Configurer la colocation sur Red Hat OpenShift avec NetApp ONTAP

NetApp Solutions

NetApp
September 25, 2024

Sommaire

Configurer la colocation sur Red Hat OpenShift avec NetApp ONTAP

Configuration d'une colocation sur Red Hat OpenShift avec NetApp

De nombreuses entreprises qui exécutent plusieurs applications ou charges de travail sur des conteneurs ont tendance à déployer un cluster Red Hat OpenShift par application ou par workload. Ils peuvent ainsi mettre en œuvre une isolation stricte pour l'application ou la charge de travail, optimiser les performances et réduire les vulnérabilités de sécurité. Toutefois, le déploiement d'un cluster Red Hat OpenShift distinct pour chaque application présente ses propres problèmes. Cette solution augmente les frais d'exploitation liés à la surveillance et à la gestion seule de chaque cluster, ce qui augmente les coûts du fait de ressources dédiées pour différentes applications et entrave l'évolutivité efficace.

Pour résoudre ces problèmes, il est possible d'exécuter toutes les applications ou charges de travail dans un seul cluster Red Hat OpenShift. Cependant, dans une telle architecture, l'isolement des ressources et les vulnérabilités liées à la sécurité des applications constituent l'un des défis majeurs. Toute vulnérabilité de sécurité dans une charge de travail pourrait naturellement se répandre sur une autre charge de travail, augmentant ainsi la zone d'impact. En outre, une application peut avoir une incidence soudaine et non contrôlée sur les performances d'une autre application, car il n'existe pas de stratégie d'allocation des ressources par défaut.

Les entreprises recherchent donc des solutions qui offrent les meilleures des deux mondes, par exemple, en leur permettant d'exécuter toutes leurs charges de travail dans un cluster unique, tout en offrant les avantages d'un cluster dédié pour chaque charge de travail.

L'une de ces solutions est utile : configurer la colocation sur Red Hat OpenShift. La colocation est une architecture qui permet à plusieurs locataires de coexister sur un même cluster avec une isolation appropriée des ressources, de la sécurité, etc. Dans ce contexte, un locataire peut être considéré comme un sous-ensemble des ressources du cluster qui sont configurées pour être utilisées par un groupe d'utilisateurs particulier à des fins exclusives. La configuration d'une colocation sur un cluster Red Hat OpenShift offre les avantages suivants :

- Réduction des dépenses d'investissement et d'exploitation en permettant le partage des ressources du cluster
- Réduisez les frais d'exploitation et de gestion
- Sécurisation des charges de travail contre toute contamination croisée des failles de sécurité
- Protection des charges de travail contre la dégradation inattendue des performances en raison des conflits des ressources

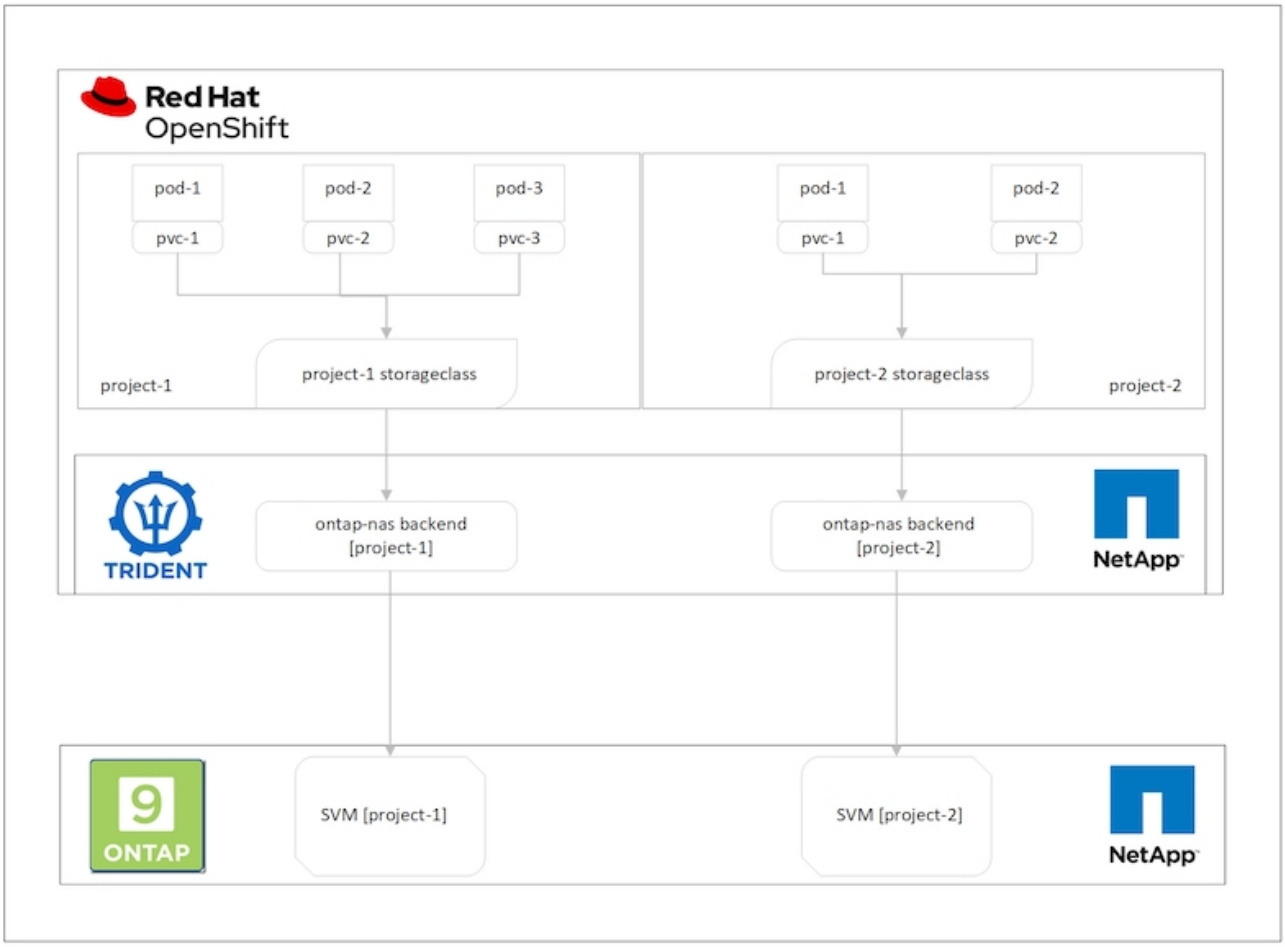
Pour un cluster OpenShift mutualisé entièrement réalisé, les quotas et les restrictions doivent être configurés pour les ressources de cluster appartenant à différents compartiments de ressources : calcul, stockage, réseau, sécurité, etc. Bien que nous aborderons certains aspects de toutes les ressources de cette solution, Nous mettons l'accent sur les bonnes pratiques d'isolation et de sécurisation des données servies ou consommées par plusieurs charges de travail sur le même cluster Red Hat OpenShift en configurant la colocation sur des ressources de stockage allouées de façon dynamique par Astra Trident et sauvegardé par NetApp ONTAP.

Architecture

Bien que Red Hat OpenShift et Astra Trident avec NetApp ONTAP ne assurent pas l'isolation des charges de travail par défaut, ils offrent un large éventail de fonctionnalités qui peuvent être utilisées pour configurer la colocation. Pour mieux comprendre comment concevoir une solution mutualisée sur un cluster Red Hat OpenShift avec Astra Trident basée sur NetApp ONTAP, nous examinons un exemple d'exigences et nous présente la configuration qui l'entoure.

Supposons qu'une entreprise exécute deux de ses charges de travail sur un cluster Red Hat OpenShift dans le cadre de deux projets sur lesquels deux équipes différentes travaillent. Les données de ces workloads résident sur des demandes de volume persistant qui sont provisionnées dynamiquement par Astra Trident sur un back-end NAS NetApp ONTAP. L'entreprise doit concevoir une solution mutualisée pour ces deux charges de travail et isoler les ressources utilisées pour ces projets afin de garantir la sécurité et la performance nécessaires. Elle est axée sur les données qui servent ces applications.

La figure suivante décrit la solution mutualisée sur un cluster Red Hat OpenShift avec Astra Trident et NetApp ONTAP.



Exigences technologiques

1. Cluster de stockage NetApp ONTAP

2. Cluster Red Hat OpenShift
3. Astra Trident

Ressources Red Hat OpenShift – Cluster

Du point de vue du cluster Red Hat OpenShift, la ressource de premier niveau à commencer est le projet. Un projet OpenShift peut être considéré comme une ressource de cluster qui divise l'ensemble du cluster OpenShift en plusieurs clusters virtuels. Ainsi, l'isolation au niveau du projet fournit une base pour la configuration de la colocation.

Ensuite, vous devez configurer RBAC dans le cluster. La meilleure pratique consiste à configurer tous les développeurs sur un seul projet ou charge de travail dans un seul groupe d'utilisateurs du fournisseur d'identités. Red Hat OpenShift permet l'intégration IDP et la synchronisation des groupes d'utilisateurs, ce qui permet d'importer les utilisateurs et les groupes du PDI dans le cluster. Les administrateurs du cluster peuvent ainsi isoler l'accès aux ressources du cluster dédiées à un projet à un ou plusieurs groupes d'utilisateurs travaillant sur ce projet, ce qui limite l'accès non autorisé aux ressources du cluster. Pour en savoir plus sur l'intégration IDP avec Red Hat OpenShift, consultez la documentation "ici".

NetApp ONTAP

Il est important d'isoler le service de stockage partagé en tant que fournisseur de stockage persistant pour un cluster Red Hat OpenShift afin de vérifier que les volumes créés sur le stockage pour chaque projet apparaissent aux hôtes comme s'ils sont créés sur un stockage distinct. Pour ce faire, créez autant de SVM (Storage Virtual machines) sur NetApp ONTAP que des projets ou des charges de travail et dédier chaque SVM à une charge de travail.

Astra Trident

Une fois que vous avez des SVM différents pour les projets créés sur NetApp ONTAP, vous devez mapper chaque SVM sur un back-end Trident différent. La configuration back-end de Trident entraîne l'allocation du stockage persistant aux ressources de cluster OpenShift, et elle requiert le mappage des détails de la SVM sur. Il doit s'agir du pilote de protocole pour le back-end au minimum. Vous pouvez également définir la manière dont les volumes sont provisionnés sur le stockage et définir des limites pour la taille des volumes ou l'utilisation des agrégats, etc. Vous trouverez des informations détaillées sur la définition des systèmes back-end Trident "ici".

Red Hat OpenShift – ressources de stockage

Une fois les systèmes back-end Trident configurés, l'étape suivante consiste à configurer les classes de stockage. Configurez autant de classes de stockage que les systèmes back-end, en donnant à chaque classe de stockage l'accès pour lancer des volumes sur un seul système back-end. Nous pouvons mapper la classe de stockage sur un back-end Trident en utilisant le paramètre storagePools lors de la définition de la classe de stockage. Les détails de la définition d'une classe de stockage sont disponibles "ici". Il existe donc un mappage un-à-un de StorageClass vers le backend Trident qui pointe vers un SVM. Ainsi, toutes les demandes de stockage traitées par la classe de stockage allouée à ce projet sont servies par la SVM dédiée à ce projet uniquement.

Comme les classes de stockage ne namesles ressources qui ne sont pas adaptées, comment pouvons-nous nous assurer que les déclarations de stockage présentées dans la classe d'un projet par des pods dans un autre espace de noms ou dans des projets sont rejetées ? La réponse est d'utiliser ResourceQuotas. ResourceQuotas sont des objets qui contrôlent l'utilisation totale des ressources par projet. Elle peut limiter le nombre ainsi que la quantité totale de ressources pouvant être consommées par des objets dans le projet. Presque toutes les ressources d'un projet peuvent être limitées à l'aide de ResourceQuotas et l'utilisation

efficace de cette solution peut aider les entreprises à réduire les coûts et les pannes dus au sur-provisionnement ou à la sur-consommation des ressources. Reportez-vous à la documentation "ici" pour en savoir plus.

Pour ce cas d'utilisation, nous devons limiter les demandes de stockage provenant de classes de stockage qui ne sont pas dédiées à leur projet dans un projet particulier. Il nous faut donc limiter les demandes de volume persistant pour d'autres classes de stockage par paramètre `<storage-class-name>.storageclass.storage.k8s.io/persistentvolumeclaims` à 0. En outre, un administrateur de cluster doit s'assurer que les développeurs d'un projet ne doivent pas avoir accès pour modifier les ResourceQuotas.

Configuration

Configuration

N'importe quelle solution mutualisée permet à aucun utilisateur d'accéder à davantage de ressources du cluster que nécessaire. Ainsi, l'ensemble des ressources à configurer dans le cadre de la configuration de colocation est divisé entre l'administrateur cluster, l'administrateur stockage et les développeurs travaillant sur chaque projet.

Le tableau suivant présente les différentes tâches à effectuer par différents utilisateurs :

Rôle	Tâches
Cluster-admin	Créez des projets pour différentes applications ou charges de travail
	Créez ClusterRoles et roles pour Storage-admin
	Créez des rôles et des roleliaisons pour les développeurs qui assignaient un accès à des projets spécifiques
	[Facultatif] configurez les projets pour planifier des pods sur des nœuds spécifiques
Storage-admin	Créez des SVM sur NetApp ONTAP
	Création des systèmes back-end Trident
	Créez des classes de stockage
	Créer des devis de ressources de stockage
Développeurs	Valider l'accès pour créer ou corriger des demandes de volume persistant ou des pods dans le projet affecté
	Valider l'accès pour créer ou corriger des demandes de volume persistant ou des pods dans un autre projet
	Validez l'accès pour afficher ou modifier des projets, des ResourceQuotas et des classes de stockage

Configuration

Vous trouverez ci-dessous les conditions préalables à la configuration de la colocation sur Red Hat OpenShift avec NetApp.

Prérequis

- Cluster NetApp ONTAP
- Cluster Red Hat OpenShift
- Trident est installé sur le cluster
- Station de travail Admin avec les outils tridentctl et oc installés et ajoutés à $PATH
- Accès administrateur à ONTAP
- L'accès cluster-admin au cluster OpenShift
- Le cluster est intégré avec Identity Provider
- Le fournisseur d'identités est configuré pour distinguer efficacement les utilisateurs de différentes équipes

Configuration : tâches d'administration du cluster

Les tâches suivantes sont réalisées par Red Hat OpenShift Cluster-admin :

1. Connectez-vous au cluster Red Hat OpenShift en tant qu'administrateur cluster.
2. Créer deux projets correspondant à différents projets.

```
oc create namespace project-1
oc create namespace project-2
```

3. Créer le rôle de développeur du projet-1.

```
cat << EOF | oc create -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: project-1
  name: developer-project-1
rules:
  - verbs:
      - '*'
    apiGroups:
      - apps
      - batch
      - autoscaling
      - extensions
      - networking.k8s.io
      - policy
```

```
      - apps.openshift.io
      - build.openshift.io
      - image.openshift.io
      - ingress.operator.openshift.io
      - route.openshift.io
      - snapshot.storage.k8s.io
      - template.openshift.io
    resources:
      - '*'
  - verbs:
      - '*'
    apiGroups:
      - ''
    resources:
      - bindings
      - configmaps
      - endpoints
      - events
      - persistentvolumeclaims
      - pods
      - pods/log
      - pods/attach
      - podtemplates
      - replicationcontrollers
      - services
      - limitranges
      - namespaces
      - componentstatuses
      - nodes
  - verbs:
      - '*'
    apiGroups:
      - trident.netapp.io
    resources:
      - tridentsnapshots
EOF
```

La définition de rôle fournie dans cette section n'est qu'un exemple. Les rôles de développeur doivent être définis en fonction des exigences de l'utilisateur final.

1. De la même façon, créez des rôles de développement pour Project-2.
2. Toutes les ressources de stockage OpenShift et NetApp sont généralement gérées par un administrateur du stockage. L'accès pour les administrateurs du stockage est contrôlé par le rôle de l'opérateur trident créé lors de l'installation de Trident. En outre, l'administrateur du stockage nécessite également l'accès à ResourceQuotas pour contrôler la consommation du stockage.

3. Créez un rôle pour gérer ResourceQuotas dans tous les projets du cluster afin de le relier à l'administrateur de stockage.

```
cat << EOF | oc create -f -
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: resource-quotas-role
rules:
  - verbs:
      - '*'
    apiGroups:
      - ''
    resources:
      - resourcequotas
  - verbs:
      - '*'
    apiGroups:
      - quota.openshift.io
    resources:
      - '*'
EOF
```

4. Assurez-vous que le cluster est intégré au fournisseur d'identité de l'entreprise et que les groupes d'utilisateurs sont synchronisés avec les groupes de clusters. L'exemple suivant montre que le fournisseur d'identités a été intégré au cluster et synchronisé avec les groupes d'utilisateurs.

```
$ oc get groups
NAME                        USERS
ocp-netapp-storage-admins   ocp-netapp-storage-admin
ocp-project-1               ocp-project-1-user
ocp-project-2               ocp-project-2-user
```

1. Configurer les liaisons ClusterRoleBindages pour les administrateurs de stockage.

```
cat << EOF | oc create -f -
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: netapp-storage-admin-trident-operator
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: ocp-netapp-storage-admins
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-operator
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: netapp-storage-admin-resource-quotas-cr
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: ocp-netapp-storage-admins
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: resource-quotas-role
EOF
```

Pour les administrateurs du stockage, deux rôles doivent être liés : trident-Operator et Resource-quotas.

1. Créer des liaisons de type rôle pour les développeurs liant le rôle développeur-projet-1 au groupe correspondant (ocp-project-1) dans Project-1.

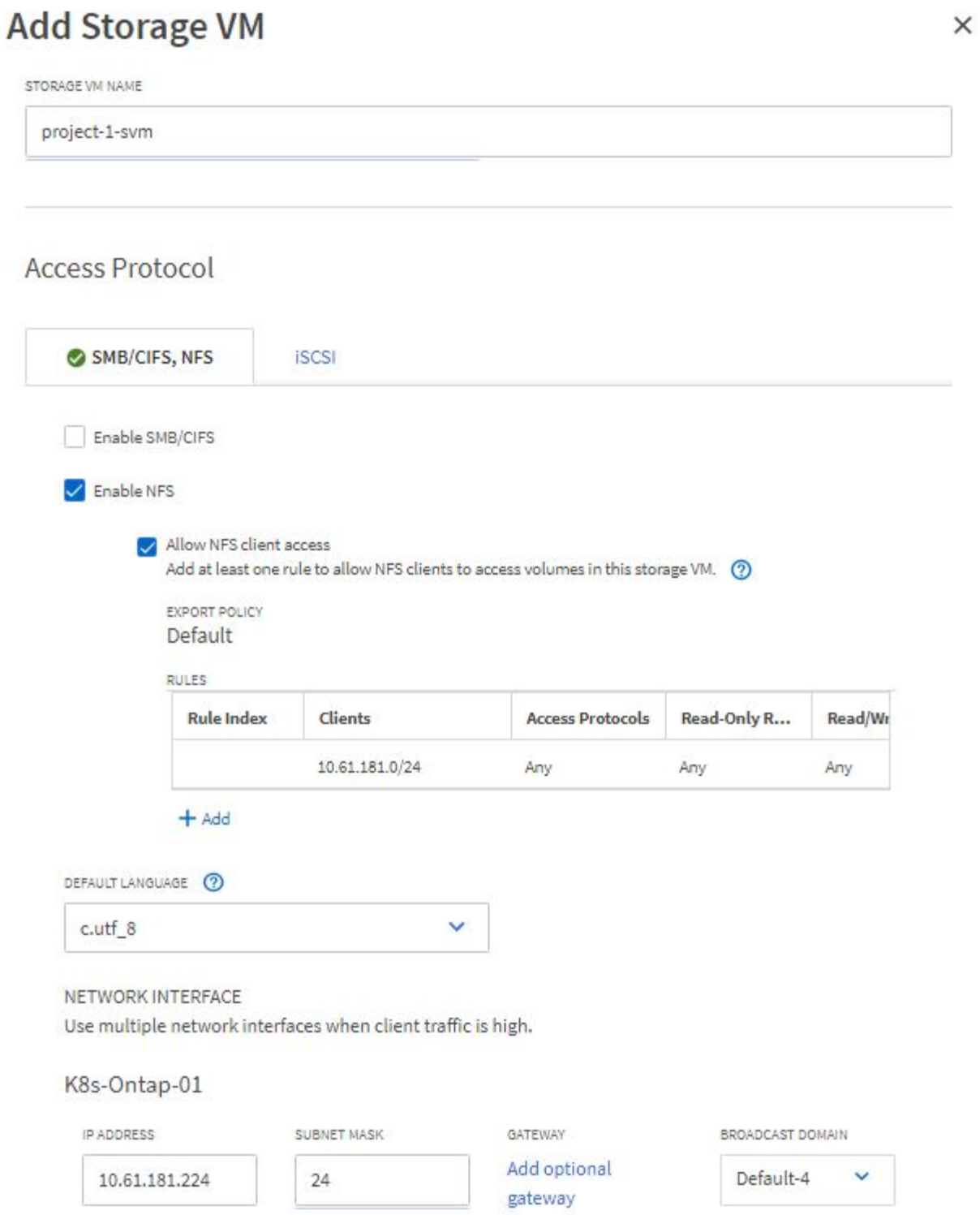
```
cat << EOF | oc create -f -
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: project-1-developer
  namespace: project-1
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: ocp-project-1
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: developer-project-1
EOF
```

2. De même, créez des liaisons de type rôle pour les développeurs qui lient les rôles de développeur au groupe d'utilisateurs correspondant dans Project-2.

Configuration : tâches d'administration du stockage

Les ressources suivantes doivent être configurées par un administrateur de stockage :

1. Connectez-vous au cluster NetApp ONTAP en tant qu'administrateur.
2. Accédez à Storage > Storage VM et cliquez sur Add. Créer deux SVM, un pour le projet-1 et l'autre pour le projet-2, en fournissant les détails requis. Créer également un compte vsadmin pour gérer le SVM et ses ressources



1. Connectez-vous au cluster Red Hat OpenShift en tant qu'administrateur du stockage.
2. Créer le backend pour projet-1 et le mapper au SVM dédié au projet NetApp recommande d'utiliser le compte vsadmin du SVM afin de connecter le backend au SVM au lieu d'utiliser l'administrateur du cluster ONTAP

```
cat << EOF | tridentctl -n trident create backend -f
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "nfs_project_1",
    "managementLIF": "172.21.224.210",
    "dataLIF": "10.61.181.224",
    "svm": "project-1-svm",
    "username": "vsadmin",
    "password": "NetApp123"
}
EOF
```

Nous utilisons le pilote ontap-nas dans cet exemple. Utilisez le pilote approprié lors de la création du back-end en fonction du cas d'utilisation.

Nous partons du principe que Trident est installé dans le projet trident.

1. Créer de la même manière le back-end Trident pour le projet-2 et le mapper sur le SVM dédié au projet-2.
2. Créez ensuite les classes de stockage. Créez la classe de stockage pour Project-1 et configurez-la pour utiliser les pools de stockage du back-end dédié au projet-1 en définissant le paramètre storagePools.

```
cat << EOF | oc create -f -
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: project-1-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
  storagePools: "nfs_project_1:.*"
EOF
```

3. De même, créez une classe de stockage pour Project-2 et configurez-la pour utiliser les pools de stockage du système back-end dédié au projet-2.
4. Créer un Resourcequota pour limiter les ressources dans le projet-1 demandant le stockage de storageclasses dédiés à d'autres projets.

```
cat << EOF | oc create -f -
kind: ResourceQuota
apiVersion: v1
metadata:
  name: project-1-sc-rq
  namespace: project-1
spec:
  hard:
    project-2-sc.storageclass.storage.k8s.io/persistentvolumeclaims: 0
EOF
```

5. De même, créez un Resourcequota pour limiter les ressources du projet 2 demandant du stockage de storageclasses dédiés à d'autres projets.

Validation

Pour valider l'architecture mutualisée configurée lors des étapes précédentes, procédez comme suit :

Valider l'accès pour créer des demandes de volume persistant ou des pods dans le projet attribué

1. Connectez-vous en tant qu'utilisateur ocp-project-1, développeur dans Project-1.
2. Vérifiez l'accès pour créer un nouveau projet.

```
oc create ns sub-project-1
```

3. Créez un PVC dans Project-1 en utilisant le storageclass affecté au projet-1.

```
cat << EOF | oc create -f -
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-pvc-project-1
  namespace: project-1
  annotations:
    trident.netapp.io/reclaimPolicy: Retain
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: project-1-sc
EOF
```

4. Vérifiez le volume persistant associé à la demande de volume persistant.

```
oc get pv
```

5. Vérifiez que le volume persistant et son volume sont créés dans un SVM dédié à Project-1 sur NetApp ONTAP.

```
volume show -vserver project-1-svm
```

6. Créez un pod dans Project-1 et montez le PVC créé à l'étape précédente.

```
cat << EOF | oc create -f -
kind: Pod
apiVersion: v1
metadata:
  name: test-pvc-pod
  namespace: project-1
spec:
  volumes:
    - name: test-pvc-project-1
      persistentVolumeClaim:
       claimName: test-pvc-project-1
  containers:
    - name: test-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: test-pvc-project-1
EOF
```

7. Vérifiez si le pod est en cours d'exécution et si il a monté le volume.

```
oc describe pods test-pvc-pod -n project-1
```

Valider l'accès pour créer des demandes de volume persistant ou des pods dans un autre projet ou utiliser des ressources dédiées à un autre projet

1. Connectez-vous en tant qu'utilisateur ocp-project-1, développeur dans Project-1.
2. Créez un PVC dans Project-1 en utilisant le storageclass affecté au projet-2.

```
cat << EOF | oc create -f -
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-pvc-project-1-sc-2
  namespace: project-1
  annotations:
    trident.netapp.io/reclaimPolicy: Retain
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: project-2-sc
EOF
```

3. Création d'une demande de volume persistant dans Project-2.

```
cat << EOF | oc create -f -
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: test-pvc-project-2-sc-1
  namespace: project-2
  annotations:
    trident.netapp.io/reclaimPolicy: Retain
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: project-1-sc
EOF
```

4. Assurez-vous que les ESV `test-pvc-project-1-sc-2` et `test-pvc-project-2-sc-1` n'ont pas été créés.

```
oc get pvc -n project-1
oc get pvc -n project-2
```

5. Créez un pod dans Project-2.

```
cat << EOF | oc create -f -
kind: Pod
apiVersion: v1
metadata:
  name: test-pvc-pod
  namespace: project-1
spec:
  containers:
    - name: test-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
EOF
```

Validez l'accès pour afficher et modifier les projets, ResourceQuotas et Storageclasses

1. Connectez-vous en tant qu'utilisateur ocp-project-1, développeur dans Project-1.
2. Vérifiez l'accès pour créer de nouveaux projets.

```
oc create ns sub-project-1
```

3. Valider l'accès pour afficher les projets.

```
oc get ns
```

4. Vérifiez si l'utilisateur peut afficher ou modifier ResourceQuotas dans Project-1.

```
oc get resourcequotas -n project-1
oc edit resourcequotas project-1-sc-rq -n project-1
```

5. Vérifiez que l'utilisateur a accès à l'affichage des données de stockage.

```
oc get sc
```

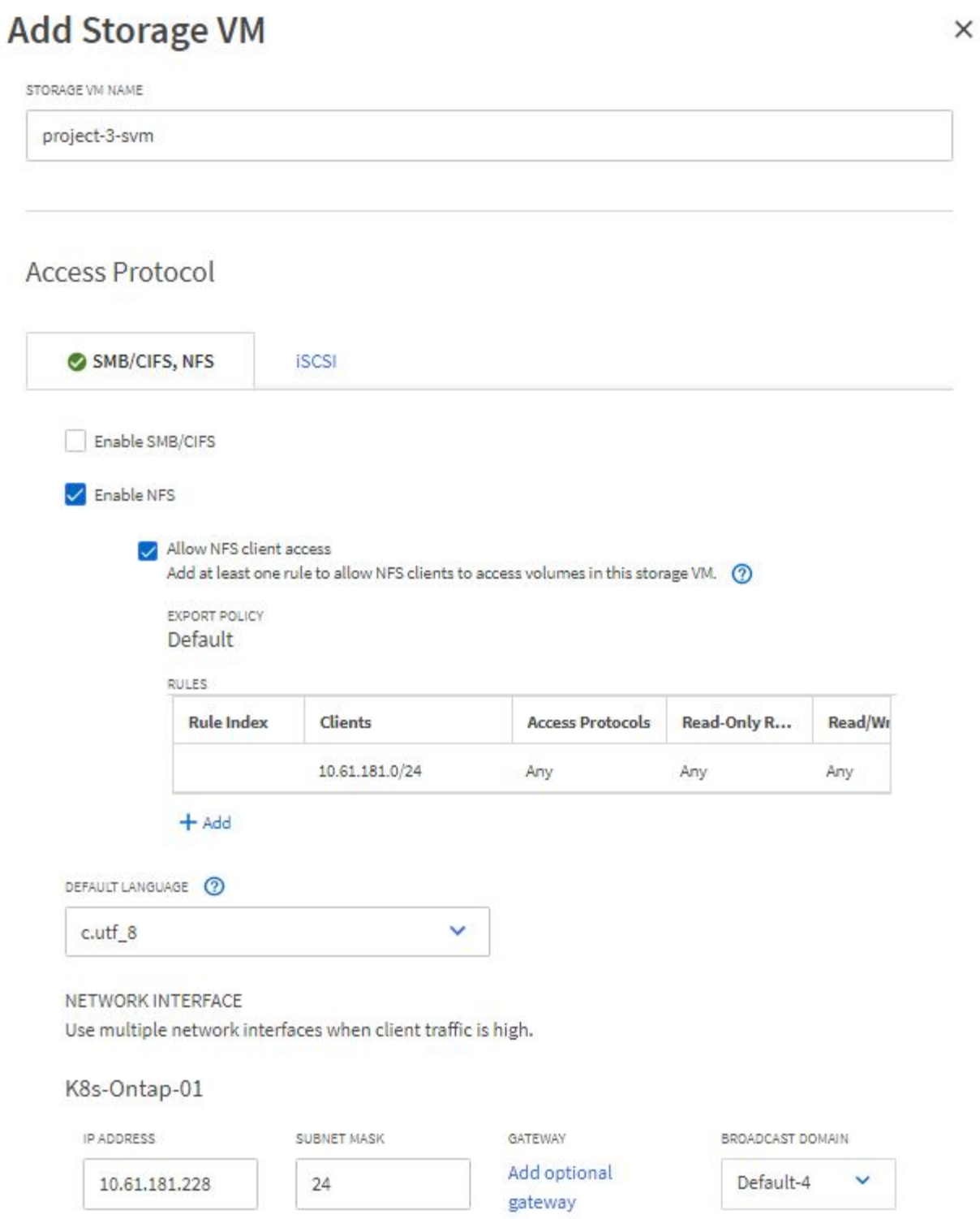
6. Vérifiez l'accès pour décrire les storageclasses.
7. Validez l'accès de l'utilisateur pour modifier les storageclasses.

```
oc edit sc project-1-sc
```

Évolutivité : ajout de projets

Dans une configuration mutualisée, l'ajout de nouveaux projets avec des ressources de stockage nécessite une configuration supplémentaire pour garantir que la colocation n'est pas respectée. Pour ajouter d'autres projets dans un cluster mutualisé, effectuez les opérations suivantes :

1. Connectez-vous au cluster NetApp ONTAP en tant qu'administrateur du stockage.
2. Accédez à `Storage → Storage VMs` et cliquez sur `Add`. Créez un nouveau SVM dédié au projet-3. Créer également un compte vsadmin pour gérer le SVM et ses ressources



1. Connectez-vous au cluster Red Hat OpenShift en tant qu'administrateur de cluster.
2. Créer un nouveau projet.

```
oc create ns project-3
```

3. Assurez-vous que le groupe d'utilisateurs du projet Project-3 est créé sur IDP et synchronisé avec le cluster OpenShift.

```
oc get groups
```

4. Créer le rôle de développeur du projet-3.

```
cat << EOF | oc create -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: project-3
  name: developer-project-3
rules:
  - verbs:
      - '*'
    apiGroups:
      - apps
      - batch
      - autoscaling
      - extensions
      - networking.k8s.io
      - policy
      - apps.openshift.io
      - build.openshift.io
      - image.openshift.io
      - ingress.operator.openshift.io
      - route.openshift.io
      - snapshot.storage.k8s.io
      - template.openshift.io
    resources:
      - '*'
  - verbs:
      - '*'
    apiGroups:
      - ''
    resources:
      - bindings
      - configmaps
      - endpoints
      - events
      - persistentvolumeclaims
      - pods
      - pods/log
      - pods/attach
```

```
      - podtemplates
      - replicationcontrollers
      - services
      - limitranges
      - namespaces
      - componentstatuses
      - nodes
  - verbs:
      - '*'
    apiGroups:
      - trident.netapp.io
    resources:
      - tridentsnapshots
EOF
```

La définition de rôle fournie dans cette section n'est qu'un exemple. Le rôle de développeur doit être défini en fonction des exigences de l'utilisateur final.

1. Créer RoleBinding pour les développeurs dans projet-3 liant le rôle développeur-projet-3 au groupe correspondant (ocp-project-3) dans projet-3.

```
cat << EOF | oc create -f -
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: project-3-developer
  namespace: project-3
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: ocp-project-3
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: developer-project-3
EOF
```

2. Connectez-vous au cluster Red Hat OpenShift en tant qu'administrateur du stockage
3. Créer un système back-end Trident et le mapper sur le SVM dédié au projet-3. NetApp recommande d'utiliser le compte vsadmin du SVM afin de connecter le backend au SVM au lieu d'utiliser l'administrateur du cluster ONTAP

```
cat << EOF | tridentctl -n trident create backend -f
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "nfs_project_3",
    "managementLIF": "172.21.224.210",
    "dataLIF": "10.61.181.228",
    "svm": "project-3-svm",
    "username": "vsadmin",
    "password": "NetApp!23"
}
EOF
```

Nous utilisons le pilote ontap-nas dans cet exemple. Utilisez le pilote approprié pour créer le back-end en fonction du cas d'utilisation.

Nous partons du principe que Trident est installé dans le projet trident.

1. Créez la classe de stockage pour Project-3 et configurez-la pour qu'elle utilise les pools de stockage du système back-end dédié au projet-3.

```
cat << EOF | oc create -f -
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: project-3-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
  storagePools: "nfs_project_3:.*"
EOF
```

2. Créer un Resourcequota pour limiter les ressources dans le projet-3 demandant du stockage de storageclasses dédié à d'autres projets.

```
cat << EOF | oc create -f -
kind: ResourceQuota
apiVersion: v1
metadata:
  name: project-3-sc-rq
  namespace: project-3
spec:
  hard:
    project-1-sc.storageclass.storage.k8s.io/persistentvolumeclaims: 0
    project-2-sc.storageclass.storage.k8s.io/persistentvolumeclaims: 0
EOF
```

3. Patch des ResourceQuotas dans d'autres projets pour limiter les ressources de ces projets à l'accès au stockage depuis le storageclass dédié au projet-3.

```
oc patch resourcequotas project-1-sc-rq -n project-1 --patch
'{"spec":{"hard":{ "project-3-
sc.storageclass.storage.k8s.io/persistentvolumeclaims": 0}}}'
oc patch resourcequotas project-2-sc-rq -n project-2 --patch
'{"spec":{"hard":{ "project-3-
sc.storageclass.storage.k8s.io/persistentvolumeclaims": 0}}}'
```