
   Effacer les rôles de compte en exécutant la commande suivante:

```
$ rosa delete account-roles --prefix <prefix> --mode auto --yes
```

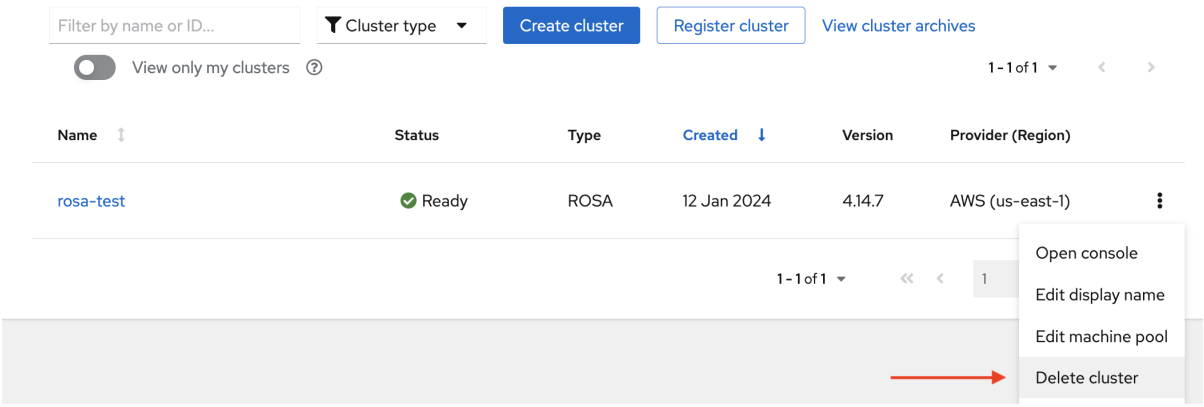
17.12.2. La suppression d'un cluster ROSA à l'aide de l'interface utilisateur

1. Connectez-vous au gestionnaire de cluster OpenShift et localisez le cluster que vous souhaitez supprimer.

2. Cliquez sur les trois points à droite du cluster.



3. Dans le menu déroulant, cliquez sur Supprimer le cluster.



4. Entrez le nom du cluster pour confirmer la suppression, puis cliquez sur Supprimer.

17.13. DIDACTICIEL: OBTENEZ DU SOUTIEN

Il est important de trouver la bonne aide lorsque vous en avez besoin. Ce sont quelques-unes des ressources à votre disposition lorsque vous avez besoin d'aide.

17.13.1. Ajout de contacts de support

Il est possible d'ajouter des adresses e-mail supplémentaires pour les communications concernant votre cluster.

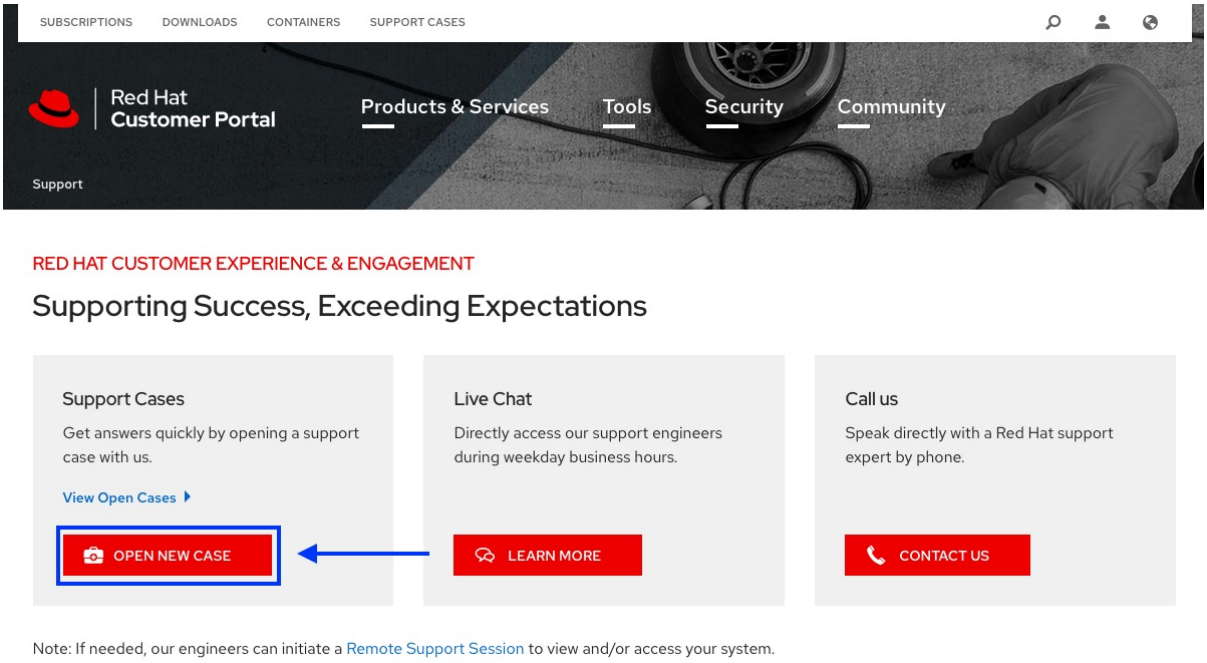
1. Dans l'interface utilisateur Red Hat OpenShift Cluster Manager (UI), cliquez sur Sélectionner le cluster.
2. Cliquez sur l'onglet Support.
3. Cliquez sur Ajouter un contact de notification et entrez les adresses e-mail supplémentaires.

17.13.2. Contacter Red Hat pour obtenir du soutien en utilisant l'interface utilisateur

1. Dans l'interface utilisateur OpenShift Cluster Manager, cliquez sur l'onglet Support.
2. Cliquez sur Ouvrir le cas d'assistance.

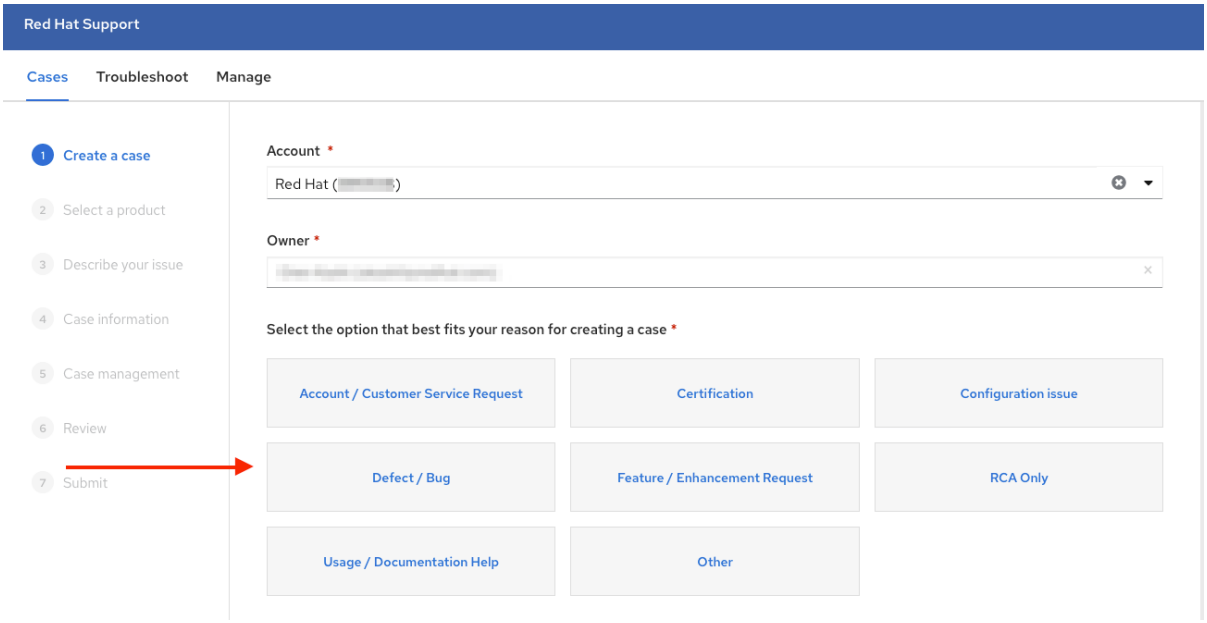
17.13.3. Contacter Red Hat pour le support à l'aide de la page de support

1. Allez à la page de support Red Hat.
2. Cliquez sur Ouvrir un nouveau cas.

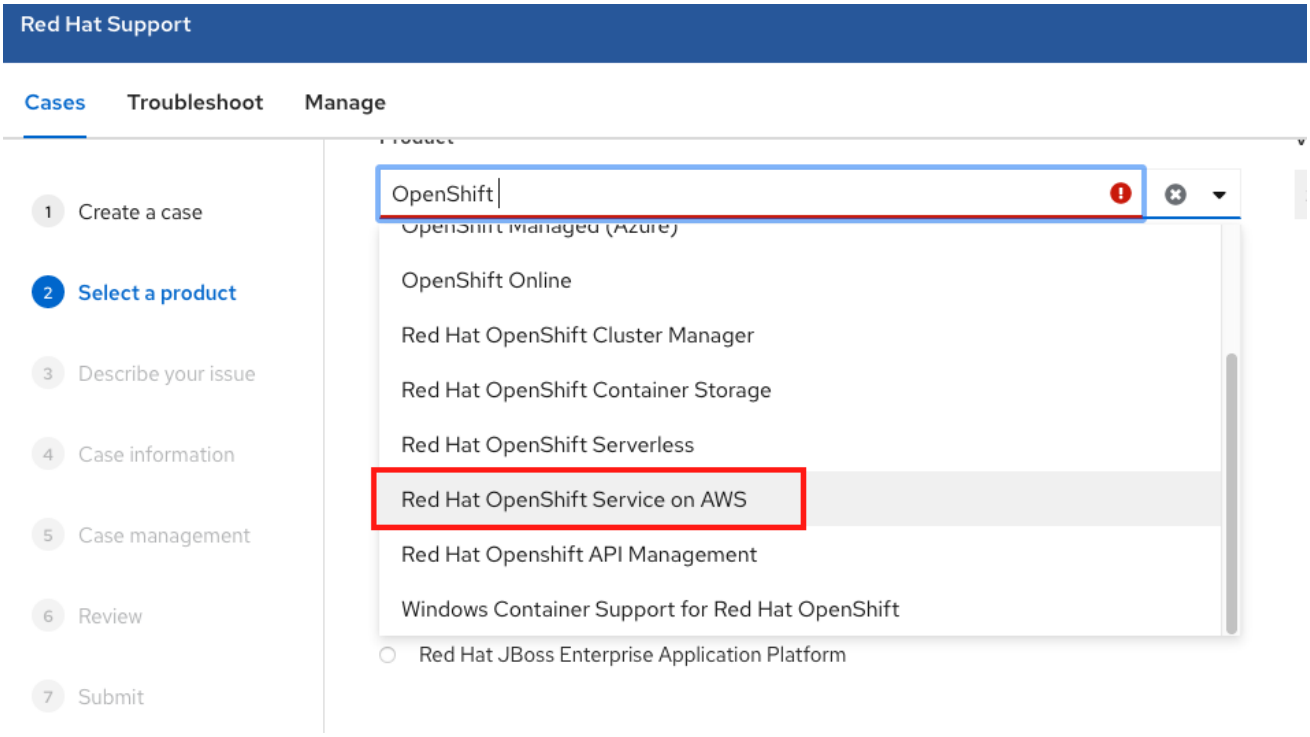


3. Connectez-vous à votre compte Red Hat.

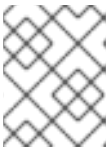
4. Choisissez la raison de contacter le support.



5. Choisissez Red Hat OpenShift Service sur AWS.

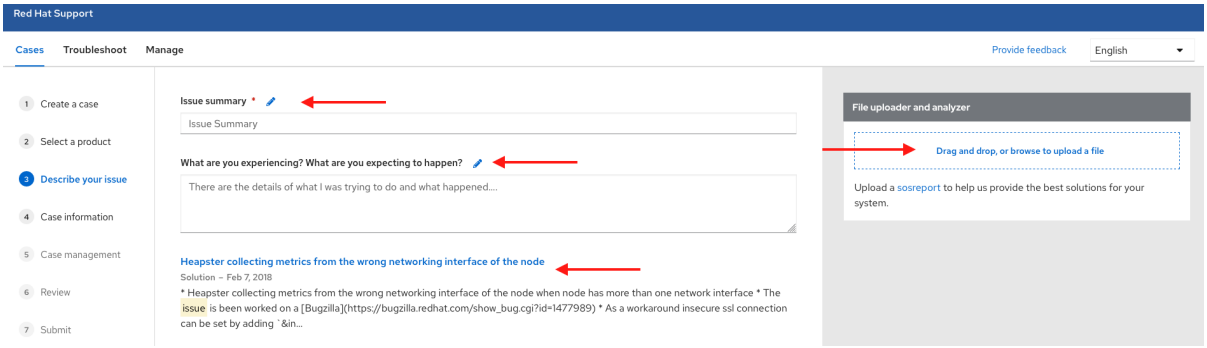


1. Cliquez sur Continuer.
2. Entrez un résumé de la question et les détails de votre demande. Envoyez des fichiers, des journaux et des captures d'écran. Le plus de détails que vous fournissez, le meilleur support Red Hat peut aider votre cas.



NOTE

Les suggestions pertinentes qui pourraient aider à résoudre votre problème apparaîtront au bas de cette page.



3. Cliquez sur Continuer.
4. De répondre aux questions dans les nouveaux champs.
5. Cliquez sur Continuer.
6. Entrez les informations suivantes concernant votre cas:
   a. Le niveau de support: Premium
   b. Gravité: Révisez les définitions de niveau de gravité de soutien Red Hat pour choisir la bonne.

c. Groupe : Si cela est lié à quelques autres cas, vous pouvez sélectionner le groupe correspondant.

d. **Langue**

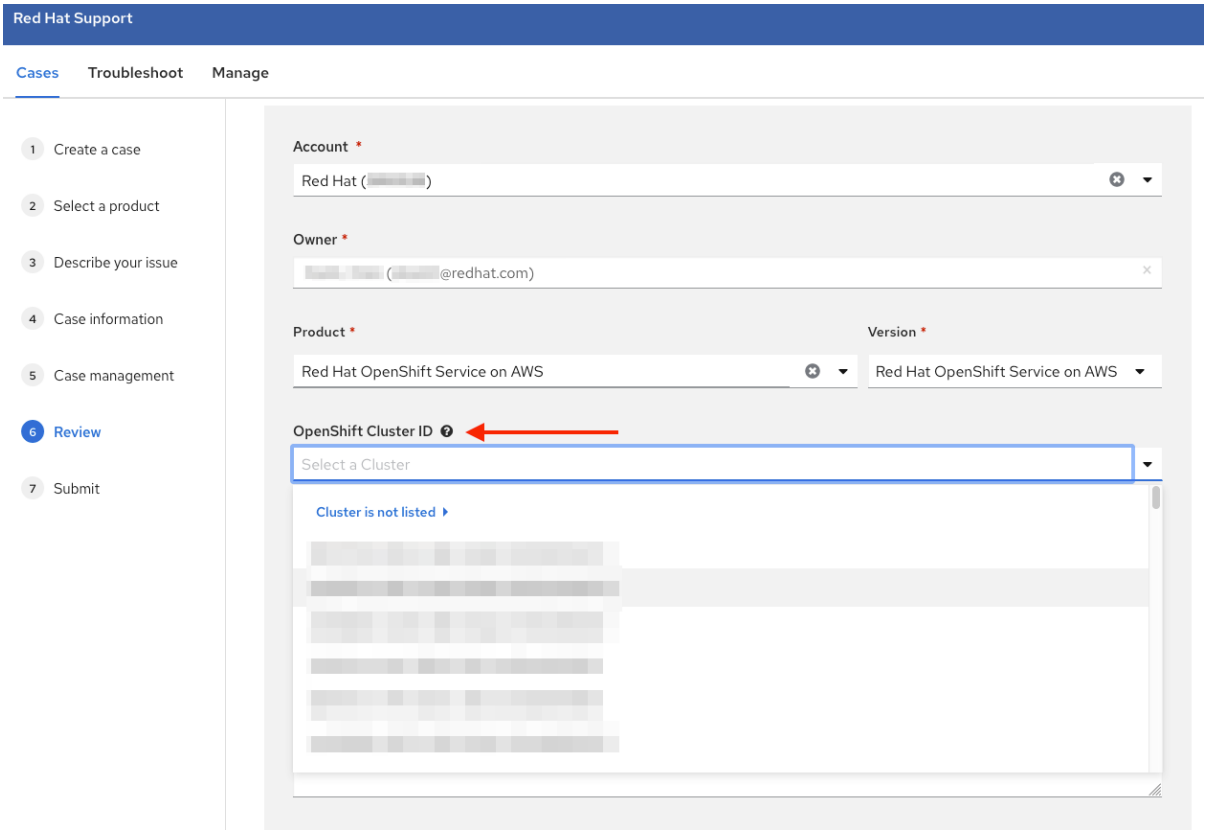
e. Envoyer des notifications: Ajoutez toute adresse e-mail supplémentaire pour rester informé de l'activité.

f. Associés Red Hat: Si vous travaillez avec n'importe qui de Red Hat et que vous voulez les garder dans la boucle, vous pouvez entrer leur adresse e-mail ici.

g. ID de cas alternatif : Si vous voulez y joindre votre propre identifiant, vous pouvez l'entrer ici.

7. Cliquez sur Continuer.

8. Dans l'écran de révision, assurez-vous de sélectionner l'identifiant de cluster correct pour lequel vous contactez le support.



9. Cliquez sur Soumettre.

10. Il vous sera contacté en fonction du temps de réponse prévu pour le niveau de gravité indiqué.

CHAPITRE 18. DÉPLOIEMENT D'UNE APPLICATION

18.1. TUTORIEL: DÉPLOIEMENT D'UNE APPLICATION

18.1.1. Introduction

Après avoir réussi à provisionner votre cluster, vous pouvez déployer une application dessus. Cette application vous permet de vous familiariser avec certaines des fonctionnalités de Red Hat OpenShift Service sur AWS (ROSA) et Kubernetes.

18.1.1.1. Aperçu du laboratoire

Dans ce laboratoire, vous compléterez les tâches suivantes conçues pour vous aider à comprendre les concepts de déploiement et d'exploitation d'applications basées sur des conteneurs:

- Déployez une application basée sur Node.js à l'aide des objets S2I et Kubernetes Deployment.
- Configurez un pipeline de livraison continue (CD) pour pousser automatiquement les changements de code source.
- Explorez l'exploitation forestière.
- Faites l'expérience de l'auto-guérison des applications.
- Explorez la gestion des configurations à travers des configmaps, des secrets et des variables d'environnement.
- Le stockage persistant permet de partager des données entre les redémarrages des pod.
- Explorez le réseautage au sein de Kubernetes et des applications.
- Familiarisez-vous avec les fonctionnalités ROSA et Kubernetes.
- Adaptez automatiquement les pods en fonction des charges de l'autoscaleur Horizontal Pod.
- Faites appel aux contrôleurs AWS pour Kubernetes (ACK) pour déployer et utiliser un seau S3.

Ce laboratoire utilise soit l'interface utilisateur Web ROSA CLI ou ROSA (UI).

18.2. TUTORIEL: DÉPLOIEMENT D'UNE APPLICATION

18.2.1. Conditions préalables

1. A Groupement de la ROSA provisoire
   Ce laboratoire suppose que vous avez accès à un cluster ROSA fourni avec succès. Lorsque vous n'avez pas encore créé de cluster ROSA, consultez le guide de démarrage rapide ROSA pour plus d'informations.

2. L'interface de ligne de commande OpenShift (CLI)
   En savoir plus, voir Commencer avec l'OpenShift CLI.

3. Compte GitHub
   Employez votre compte GitHub existant ou inscrivez-vous à https://github.com/signup.

18.2.1.1. Comprendre l'association de compte AWS

Avant de pouvoir utiliser Red Hat OpenShift Cluster Manager sur la console de cloud hybride Red Hat pour créer des clusters Red Hat OpenShift sur AWS (ROSA) qui utilisent le Service de jetons de sécurité AWS (STS), vous devez associer votre compte AWS à votre organisation Red Hat. Il est possible d'associer votre compte en créant et en liant les rôles IAM suivants.

Le rôle de gestionnaire de cluster OpenShift

Créez un rôle OpenShift Cluster Manager IAM et liez-le à votre organisation Red Hat.
Il est possible d'appliquer des autorisations de base ou administratives au rôle OpenShift Cluster Manager. Les autorisations de base permettent la maintenance des clusters à l'aide d'OpenShift Cluster Manager. Les autorisations administratives permettent le déploiement automatique des rôles d'opérateur spécifiques au cluster et du fournisseur OpenID Connect (OIDC) à l'aide d'OpenShift Cluster Manager.

Le rôle de l'utilisateur

Créez un rôle IAM utilisateur et liez-le à votre compte d'utilisateur Red Hat. Le compte d'utilisateur Red Hat doit exister dans l'organisation Red Hat qui est liée à votre rôle OpenShift Cluster Manager. Le rôle d'utilisateur est utilisé par Red Hat pour vérifier votre identité AWS lorsque vous utilisez la console cloud hybride OpenShift Cluster Manager pour installer un cluster et les ressources STS requises.

Associer votre compte AWS à votre organisation Red Hat
Avant d'utiliser Red Hat OpenShift Cluster Manager sur la console de cloud hybride Red Hat pour créer des clusters Red Hat OpenShift sur AWS (ROSA) qui utilisent le service de jetons de sécurité AWS (STS), créer un rôle OpenShift Cluster Manager IAM et le lier à votre organisation Red Hat. Ensuite, créez un rôle IAM utilisateur et liez-le à votre compte d'utilisateur Red Hat dans la même organisation Red Hat.

Procédure

1. Créez un rôle OpenShift Cluster Manager et liez-le à votre organisation Red Hat:

NOTE

Afin d'activer le déploiement automatique des rôles d'opérateur spécifiques au cluster et du fournisseur OpenID Connect (OIDC) à l'aide de la console de cloud hybride OpenShift Cluster Manager, vous devez appliquer les privilèges administratifs au rôle en choisissant la commande de rôle OCM Admin dans l'étape Comptes et rôles de la création d'un cluster ROSA. En savoir plus sur les privilèges de base et administratifs pour le rôle de gestionnaire de cluster OpenShift, voir Comprendre l'association de compte AWS.

NOTE

Lorsque vous choisissez la commande de rôle OCM de base dans l'étape Comptes et rôles de la création d'un cluster ROSA dans la console de cloud hybride OpenShift Cluster Manager, vous devez déployer un cluster ROSA en mode manuel. Dans une étape ultérieure, vous serez invité à configurer les rôles d'opérateur spécifiques au cluster et le fournisseur OpenID Connect (OIDC).

```
$ rosa create ocm-role
```

Choisissez les valeurs par défaut dans les instructions pour créer rapidement et lier le rôle.

2. Créez un rôle d'utilisateur et liez-le à votre compte d'utilisateur Red Hat:

```
$ rosa create user-role
```

Choisissez les valeurs par défaut dans les instructions pour créer rapidement et lier le rôle.

NOTE

Le compte d'utilisateur Red Hat doit exister dans l'organisation Red Hat qui est liée à votre rôle OpenShift Cluster Manager.

18.3. TUTORIEL: DÉPLOIEMENT D'UNE APPLICATION

18.3.1. Aperçu du laboratoire

18.3.1.1. Les ressources du laboratoire

- Code source pour l'application OSToy
- Image de conteneur frontal OSToy
- Image de conteneur OSToy microservice
- Déploiement Définition des fichiers YAML:

accueil > ostoy-frontend-deployment.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: ostoy-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ostoy-frontend
  labels:
    app: ostoy
spec:
  selector:
    matchLabels:
      app: ostoy-frontend
  strategy:
    type: Recreate
  replicas: 1
  template:
```

```
  metadata:
    labels:
      app: ostoy-frontend
  spec:
    # Uncomment to use with ACK portion of the workshop
    # If you chose a different service account name please replace it.
    # serviceAccount: ostoy-sa
    containers:
    - name: ostoy-frontend
      securityContext:
        allowPrivilegeEscalation: false
        runAsNonRoot: true
        seccompProfile:
          type: RuntimeDefault
        capabilities:
          drop:
          - ALL
      image: quay.io/ostoylab/ostoy-frontend:1.6.0
      imagePullPolicy: IfNotPresent
      ports:
      - name: ostoy-port
        containerPort: 8080
      resources:
        requests:
          memory: "256Mi"
          cpu: "100m"
        limits:
          memory: "512Mi"
          cpu: "200m"
      volumeMounts:
      - name: configvol
        mountPath: /var/config
      - name: secretvol
        mountPath: /var/secret
      - name: datavol
        mountPath: /var/demo_files
      livenessProbe:
        httpGet:
          path: /health
          port: 8080
        initialDelaySeconds: 10
        periodSeconds: 5
      env:
      - name: ENV_TOY_SECRET
        valueFrom:
          secretKeyRef:
            name: ostoy-secret-env
            key: ENV_TOY_SECRET
      - name: MICROSERVICE_NAME
        value: OSTOY_MICROSERVICE_SVC
      - name: NAMESPACE
        valueFrom:
          fieldRef:
            fieldPath: metadata.namespace
    volumes:
      - name: configvol
```

```
          configMap:
            name: ostoy-configmap-files
        - name: secretvol
          secret:
            defaultMode: 420
            secretName: ostoy-secret
        - name: datavol
          persistentVolumeClaim:
            claimName: ostoy-pvc
---
apiVersion: v1
kind: Service
metadata:
  name: ostoy-frontend-svc
  labels:
    app: ostoy-frontend
spec:
  type: ClusterIP
  ports:
    - port: 8080
      targetPort: ostoy-port
      protocol: TCP
      name: ostoy
  selector:
    app: ostoy-frontend
---
apiVersion: route.openshift.io/v1
kind: Route
metadata:
  name: ostoy-route
spec:
  to:
    kind: Service
    name: ostoy-frontend-svc
---
apiVersion: v1
kind: Secret
metadata:
  name: ostoy-secret-env
type: Opaque
data:
  ENV_TOY_SECRET: VGhpcyBpcyBhIHRlc3Q=
---
kind: ConfigMap
apiVersion: v1
metadata:
  name: ostoy-configmap-files
data:
  config.json:  '{ "default": "123" }'
---
apiVersion: v1
kind: Secret
metadata:
  name: ostoy-secret
data:
  secret.txt:
```

```
VVNFUk5BTUU9bXlfdXNlcgpQQVNTV09SRD1AT3RCbCVYQXAhIzYzMlk1RndDQE1UUWsK
U01UUD1sb2NhbGhvc3QKU01UUF9QT1JUPTI1
type: Opaque
```

accueil > ostoy-microservice-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ostoy-microservice
  labels:
    app: ostoy
spec:
  selector:
    matchLabels:
      app: ostoy-microservice
  replicas: 1
  template:
    metadata:
      labels:
        app: ostoy-microservice
    spec:
      containers:
      - name: ostoy-microservice
        securityContext:
          allowPrivilegeEscalation: false
          runAsNonRoot: true
          seccompProfile:
            type: RuntimeDefault
          capabilities:
            drop:
            - ALL
        image: quay.io/ostoylab/ostoy-microservice:1.5.0
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 8080
          protocol: TCP
        resources:
          requests:
            memory: "128Mi"
            cpu: "50m"
          limits:
            memory: "256Mi"
            cpu: "100m"
---
apiVersion: v1
kind: Service
metadata:
  name: ostoy-microservice-svc
  labels:
    app: ostoy-microservice
spec:
  type: ClusterIP
  ports:
    - port: 8080
```

```
    targetPort: 8080
    protocol: TCP
  selector:
    app: ostoy-microservice
```

- Le seau S3 manifeste pour ACK S3

  ajouter au panier S3-bucket.yaml

```
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: ostoy-bucket
  namespace: ostoy
spec:
  name: ostoy-bucket
```

NOTE

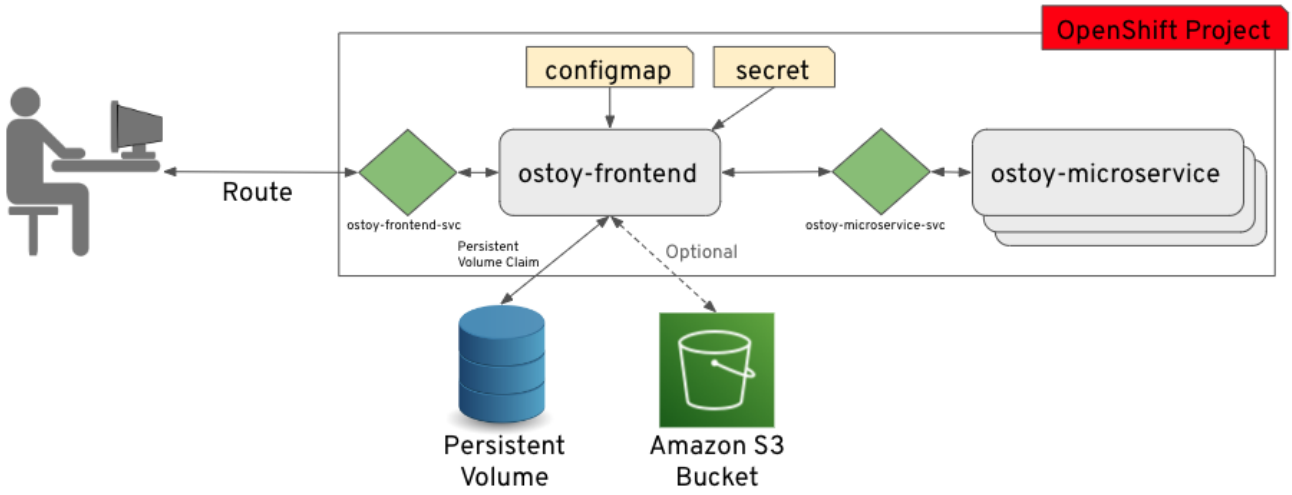
Afin de simplifier le déploiement de l'application OSToy, tous les objets requis dans les manifestes de déploiement ci-dessus sont regroupés. Dans le cas d'un déploiement d'entreprise typique, un fichier manifeste distinct pour chaque objet Kubernetes est recommandé.

18.3.1.2. À propos de l'application OSToy

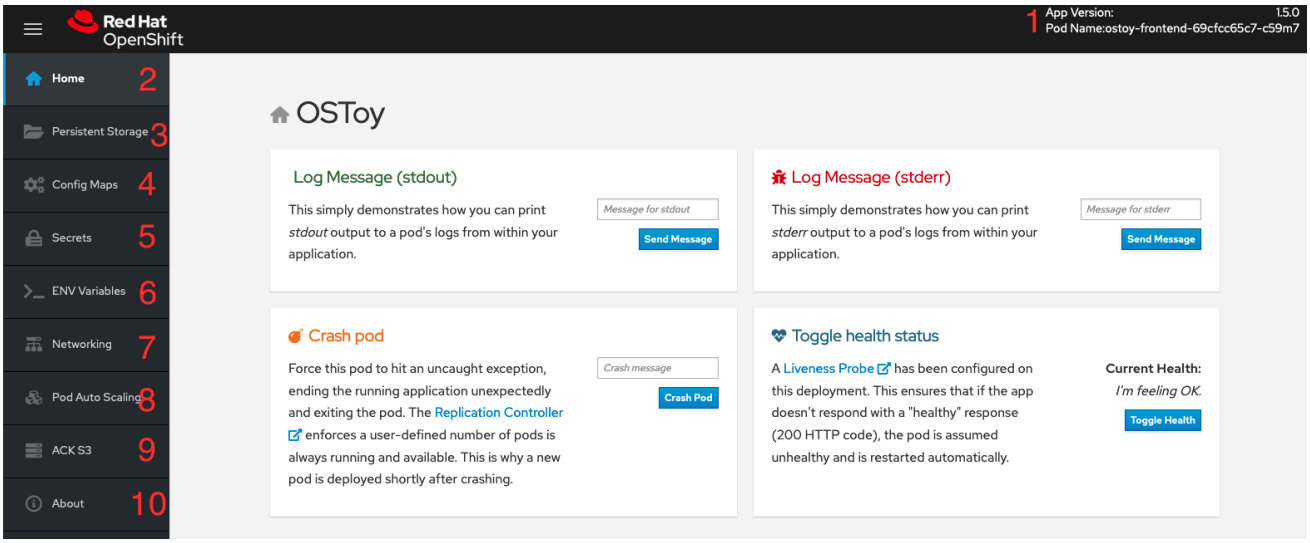
OSToy est une application Node.js simple que vous allez déployer dans un cluster ROSA pour vous aider à explorer les fonctionnalités de Kubernetes. Cette application dispose d'une interface utilisateur où vous pouvez:

- Écrivez des messages dans le journal (stdout / stderr).
- Bloquez intentionnellement l'application pour voir l'auto-guérison.
- Basculer une sonde de vivacité et surveiller le comportement d'OpenShift.
- Lisez les cartes de configuration, les secrets et les variables env.
- En cas de connexion au stockage partagé, lisez et écrivez des fichiers.
- Contrôlez la connectivité réseau, le DNS intra-cluster et l'intra-communication avec le microservice inclus.
- Augmentez la charge pour afficher la mise à l'échelle automatique des gousses pour gérer la charge à l'aide de l'Autoscaler Horizontal Pod.
- Facultatif : Connectez-vous à un seau AWS S3 pour lire et écrire des objets.

18.3.1.3. Diagramme d'application OSToy



18.3.1.4. Comprendre l'interface utilisateur d'OSToy



1. Affiche le nom de pod qui a servi votre navigateur la page.
2. Accueil: La page principale de l'application où vous pouvez effectuer certaines des fonctions listées que nous explorerons.
3. Stockage persistant : vous permet d'écrire des données sur le volume persistant lié à cette application.
4. Configuration Maps: affiche le contenu des configmaps disponibles pour l'application et les paires key:value.
5. Les secrets : Montre le contenu des secrets disponibles pour l'application et les paires key:value.
6. ENV Variables: affiche les variables d'environnement disponibles pour l'application.
7. Le réseautage : Outils pour illustrer le réseautage au sein de l'application.
8. La mise à l'échelle automatique de Pod: Outil pour augmenter la charge des pods et tester l'HPA.
9. ACK S3: Option: Intégrer avec AWS S3 pour lire et écrire des objets dans un seau.

NOTE

Afin de voir la section "ACK S3" d'OSToy, vous devez compléter la section ACK de cet atelier. Lorsque vous décidez de ne pas compléter cette section, l'application OSToy fonctionnera toujours.

10. À propos : Affiche plus d'informations sur l'application.

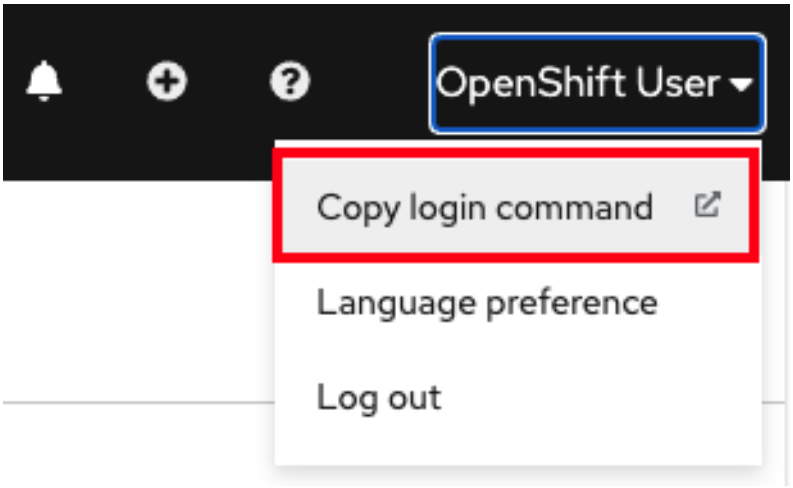
18.4. TUTORIEL: DÉPLOIEMENT D'UNE APPLICATION

18.4.1. Déploiement de l'application OSToy avec Kubernetes

Il est possible de déployer l'application OSToy en créant et en stockant les images pour les conteneurs microservice front-end et back-end dans un référentiel d'images. Ensuite, vous pouvez créer des déploiements Kubernetes pour déployer l'application.

18.4.1.1. La récupération de la commande login

1. Lorsque vous n'êtes pas connecté au CLI, accédez à votre cluster avec la console Web.
2. Cliquez sur la flèche déroulante à côté de votre nom de connexion en haut à droite, puis sélectionnez Copier la commande de connexion.



Le nouvel onglet s'ouvre.

3. Choisissez votre méthode d'authentification.
4. Cliquez sur Affichage du jeton.
5. Copiez la commande sous Connexion avec ce jeton.
6. À partir de votre terminal, collez et exécutez la commande copiée. En cas de succès de la connexion, vous verrez le message de confirmation suivant:

```
$ oc login --token=<your_token> --server=https://api.osd4-
demo.abc1.p1.openshiftapps.com:6443
Logged into "https://api.myrosacluster.abcd.p1.openshiftapps.com:6443" as "rosa-user"
using the token provided.
```

```
You don't have any projects. You can try to create a new project, by running

oc new-project <project name>
```

18.4.1.2. Créer un nouveau projet

18.4.1.2.1. En utilisant le CLI

1. Créez un nouveau projet nommé ostoy dans votre cluster en exécutant la commande suivante:

```
$ oc new-project ostoy
```

Exemple de sortie

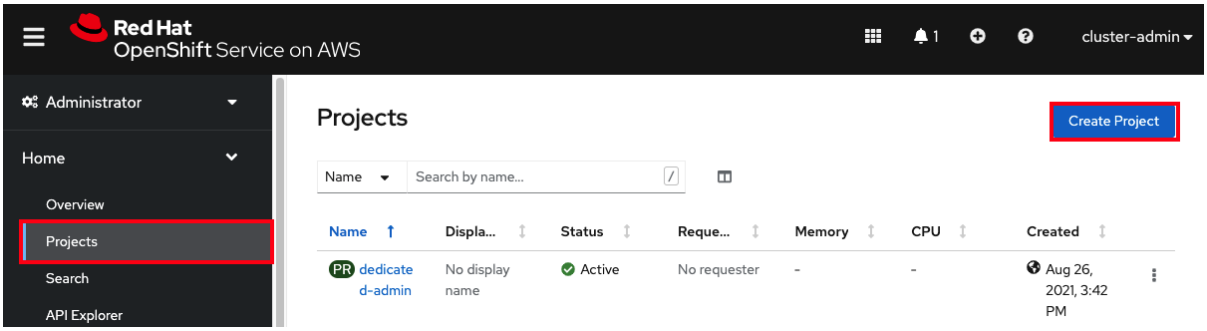
```
Now using project "ostoy" on server
"https://api.myrosacluster.abcd.p1.openshiftapps.com:6443".
```

2. Facultatif: alternativement, créez un nom de projet unique en exécutant la commande suivante:

```
$ oc new-project ostoy-$(uuidgen | cut -d - -f 2 | tr '[:upper:]' '[:lower:]')
```

18.4.1.2.2. À l'aide de la console web

1. À partir de la console web, cliquez sur Home → Projets.
2. Dans la page Projets, cliquez sur Créer un projet.



18.4.1.3. Déploiement du microservice back-end

Le microservice sert les requêtes Web internes et renvoie un objet JSON contenant le nom d'hôte actuel et une chaîne de couleurs générée au hasard.

- Déployez le microservice en exécutant la commande suivante depuis votre terminal:

```
$ oc apply -f https://raw.githubusercontent.com/openshift-cs/rosaworkshop/master/rosa-workshop/ostoy/yaml/ostoy-microservice-deployment.yaml
```

Exemple de sortie

```
$ oc apply -f https://raw.githubusercontent.com/openshift-cs/rosaworkshop/master/rosa-workshop/ostoy/yaml/ostoy-microservice-deployment.yaml
deployment.apps/ostoy-microservice created
service/ostoy-microservice-svc created
```

18.4.1.4. Déploiement du service front-end

Le déploiement front-end utilise le front-end Node.js pour l'application et les objets Kubernetes supplémentaires.

Le fichier ostoy-frontend-deployment.yaml montre que le déploiement front-end définit les caractéristiques suivantes:

- Revendication de volume persistant
- L'objet de déploiement
- Le service
- Itinéraire
- ConfigMaps
- Les secrets
    - Déployez l'application front-end et créez tous les objets en entrant la commande suivante:

```
$ oc apply -f https://raw.githubusercontent.com/openshift-cs/rosaworkshop/master/rosa-workshop/ostoy/yaml/ostoy-frontend-deployment.yaml
```

Exemple de sortie

```
persistentvolumeclaim/ostoy-pvc created
deployment.apps/ostoy-frontend created
service/ostoy-frontend-svc created
route.route.openshift.io/ostoy-route created
configmap/ostoy-configmap-env created
secret/ostoy-secret-env created
configmap/ostoy-configmap-files created
secret/ostoy-secret created
```

Il faut voir tous les objets créés avec succès.

18.4.1.5. J'obtiens la route

Il faut obtenir l'itinéraire pour accéder à l'application.

- Accédez à l'itinéraire de votre application en exécutant la commande suivante:

```
$ oc get route
```

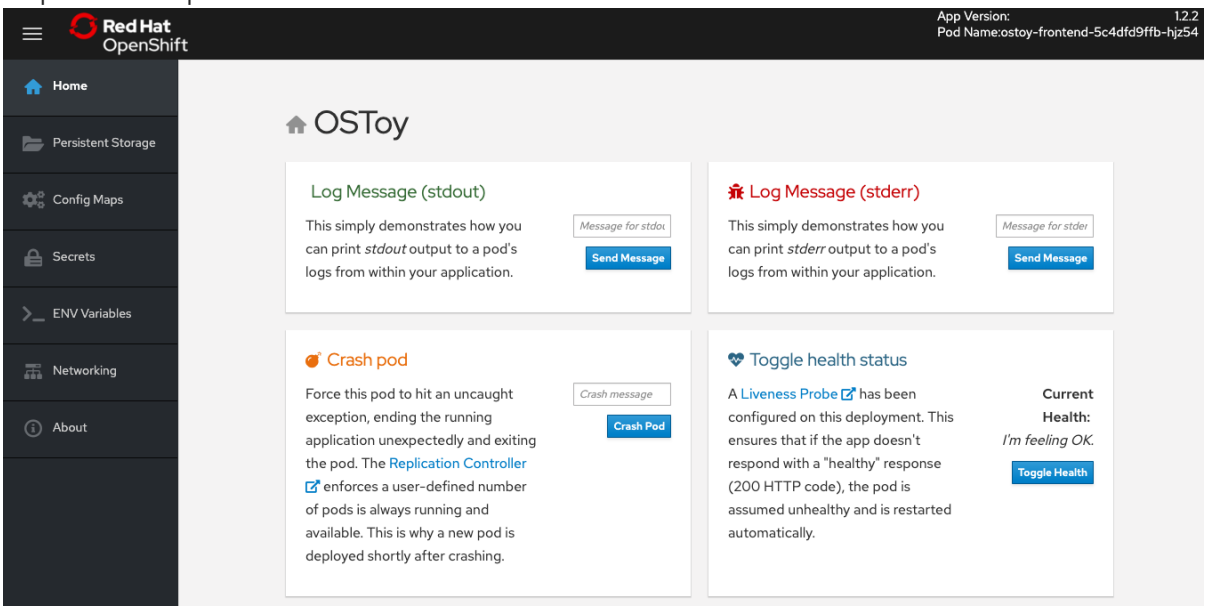
Exemple de sortie

```
NAME          HOST/PORT                                                    PATH   SERVICES          PORT
TERMINATION   WILDCARD
ostoy-route   ostoy-route-ostoy.apps.<your-rosa-cluster>.abcd.p1.openshiftapps.com
ostoy-frontend-svc   <all>                 None
```

18.4.1.6. Affichage de l'application

1. Copiez la sortie de l'URL ostoy-route-ostoy.apps.<your-rosa-cluster>.abcd.p1.openshiftapps.com depuis l'étape précédente.

2. Collez l'URL copiée dans votre navigateur Web et appuyez sur Entrée. Consultez la page d'accueil de votre application. Dans le cas où la page ne se charge pas, assurez-vous d'utiliser http et non https.

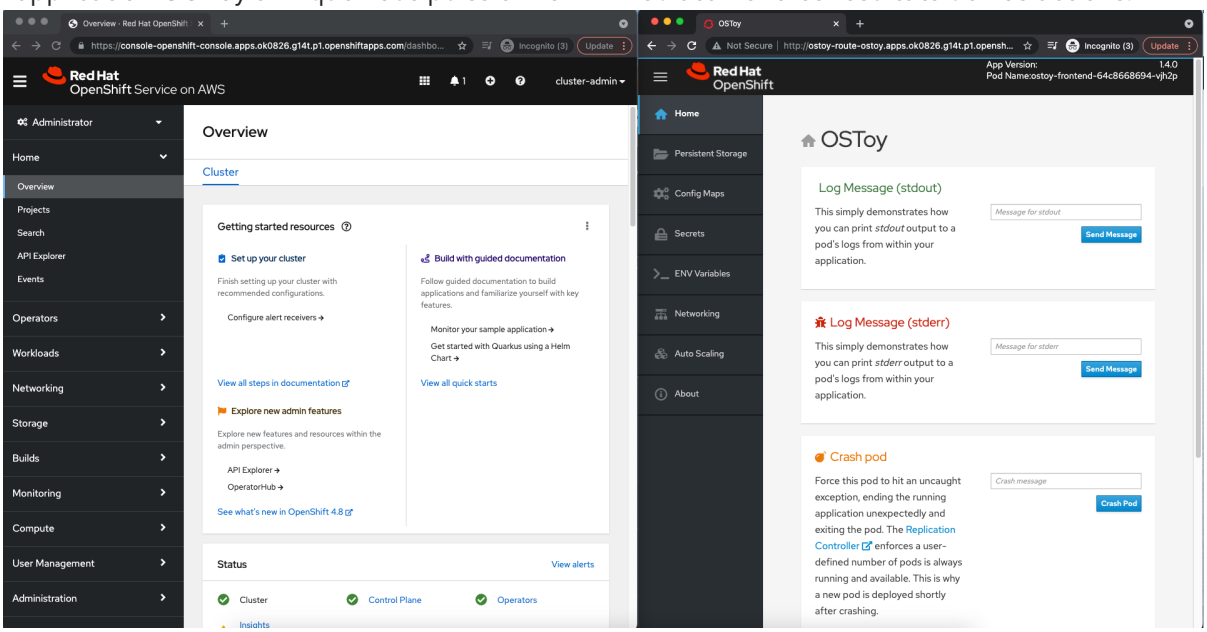


18.5. DIDACTICIEL: VÉRIFICATION DE LA SANTÉ

Il est possible de voir comment Kubernetes réagit à l'échec de la pod en plantant intentionnellement votre gousse et en le rendant insensible aux sondes de vivacité Kubernetes.

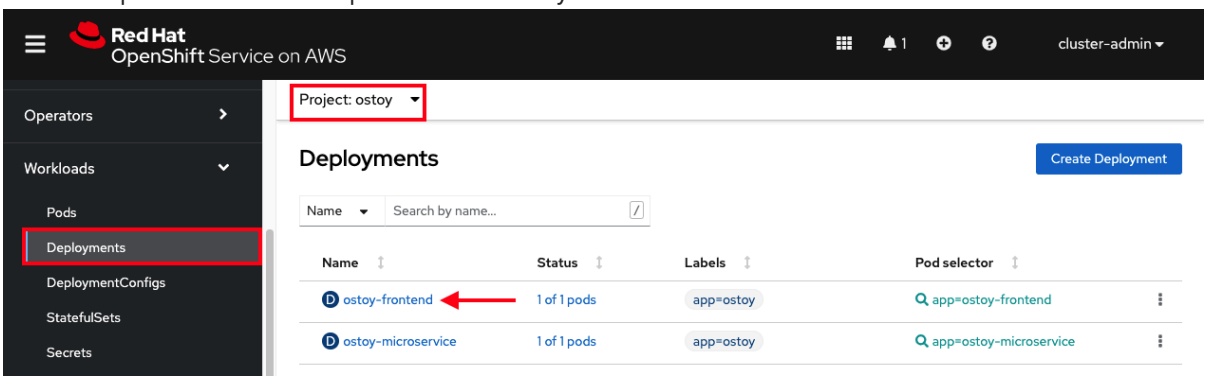
18.5.1. La préparation de votre bureau

1. Divisez votre écran de bureau entre la console Web OpenShift et la console Web de l'application OSToy afin que vous puissiez voir immédiatement les résultats de vos actions.



Lorsque vous ne pouvez pas diviser votre écran, ouvrez la console Web de l'application OSToy dans un autre onglet afin que vous puissiez passer rapidement à la console Web OpenShift après avoir activé les fonctionnalités de l'application.

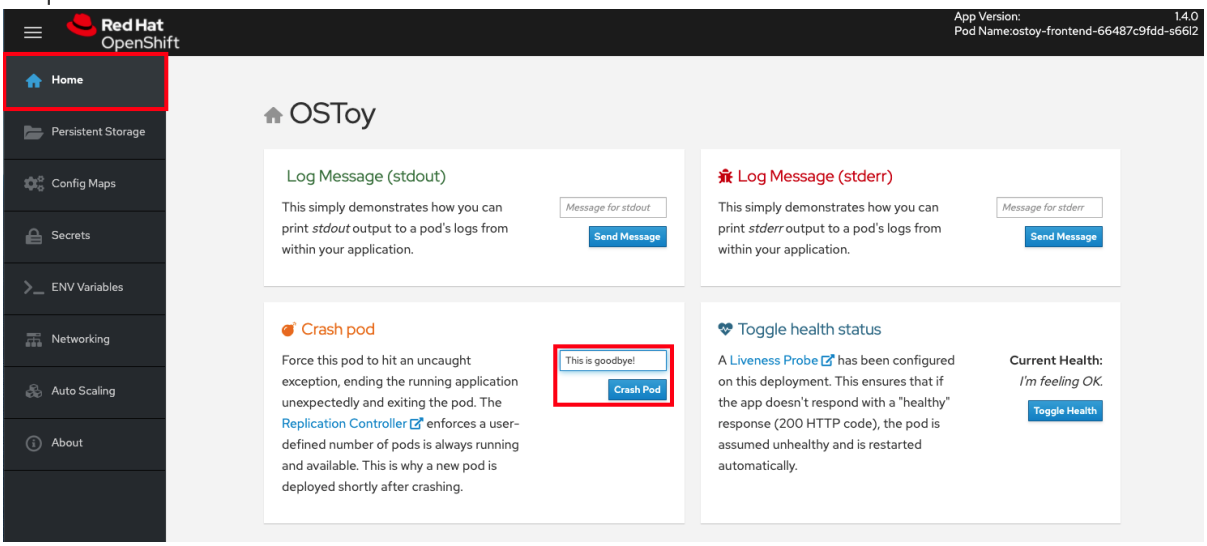
2. Dans la console Web OpenShift, sélectionnez Charges de travail > Déploiements > ostoy-frontend pour afficher le déploiement OSToy.



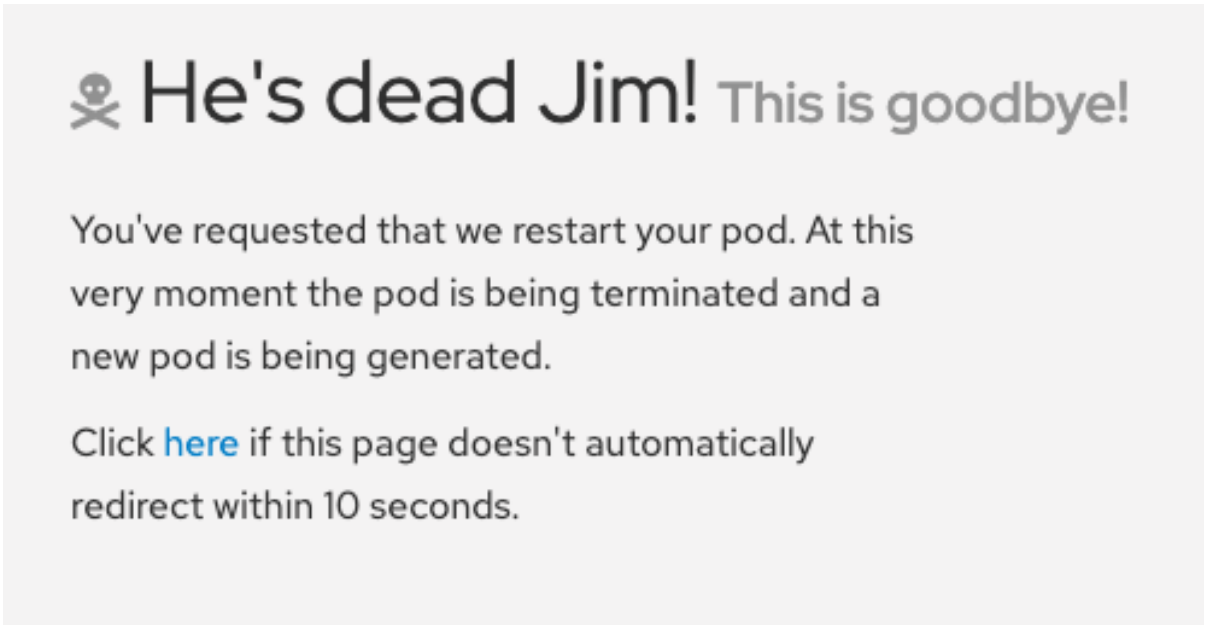
18.5.2. Écraser la gousse

1. À partir de la console Web de l'application OSToy, cliquez sur Accueil dans le menu de gauche, et entrez un message dans la boîte Crash Pod, par exemple, C'est au revoir!.

2. Cliquez sur Crash Pod.

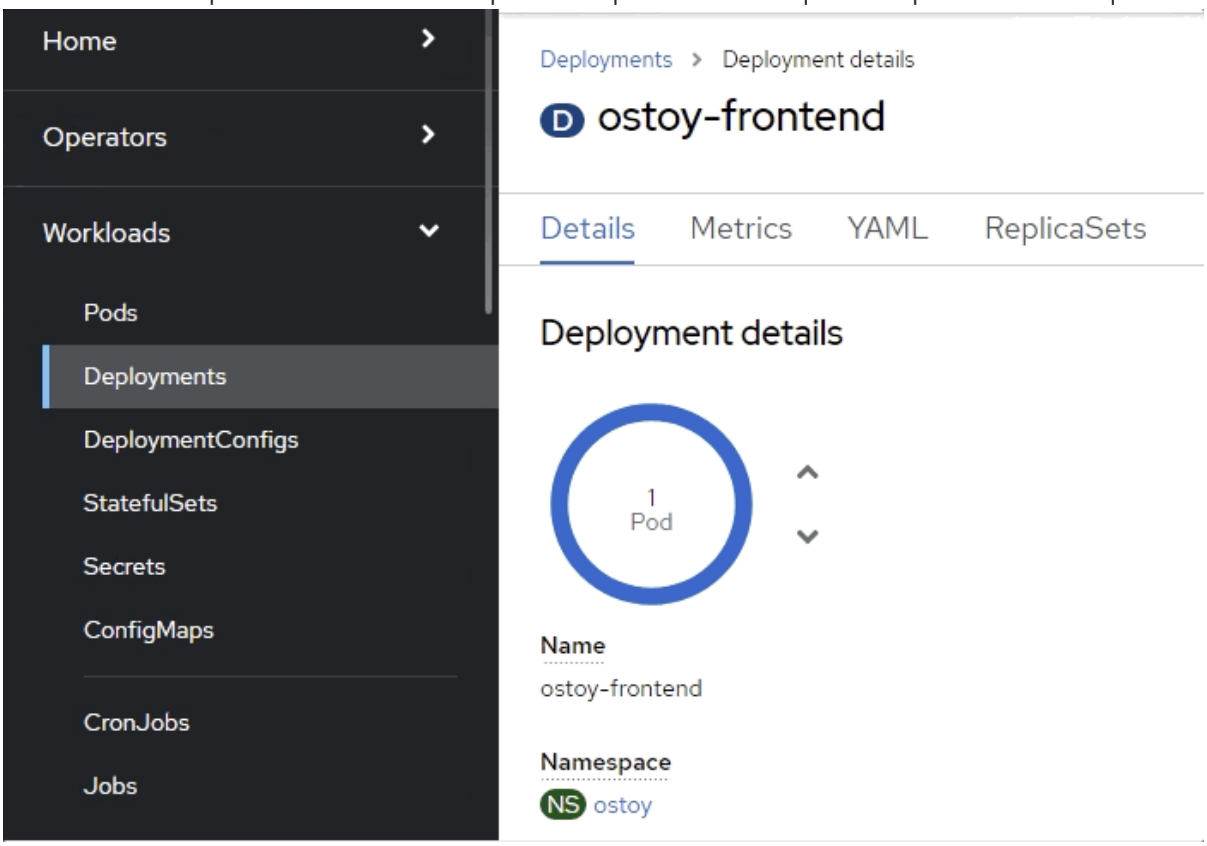


La gousse s'écrase et les Kubernetes devraient redémarrer la gousse.



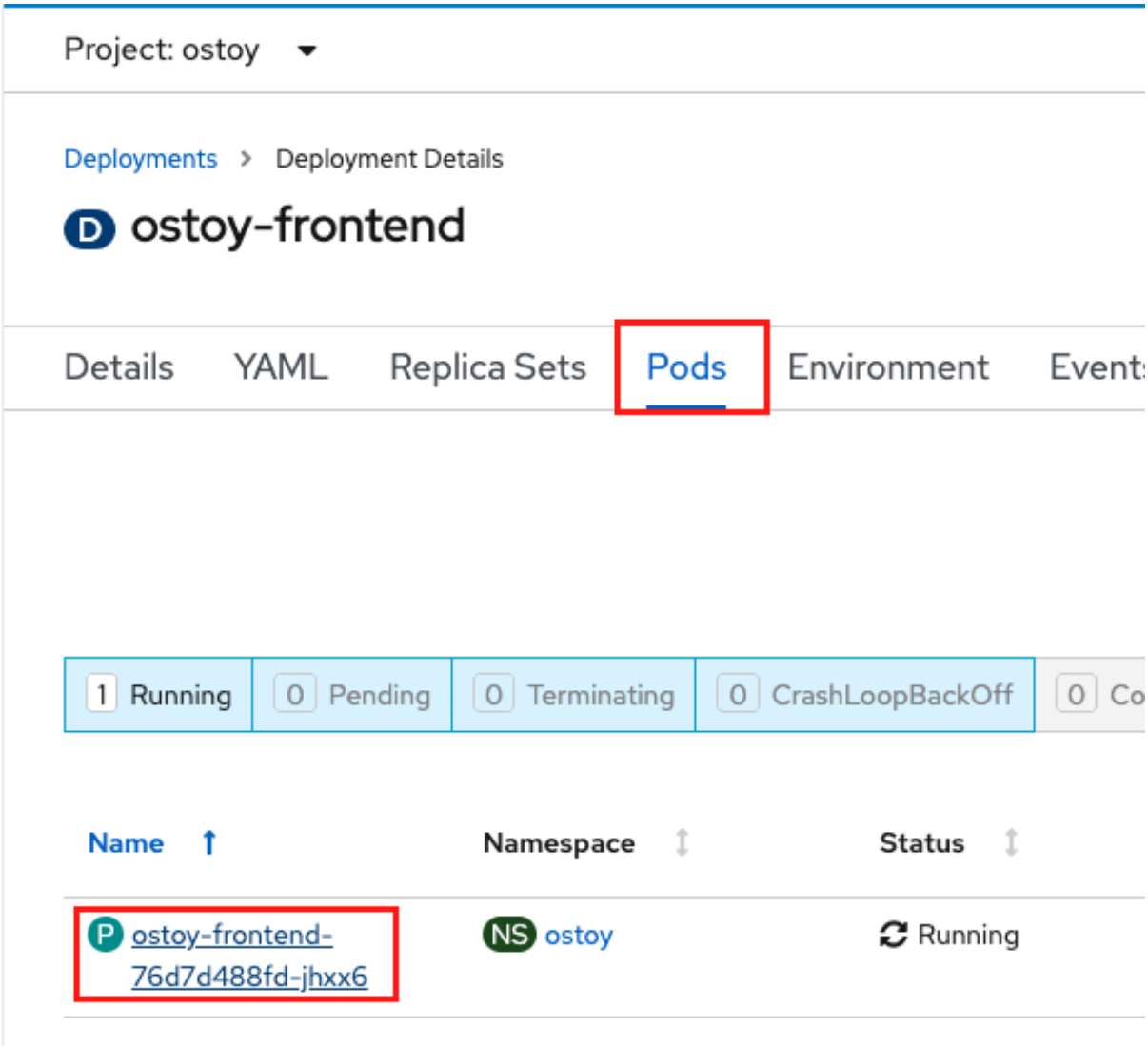
18.5.3. Affichage de la gousse relancée

1. À partir de la console Web OpenShift, passez rapidement à l'écran Déploiements. « vous verrez que la gousse devient jaune, ce qui signifie qu'elle est en baisse. Il devrait rapidement revivre et devenir bleu. Le processus de réveil se produit rapidement afin que vous puissiez le manquer.

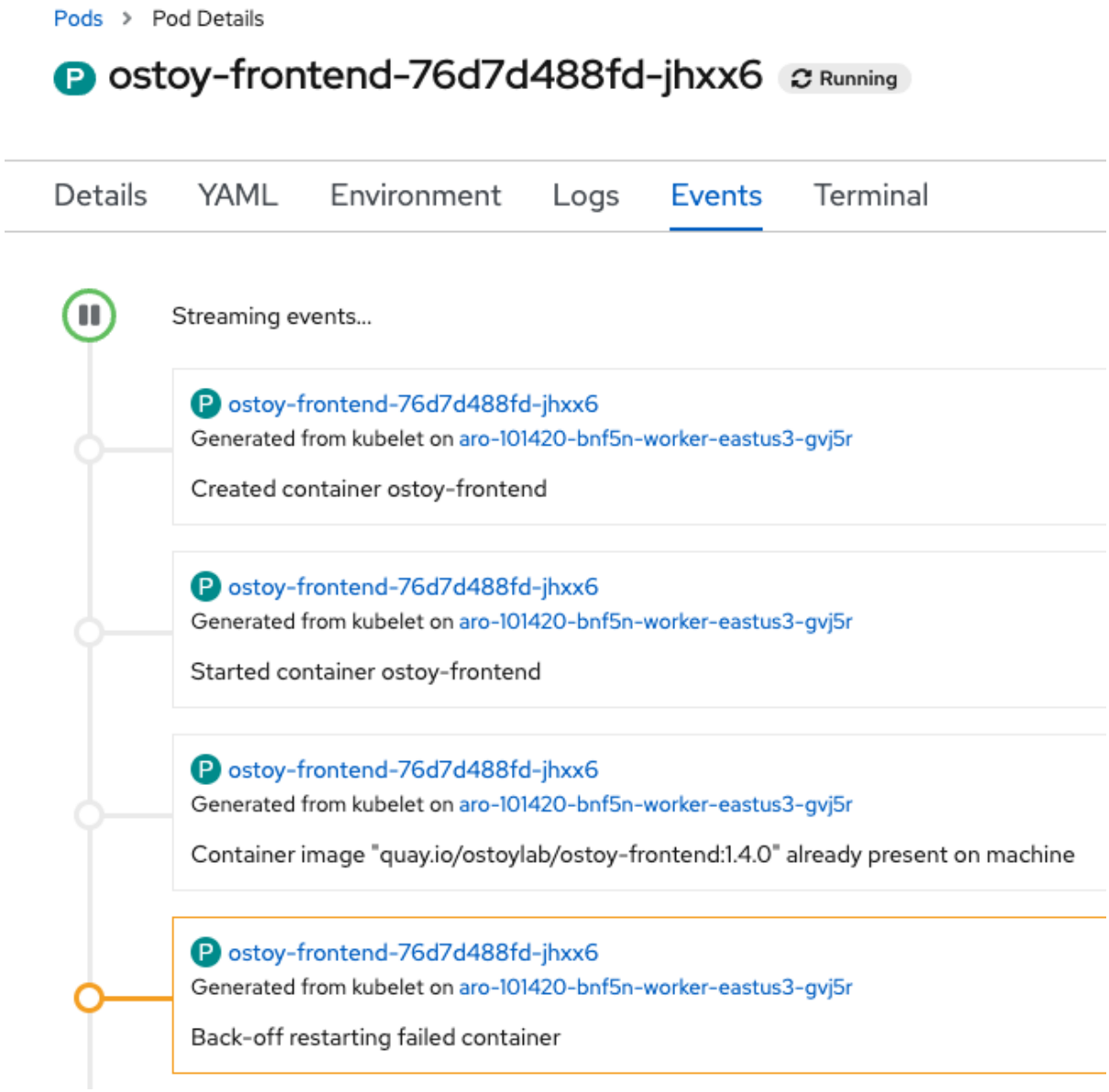


La vérification

1. À partir de la console Web, cliquez sur Pods > ostoy-frontend-xxxxxxx-xxxx pour changer à l'écran des pods.



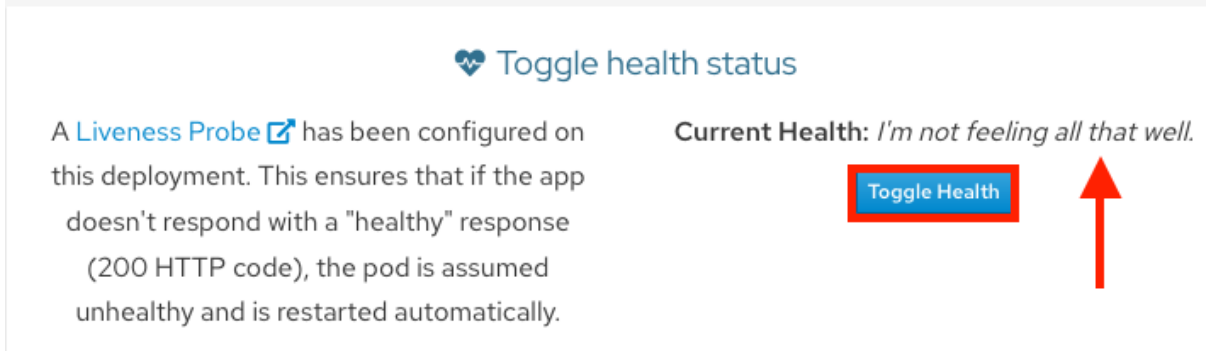
2. Cliquez sur le sous-onglet Événements et vérifiez que le conteneur s'est écrasé et redémarré.



18.5.4. Faire le mauvais fonctionnement de l'application

Gardez la page d'événements de pod ouverte à partir de la procédure précédente.

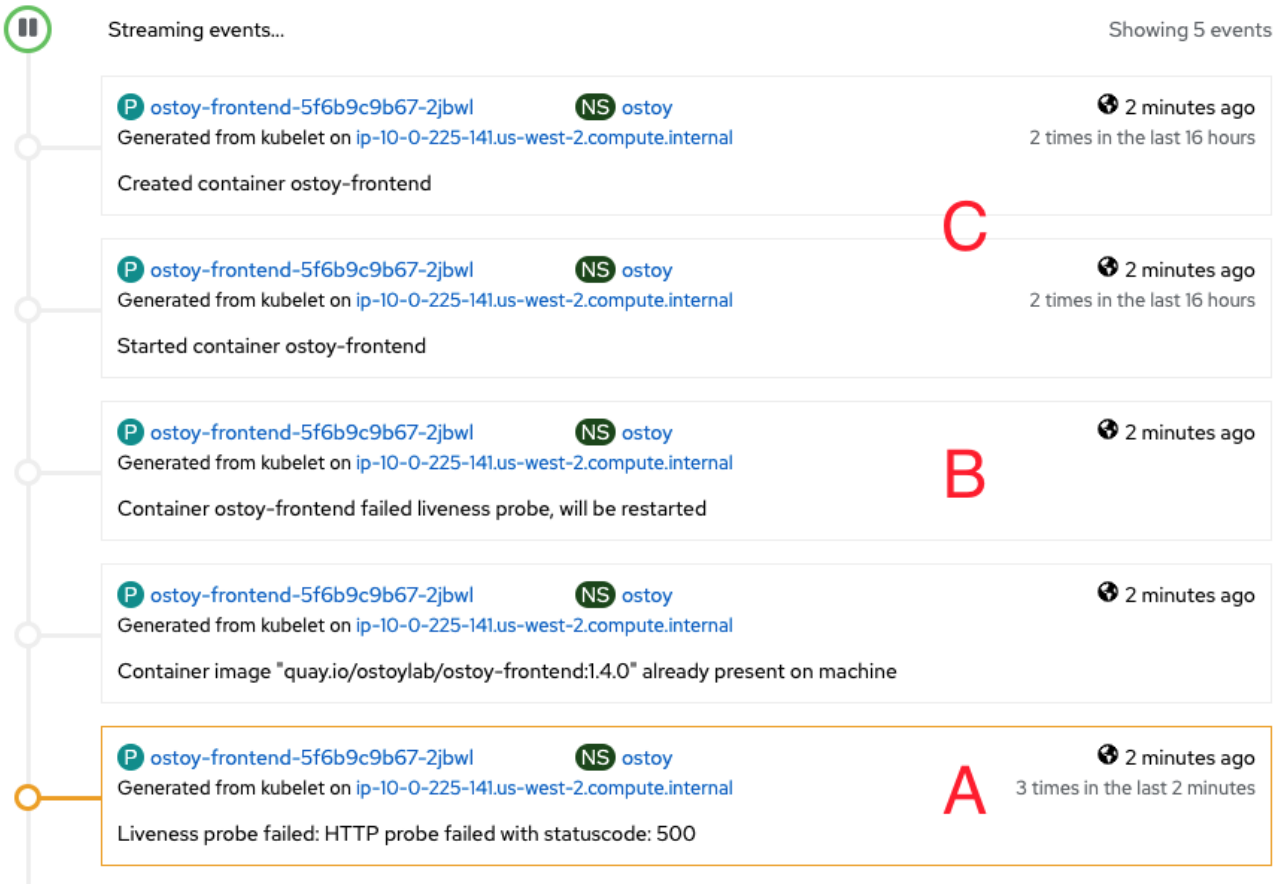
- À partir de l'application OSToy, cliquez sur Toggle Health in the Toggle Health Status tile. La santé actuelle passe à je ne me sens pas si bien.



La vérification

Après l'étape précédente, l'application cesse de répondre avec un code HTTP 200. Après 3 échecs consécutifs, Kubernetes arrêtera le pod et le redémarrera. De la console Web, retournez à la page des événements de pod et vous verrez que la sonde de vivacité a échoué et que le pod a redémarré.

L'image suivante montre un exemple de ce que vous devriez voir sur la page des événements de votre pod.



A. La gousse a trois échecs consécutifs.

B. Kubernetes arrête le pod.

C. Kubernetes redémarre le pod.

18.6. DIDACTICIEL: VOLUMES PERSISTANTS POUR LE STOCKAGE EN CLUSTER

Le service OpenShift Red Hat sur AWS (ROSA) (architecture classique) et le service OpenShift Red Hat sur AWS (ROSA) prennent en charge le stockage de volumes persistants avec Amazon Web Services (AWS) Elastic Block Store (EBS) ou AWS Elastic File System (EFS).

18.6.1. En utilisant des volumes persistants

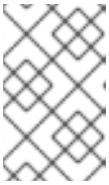
Les procédures suivantes permettent de créer un fichier, de le stocker sur un volume persistant dans votre cluster et de confirmer qu'il existe toujours après l'échec de la pod et la recréation.

18.6.1.1. Affichage d'une revendication de volume persistante

1. Accédez à la console Web OpenShift du cluster.

2. Cliquez sur Stockage dans le menu de gauche, puis cliquez sur PersistentVolumeClaims pour voir une liste de toutes les revendications de volume persistantes.

3. Cliquez sur une revendication de volume de persistance pour voir la taille, le mode d'accès, la classe de stockage et d'autres détails de réclamation supplémentaires.



NOTE

Le mode d'accès est ReadWriteOnce (RWO). Cela signifie que le volume ne peut être monté qu'à un nœud et que le pod ou les gousses peuvent lire et écrire au volume.

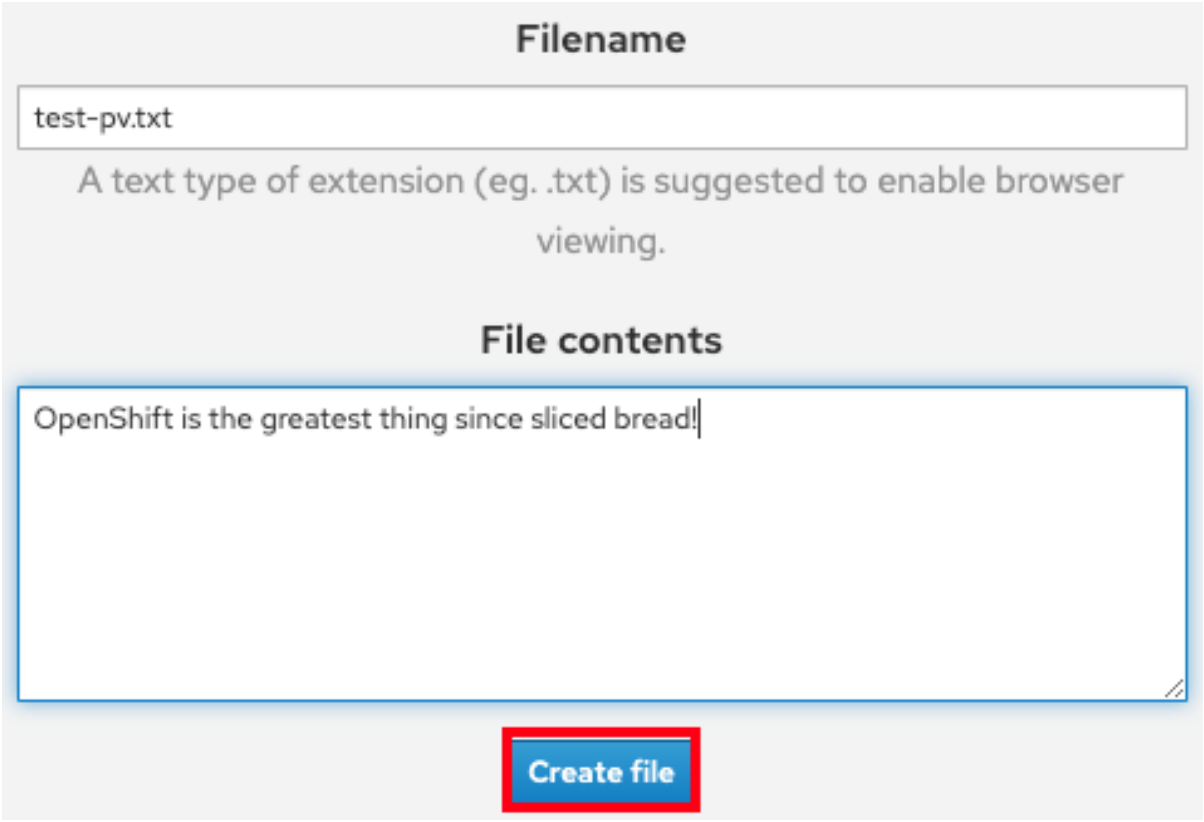
18.6.1.2. Stocker votre fichier

1. Dans la console d'application OSToy, cliquez sur Stockage persistant dans le menu de gauche.

2. Dans la zone Nom de fichier, entrez un nom de fichier avec une extension .txt, par exemple test-pv.txt.

3. Dans la zone de contenu du fichier, entrez une phrase de texte, par exemple OpenShift est la meilleure chose depuis le pain tranché!.

4. Cliquez sur Créer un fichier.



5. Faites défiler jusqu'aux fichiers existants sur la console de l'application OSToy.

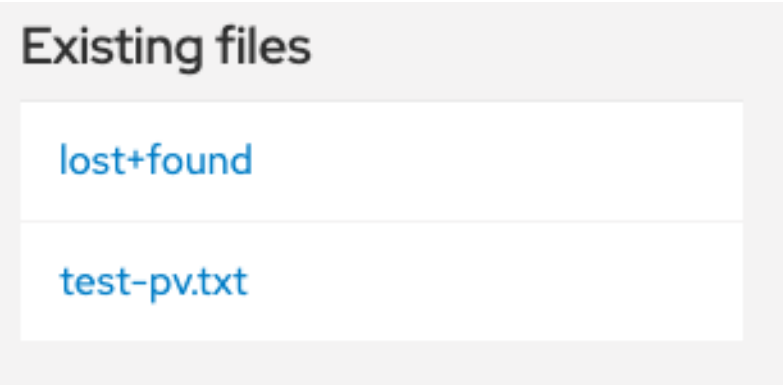
6. Cliquez sur le fichier que vous avez créé pour voir le nom et le contenu du fichier.

18.6.1.3. Écraser la gousse

1. Dans la console de l'application OSToy, cliquez sur Accueil dans le menu de gauche.
2. Cliquez sur Crash Pod.

18.6.1.4. Confirmation du stockage persistant

1. Attendez que la gousse recrée.
2. Dans la console de l'application OSToy, cliquez sur Stockage persistant dans le menu de gauche.
3. Cherchez le fichier que vous avez créé et ouvrez-le pour afficher et confirmer le contenu.



La vérification

Le fichier YAML de déploiement montre que nous avons monté le répertoire /var/demo_files à notre revendication de volume persistante.

1. Il suffit de récupérer le nom de votre pod front-end en exécutant la commande suivante:

```
$ oc get pods
```

2. Démarrez une session shell sécurisée (SSH) dans votre conteneur en exécutant la commande suivante:

```
$ oc rsh <pod_name>
```

3. Allez dans le répertoire en exécutant la commande suivante:

```
$ cd /var/demo_files
```

4. Facultatif: Voir tous les fichiers que vous avez créés en exécutant la commande suivante:

```
$ ls
```

5. Ouvrez le fichier pour afficher le contenu en exécutant la commande suivante:

```
$ cat test-pv.txt
```

6. Assurez-vous que la sortie est le texte que vous avez entré dans la console de l'application OSToy.

Exemple de terminal

```
$ oc get pods
NAME                                READY    STATUS    RESTARTS   AGE
ostoy-frontend-5fc8d486dc-wsw24     1/1      Running   0          18m
ostoy-microservice-6cf764974f-hx4qm 1/1      Running   0          18m

$ oc rsh ostoy-frontend-5fc8d486dc-wsw24

$ cd /var/demo_files/

$ ls
lost+found   test-pv.txt

$ cat test-pv.txt
OpenShift is the greatest thing since sliced bread!
```

18.6.1.5. Fin de la session

- Entrez la sortie dans votre terminal pour quitter la session et retourner au CLI.

18.6.2. Ressources supplémentaires

- En savoir plus sur le stockage en volume persistant, voir Comprendre le stockage persistant.
- En savoir plus sur les options de stockage ROSA, voir Aperçu du stockage.

18.7. DIDACTICIEL: CONFIGMAPS, SECRETS ET VARIABLES D'ENVIRONNEMENT

Ce tutoriel montre comment configurer l'application OSToy en utilisant des variables de configuration des cartes, des secrets et de l'environnement. Consultez ces sujets liés pour plus d'informations.

18.7.1. Configuration à l'aide de ConfigMaps

Les cartes de configuration vous permettent de découpler les artefacts de configuration du contenu de l'image du conteneur pour garder les applications conteneurisées portables.

Procédure

- Dans l'application OSToy, dans le menu de gauche, cliquez sur Config Maps, en affichant le contenu de la carte de configuration disponible pour l'application OSToy. L'extrait de code montre un exemple de configuration de la carte de configuration:

Exemple de sortie

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: ostoy-configmap-files
data:
  config.json:  '{ "default": "123" }'
```

18.7.2. Configuration à l'aide de secrets

Les objets Kubernetes Secret vous permettent de stocker et de gérer des informations sensibles, telles que les mots de passe, les jetons OAuth et les clés SSH. La mise en secret de ces informations est plus sûre et plus flexible que de les mettre en texte clair dans une définition de pod ou une image de conteneur.

Procédure

- Dans l'application OSToy, dans le menu de gauche, cliquez sur Secrets, affichant le contenu des secrets disponibles pour l'application OSToy. L'extrait de code montre un exemple de configuration secrète:

  Exemple de sortie

```
USERNAME=my_user
PASSWORD=VVNFUk5BTUU9bXlfdXNlcgpQQVNTV09SRD1AT3RCbCVYQXAhIzYzMlk1Rn
dDQE1UUWsKU01UUD1sb2NhbGhvc3QKU01UUF9QT1JUPTI1
SMTP=localhost
SMTP_PORT=25
```

18.7.3. Configuration à l'aide de variables d'environnement

L'utilisation de variables d'environnement est un moyen facile de changer le comportement de l'application sans nécessiter de modifications de code. Il permet aux différents déploiements d'une même application de se comporter potentiellement différemment en fonction des variables d'environnement. Le service OpenShift Red Hat sur AWS facilite la définition, la visualisation et la mise à jour des variables d'environnement pour les pods ou les déploiements.

Procédure

- Dans l'application OSToy, dans le menu de gauche, cliquez sur Variables ENV, affichant les variables d'environnement disponibles pour l'application OSToy. L'extrait de code montre un exemple de configuration de variable environnementale:

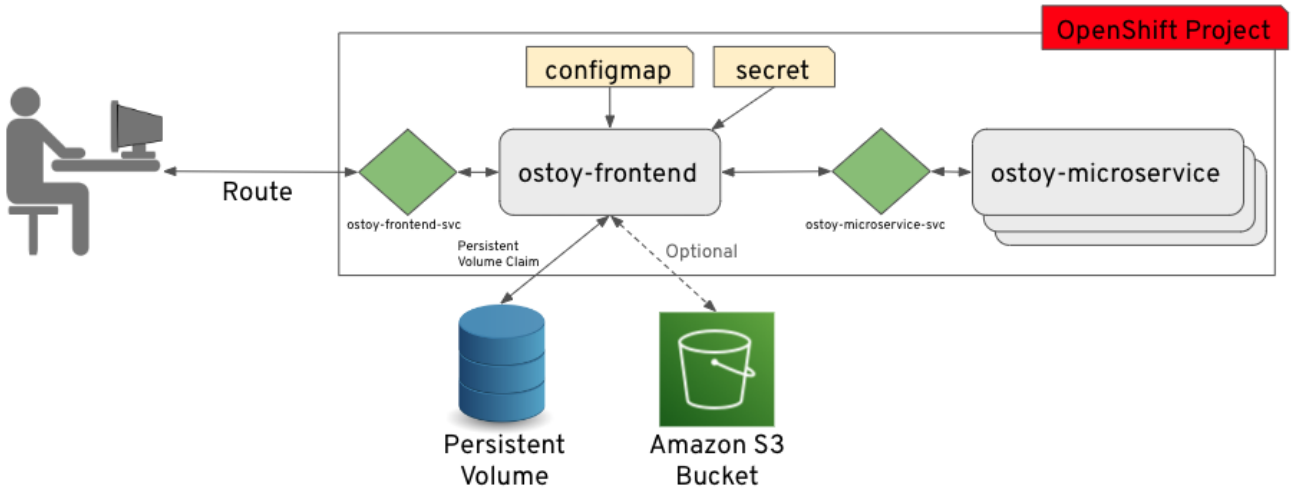
  Exemple de sortie

```
{
  "npm_config_local_prefix": "/opt/app-root/src",
  "STI_SCRIPTS_PATH": "/usr/libexec/s2i",
  "npm_package_version": "1.7.0",
  "APP_ROOT": "/opt/app-root",
  "NPM_CONFIG_PREFIX": "/opt/app-root/src/.npm-global",
  "OSTOY_MICROSERVICE_PORT_8080_TCP_PORT": "8080",
  "NODE": "/usr/bin/node",
  "LD_PRELOAD": "libnss_wrapper.so",
```

```
  "KUBERNETES_SERVICE_HOST": "172.30.0.1",
  "OSTOY_MICROSERVICE_PORT": "tcp://172.30.60.255:8080",
  "OSTOY_PORT": "tcp://172.30.152.25:8080",
  "npm_package_name": "ostoy",
  "OSTOY_SERVICE_PORT_8080_TCP": "8080",
  "_": "/usr/bin/node"
  "ENV_TOY_CONFIGMAP": "ostoy-configmap -env"
}
```


18.8. DIDACTICIEL: NETWORKING

Ce tutoriel montre comment l'application OSToy utilise le réseau intra-cluster pour séparer les fonctions en utilisant des microservices et visualiser l'échelle des pods.



Le diagramme montre qu'il y a au moins deux pods distincts, chacun avec son propre service.

Il s'agit d'une application web frontale avec un service et un itinéraire accessible au public. L'autre pod fonctionne comme le microservice backend avec un objet de service afin que le pod avant puisse communiquer avec le microservice. Cette communication se produit à travers les gousses si plus d'un. En raison de ces limites de communication, ce microservice n'est pas accessible de l'extérieur de ce cluster ou d'autres espaces de noms ou projets si ceux-ci sont configurés. Le seul but de ce microservice est de servir les requêtes Web internes et de renvoyer un objet JSON contenant le nom d'hôte actuel, qui est le nom du pod, et une chaîne de couleurs générée au hasard. Cette chaîne de couleur est utilisée pour afficher une boîte avec cette couleur affichée dans la tuile intitulée « Communication intra-cluster ».

En savoir plus sur les limites du réseau, voir À propos de la politique de réseau.

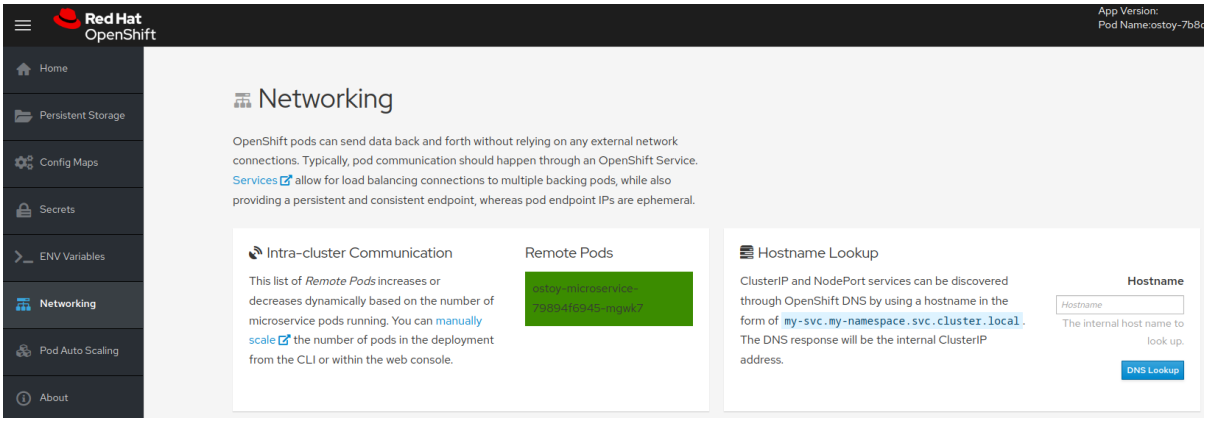
18.8.1. La mise en réseau intra-cluster

Dans votre application OSToy, vous pouvez visualiser vos configurations de réseautage.

Procédure

1. Dans l'application OSToy, cliquez sur Networking dans le menu de gauche.

2. Examinez la configuration de mise en réseau. La bonne tuile intitulée "Hostname Lookup" illustre comment le nom de service créé pour un pod peut être utilisé pour traduire en une adresse ClusterIP interne.



3. Entrez le nom du microservice créé dans la tuile droite ("Hostname Lookup") en suivant le format de <service_name>.<namespace>.svc.cluster.local. Ce nom de service se trouve dans la définition de service de ostoy-microservice.yaml en exécutant la commande suivante:

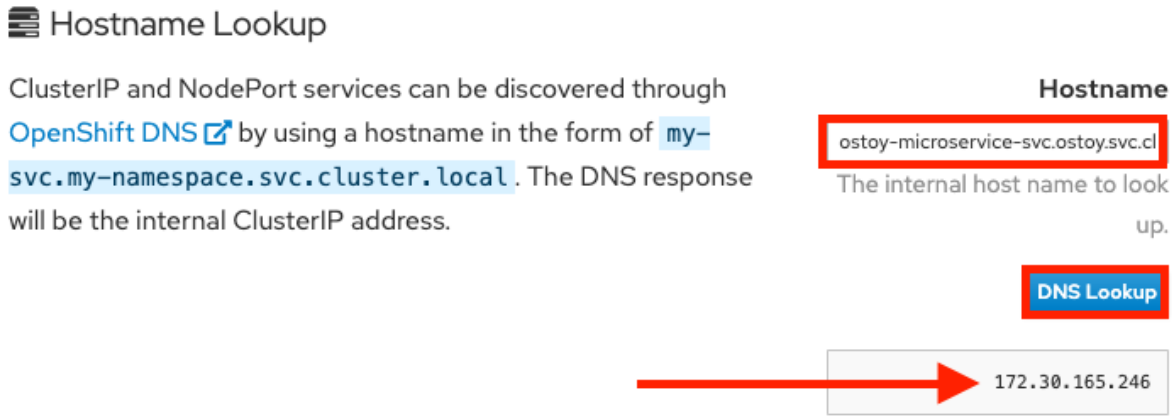
```
$ oc get service <name_of_service> -o yaml
```

Exemple de sortie

```
apiVersion: v1
kind: Service
metadata:
  name: ostoy-microservice-svc
  labels:
    app: ostoy-microservice
spec:
  type: ClusterIP
  ports:
    - port: 8080
      targetPort: 8080
      protocol: TCP
  selector:
    app: ostoy-microservice
```

Dans cet exemple, le nom d'hôte complet est ostoy-microservice-svc.ostoy.svc.cluster.local.

4. Il y a une adresse IP retournée. Dans cet exemple, il est 172.30.165.246. Il s'agit de l'adresse IP intra-cluster, qui n'est accessible qu'à partir du cluster.



18.9. TUTORIEL: MISE À L'ÉCHELLE D'UNE APPLICATION

18.9.1. La mise à l'échelle

Il est possible d'adapter manuellement ou automatiquement vos pods à l'aide de l'Autoscaler Horizontal Pod (HPA). Il est également possible de mettre à l'échelle vos nœuds de cluster.

18.9.1.1. Evoluation manuelle des pod

Il est possible de mettre à l'échelle manuellement les pods de votre application en utilisant l'une des méthodes suivantes:

- Changer votre définition de ReplicaSet ou de déploiement
- En utilisant la ligne de commande
- À l'aide de la console web

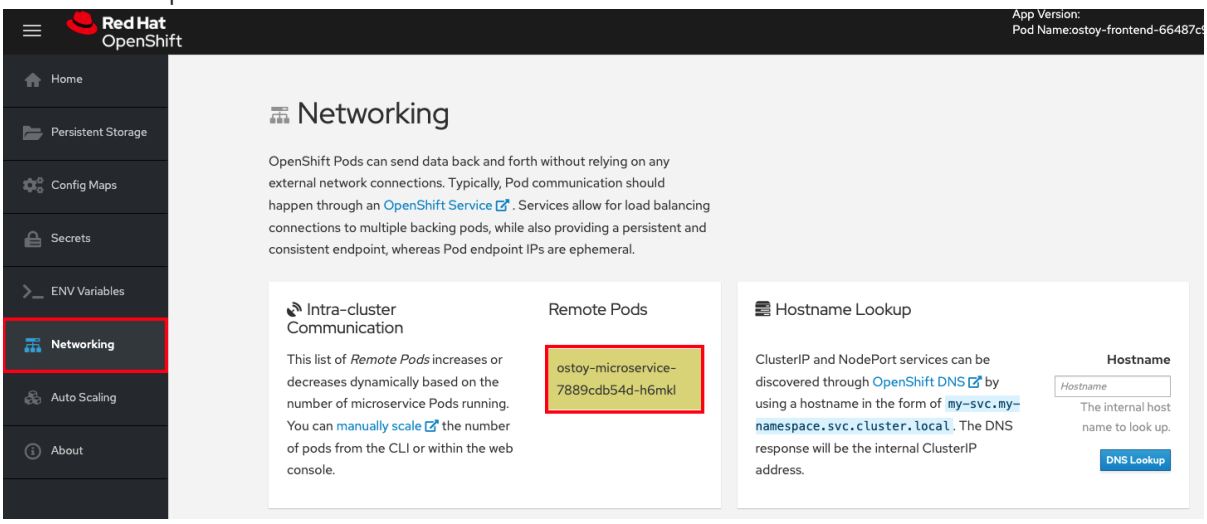
Cet atelier commence par l'utilisation d'un seul pod pour le microservice. En définissant une réplique de 1 dans votre définition de déploiement, le contrôleur de réplication Kubernetes s'efforce de garder un pod en vie. Ensuite, vous apprenez à définir l'autoscaling de pod en utilisant le Horizontal Pod Autoscaler (HPA) qui est basé sur la charge et mettra à l'échelle plus de pods, au-delà de votre définition initiale, si une charge élevée est expérimentée.

Conditions préalables

- Cluster ROSA actif
- A déloyed l'application OSToy

Procédure

1. Dans l'application OSToy, cliquez sur l'onglet Networking dans le menu navigation.
2. Dans la section "Communication intra-cluster", localisez la boîte située sous "Pods à distance" qui change au hasard les couleurs. À l'intérieur de la boîte, vous voyez le nom de la gousse du microservice. Il n'y a qu'une seule boîte dans cet exemple parce qu'il n'y a qu'un seul microservice pod.



3. Confirmez qu'il n'y a qu'un seul pod pour le microservice en exécutant la commande suivante:

```
$ oc get pods
```

Exemple de sortie

```
NAME                              READY    STATUS    RESTARTS   AGE
ostoy-frontend-679cb85695-5cn7x       1/1       Running   0          1h
ostoy-microservice-86b4c6f559-p594d   1/1       Running   0          1h
```

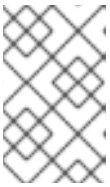
4. Enregistrez-le sur votre ordinateur local et enregistrez-le sur votre machine locale.

5. Changer la définition de déploiement en trois pods au lieu d'un en utilisant l'exemple suivant:

```
spec:
   selector:
     matchLabels:
       app: ostoy-microservice
   replicas: 3
```

6. Appliquez les modifications de réplique en exécutant la commande suivante:

```
$ oc apply -f ostoy-microservice-deployment.yaml
```



NOTE

En outre, vous pouvez modifier le fichier ostoy-microservice-deployment.yaml dans la console Web OpenShift en allant dans l'onglet Charges de travail > Déploiements > ostoy-microservice > onglet YAML.

7. Confirmez qu'il y a maintenant 3 pods en exécutant la commande suivante:

```
$ oc get pods
```

La sortie montre qu'il y a maintenant 3 pods pour le microservice au lieu d'un seul.

Exemple de sortie

```
NAME                              READY   STATUS    RESTARTS   AGE
ostoy-frontend-5fbcc7d9-rzlgz         1/1     Running   0          26m
ostoy-microservice-6666dcf455-2lcv4   1/1     Running   0          81s
ostoy-microservice-6666dcf455-5z56w   1/1     Running   0          81s
ostoy-microservice-6666dcf455-tqzmn   1/1     Running   0          26m
```

8. Faites évoluer l'application en utilisant le CLI ou en utilisant l'interface utilisateur web:

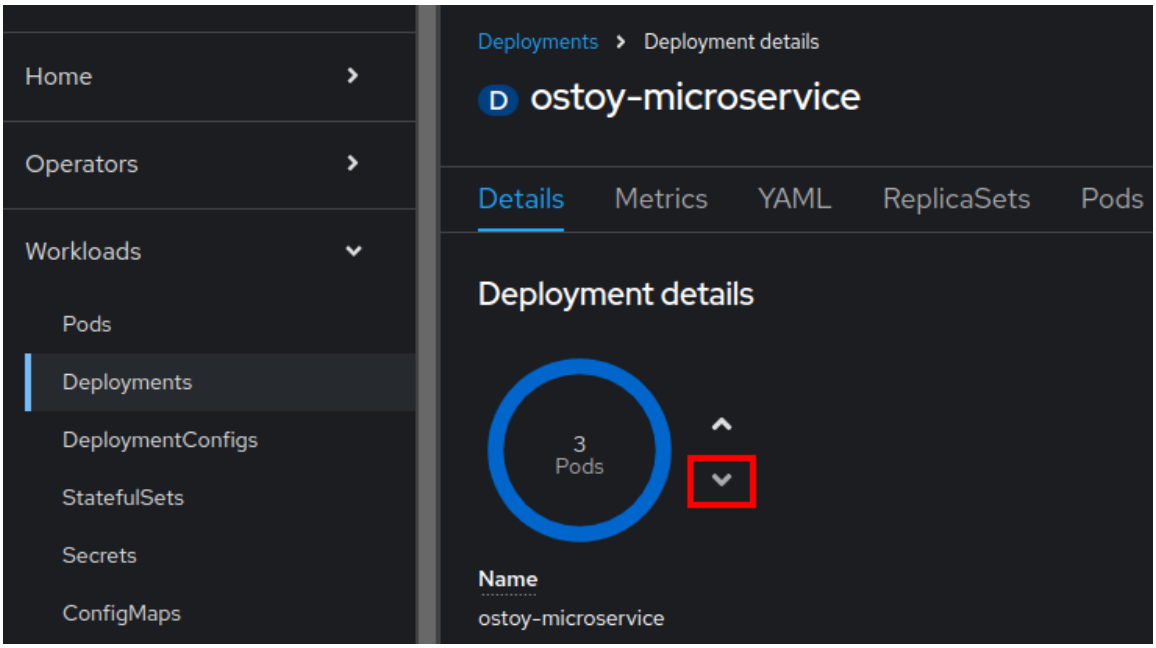
   - Dans le CLI, diminuer le nombre de pods de 3 à 2 en exécutant la commande suivante:

     ```
     $ oc scale deployment ostoy-microservice --replicas=2
     ```

   - Dans le menu de navigation de l'interface utilisateur de la console Web OpenShift, cliquez sur Charges de travail > Déploiements > ostoy-microservice.

   - À gauche de la page, localisez le cercle bleu avec une étiquette "3 Pod" au milieu.

- La sélection des flèches à côté du cercle met à l'échelle le nombre de gousses. Choisissez la flèche vers le bas à 2.



La vérification

Consultez le compte de vos pods en utilisant le CLI, l'interface utilisateur Web ou l'application OSToy:

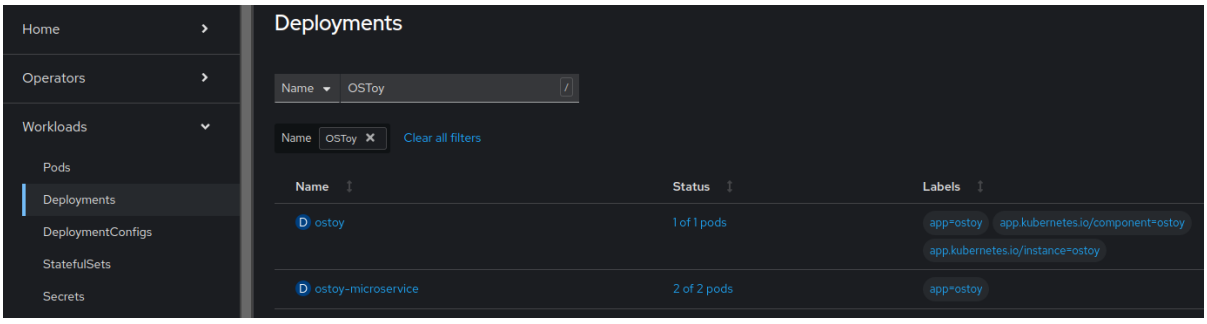
- À partir du CLI, confirmez que vous utilisez deux pods pour le microservice en exécutant la commande suivante:

```
$ oc get pods
```

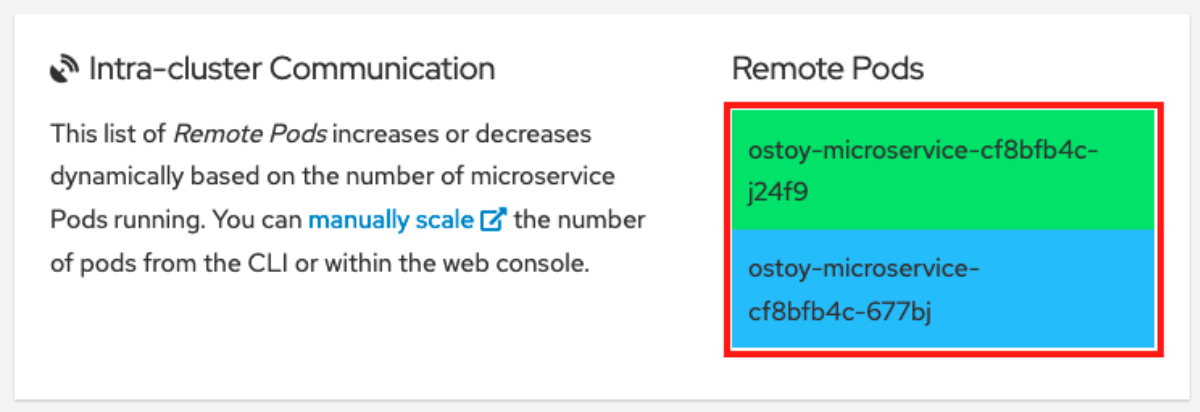
Exemple de sortie

```
NAME                                READY   STATUS    RESTARTS   AGE
ostoy-frontend-5fbcc7d9-rzlgz       1/1     Running   0          75m
ostoy-microservice-6666dcf455-2lcv4 1/1     Running   0          50m
ostoy-microservice-6666dcf455-tqzmn 1/1     Running   0          75m
```

- Dans l'interface utilisateur Web, sélectionnez Charges de travail > Déploiements > ostoy-microservice.



- En outre, vous pouvez confirmer qu'il existe deux pods en sélectionnant Networking dans le menu de navigation de l'application OSToy. Il devrait y avoir deux boîtes colorées pour les deux gousses.

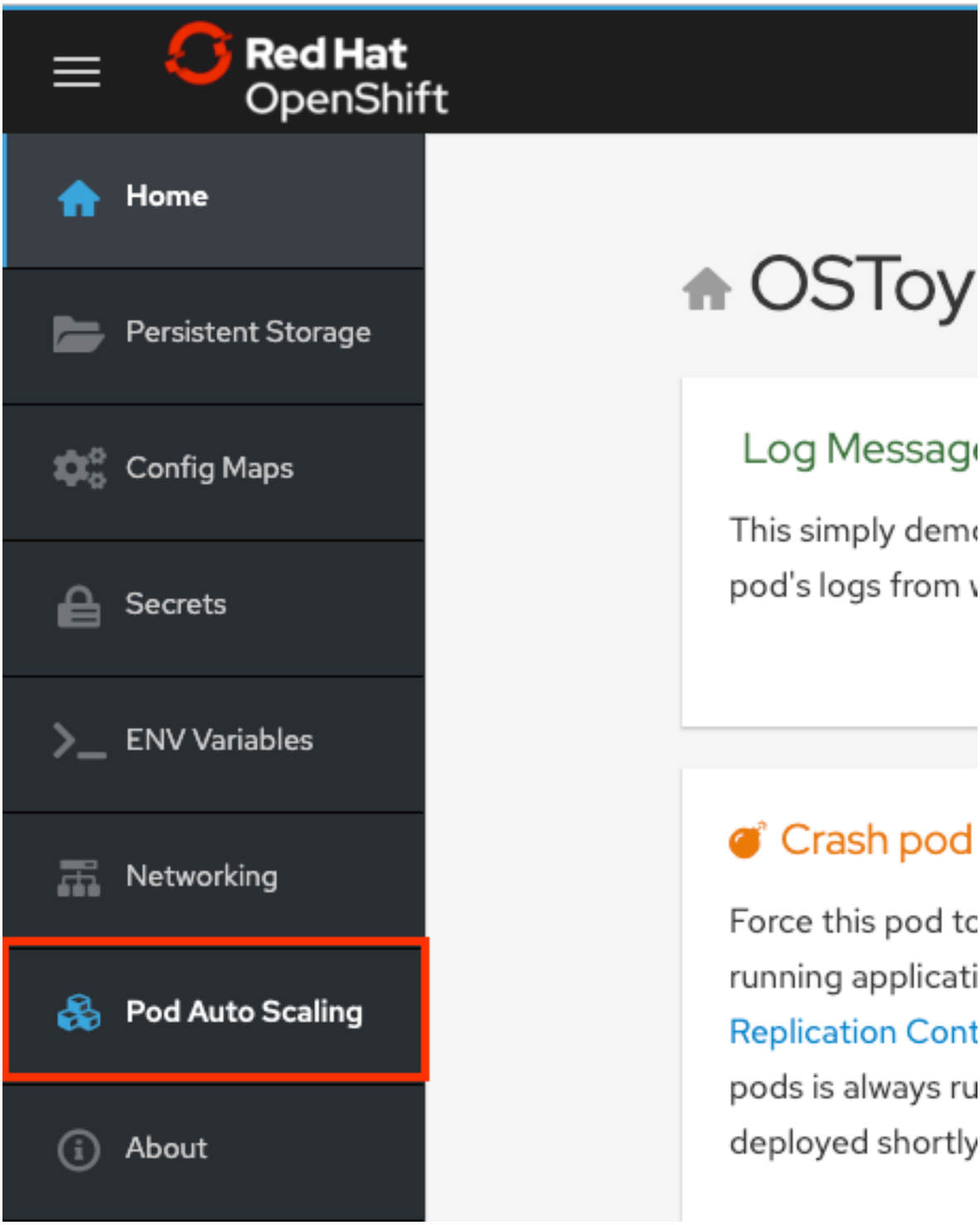


18.9.1.2. La pod Autoscaling

Le service OpenShift Red Hat sur AWS offre un Autoscaler Horizontal Pod (HPA). La HPA utilise des mesures pour augmenter ou diminuer le nombre de gousses si nécessaire.

Procédure

1. Dans le menu de navigation de l'interface utilisateur Web, sélectionnez Pod Auto Scaling.



2. Créez l'HPA en exécutant la commande suivante:

```
$ oc autoscale deployment/ostoy-microservice --cpu-percent=80 --min=1 --max=10
```

Cette commande crée une HPA qui maintient entre 1 et 10 répliques des pods contrôlés par le déploiement ostoy-microservice. Bien que le déploiement, HPA augmente et diminue le nombre de répliques pour maintenir l'utilisation moyenne du CPU dans toutes les gousses à 80% et 40 millicores.

3. Dans la page Pod Auto Scaling > Horizontal Pod Autoscaling, sélectionnez Augmenter la charge.

IMPORTANT

Étant donné que l'augmentation de la charge génère des calculs intensifs de CPU, la page peut devenir insensible. C'est une réponse attendue. Cliquez sur Augmenter la charge une seule fois. Consultez le référentiel GitHub du microservice pour plus d'informations sur le processus.

Après quelques minutes, les nouveaux pods s'affichent sur la page représentée par des boîtes colorées.

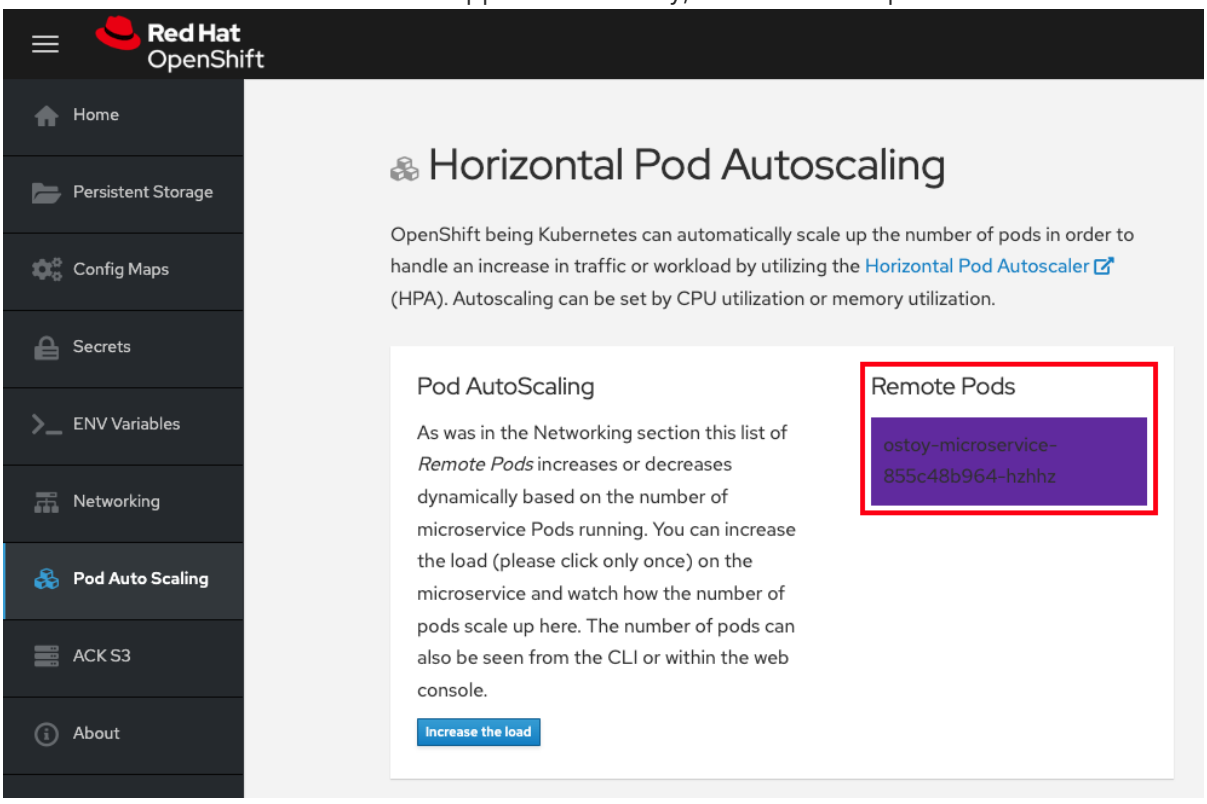
NOTE

La page peut faire l'expérience d'un décalage.

La vérification

Consultez le comptage de vos gousses avec l'une des méthodes suivantes:

- Dans l'interface utilisateur Web de l'application OSToy, voir la boîte de pods à distance:



Étant donné qu'il n'y a qu'une seule gousse, l'augmentation de la charge de travail devrait déclencher une augmentation des gousses.

- Dans le CLI, exécutez la commande suivante:

```
oc get pods --field-selector=status.phase=Running | grep microservice
```

Exemple de sortie

```
ostoy-microservice-79894f6945-cdmbd   1/1     Running   0          3m14s
ostoy-microservice-79894f6945-mgwk7   1/1     Running   0          4h24m
ostoy-microservice-79894f6945-q925d   1/1     Running   0          3m14s
```

- Il est également possible de vérifier la mise à l'échelle automatique à partir du gestionnaire de cluster OpenShift

1. Dans le menu de navigation de la console Web OpenShift, cliquez sur Observer > Tableau de bord.

2. Dans le tableau de bord, sélectionnez Kubernetes / Compute Resources / Namespace (Pods) et votre espace de noms.

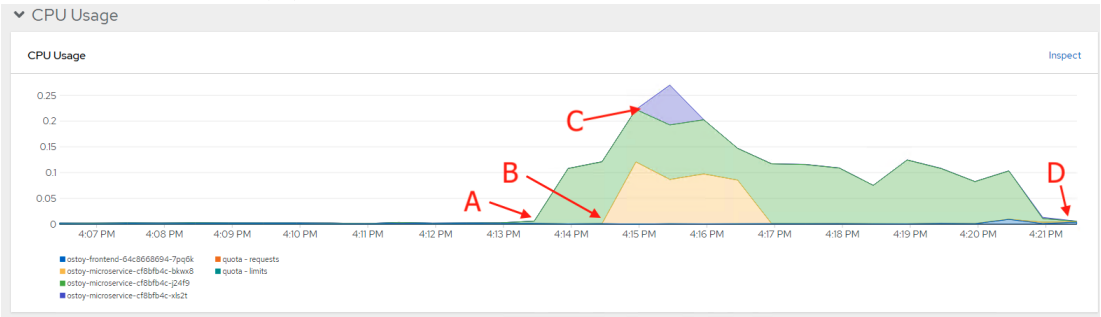
3. Le graphique apparaît montrant l'utilisation de vos ressources à travers le CPU et la mémoire. Le graphique supérieur montre la consommation récente de CPU par pod et le graphique inférieur indique l'utilisation de la mémoire. Ce qui suit répertorie les appels dans le graphique:

a. La charge a augmenté (A).

b. Deux nouveaux pods ont été créés (B et C).

c. L'épaisseur de chaque graphique représente la consommation de CPU et indique quelles gousses manipulaient plus de charge.

d. La charge a diminué (D), et les gousses ont été supprimées.



18.9.1.3. La mise à l'échelle des nœuds

Le service Red Hat OpenShift sur AWS vous permet d'utiliser l'autoscaling des nœuds. Dans ce scénario, vous allez créer un nouveau projet avec un travail qui a une charge de travail importante que le cluster ne peut pas gérer. Avec la mise à l'échelle automatique activée, lorsque la charge est supérieure à votre capacité actuelle, le cluster créera automatiquement de nouveaux nœuds pour gérer la charge.

Conditions préalables

- L'autoscaling est activé sur vos pools de machines.

Procédure

1. Créez un nouveau projet appelé autoscale-ex en exécutant la commande suivante:

```
$ oc new-project autoscale-ex
```

2. Créez le travail en exécutant la commande suivante:

```
$ oc create -f https://raw.githubusercontent.com/openshift-cs/rosaworkshop/master/rosa-workshop/ostoy/yaml/job-work-queue.yaml
```

3. Après quelques minuts, exécutez la commande suivante pour voir les pods:

```
$ oc get pods
```

Exemple de sortie

```
NAME                READY   STATUS    RESTARTS   AGE
work-queue-5x2nq-24xxn   0/1     Pending   0          10s
work-queue-5x2nq-57zpt   0/1     Pending   0          10s
work-queue-5x2nq-58bvs   0/1     Pending   0          10s
work-queue-5x2nq-6c5tl   1/1     Running   0          10s
work-queue-5x2nq-7b84p   0/1     Pending   0          10s
work-queue-5x2nq-7hktm   0/1     Pending   0          10s
work-queue-5x2nq-7md52   0/1     Pending   0          10s
work-queue-5x2nq-7qgmp   0/1     Pending   0          10s
work-queue-5x2nq-8279r   0/1     Pending   0          10s
work-queue-5x2nq-8rkj2   0/1     Pending   0          10s
work-queue-5x2nq-96cdl   0/1     Pending   0          10s
work-queue-5x2nq-96tfr   0/1     Pending   0          10s
```

4. Étant donné qu'il y a beaucoup de pods dans un état en attente, ce statut devrait déclencher l'autoscaler pour créer plus de nœuds dans votre pool de machines. Laissez le temps de créer ces nœuds de travail.

5. Après quelques minutes, utilisez la commande suivante pour voir combien de nœuds de travail vous avez maintenant:

```
$ oc get nodes
```

Exemple de sortie

```
NAME                                         STATUS   ROLES          AGE     VERSION
ip-10-0-138-106.us-west-2.compute.internal   Ready    infra,worker   22h     v1.23.5+3afdacb
ip-10-0-153-68.us-west-2.compute.internal    Ready    worker         2m12s   v1.23.5+3afdacb
ip-10-0-165-183.us-west-2.compute.internal   Ready    worker         2m8s    v1.23.5+3afdacb
ip-10-0-176-123.us-west-2.compute.internal   Ready    infra,worker   22h     v1.23.5+3afdacb
ip-10-0-195-210.us-west-2.compute.internal   Ready    master         23h     v1.23.5+3afdacb
ip-10-0-196-84.us-west-2.compute.internal    Ready    master         23h     v1.23.5+3afdacb
ip-10-0-203-104.us-west-2.compute.internal   Ready    worker         2m6s    v1.23.5+3afdacb
ip-10-0-217-202.us-west-2.compute.internal   Ready    master         23h     v1.23.5+3afdacb
ip-10-0-225-141.us-west-2.compute.internal   Ready    worker         23h     v1.23.5+3afdacb
ip-10-0-231-245.us-west-2.compute.internal   Ready    worker         2m11s   v1.23.5+3afdacb
ip-10-0-245-27.us-west-2.compute.internal    Ready    worker         2m8s    v1.23.5+3afdacb
ip-10-0-245-7.us-west-2.compute.internal     Ready    worker         23h     v1.23.5+3afdacb
```

Les nœuds de travail ont été créés automatiquement pour gérer la charge de travail.

6. De retour à l'application OSToy en entrant la commande suivante:

```
$ oc project ostoy
```

18.10. DIDACTICIEL: LOGGING

Il existe différentes méthodes pour afficher vos journaux dans Red Hat OpenShift Service sur AWS (ROSA). Les procédures suivantes permettent de transmettre les journaux à AWS CloudWatch et de visualiser les journaux directement via le pod en utilisant les journaux oc.

NOTE

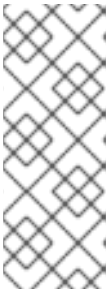
La ROSA n'est pas préconfigurée avec une solution de journalisation.

18.10.1. Envoi de journaux vers CloudWatch

Installez le service d'extension de journalisation pour transférer les journaux à AWS CloudWatch.

1. Exécutez le script suivant pour configurer votre cluster ROSA pour transférer les journaux vers CloudWatch:

```
$ curl https://raw.githubusercontent.com/openshift-cs/rosaworkshop/master/rosa-
workshop/ostoy/resources/configure-cloudwatch.sh | bash
```



NOTE

Configurer ROSA pour envoyer des journaux à CloudWatch va au-delà du cadre de ce tutoriel. L'intégration à AWS et l'activation de la journalisation CloudWatch sont des aspects importants de ROSA, de sorte qu'un script est inclus pour simplifier le processus de configuration. Le script configurera automatiquement AWS CloudWatch. Il est possible d'examiner le script pour comprendre les étapes impliquées.

Exemple de sortie

```
Varaibles are set...ok.
Policy already exists...ok.
Created RosaCloudWatch-mycluster role.
Attached role policy.
Deploying the Red Hat OpenShift Logging Operator
namespace/openshift-logging configured
operatorgroup.operators.coreos.com/cluster-logging created
subscription.operators.coreos.com/cluster-logging created
Waiting for Red Hat OpenShift Logging Operator deployment to complete...
Red Hat OpenShift Logging Operator deployed.
secret/cloudwatch-credentials created
clusterlogforwarder.logging.openshift.io/instance created
clusterlogging.logging.openshift.io/instance created
Complete.
```

2. Après quelques minutes, vous devriez commencer à voir les groupes de journaux à l'intérieur d'AWS CloudWatch. Exécutez la commande suivante pour voir les groupes de journaux:

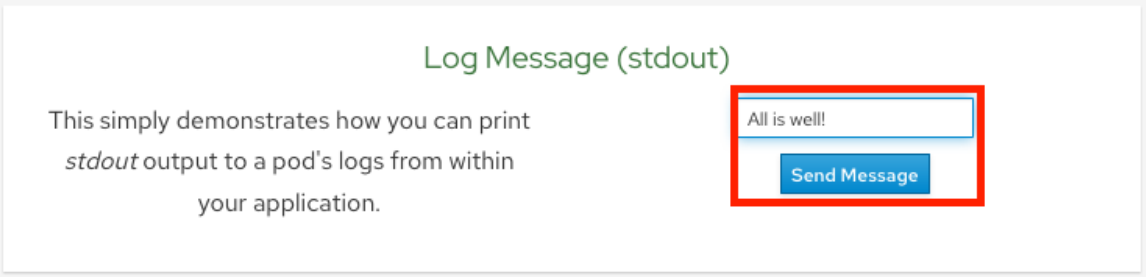
```
$ aws logs describe-log-groups --log-group-name-prefix rosa-mycluster
```

Exemple de sortie

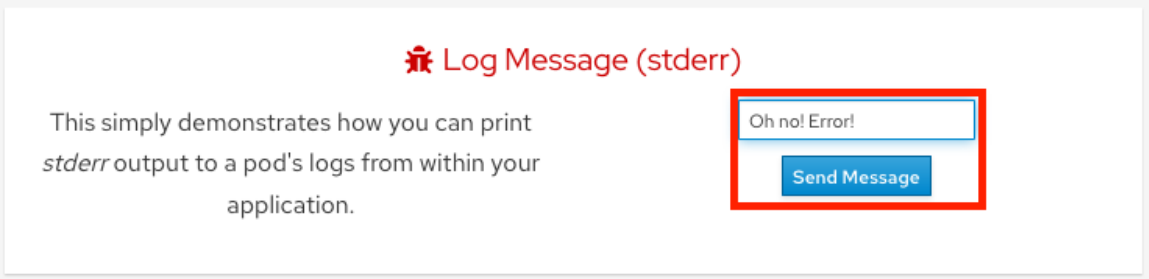
```
{
    "logGroups": [
        {
            "logGroupName": "rosa-mycluster.application",
            "creationTime": 1724104537717,
            "metricFilterCount": 0,
            "arn": "arn:aws:logs:us-west-2:000000000000:log-group:rosa-mycluster.application:*",
            "storedBytes": 0,
            "logGroupClass": "STANDARD",
            "logGroupArn": "arn:aws:logs:us-west-2:000000000000:log-group:rosa-
mycluster.application"
        },
        {
            "logGroupName": "rosa-mycluster.audit",
            "creationTime": 1724104152968,
            "metricFilterCount": 0,
            "arn": "arn:aws:logs:us-west-2:000000000000:log-group:rosa-mycluster.audit:*",
            "storedBytes": 0,
            "logGroupClass": "STANDARD",
            "logGroupArn": "arn:aws:logs:us-west-2:000000000000:log-group:rosa-
mycluster.audit"
        },
```

18.10.2. La sortie des données dans les flux et les journaux

1. Envoyez un message à stdout.
   a. Dans l'application OSToy, cliquez sur Accueil, puis cliquez sur la zone de message pour Log Message (stdout).
   b. Écrivez un message à la sortie sur le flux stdout, par exemple "Tout va bien!".
   c. Cliquez sur Envoyer un message.



2. Envoyez un message à stderr.
   a. Cliquez sur la zone de message pour Log Message (stderr).
   b. Écrivez un message à la sortie sur le flux stderr, par exemple "Oh non! Erreur ! ».
   c. Cliquez sur Envoyer un message.



18.10.3. Affichage des journaux de l'application à l'aide de la commande oc

1. Entrez la commande suivante dans l'interface de ligne de commande (CLI) pour récupérer le nom de votre pod frontend:

```
$ oc get pods -o name
```

Exemple de sortie

```
pod/ostoy-frontend-679cb85695-5cn7x 1
pod/ostoy-microservice-86b4c6f559-p594d
```

1 Le nom du pod est ostoy-frontend-679cb85695-5cn7x.

2. Exécutez la commande suivante pour voir les messages stdout et stderr:

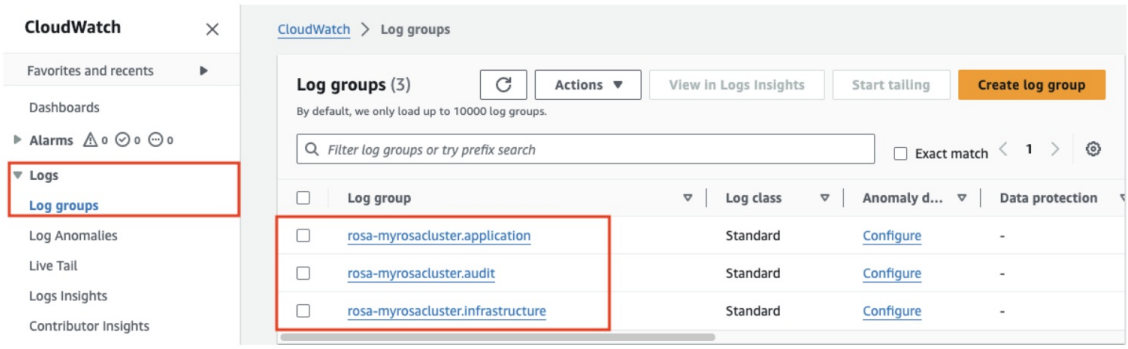
```
$ oc logs <pod-name>
```

Exemple de sortie

```
$ oc logs ostoy-frontend-679cb85695-5cn7x
[...]
ostoy-frontend-679cb85695-5cn7x: server starting on port 8080
Redirecting to /home
stdout: All is well!
stderr: Oh no! Error!
```

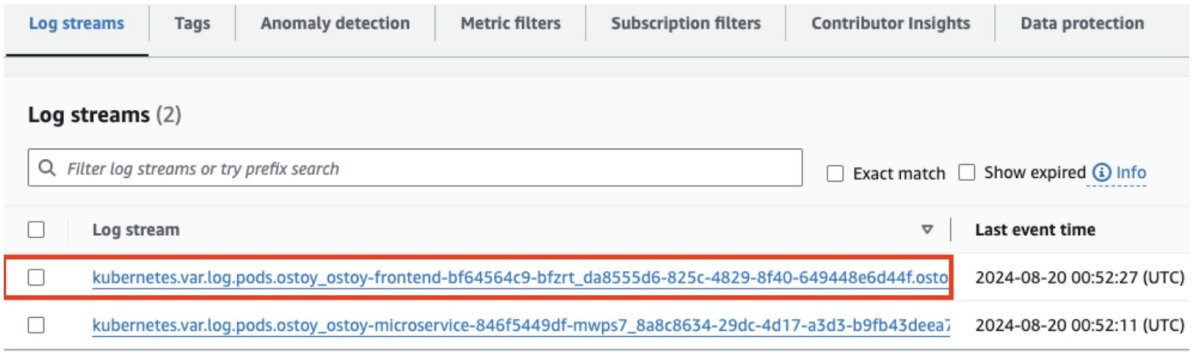
18.10.4. Affichage des journaux avec CloudWatch

1. Accédez à CloudWatch sur la console web AWS.
2. Dans le menu de gauche, cliquez sur Logs, puis Log Groups pour voir les différents groupes de journaux. Il faut voir 3 groupes:
   - **ajouter au panier Rosa-<cluster-name>.application**
   - **> Rosa-<cluster-name>.audit**
   - **caractéristiques de Rosa-<cluster-name>.infrastructure**



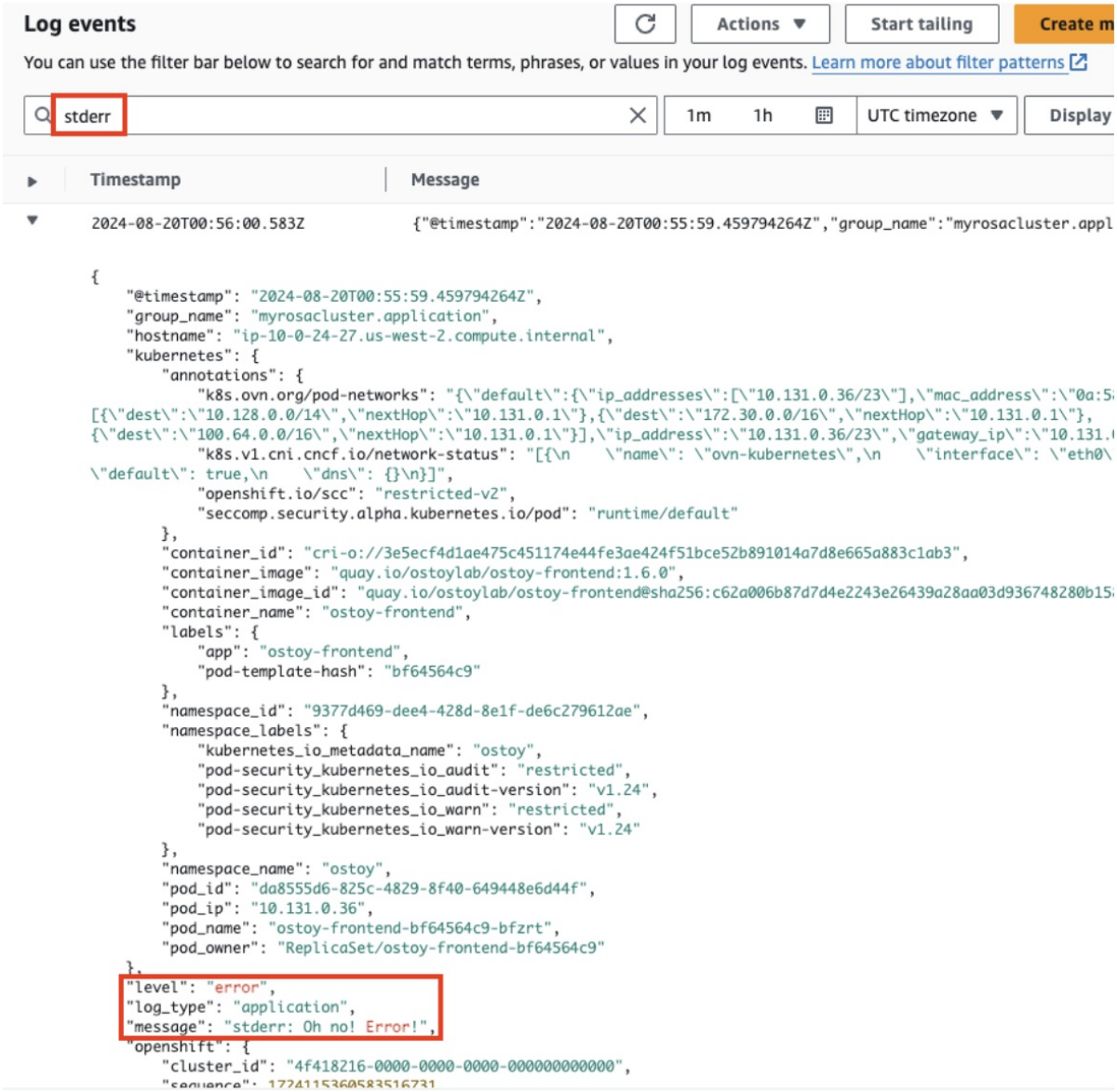
3. Cliquez sur rosa-<cluster-name>.application.

4. Cliquez sur le flux de journal pour le pod frontend.



5. Filtre pour stdout et stderr.

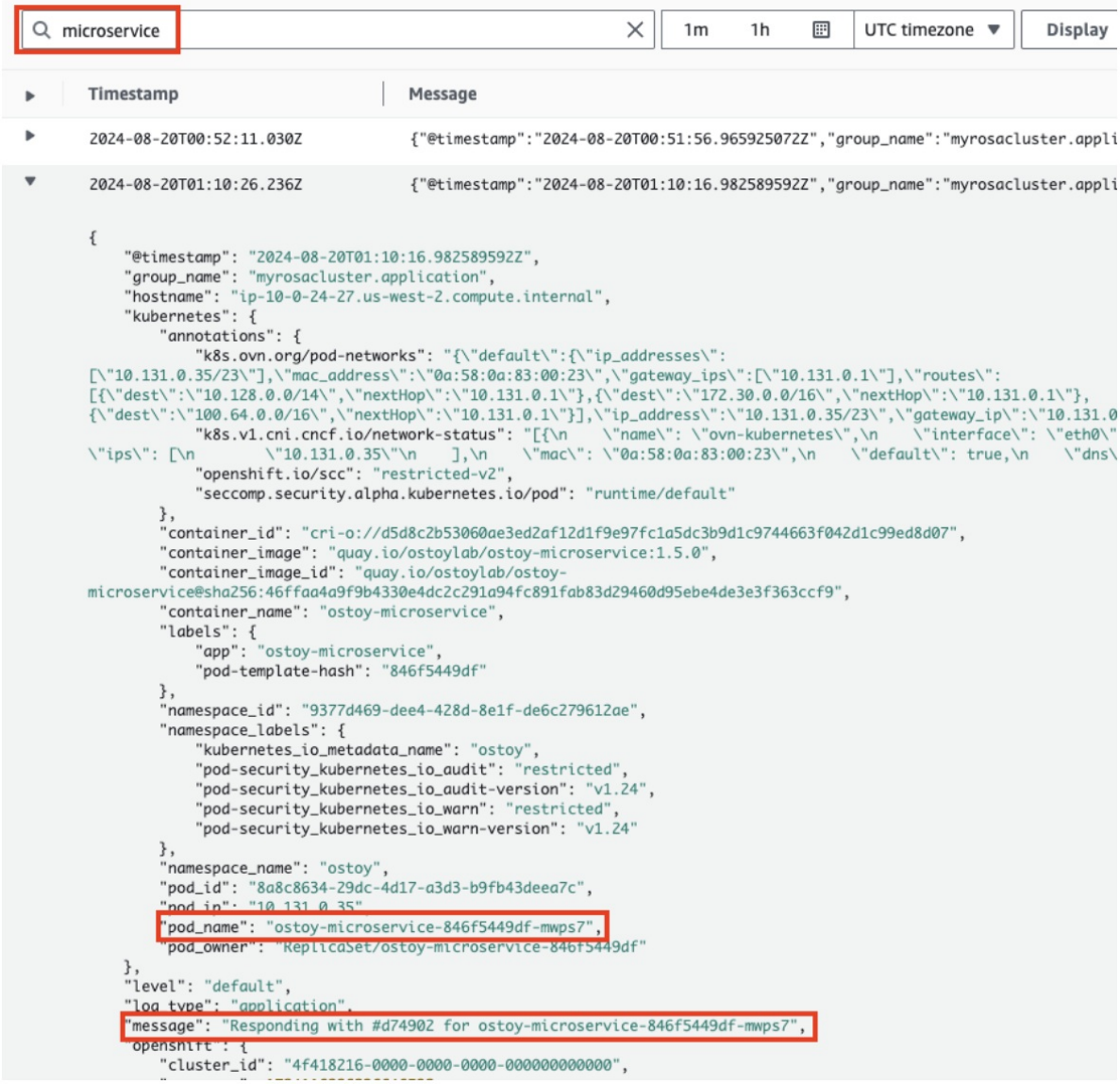
6. Développez la ligne pour afficher les messages que vous avez saisis plus tôt et d'autres informations pertinentes.



7. Entrez dans les flux de journaux et sélectionnez le microservice.

8. Entrez "microservice" dans la barre de recherche pour voir d'autres messages dans vos journaux.

9. Développez l'une des entrées pour voir la couleur de la gousse frontale reçue du microservice et quel pod a envoyé cette couleur à la gousse frontale.



Ressources supplémentaires

- Ce qu'est Amazon CloudWatch

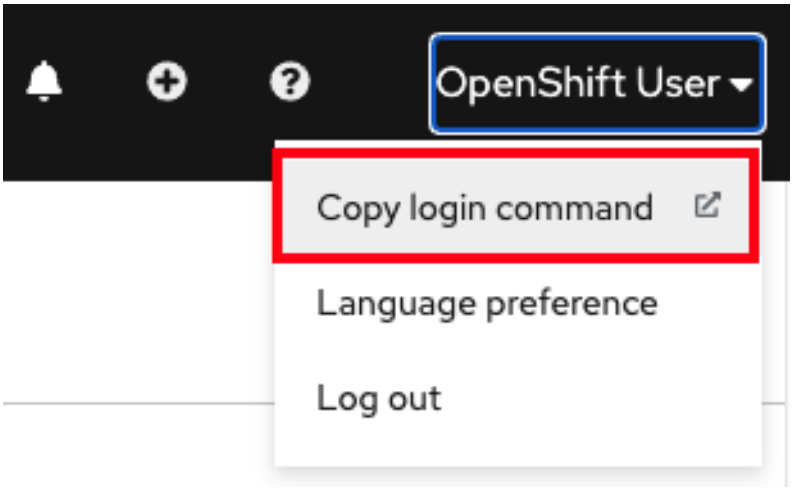
18.11. TUTORIEL: DÉPLOIEMENTS S2I

Il existe plusieurs méthodes pour déployer des applications dans OpenShift. Ce tutoriel explore à l'aide du constructeur intégré Source-à-Image (S2I). Comme mentionné dans la section Concepts OpenShift, S2I est un outil pour construire des images de conteneurs reproductibles au format Docker.

18.11.1. Conditions préalables

Les exigences suivantes doivent être remplies avant de pouvoir utiliser ce tutoriel.

1. « vous avez créé un cluster ROSA.
2. Enregistrez votre commande de connexion
   a. Lorsque vous n'êtes pas connecté via le CLI, dans OpenShift Cluster Manager, cliquez sur la flèche déroulante à côté de votre nom en haut à droite et sélectionnez Copier la commande de connexion.



b. Le nouvel onglet s'ouvre. Entrez votre nom d'utilisateur et votre mot de passe, puis sélectionnez la méthode d'authentification.

c. Cliquez sur Affichage du jeton

d. Copiez la commande sous « Connectez-vous avec ce jeton ».

e. Connectez-vous à l'interface de ligne de commande (CLI) en exécutant la commande copiée dans votre terminal. Il faut voir quelque chose de similaire à ce qui suit:

```
$ oc login --token=RYhFIXXXXXXXXXXX --server=https://api.osd4-
demo.abc1.p1.openshiftapps.com:6443
```

Exemple de sortie

```
Logged into "https://api.myrosacluster.abcd.p1.openshiftapps.com:6443" as "rosa-user"
using the token provided.

You don't have any projects. You can try to create a new project, by running

oc new-project <project name>
```

3. Créez un nouveau projet à partir du CLI en exécutant la commande suivante:

```
$ oc new-project ostoy-s2i
```

18.11.2. Fourche le référentiel OSToy

La section suivante se concentre sur le déclenchement de builds automatisés en fonction des modifications apportées au code source. Il faut configurer un webhook GitHub pour déclencher les builds S2I lorsque vous poussez du code dans votre Repo GitHub. Afin de configurer le webhook, vous devez d'abord fourcher le repo.

NOTE

Dans ce guide, remplacez <UserName> par votre propre nom d'utilisateur GitHub pour les URL suivantes.

18.11.3. En utilisant S2i pour déployer OSToy sur votre cluster

1. Ajouter secret à OpenShift
   L'exemple imite un fichier .env et montre à quel point il est facile de les déplacer directement dans un environnement OpenShift. Les fichiers peuvent même être renommés dans le Secret. Dans votre CLI entrez la commande suivante, remplaçant <UserName> par votre nom d'utilisateur GitHub:

   ```
   $ oc create -f
   https://raw.githubusercontent.com/<UserName>/ostoy/master/deployment/yaml/secret.yaml
   ```

2. Ajouter ConfigMap à OpenShift
   L'exemple imite un fichier de configuration HAProxy, et est généralement utilisé pour remplacer les configurations par défaut dans une application OpenShift. Les fichiers peuvent même être renommés dans le ConfigMap.

   Dans votre CLI entrez la commande suivante, remplaçant <UserName> par votre nom d'utilisateur GitHub:

   ```
   $ oc create -f
   https://raw.githubusercontent.com/<UserName>/ostoy/master/deployment/yaml/configmap.yaml
   ```

3. Déployer le microservice
   Il faut d'abord déployer le microservice pour s'assurer que les variables d'environnement SERVICE sont disponibles à partir de l'application UI. --context-dir est utilisé ici pour construire uniquement l'application définie dans le répertoire microservice dans le référentiel git. En utilisant l'étiquette de l'application nous permet de nous assurer que l'application d'interface utilisateur et le microservice sont tous deux regroupés dans l'interface utilisateur OpenShift. Exécutez la commande suivante dans le CLI pour créer le microservice, en remplaçant <UserName> par votre nom d'utilisateur GitHub:

   ```
   $ oc new-app https://github.com/<UserName>/ostoy \
      --context-dir=microservice \
      --name=ostoy-microservice \
      --labels=app=ostoy
   ```

   Exemple de sortie

   ```
   --> Creating resources with label app=ostoy ...
      imagestream.image.openshift.io "ostoy-microservice" created
      buildconfig.build.openshift.io "ostoy-microservice" created
      deployment.apps "ostoy-microservice" created
      service "ostoy-microservice" created
   --> Success
      Build scheduled, use 'oc logs -f buildconfig/ostoy-microservice' to track its progress.
      Application is not exposed. You can expose services to the outside world by executing one
   or more of the commands below:
       'oc expose service/ostoy-microservice'
      Run 'oc status' to view your app.
   ```

4. Consultez l'état du microservice
   Avant de passer à l'étape suivante, nous devrions être sûrs que le microservice a été créé et fonctionne correctement en exécutant la commande suivante:

   ```
   $ oc status
   ```

Exemple de sortie

```
In project ostoy-s2i on server https://api.myrosacluster.g14t.p1.openshiftapps.com:6443

svc/ostoy-microservice - 172.30.47.74:8080
  dc/ostoy-microservice deploys istag/ostoy-microservice:latest <-
    bc/ostoy-microservice source builds https://github.com/UserName/ostoy on
openshift/nodejs:14-ubi8
    deployment #1 deployed 34 seconds ago - 1 pod
```

Attendez que vous voyez qu'il a été déployé avec succès. Il est également possible de vérifier cela via l'interface utilisateur Web.

5. Déployer l'interface utilisateur frontale
   L'application a été conçue pour s'appuyer sur plusieurs variables d'environnement pour définir des paramètres externes. Attachez le Secret et ConfigMap précédemment créés par la suite, ainsi que la création d'un Volume Persistent. Inscrivez ce qui suit dans le CLI:

```
$ oc new-app https://github.com/<UserName>/ostoy \
    --env=MICROSERVICE_NAME=OSTOY_MICROSERVICE
```

Exemple de sortie

```
--> Creating resources ...
    imagestream.image.openshift.io "ostoy" created
    buildconfig.build.openshift.io "ostoy" created
    deployment.apps "ostoy" created
    service "ostoy" created
--> Success
    Build scheduled, use 'oc logs -f buildconfig/ostoy' to track its progress.
    Application is not exposed. You can expose services to the outside world by executing one
or more of the commands below:
     'oc expose service/ostoy'
    Run 'oc status' to view your app.
```

6. Actualiser le déploiement
   Actualisez le déploiement pour utiliser une stratégie de déploiement « Recréer » (par opposition à la valeur par défaut de RollingUpdate) pour des déploiements cohérents avec des volumes persistants. Le raisonnement ici est que le PV est soutenu par EBS et en tant que tel ne prend en charge que la méthode RWO. « si le déploiement est mis à jour sans que toutes les gousses existantes ne soient tuées, il pourrait ne pas être en mesure de planifier un nouveau pod et de créer un PVC pour le PV car il est toujours lié à la gousse existante. » Lorsque vous utilisez EFS, vous n'avez pas à modifier cela.

```
$ oc patch deployment ostoy --type=json -p \
    '[{"op": "replace", "path": "/spec/strategy/type", "value": "Recreate"}, {"op": "remove", "path":
"/spec/strategy/rollingUpdate"}]'
```

7. Définir une sonde Liveness
   Créez une sonde de vie sur le déploiement pour s'assurer que le pod est redémarré si quelque chose n'est pas sain dans l'application. Inscrivez ce qui suit dans le CLI:

```
$ oc set probe deployment ostoy --liveness --get-url=http://:8080/health
```

8. Joindre Secret, ConfigMap et PersistentVolume au déploiement
   Exécutez les commandes suivantes attachez votre secret, ConfigMap et PersistentVolume:

   a. Attachez le secret

   ```
   $ oc set volume deployment ostoy --add \
      --secret-name=ostoy-secret \
      --mount-path=/var/secret
   ```

   b. Attachez ConfigMap

   ```
   $ oc set volume deployment ostoy --add \
      --configmap-name=ostoy-config \
      -m /var/config
   ```

   c. Créer et joindre PersistentVolume

   ```
   $ oc set volume deployment ostoy --add \
      --type=pvc \
      --claim-size=1G \
      -m /var/demo_files
   ```

9. Exposer l'application UI en tant que route OpenShift
   Exécutez la commande suivante pour déployer ceci en tant qu'application HTTPS qui utilise les certificats de wildcard TLS inclus:

   ```
   $ oc create route edge --service=ostoy --insecure-policy=Redirect
   ```

10. Accédez à votre application avec les méthodes suivantes:

    - L'exécution de la commande suivante ouvre un navigateur Web avec votre application OSToy:

    ```
    $ python -m webbrowser "$(oc get route ostoy -o template --
    template='https://{{.spec.host}}')"
    ```

    - Il est possible d'obtenir l'itinéraire de l'application et de copier et coller l'itinéraire dans votre navigateur en exécutant la commande suivante:

    ```
    $ oc get route
    ```

18.12. DIDACTICIEL: UTILISER DES WEBHOOKS SOURCE-TO-IMAGE (S2I) POUR UN DÉPLOIEMENT AUTOMATISÉ

Déclenchez automatiquement un build et déployez à chaque fois que vous modifiez le code source à l'aide d'un webhook. Consultez Triggering Builds pour plus d'informations sur ce processus.

Procédure

1. Afin d'obtenir le secret de déclenchement GitHub webhook, dans votre terminal, exécutez la commande suivante:

```
$ oc get bc/ostoy-microservice -o=jsonpath='{.spec.triggers..github.secret}'
```

Exemple de sortie

```
`o_3x9M1qoI2Wj_cz1WiK`
```

IMPORTANT

Il faut utiliser ce secret dans une étape ultérieure de ce processus.

2. Afin d'obtenir l'URL de déclenchement GitHub webhook à partir du buildconfig de l'OSToy, exécutez la commande suivante:

```
$ oc describe bc/ostoy-microservice
```

Exemple de sortie

```
[...]
Webhook GitHub:
 URL: https://api.demo1234.openshift.com:443/apis/build.openshift.io/v1/namespaces/ostoy-s2i/buildconfigs/ostoy/webhooks/<secret>/github
[...]
```

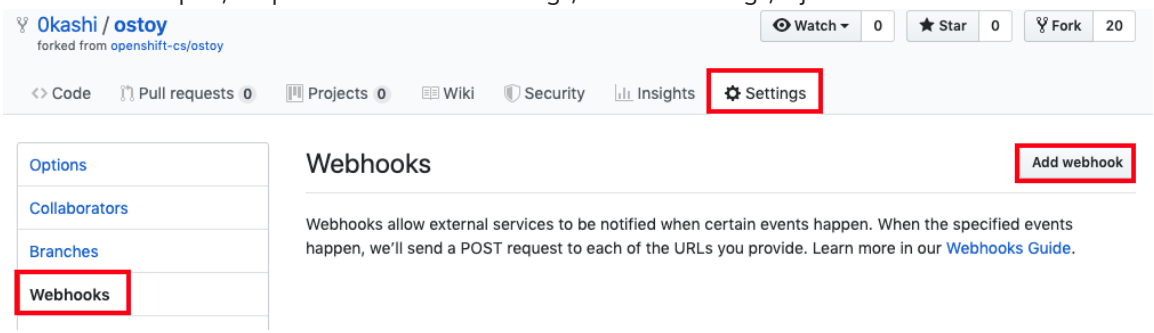
3. Dans l'URL GitHub webhook, remplacez le texte <secret> par le secret que vous avez récupéré. L'URL ressemblera à l'exemple suivant:

Exemple de sortie

```
https://api.demo1234.openshift.com:443/apis/build.openshift.io/v1/namespaces/ostoy-s2i/buildconfigs/ostoy-microservice/webhooks/o_3x9M1qoI2Wj_czR1WiK/github
```

4. Configurez l'URL Webhook dans le référentiel GitHub.

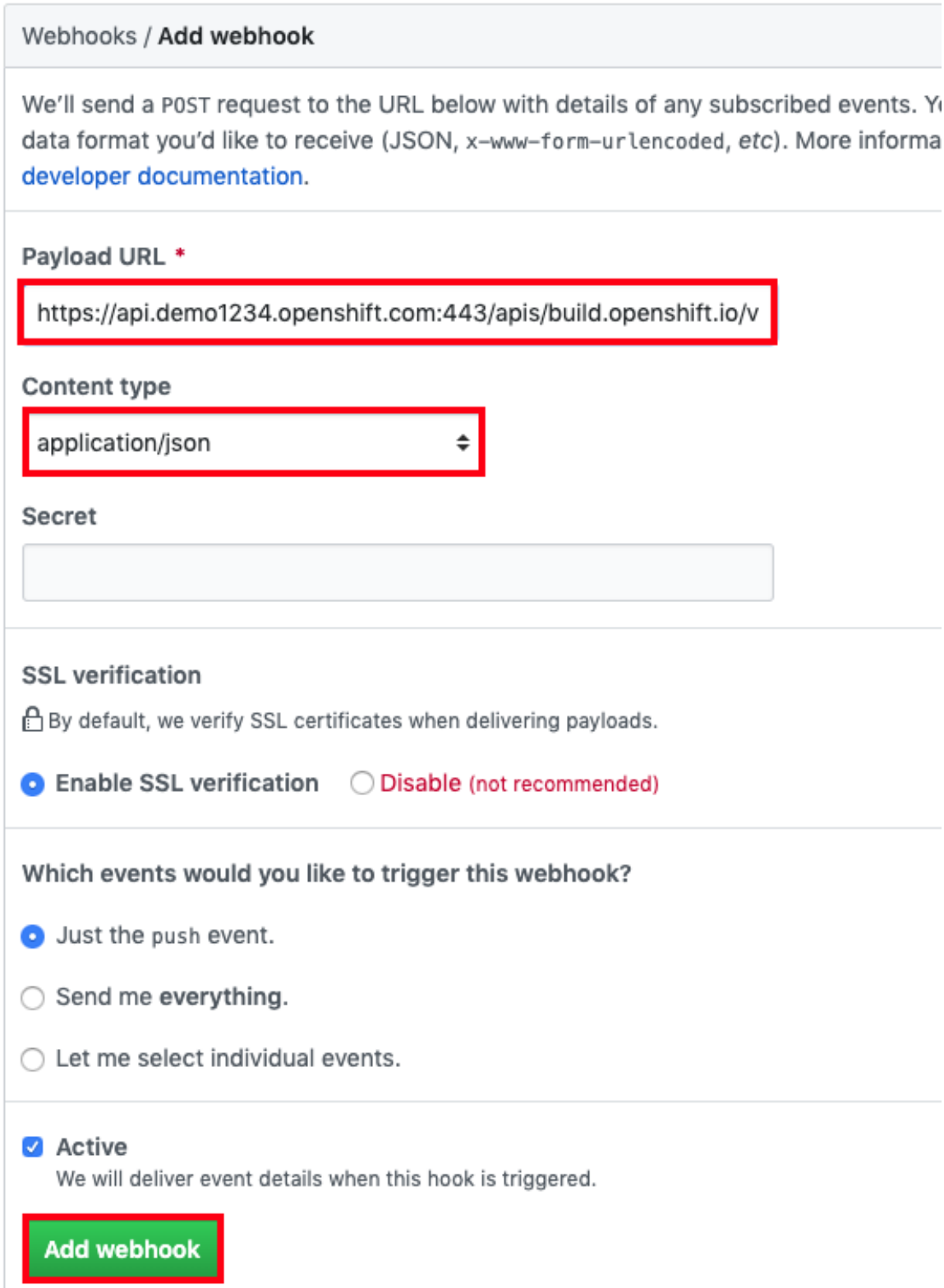
   a. Dans votre dépôt, cliquez sur Paramètres > Webhooks > Ajouter webhook.



   b. Collez l'URL GitHub Webhook avec le secret inclus dans le champ "URL de chargement".

   c. Changer le "type de contenu" en application/json.

   d. Cliquez sur le bouton Ajouter webhook.



Il faut voir un message de GitHub indiquant que votre webhook a été configuré avec succès. Désormais, chaque fois que vous poussez un changement dans votre référentiel GitHub, une nouvelle version démarre automatiquement, et lors d'une construction réussie, un nouveau déploiement commence.

5. Apportez un changement dans le code source. Les modifications déclenchent automatiquement une construction et un déploiement. Dans cet exemple, les couleurs qui indiquent l'état de votre application OSToy sont sélectionnées au hasard. Afin de tester la configuration, changez la boîte pour afficher uniquement l'échelle de gris.

   a. Accédez au code source de votre référentiel https://github.com/<nom d'utilisateur>/ostoy/blob/master/microservice/app.js.

b. Éditez le fichier.

c. Commenter la ligne 8 (contenant laisser aléatoireColor = getRandomColor();).

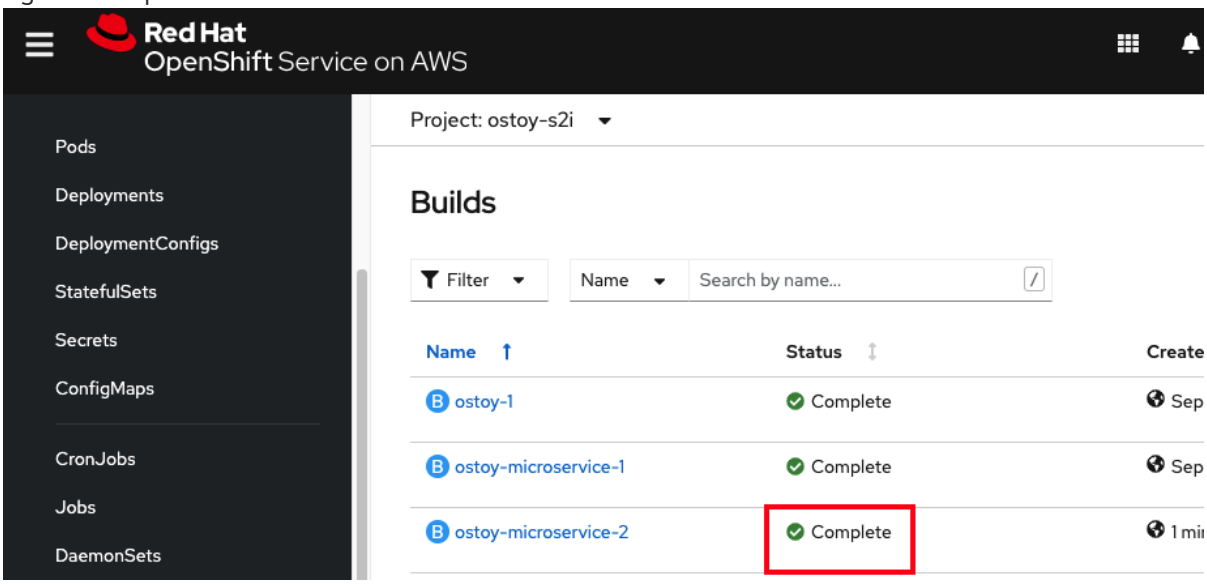
d. Ligne de décommentation 9 (contenant laisser aléatoireColor = getRandomGrayScaleColor();).

```
7  app.get('/', function(request, response) {
8  //let randomColor = getRandomColor(); // <-- comment this
9  let randomColor = getRandomGrayScaleColor(); // <-- uncomment this
10
11  response.writeHead(200, {'Content-Type': 'application/json'});
```

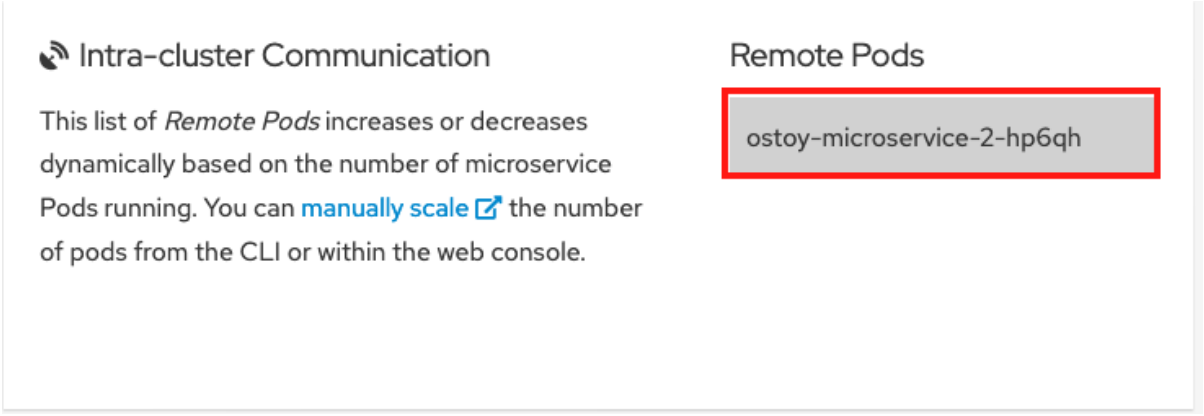
e. Entrez un message pour la mise à jour, tel que "change de boîte en couleurs à échelle de gris".

f. Cliquez sur Commiter en bas pour effectuer les modifications à la branche principale.

6. Dans l'interface utilisateur Web de votre cluster, cliquez sur Builds > Builds pour déterminer l'état de la construction. Après la fin de cette construction, le déploiement commence. Il est également possible de vérifier l'état en exécutant l'état d'oc dans votre terminal.



7. Après la fin du déploiement, retournez à l'application OSToy dans votre navigateur. Accédez à l'élément de menu Networking à gauche. La couleur de la boîte est maintenant limitée aux couleurs de l'échelle de gris seulement.



18.13. TUTORIEL : INTÉGRER AVEC AWS SERVICES

Bien que l'application OSToy fonctionne indépendamment, de nombreuses applications du monde réel nécessitent des services externes tels que des bases de données, des magasins d'objets ou des services de messagerie.

Les objectifs

- Découvrez comment intégrer l'application OSToy à d'autres services Amazon Web Services (AWS), en particulier AWS S3 Storage. À la fin de cette section, l'application créera et lira en toute sécurité des objets à partir d'AWS S3 Storage.
- Faites appel au contrôleur Amazon pour Kubernetes (ACK) pour créer les services nécessaires à notre application directement à partir de Kubernetes.
- Les rôles de gestion de l'identité et de l'accès (IAM) sont utilisés pour les comptes de service pour gérer l'accès et l'authentification.
- Créez un fichier texte de base et enregistrez-le dans un seau S3.
- Confirmez que le fichier a été ajouté avec succès et peut être lu à partir du seau.

18.13.1. Contrôleur Amazon pour Kubernetes (ACK)

L'ACK permet de créer et d'utiliser les services AWS directement à partir de Kubernetes. Il est possible de déployer vos applications directement dans le framework Kubernetes en utilisant une structure familière pour définir et créer des services AWS tels que S3 buckets ou Relational Database Service (RDS).

Avec ACK, vous pouvez créer un seau S3, l'intégrer à l'application OSToy, y télécharger un fichier et afficher le fichier dans votre application.

18.13.2. IAM rôles pour les comptes de service

Il est possible d'utiliser les rôles IAM pour les comptes de service pour attribuer des rôles IAM directement aux comptes de service Kubernetes. Il est possible de l'utiliser pour autoriser le contrôleur ACK à déployer des services sur votre compte AWS. Les rôles IAM pour les comptes de service permettent d'automatiser la gestion et la rotation des informations d'identification temporaires.

Les pods reçoivent un jeton Web JSON (OIDC) valide et le transmettent à l'opération AWS STS AssumeRoleWithWebIdentity API pour recevoir des informations d'identification de rôle temporaires IAM. Le processus repose sur l'identité de la pod EKS mutant le webhook qui modifie les pods nécessitant un accès AWS IAM.

Les rôles de l'IAM pour les comptes de services respectent les meilleures pratiques suivantes:

- Le principe du moindre privilège : Vous pouvez créer des autorisations IAM pour les rôles AWS qui n'autorisent qu'un accès limité. Ces autorisations sont limitées au compte de service associé au rôle et seuls les pods qui utilisent ce compte de service ont accès.
- Isolation d'identification: Un pod ne peut récupérer que des informations d'identification pour le rôle IAM associé au compte de service que le pod utilise.
- Audit : Tous les accès aux ressources AWS peuvent être consultés dans CloudTrail.

Ressources supplémentaires

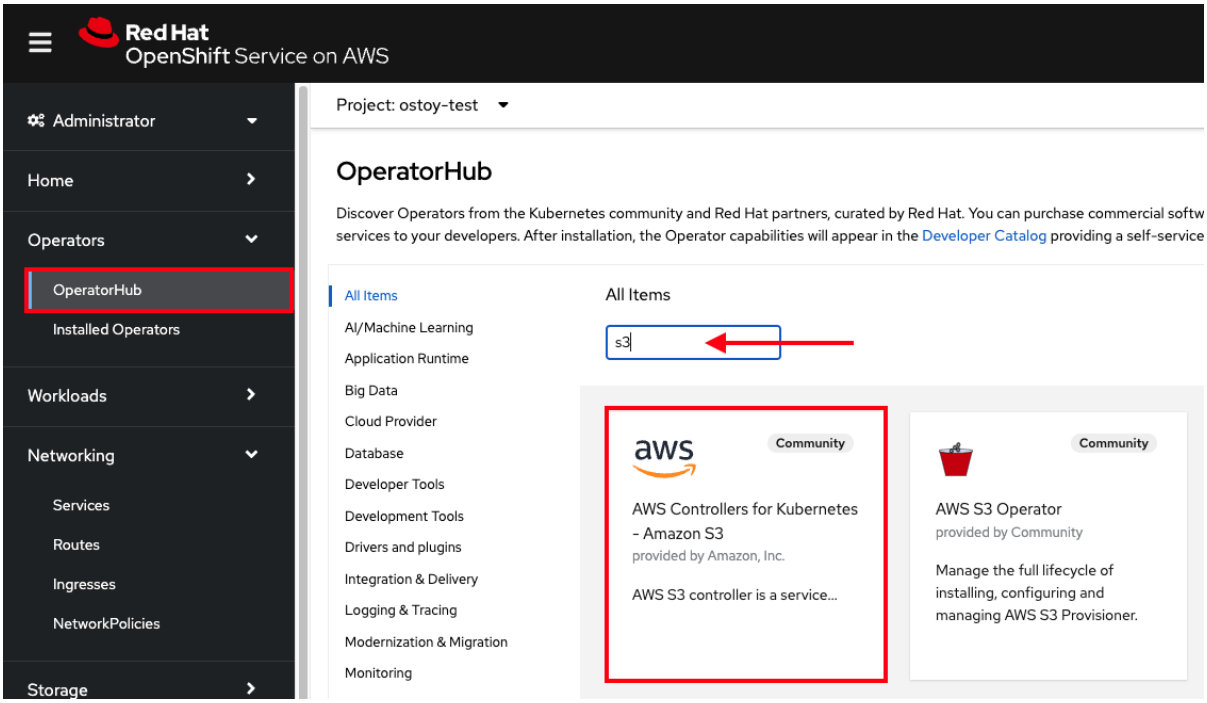
- La page de documentation ACK

18.13.3. Installation du contrôleur ACK

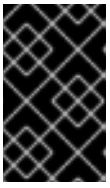
Installez le contrôleur ACK pour créer et supprimer des seaux dans le service S3 en utilisant une ressource personnalisée Kubernetes pour le seau. L'installation du contrôleur créera également l'espace de noms et le compte de service requis.

« nous utiliserons un opérateur pour faciliter la tâche. L'installation de l'opérateur créera également un espace de noms ack-system et un contrôleur de compte de service pour vous.

1. Connectez-vous à la console de cluster.
2. Dans le menu de gauche, cliquez sur Opérateurs, puis OperatorHub.
3. Dans la zone de filtre, entrez "S3" et sélectionnez AWS Controller pour Kubernetes - Amazon S3.



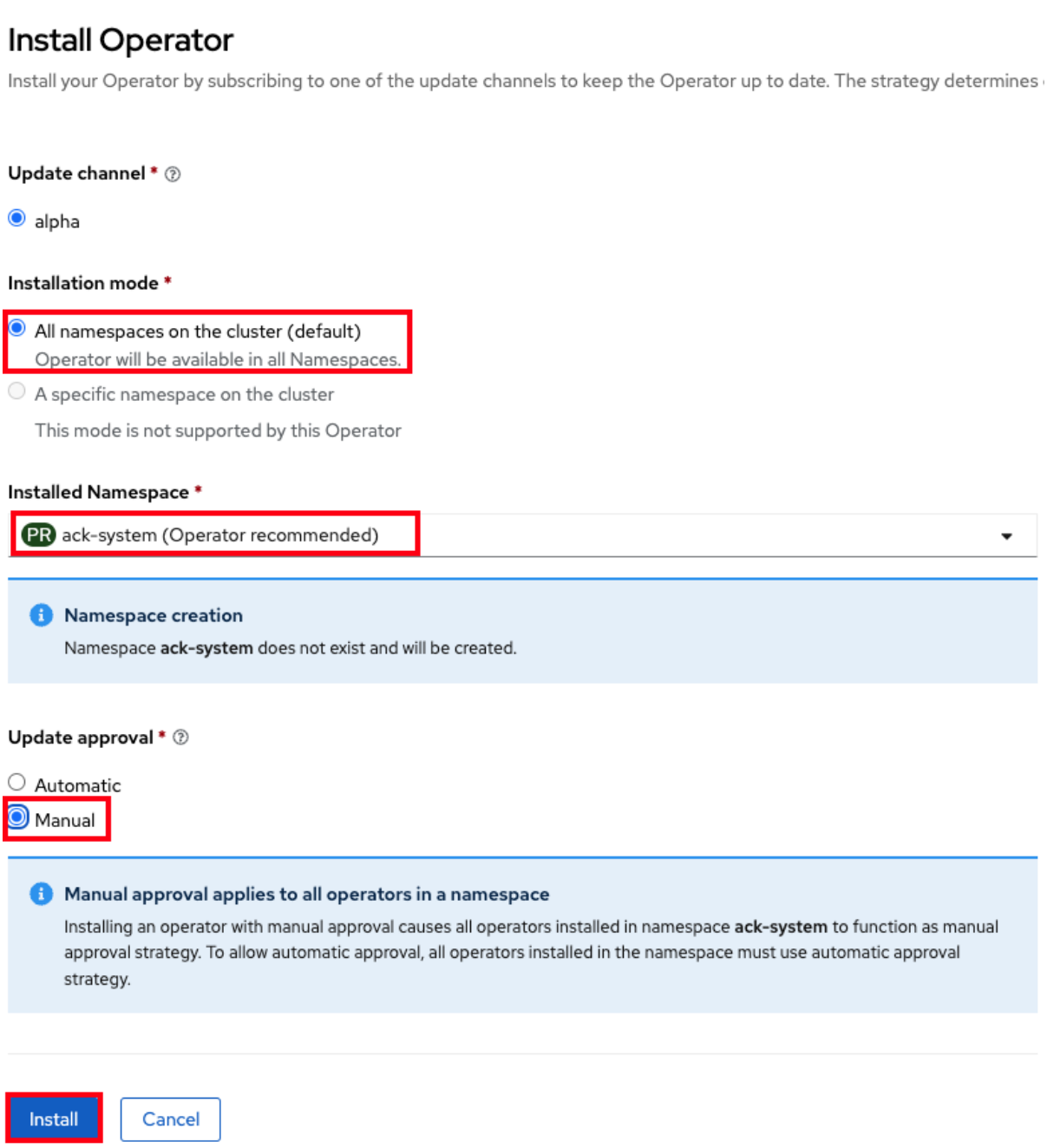
4. En cas d'apparition d'une pop-up sur les opérateurs communautaires, cliquez sur Continuer.
5. Cliquez sur **Install**.
6. Choisissez Tous les espaces de noms du cluster sous « Mode d'installation ».
7. Choisissez ack-system sous "Installed Namespace".
8. Choisissez le manuel sous la rubrique « Mise à jour de l'approbation ».



IMPORTANT

Assurez-vous que le mode manuel est sélectionné afin que les modifications apportées au compte de service ne soient pas écrasées par une mise à jour automatique de l'opérateur.

9. Cliquez sur **Install**.
   Les paramètres devraient ressembler à l'image ci-dessous.



10. Cliquez sur Approuver.

11. L'installation commence mais ne sera pas terminée tant que vous n'avez pas créé un rôle et une politique IAM pour le contrôleur ACK.

18.13.4. Créer un rôle et une politique IAM pour le contrôleur ACK

1. Exécutez l'un des scripts suivants pour créer le rôle AWS IAM pour le contrôleur ACK et assigner la stratégie S3:

   - Automatiquement télécharger le script setup-s3-ack-controller.sh, qui automatise le processus pour vous.

   - Exécutez le script suivant dans votre interface de ligne de commande (CLI):

     ```
     $ curl https://raw.githubusercontent.com/openshift-cs/rosaworkshop/master/rosa-workshop/ostoy/resources/setup-s3-ack-controller.sh | bash
     ```

2. Lorsque le script est terminé, il redémarre le déploiement qui met à jour les pods du contrôleur de service avec les rôles IAM pour les variables d'environnement des comptes de service.

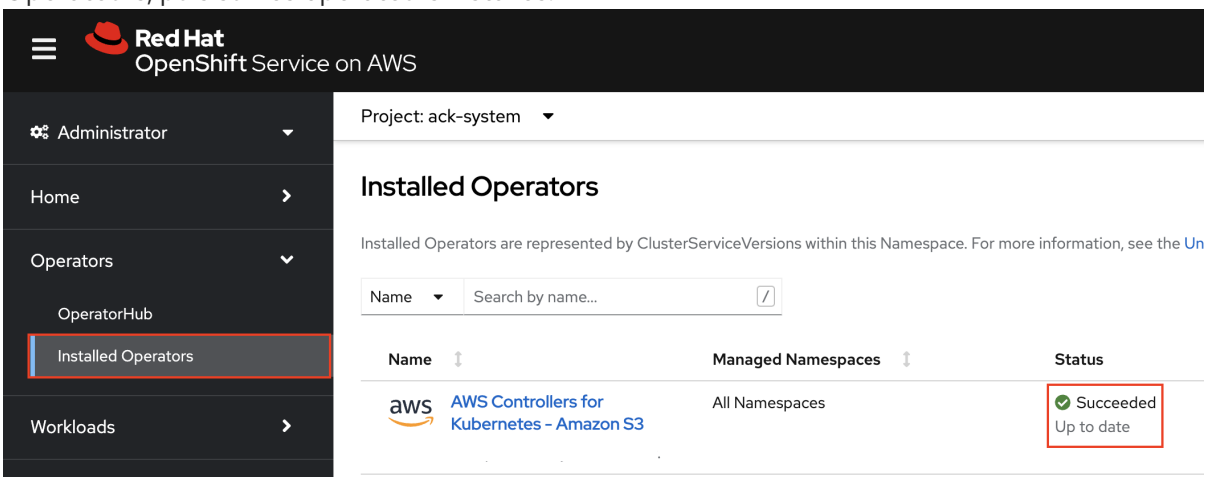
3. Confirmez que les variables d'environnement sont définies en exécutant la commande suivante:

```
$ oc describe pod ack-s3-controller -n ack-system | grep "^\s*AWS_"
```

Exemple de sortie

```
AWS_ROLE_ARN:                 arn:aws:iam::000000000000:role/ack-s3-controller
AWS_WEB_IDENTITY_TOKEN_FILE:
/var/run/secrets/eks.amazonaws.com/serviceaccount/token
```

4. Confirmez la configuration réussie du contrôleur ACK dans la console Web en cliquant sur Opérateurs, puis sur les opérateurs installés.



5. Lorsque vous ne voyez pas une installation réussie de l'opérateur et les variables d'environnement, redémarrez manuellement le déploiement en exécutant la commande suivante:

```
$ oc rollout restart deployment ack-s3-controller -n ack-system
```

18.13.5. Configuration de l'accès pour l'application

Il est possible de créer un compte de rôle et de service AWS IAM afin que OSToy puisse lire et écrire des objets dans un seau S3.

1. Créez un nouveau projet unique pour OSToy en exécutant la commande suivante:

```
$ oc new-project ostoy-$(uuidgen | cut -d - -f 2 | tr '[:upper:]' '[:lower:]')
```

2. Enregistrez le nom de l'espace de noms et du projet dans une variable d'environnement en exécutant la commande suivante:

```
$ export OSTOY_NAMESPACE=$(oc config view --minify -o 'jsonpath={..namespace}')
```

18.13.6. Créer un rôle AWS IAM

1. Bénéficiez de l'ID de votre compte AWS en exécutant la commande suivante:

```
$ export AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query Account --output text)
```

2. Obtenez le fournisseur OIDC en exécutant la commande suivante, en remplaçant <cluster-name> par le nom de votre cluster:

```
$ export OIDC_PROVIDER=$(rosa describe cluster -c <cluster-name> -o yaml | awk
'/oidc_endpoint_url/ {print $2}' | cut -d '/' -f 3,4)
```

3. Créez le fichier de stratégie de confiance en exécutant la commande suivante:

```
$ cat <<EOF > ./ostoy-sa-trust.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::${AWS_ACCOUNT_ID}:oidc-provider/${OIDC_PROVIDER}"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "${OIDC_PROVIDER}:sub": "system:serviceaccount:${OSTOY_NAMESPACE}:ostoy-
sa"
        }
      }
    }
  ]
}
EOF
```

4. Créez le rôle AWS IAM à utiliser avec votre compte de service en exécutant la commande suivante:

```
$ aws iam create-role --role-name "ostoy-sa-role" --assume-role-policy-document file://ostoy-
sa-trust.json
```

18.13.7. Attacher la politique S3 au rôle de l'IAM

1. Bénéficiez de la stratégie d'accès complet S3 ARN en exécutant la commande suivante:

```
$ export POLICY_ARN=$(aws iam list-policies --query 'Policies[?
PolicyName==`AmazonS3FullAccess`].Arn' --output text)
```

2. Attachez la stratégie au rôle AWS IAM en exécutant la commande suivante:

```
$ aws iam attach-role-policy --role-name "ostoy-sa-role" --policy-arn "${POLICY_ARN}"
```

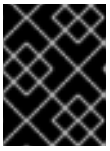
18.13.8. Créer le compte de service pour votre pod

1. Bénéficiez de l'ARN pour le rôle AWS IAM que nous avons créé afin qu'il soit inclus comme annotation lorsque vous créez votre compte de service en exécutant la commande suivante:

```
$ export APP_IAM_ROLE_ARN=$(aws iam get-role --role-name=ostoy-sa-role --query
Role.Arn --output text)
```

2. Créez le compte de service en exécutant la commande suivante:

```
$ cat <<EOF | oc apply -f -
apiVersion: v1
kind: ServiceAccount
metadata:
  name: ostoy-sa
  namespace: ${OSTOY_NAMESPACE}
  annotations:
    eks.amazonaws.com/role-arn: "$APP_IAM_ROLE_ARN"
EOF
```



IMPORTANT

Il ne faut pas changer le nom du compte de service de "ostoy-sa" ou vous devrez modifier la relation de confiance pour le rôle AWS IAM.

3. Accordez au compte de service le rôle restreint en exécutant la commande suivante:

```
$ oc adm policy add-scc-to-user restricted
system:serviceaccount:${OSTOY_NAMESPACE}:ostoy-sa
```

4. Confirmez que l'annotation a été couronnée de succès en exécutant la commande suivante:

```
$ oc describe serviceaccount ostoy-sa -n ${OSTOY_NAMESPACE}
```

Exemple de sortie

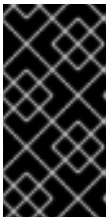
```
Name:                ostoy-sa
Namespace:           ostoy
Labels:              <none>
Annotations:         eks.amazonaws.com/role-arn: arn:aws:iam::000000000000:role/ostoy-sa-
role
Image pull secrets:  ostoy-sa-dockercfg-b2l94
Mountable secrets:   ostoy-sa-dockercfg-b2l94
Tokens:              ostoy-sa-token-jlc6d
Events:              <none>
```

18.13.9. Créer un seau S3

1. Créez un seau S3 à l'aide d'un fichier manifeste en exécutant la commande suivante:

```
$ cat <<EOF | oc apply -f -
apiVersion: s3.services.k8s.aws/v1alpha1
kind: Bucket
metadata:
  name: ${OSTOY_NAMESPACE}-bucket
  namespace: ${OSTOY_NAMESPACE}
```

```
spec:
  name: ${OSTOY_NAMESPACE}-bucket
EOF
```



IMPORTANT

L'application OSToy s'attend à trouver un seau nommé <namespace>-bucket. Lorsque vous utilisez autre chose que l'espace de noms de votre projet OSToy, cette fonctionnalité ne fonctionnera pas. Ainsi, si notre projet est "ostoy", la valeur du nom doit être ostoy-bucket.

2. Confirmez que le seau a été créé en exécutant la commande suivante:

```
$ aws s3 ls | grep ${OSTOY_NAMESPACE}-bucket
```

18.13.10. Le redéploiement de l'application OSToy avec le nouveau compte de service

1. Exécutez votre pod avec le compte de service que vous avez créé.
2. Déployez le microservice en exécutant la commande suivante:

```
$ - oc apply -f https://raw.githubusercontent.com/openshift-cs/rosaworkshop/master/rosa-workshop/ostoy/yaml/ostoy-microservice-deployment.yaml
```

3. Déployez l'ostoy-frontend en exécutant la commande suivante:

```
$ - oc apply -f https://raw.githubusercontent.com/openshift-cs/rosaworkshop/master/rosa-workshop/ostoy/yaml/ostoy-frontend-deployment.yaml
```

4. Corrigez le déploiement ostoy-frontend en exécutant la commande suivante:

```
$ oc patch deploy ostoy-frontend -n ${OSTOY_NAMESPACE} --type=merge --patch '{"spec": {"template": {"spec":{"serviceAccount":"ostoy-sa"}}}}'
```

Exemple de sortie

```
spec:
  # Uncomment to use with ACK portion of the workshop
  # If you chose a different service account name please replace it.
  serviceAccount: ostoy-sa
  containers:
  - name: ostoy-frontend
    image: quay.io/ostoylab/ostoy-frontend:1.6.0
    imagePullPolicy: IfNotPresent
[...]
```

5. Attendez que le pod soit mis à jour.

18.13.11. Confirmation des variables d'environnement

- La commande suivante permet de décrire les pods et de vérifier l'existence de variables d'environnement AWS_WEB_IDENTITY_TOKEN_FILE et AWS_ROLE_ARN pour notre application:

```
$ oc describe pod ostoy-frontend -n ${OSTOY_NAMESPACE} | grep "^\s*AWS_"
```

Exemple de sortie

```
AWS_ROLE_ARN:                 arn:aws:iam::000000000000:role/ostoy-sa
AWS_WEB_IDENTITY_TOKEN_FILE:
/var/run/secrets/eks.amazonaws.com/serviceaccount/token
```

18.13.12. Affichage du contenu du seau à travers OSToy

Consultez votre application pour afficher le contenu de votre seau S3.

1. Accédez à l'itinéraire de l'application nouvellement déployée en exécutant la commande suivante:

```
$ oc get route ostoy-route -n ${OSTOY_NAMESPACE} -o jsonpath='{.spec.host}{"\n"}'
```

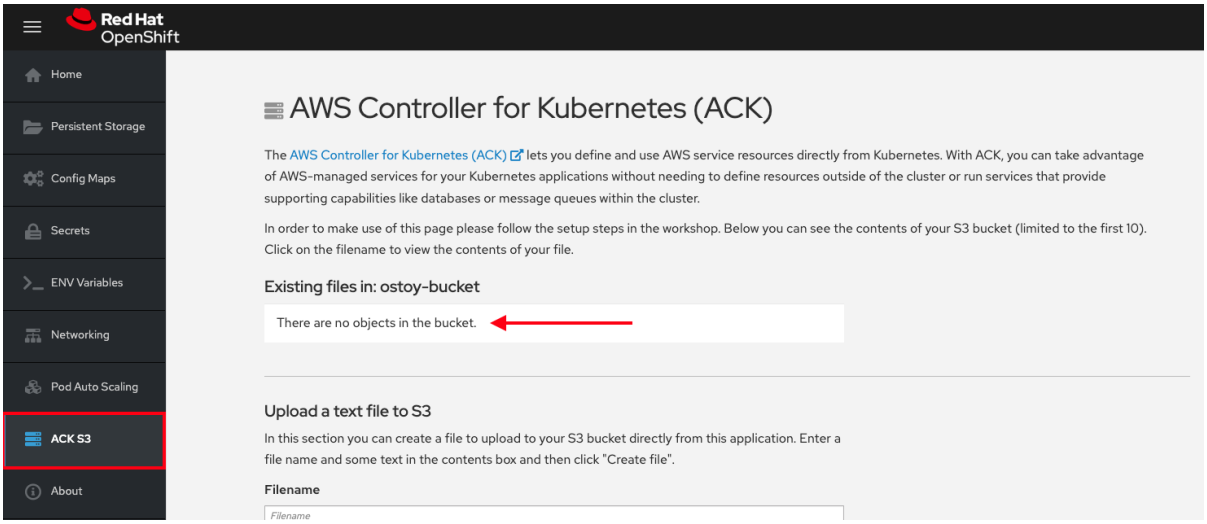
2. Ouvrez un nouvel onglet du navigateur et entrez l'itinéraire obtenu à l'étape précédente.

IMPORTANT

Assurez-vous d'utiliser http:// et non https://.

3. Cliquez sur ACK S3 dans le menu de gauche dans OSToy.

4. Comme il s'agit d'un nouveau seau, le seau doit être vide.

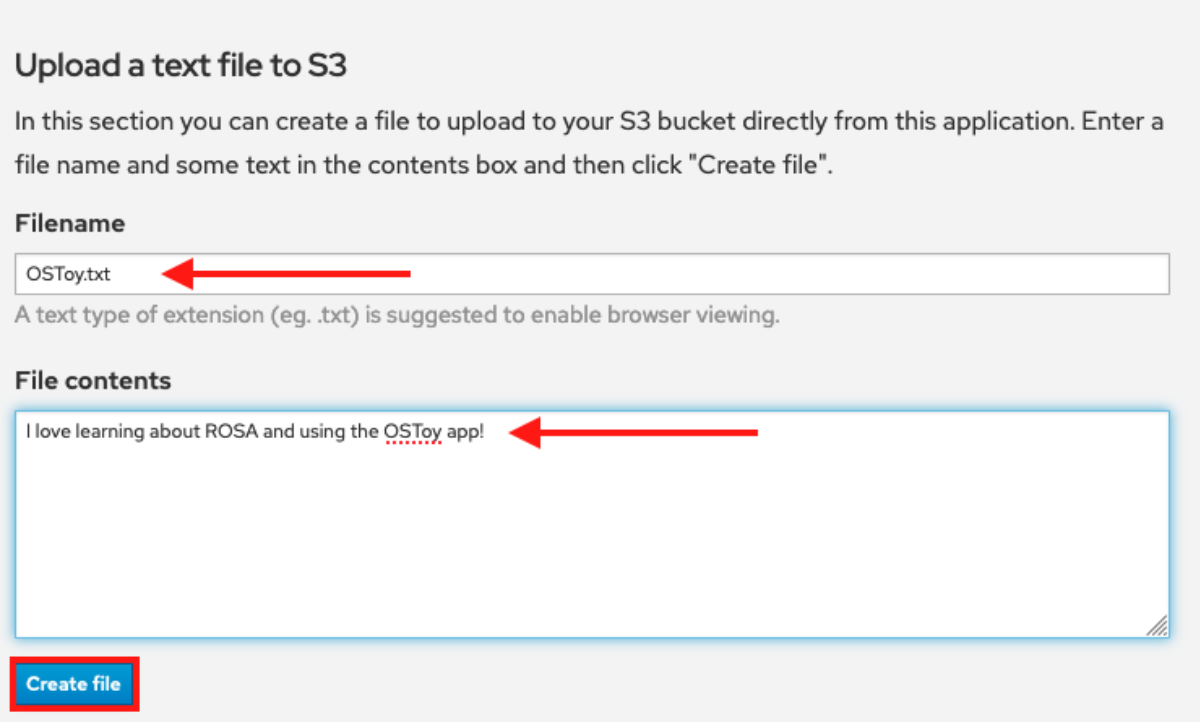


18.13.13. Créer des fichiers dans votre seau S3

Faites appel à OStoy pour créer un fichier et le télécharger sur le seau S3. Bien que S3 puisse accepter n'importe quel type de fichier, pour ce tutoriel, utilisez des fichiers texte afin que le contenu puisse facilement être rendu dans le navigateur.

1. Cliquez sur ACK S3 dans le menu de gauche dans OSToy.

2. Faites défiler vers le bas pour télécharger un fichier texte vers S3.
3. Entrez un nom de fichier pour votre fichier.
4. Entrez le contenu de votre fichier.
5. Cliquez sur Créer un fichier.



6. Faites défiler la section supérieure pour les fichiers existants et confirmez que le fichier que vous venez de créer est là.
7. Cliquez sur le nom du fichier pour afficher le fichier.

8. Confirmez avec AWS CLI en exécutant la commande suivante pour répertorier le contenu de votre seau:

```
$ aws s3 ls s3://${OSTOY_NAMESPACE}-bucket
```

Exemple de sortie

```
$ aws s3 ls s3://ostoy-bucket
2023-05-04 22:20:51         51 OSToy.txt
```