OpenShift Dedicated 4

Le suivi

Le suivi des projets dans OpenShift Dedicated

Last Updated: 2025-05-27

OpenShift Dedicated 4 Le suivi

Le suivi des projets dans OpenShift Dedicated

Notice légale

Résumé

Ce document fournit des informations sur les projets de surveillance dans OpenShift Dedicated.

Table des matières

CHAPITRE 1. APERÇU DU SUIVI

1.1. À PROPOS D'OPENSHIFT SURVEILLANCE DÉDIÉE

Dans OpenShift Dedicated, vous pouvez surveiller vos propres projets indépendamment des métriques de la plate-forme Red Hat Site Reliability Engineering (SRE). Il est possible de surveiller vos propres projets sans avoir besoin d'une solution de surveillance supplémentaire.

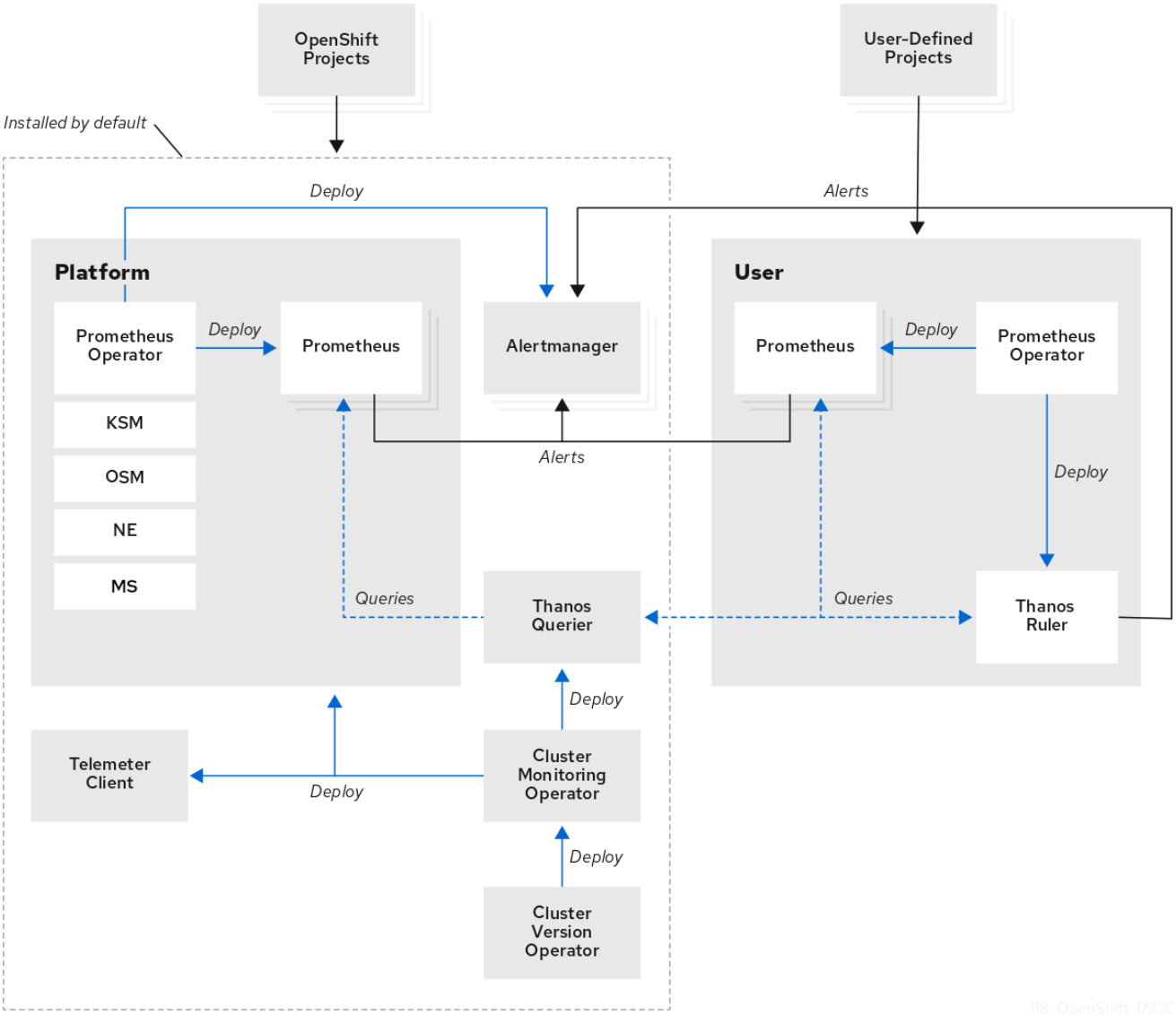
La pile de surveillance OpenShift Dedicated est basée sur le projet open source Prometheus et son écosystème plus large.

1.2. COMPRENDRE LA PILE DE SURVEILLANCE

La pile de surveillance comprend les composants suivants:

- Composants de surveillance de plate-forme par défaut. Lors d'une installation dédiée à OpenShift, un ensemble de composants de surveillance de plate-forme sont installés par défaut dans le projet de surveillance openshift. Les ingénieurs de fiabilité du site Red Hat (SRE) utilisent ces composants pour surveiller les composants principaux des clusters, y compris les services Kubernetes. Cela inclut des métriques critiques, telles que CPU et mémoire, collectées à partir de toutes les charges de travail dans chaque espace de noms.
  Ces composants sont illustrés dans la section Installé par défaut dans le diagramme suivant.

- Composants pour le suivi des projets définis par l'utilisateur. Dans le cadre d'une installation dédiée à OpenShift, un ensemble de composants de surveillance de projet définis par l'utilisateur sont installés par défaut dans le cadre d'un projet de surveillance de la charge de travail ouvert par l'utilisateur. Ces composants peuvent être utilisés pour surveiller les services et les pods dans les projets définis par l'utilisateur. Ces composants sont illustrés dans la section Utilisateur dans le diagramme suivant.

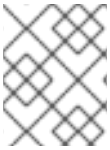


1.2.1. Cibles de surveillance par défaut

Ce qui suit sont des exemples de cibles surveillées par Red Hat Site Reliability Engineers (SRE) dans votre cluster OpenShift dédié:

- CoreDNS
- etc.
- HAProxy
- Enregistrement d'images
- Kubelets
- Kubernetes serveur API
- Gestionnaire de contrôleur Kubernetes
- Kubernetes planificateur
- Le serveur d'API OpenShift

- Gestionnaire de contrôleur OpenShift
- Gestionnaire du cycle de vie de l'opérateur (OLM)



NOTE

La liste exacte des cibles peut varier en fonction des capacités de votre cluster et des composants installés.

Ressources supplémentaires

- L'obtention d'informations détaillées sur une cible de mesures

1.2.2. Composants pour le suivi des projets définis par l'utilisateur

Le logiciel OpenShift Dedicated inclut une amélioration optionnelle de la pile de surveillance qui vous permet de surveiller les services et les pods dans les projets définis par l'utilisateur. Cette fonctionnalité comprend les composants suivants:

Tableau 1.1. Composants pour le suivi des projets définis par l'utilisateur

Composante	Description
L'opérateur Prometheus	L'opérateur Prometheus (PO) dans le projet openshift-user-workload-monitoring crée, configure et gère les instances Prometheus et Thanos Ruler dans le même projet.
Le Prométhée	Le Prometheus est le système de surveillance par lequel la surveillance est assurée pour les projets définis par l'utilisateur. Le Prometheus envoie des alertes à Alertmanager pour le traitement.
Le chef de Thanos	Le Thanos Ruler est un moteur d'évaluation de règles pour Prometheus qui est déployé en tant que processus séparé. Dans OpenShift Dedicated , Thanos Ruler fournit une évaluation des règles et des alertes pour le suivi des projets définis par l'utilisateur.
Alertmanager	Le service Alertmanager gère les alertes reçues de Prometheus et Thanos Ruler. Alertmanager est également responsable de l'envoi d'alertes définies par l'utilisateur aux systèmes de notification externes. Le déploiement de ce service est facultatif.

L'ensemble de ces composants sont surveillés par la pile et sont automatiquement mis à jour lorsque OpenShift Dedicated est mis à jour.

1.2.3. Cibles de suivi pour les projets définis par l'utilisateur

La surveillance est activée par défaut pour les projets définis par l'utilisateur OpenShift. Il est possible de surveiller:

- Les métriques fournies par l'intermédiaire de points de terminaison de service dans les projets définis par l'utilisateur.
- Des pods s'exécutant dans des projets définis par l'utilisateur.

1.2.4. La pile de surveillance dans les clusters à haute disponibilité

Dans les clusters multi-nœuds par défaut, les composants suivants s'exécutent en mode haute disponibilité (HA) pour éviter la perte de données et l'interruption de service:

- Le Prométhée
- Alertmanager
- Le chef de Thanos

Le composant est reproduit sur deux pods, chacun fonctionnant sur un nœud séparé. Cela signifie que la pile de surveillance peut tolérer la perte d'une gousse.

Le Prométhée en mode HA

- Les deux répliques grattent indépendamment les mêmes cibles et évaluent les mêmes règles.
- Les répliques ne communiquent pas entre elles. Les données peuvent donc différer entre les pods.

Alertmanager en mode HA

- Les deux répliques synchronisent les états de notification et de silence les uns avec les autres. Cela garantit que chaque notification est envoyée au moins une fois.
- Lorsque les répliques ne parviennent pas à communiquer ou s'il y a un problème du côté de la réception, des notifications sont toujours envoyées, mais elles peuvent être dupliquées.

IMPORTANT

Les Prometheus, Alertmanager et Thanos Ruler sont des composants d'état. Afin d'assurer une disponibilité élevée, vous devez les configurer avec un stockage persistant.

Ressources supplémentaires

- Détection et support de clusters à haute disponibilité ou à un seul nœud
- Configuration du stockage persistant
- Configuration de la pile de surveillance

1.3. GLOSSAIRE DES TERMES COMMUNS POUR OPENSHIFT SURVEILLANCE DÉDIÉE

Ce glossaire définit des termes communs qui sont utilisés dans l'architecture dédiée OpenShift.

Alertmanager

Alertmanager gère les alertes reçues de Prometheus. Alertmanager est également responsable de l'envoi des alertes aux systèmes de notification externes.

Les règles d'alerte

Les règles d'alerte contiennent un ensemble de conditions qui définissent un état particulier au sein d'un cluster. Les alertes sont déclenchées lorsque ces conditions sont vraies. Il est possible d'attribuer une règle d'alerte à une sévérité qui définit la manière dont les alertes sont acheminées.

Opérateur de surveillance des clusters

L'opérateur de surveillance des grappes (CMO) est un élément central de la pile de surveillance. Il déploie et gère des instances Prometheus telles que, le Thanos Querier, le Telemeter Client, et des cibles métriques pour s'assurer qu'elles sont à jour. Le CMO est déployé par l'opérateur de versions de cluster (CVO).

L'opérateur de la version cluster

Le CVO (Cluster Version Operator) gère le cycle de vie des opérateurs de clusters, dont beaucoup sont installés dans OpenShift Dedicated par défaut.

configuration de la carte

La carte de configuration fournit un moyen d'injecter des données de configuration dans des pods. Les données stockées dans une carte de configuration peuvent être référencées dans un volume de type ConfigMap. Les applications qui s'exécutent dans un pod peuvent utiliser ces données.

Conteneur

Le conteneur est une image légère et exécutable qui comprend des logiciels et toutes ses dépendances. Les conteneurs virtualisent le système d'exploitation. En conséquence, vous pouvez exécuter des conteneurs n'importe où d'un centre de données vers un cloud public ou privé ainsi que l'ordinateur portable d'un développeur.

les ressources personnalisées (CR)

A CR est une extension de l'API Kubernetes. Créez des ressources personnalisées.

etc.

etcd est le magasin de valeur clé pour OpenShift Dedicated, qui stocke l'état de tous les objets de ressource.

Fluentd

Fluentd est un collectionneur de journaux qui réside sur chaque nœud dédié OpenShift. Il rassemble les journaux d'applications, d'infrastructures et d'audits et les transmet à différents extrants.

NOTE

Fluentd est déprécié et devrait être retiré dans une version ultérieure. Le Red Hat fournit des corrections de bogues et une prise en charge de cette fonctionnalité pendant le cycle de vie de la version actuelle, mais cette fonctionnalité ne reçoit plus d'améliorations. Comme alternative à Fluentd, vous pouvez utiliser Vector à la place.

Kubelets

Fonctionne sur les nœuds et lit le conteneur se manifeste. Assure que les conteneurs définis ont commencé et sont en cours d'exécution.

Kubernetes serveur API

Kubernetes API serveur valide et configure les données pour les objets API.

Gestionnaire de contrôleur Kubernetes

Le gestionnaire du contrôleur Kubernetes régit l'état du cluster.

Kubernetes planificateur

Kubernetes programmateur alloue des pods aux nœuds.

étiquettes

Les étiquettes sont des paires clé-valeur que vous pouvez utiliser pour organiser et sélectionner des sous-ensembles d'objets tels qu'un pod.

Le serveur de métriques

Le composant de surveillance Metrics Server recueille les métriques des ressources et les expose dans le service API Metrics.k8s.io pour une utilisation par d'autres outils et API, ce qui libère la pile de la plateforme principale Prometheus de gérer cette fonctionnalité.

le nœud

D'une machine ouvrier dans le cluster dédié OpenShift. Le nœud est soit une machine virtuelle (VM) soit une machine physique.

Exploitant

La méthode préférée d'emballage, de déploiement et de gestion d'une application Kubernetes dans un cluster dédié OpenShift. L'opérateur utilise les connaissances opérationnelles humaines et l'encode dans un logiciel qui est emballé et partagé avec les clients.

Gestionnaire du cycle de vie de l'opérateur (OLM)

Il vous aide à installer, mettre à jour et gérer le cycle de vie des applications natives Kubernetes. L'OLM est une boîte à outils open source conçue pour gérer les opérateurs de manière efficace, automatisée et évolutive.

Le stockage persistant

Enregistre les données même après l'arrêt de l'appareil. Kubernetes utilise des volumes persistants pour stocker les données de l'application.

Allégation de volume persistant (PVC)

Il est possible d'utiliser un PVC pour monter un Volume Persistent dans un Pod. Il est possible d'accéder au stockage sans connaître les détails de l'environnement cloud.

la pod

Le pod est la plus petite unité logique de Kubernetes. La gousse est composée d'un ou de plusieurs contenants à utiliser dans un nœud ouvrier.

Le Prométhée

Le Prometheus est le système de surveillance sur lequel repose la pile de surveillance OpenShift Dedicated. Le Prometheus est une base de données de séries chronologiques et un moteur d'évaluation des règles pour les métriques. Le Prometheus envoie des alertes à Alertmanager pour le traitement.

L'opérateur Prometheus

L'opérateur Prometheus (PO) dans le projet de surveillance openshift crée, configure et gère les instances de la plateforme Prometheus et Alertmanager. Il génère également automatiquement des configurations cibles de surveillance basées sur les requêtes d'étiquettes Kubernetes.

Les silences

Le silence peut être appliqué à une alerte pour empêcher l'envoi de notifications lorsque les conditions d'une alerte sont vraies. Après la notification initiale, vous pouvez réduire une alerte pendant que vous travaillez à résoudre le problème sous-jacent.

le stockage

Le logiciel OpenShift Dedicated prend en charge de nombreux types de stockage sur AWS et GCP. Dans un cluster dédié OpenShift, vous pouvez gérer le stockage de conteneurs pour des données persistantes et non persistantes.

Le chef de Thanos

Le Thanos Ruler est un moteur d'évaluation de règles pour Prometheus qui est déployé en tant que processus séparé. Dans OpenShift Dedicated, Thanos Ruler fournit une évaluation des règles et des alertes pour le suivi des projets définis par l'utilisateur.

Le vecteur

Le vecteur est un collecteur de journaux qui se déploie sur chaque nœud dédié OpenShift. Il collecte les données de log de chaque nœud, transforme les données et les transmet aux sorties configurées.

console Web

Interface utilisateur (UI) pour gérer OpenShift Dedicated.

CHAPITRE 2. ACCÈS AU SUIVI POUR LES PROJETS DÉFINIS PAR L'UTILISATEUR

Lorsque vous installez un cluster dédié OpenShift, la surveillance des projets définis par l'utilisateur est activée par défaut. Avec la surveillance des projets définis par l'utilisateur activé, vous pouvez surveiller vos propres projets OpenShift dédiés sans avoir besoin d'une solution de surveillance supplémentaire.

L'utilisateur admin dédié dispose d'autorisations par défaut pour configurer et accéder à la surveillance des projets définis par l'utilisateur.

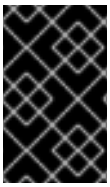
NOTE

Les instances Prometheus personnalisées et l'opérateur Prometheus installé via Operator Lifecycle Manager (OLM) peuvent causer des problèmes avec la surveillance de projet définie par l'utilisateur si elle est activée. Les instances Prometheus personnalisées ne sont pas prises en charge.

En option, vous pouvez désactiver la surveillance des projets définis par l'utilisateur pendant ou après l'installation d'un cluster.

CHAPITRE 3. CONFIGURATION DE LA PILE DE SURVEILLANCE

Cette section explique quelle configuration est prise en charge, montre comment configurer la pile de surveillance pour les projets définis par l'utilisateur et présente plusieurs scénarios de configuration communs.



IMPORTANT

Les paramètres de configuration de la pile de surveillance ne sont pas tous exposés. Les paramètres et les champs listés dans la référence de la carte de configuration pour l'opérateur de surveillance du cluster sont pris en charge pour la configuration.

3.1. ENTRETIEN ET PRISE EN CHARGE DE LA SURVEILLANCE

Les options de configuration de la pile de surveillance ne sont pas toutes exposées. La seule façon prise en charge de configurer OpenShift Dedicated monitoring consiste à configurer l'opérateur de surveillance du cluster (CMO) en utilisant les options décrites dans la référence de la carte de configuration pour l'opérateur de surveillance du cluster. **Il ne faut pas utiliser d'autres configurations, car elles ne sont pas prises en charge.**

Les paradigmes de configuration peuvent changer entre les versions de Prometheus, et de tels cas ne peuvent être traités que si toutes les possibilités de configuration sont contrôlées. Lorsque vous utilisez des configurations autres que celles décrites dans la référence de la carte Config pour l'opérateur de surveillance du cluster, vos modifications disparaîtront parce que le CMO réconcilie automatiquement toutes les différences et réinitialise toute modification non prise en charge à l'état initialement défini par défaut et par conception.

IMPORTANT

L'installation d'une autre instance Prometheus n'est pas prise en charge par les ingénieurs de fiabilité du site Red Hat (SRE).

3.1.1. Considérations d'appui au suivi

NOTE

La rétrocompatibilité pour les métriques, les règles d'enregistrement ou les règles d'alerte n'est pas garantie.

Les modifications suivantes ne sont explicitement pas prises en charge:

- **Installation d'instances Prometheus personnalisées sur OpenShift Dedicated.**L'instance personnalisée est une ressource personnalisée (CR) de Prometheus gérée par l'opérateur Prometheus.
- **La modification des composants de surveillance de la plate-forme par défaut**Il ne faut pas modifier aucun des composants définis dans la carte de configuration cluster-monitoring-config. Le Red Hat SRE utilise ces composants pour surveiller les composants principaux du cluster et les services Kubernetes.

3.1.2. La matrice de version de support pour les composants de surveillance

La matrice suivante contient des informations sur les versions des composants de surveillance d'OpenShift Dedicated 4.12 et des versions ultérieures:

Tableau 3.1. Les versions dédiées et composants d'OpenShift

Dédié à OpenShift	L'opérateur Prometheus	Le Prométhée	Le serveur de métriques	Alertmanager	Kube-state-metrics agent	la surveillance-plugin	agent d'exportation de nœuds	♪ Thanos ♪
4.18	0,78.1	2.55.1	0,7.2	0,27,0	2.13.0	1.0.0.	1.8.2	0.36.1
4.17	0,75.2	2.53.1	0,7.1	0,27,0	2.13.0	1.0.0.	1.8.2	0.35.1
4.16	0,73.2	2.52.0	0,7.1	0,26.0	2.12.0	1.0.0.	1.8.0	0.35.0
4.15	0,70,0	2.48.0	0.6.4	0,26.0	2.10.1	1.0.0.	1.7.0	0.32.5
4.14	0.67.1	2.46.0	C) N/A	0,25,0	2.9.2	1.0.0.	1.6.1	0.30.2
4.13	0.63.0	2.42.0	C) N/A	0,25,0	2.8.1	C) N/A	1.5.0	0.30.2
4.12	0.60.1	2.39.1	C) N/A	0.24.0	2.6.0	C) N/A	1.4.0	0,28.1

NOTE

L'agent openshift-state-metrics et Telemeter Client sont des composants spécifiques à OpenShift. Leurs versions correspondent donc aux versions d'OpenShift Dedicated.

3.2. CONFIGURATION DE LA PILE DE SURVEILLANCE

Dans OpenShift Dedicated, vous pouvez configurer la pile qui surveille les charges de travail des projets définis par l'utilisateur en utilisant l'objet ConfigMap. Configurez les cartes configurer l'opérateur de surveillance des clusters (CMO), qui à son tour configure les composants de la pile.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez l'objet ConfigMap.
    a. Éditer l'objet ConfigMap de l'utilisateur-workload-monitoring-config dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

b. Ajoutez votre configuration sous data/config.yaml en tant que paire de clés-valeur <component_name>: <component_configuration>:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    <component>:
      <configuration_for_the_component>
```

<composant> et <configuration_for_the_component> en conséquence.

L'exemple suivant de l'objet ConfigMap configure une période de conservation des données et des demandes minimales de ressources de conteneur pour Prometheus. Cela concerne l'instance Prometheus qui surveille uniquement les projets définis par l'utilisateur:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus: 1
      retention: 24h 2
      resources:
        requests:
          cpu: 200m 3
          memory: 2Gi 4
```

1. Définit le composant Prometheus et les lignes suivantes définissent sa configuration.
2. Configure une période de conservation des données de vingt-quatre heures pour l'instance Prometheus qui surveille les projets définis par l'utilisateur.
3. Définit une demande de ressources minimale de 200 millicores pour le conteneur Prometheus.
4. Définit une demande minimale de ressources de Pod de 2 GiB de mémoire pour le conteneur Prometheus.

2. Enregistrez le fichier pour appliquer les modifications à l'objet ConfigMap.

AVERTISSEMENT

Différentes modifications de configuration de l'objet ConfigMap donnent des résultats différents:

- Les pods ne sont pas redéployés. Il n'y a donc pas de panne de service.
- Les gousses touchées sont redéployées:
  - Dans le cas des clusters à nœud unique, il en résulte une panne de service temporaire.
  - Dans le cas des grappes à nœuds multiples, en raison de leur disponibilité élevée, les gousses touchées sont progressivement déployées et la pile de surveillance reste disponible.
  - La configuration et le redimensionnement d'un volume persistant entraînent toujours une panne de service, quelle que soit la disponibilité élevée.

Chaque procédure qui nécessite un changement dans la carte de configuration inclut le résultat attendu.

Ressources supplémentaires

- Configuration de référence pour la carte de configuration de la configuration de la charge de travail-surveillance de l'utilisateur

3.3. COMPOSANTS DE SURVEILLANCE CONFIGURABLES

Ce tableau montre les composants de surveillance que vous pouvez configurer et les clés utilisées pour spécifier les composants de la carte de configuration de la configuration de la charge de travail-surveillance de l'utilisateur.

AVERTISSEMENT

Il ne faut pas modifier les composants de surveillance de l'objet ConfigMap de cluster-monitoring-config. Les ingénieurs de fiabilité du site Red Hat (SRE) utilisent ces composants pour surveiller les composants principaux du cluster et les services Kubernetes.

Tableau 3.2. Composants de surveillance configurables pour les projets définis par l'utilisateur

Composante	clé de configuration de configuration de la charge de travail-surveillance de l'utilisateur
L'opérateur Prometheus	**accueil > PrometheusOperator**
Le Prométhée	**le Prométhée**
Alertmanager	**Alertmanager**
Le chef de Thanos	**le thanosRuler**

3.4. EN UTILISANT DES SÉLECTEURS DE NŒUDS POUR DÉPLACER LES COMPOSANTS DE SURVEILLANCE

En utilisant la contrainte nodeSelector avec des nœuds étiquetés, vous pouvez déplacer n'importe quel composant de la pile de surveillance vers des nœuds spécifiques. Ce faisant, vous pouvez contrôler le placement et la distribution des composants de surveillance sur un cluster.

En contrôlant le placement et la distribution des composants de surveillance, vous pouvez optimiser l'utilisation des ressources du système, améliorer les performances et séparer les charges de travail en fonction d'exigences ou de politiques spécifiques.

Comment les sélecteurs de nœuds fonctionnent avec d'autres contraintes

Lorsque vous déplacez les composants de surveillance en utilisant des contraintes de sélecteur de nœuds, sachez que d'autres contraintes pour contrôler la planification des pod peuvent exister pour un cluster:

- Des contraintes de propagation de la topologie pourraient être en place pour contrôler le placement des pods.
- Des règles anti-affinité dures sont en place pour Prometheus, Alertmanager et d'autres composants de surveillance pour s'assurer que plusieurs pods pour ces composants sont toujours répartis entre différents nœuds et sont donc toujours très disponibles.

Lors de la planification des pods sur les nœuds, le planificateur de pod tente de satisfaire toutes les contraintes existantes lors de la détermination du placement de la gousse. C'est-à-dire que toutes les contraintes se composent lorsque le planificateur de pod détermine quels pods seront placés sur quels nœuds.

Donc, si vous configurez une contrainte de sélecteur de nœuds mais que les contraintes existantes ne peuvent pas toutes être satisfaites, le programmeur de pod ne peut pas correspondre à toutes les contraintes et ne programmera pas un pod pour le placement sur un nœud.

Afin de maintenir la résilience et la disponibilité élevée pour les composants de surveillance, assurez-vous que suffisamment de nœuds sont disponibles et correspondent à toutes les contraintes lorsque vous configurez une contrainte de sélecteur de nœud pour déplacer un composant.

Ressources supplémentaires

- En utilisant des contraintes de propagation de la topologie de la pod pour la surveillance
- Documentation Kubernetes sur les sélecteurs de nœuds

3.4.1. Déplacement des composants de surveillance vers différents nœuds

Il est possible de déplacer l'un des composants qui surveillent les charges de travail des projets définis par l'utilisateur vers des nœuds de travail spécifiques.

AVERTISSEMENT

Il n'est pas permis de déplacer les composants vers des nœuds d'avion ou d'infrastructure de contrôle.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Lorsque vous ne l'avez pas encore fait, ajoutez une étiquette aux nœuds sur lesquels vous souhaitez exécuter les composants de surveillance:

```
$ oc label nodes <node_name> <node_label> 1
```

1 <node_name> par le nom du nœud où vous souhaitez ajouter l'étiquette. <node_label> par le nom de l'étiquette recherchée.

2. Éditer l'objet ConfigMap de l'utilisateur-workload-monitoring-config dans le projet openshift-user-workload-monitoring:

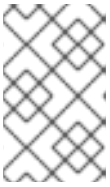
```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

3. Indiquez les étiquettes des nœuds pour la contrainte nodeSelector pour le composant sous data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    # ...
    <component>: 1
      nodeSelector:
        <node_label_1> 2
        <node_label_2> 3
    # ...
```

-

1. <composant> par le nom approprié du composant de la pile de surveillance.
2. Remplacez <node_label_1> par l'étiquette que vous avez ajoutée au nœud.
3. Facultatif: Spécifiez des étiquettes supplémentaires. Lorsque vous spécifiez des étiquettes supplémentaires, les pods du composant ne sont programmés que sur les nœuds qui contiennent toutes les étiquettes spécifiées.



NOTE

Lorsque les composants de surveillance restent dans un état en attente après avoir configuré la contrainte nodeSelector, vérifiez les événements de pod pour détecter les erreurs liées aux taintes et aux tolérances.

4. Enregistrez le fichier pour appliquer les modifications. Les composants spécifiés dans la nouvelle configuration sont automatiquement déplacés vers les nouveaux nœuds, et les pods affectés par la nouvelle configuration sont redéployés.

Ressources supplémentaires

- Consultez la documentation Kubernetes pour plus de détails sur la contrainte nodeSelector

3.5. ATTRIBUTION DES TOLÉRANCES AUX COMPOSANTS DE SURVEILLANCE

Il est possible d'attribuer des tolérances aux composants qui surveillent les projets définis par l'utilisateur, afin de les déplacer vers des nœuds de travail contaminés. La planification n'est pas autorisée sur les nœuds d'avion de contrôle ou d'infrastructure.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe dans l'espace de noms openshift-user-workload-monitoring. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Indiquez les tolérances pour le composant:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
```

```
data:
  config.yaml: |
    <component>:
      tolerations:
        <toleration_specification>
```

<composant> et <tolérance_spécification> en conséquence.

À titre d'exemple, oc adm taint nodes node1 key1=value1:NoSchedule ajoute un taint à node1 avec la clé1 et la valeur1. Cela empêche les composants de surveillance de déployer des pods sur node1 à moins qu'une tolérance ne soit configurée pour cette tainte. L'exemple suivant configure le composant thanosRuler pour tolérer l'exemplaire:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    thanosRuler:
      tolerations:
      - key: "key1"
        operator: "Equal"
        value: "value1"
        effect: "NoSchedule"
```

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.

Ressources supplémentaires

- Consultez la documentation Kubernetes sur les taintes et les tolérances

3.6. GESTION DES RESSOURCES CPU ET MÉMOIRE POUR LA SURVEILLANCE DES COMPOSANTS

Assurez-vous que les conteneurs qui exécutent des composants de surveillance disposent de suffisamment de ressources CPU et mémoire en spécifiant les valeurs pour les limites de ressources et les demandes pour ces composants.

Il est possible de configurer ces limites et demandes de composants principaux de surveillance de la plate-forme dans l'espace de noms openshift-monitoring et pour les composants qui surveillent les projets définis par l'utilisateur dans l'espace de noms openshift-Workload-monitoring.

3.6.1. À propos de spécifier les limites et les demandes de composants de surveillance

Configurez les limites de ressources et les demandes pour les composants de surveillance de la plate-forme de base suivants:

- Alertmanager
- Kube-state-metrics

- la surveillance-plugin
- exportateur de nœuds
- caractéristiques OpenShift-state-metrics
- Le Prométhée
- Le serveur de métriques
- L'opérateur Prometheus et son service webhook d'admission
- Client de télémètre
- À propos de Thanos Querier

Configurez les limites de ressources et les demandes pour les composants suivants qui surveillent les projets définis par l'utilisateur:

- Alertmanager
- Le Prométhée
- Le chef de Thanos

En définissant les limites de ressources, vous limitez l'utilisation des ressources d'un conteneur, ce qui empêche le conteneur de dépasser les valeurs maximales spécifiées pour les ressources CPU et mémoire.

En définissant les demandes de ressources, vous spécifiez qu'un conteneur ne peut être programmé que sur un nœud disposant de suffisamment de ressources CPU et mémoire disponibles pour correspondre aux ressources demandées.

3.6.2. Définir les limites et les demandes

Afin de configurer les ressources CPU et mémoire, spécifiez les valeurs des limites de ressources et des requêtes dans l'objet ConfigMap de l'utilisateur-workload-monitoring dans l'espace de noms openshift-user-workload-monitoring.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster en tant qu'utilisateur avec le rôle cluster-admin ou en tant qu'utilisateur avec le rôle utilisateur-téléchargement-surveillance-config-edit dans le projet openshift-user-workload-monitoring.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajoutez des valeurs pour définir les limites de ressources et les demandes pour chaque composant que vous souhaitez configurer.

IMPORTANT

Assurez-vous que la valeur définie pour une limite est toujours supérieure à la valeur définie pour une demande. Dans le cas contraire, une erreur se produira et le conteneur ne fonctionnera pas.

Exemple de définition des limites de ressources et des demandes

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    alertmanager:
      resources:
        limits:
          cpu: 500m
          memory: 1Gi
        requests:
          cpu: 200m
          memory: 500Mi
    prometheus:
      resources:
        limits:
          cpu: 500m
          memory: 3Gi
        requests:
          cpu: 200m
          memory: 500Mi
    thanosRuler:
      resources:
        limits:
          cpu: 500m
          memory: 1Gi
        requests:
          cpu: 200m
          memory: 500Mi
```

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.

3.7. CONFIGURATION DU STOCKAGE PERSISTANT

Exécutez la surveillance des clusters avec un stockage persistant pour obtenir les avantages suivants:

- De protéger vos métriques et d'alerter les données contre la perte de données en les stockant dans un volume persistant (PV). En conséquence, ils peuvent survivre aux gousses redémarrées ou recréées.
- Évitez d'obtenir des notifications en double et de perdre des silences pour les alertes lorsque les pods Alertmanager sont redémarrés.

Dans les environnements de production, il est fortement recommandé de configurer le stockage persistant.

IMPORTANT

Dans les clusters multi-nœuds, vous devez configurer le stockage persistant pour Prometheus, Alertmanager et Thanos Ruler pour assurer une disponibilité élevée.

3.7.1. Conditions de stockage persistantes

- Employez le type de stockage de bloc.

3.7.2. Configuration d'une revendication de volume persistante

Afin d'utiliser un volume persistant (PV) pour la surveillance des composants, vous devez configurer une revendication de volume persistant (PVC).

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajoutez votre configuration PVC pour le composant sous data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    <component>: 1
      volumeClaimTemplate:
        spec:
          storageClassName: <storage_class> 2
          resources:
            requests:
              storage: <amount_of_storage> 3
```

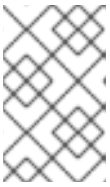
1. Indiquez le composant de surveillance pour lequel vous souhaitez configurer le PVC.
2. Indiquez une classe de stockage existante. Lorsqu'une classe de stockage n'est pas spécifiée, la classe de stockage par défaut est utilisée.

3 Indiquez la quantité de stockage requise.

L'exemple suivant configure un PVC qui revendique un stockage persistant pour Thanos Ruler:

Exemple de configuration PVC

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    thanosRuler:
      volumeClaimTemplate:
        spec:
          storageClassName: my-storage-class
          resources:
            requests:
              storage: 10Gi
```



NOTE

Les exigences de stockage pour le composant thanosRuler dépendent du nombre de règles évaluées et du nombre d'échantillons générés par chaque règle.

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés et la nouvelle configuration de stockage est appliquée.

AVERTISSEMENT

Lorsque vous mettez à jour la carte de configuration avec une configuration PVC, l'objet StatefulSet affecté est recréé, ce qui entraîne une panne de service temporaire.

Ressources supplémentaires

- Description PersistentVolumeClaims (documentation Kubernetes sur la façon de spécifier volumeClaimTemplate)

3.7.3. Durée de conservation et taille pour les métriques Prometheus

Par défaut, Prometheus conserve les données métriques pour les durées suivantes:

- Base de surveillance de la plate-forme: 15 jours
- Contrôle des projets définis par l'utilisateur: 24 heures

Il est possible de modifier le délai de conservation de l'instance Prometheus afin de modifier la rapidité avec laquelle les données sont supprimées. De plus, vous pouvez définir la quantité maximale d'espace disque utilisée par les données métriques conservées. Lorsque les données atteignent cette limite de taille, Prometheus supprime d'abord les données les plus anciennes jusqu'à ce que l'espace disque utilisé soit de nouveau inférieur à la limite.

À noter les comportements suivants de ces paramètres de conservation des données:

- La stratégie de rétention basée sur la taille s'applique à tous les répertoires de blocs de données du répertoire /prometheus, y compris les blocs persistants, les données du journal d'écriture-ahead (WAL) et les morceaux m-mapped.
- Les données dans les répertoires /wal et /head_chunks comptent vers la limite de taille de rétention, mais Prometheus ne purge jamais les données de ces répertoires en fonction des politiques de conservation basées sur la taille ou le temps. Ainsi, si vous définissez une limite de taille de rétention inférieure à la taille maximale définie pour les répertoires /wal et /head_chunks, vous avez configuré le système pour ne pas conserver de blocs de données dans les répertoires de données /prometheus.
- La politique de conservation basée sur la taille n'est appliquée que lorsque Prometheus coupe un nouveau bloc de données, qui se produit toutes les deux heures après que le WAL contient au moins trois heures de données.
- Lorsque vous ne définissez pas explicitement des valeurs de rétention ou de rétention, le temps de rétention par défaut est de 15 jours pour la surveillance de la plate-forme de base et de 24 heures pour la surveillance de projet définie par l'utilisateur. La taille de rétention n'est pas définie.
- Lorsque vous définissez des valeurs pour la rétention et la taille de rétention, les deux valeurs s'appliquent. Lorsque des blocs de données dépassent le temps de conservation défini ou la limite de taille définie, Prometheus purge ces blocs de données.
- Lorsque vous définissez une valeur pour la taille de rétention et que vous ne définissez pas la rétention, seule la valeur de la taille de rétention s'applique.
- Dans le cas où vous ne définissez pas une valeur pour la taille de la rétention et que vous ne définissez qu'une valeur de rétention, seule la valeur de rétention s'applique.
- Lorsque vous définissez la valeur de rétention ou de rétention sur 0, les paramètres par défaut s'appliquent. Les paramètres par défaut fixent le temps de rétention à 15 jours pour la surveillance de la plate-forme de base et 24 heures pour la surveillance de projet définie par l'utilisateur. La taille de la rétention n'est pas définie par défaut.

NOTE

Le compactage des données se produit toutes les deux heures. Ainsi, un volume persistant (PV) pourrait se remplir avant le compactage, dépassant potentiellement la limite de taille de rétention. Dans de tels cas, l'alerte KubePersistentVolumeFillingUp s'allume jusqu'à ce que l'espace sur un PV soit inférieur à la limite de taille de rétention.

3.7.4. La modification du temps de rétention et de la taille des données métriques de Prometheus

Par défaut, Prometheus conserve les données métriques pendant 24 heures pour la surveillance des projets définis par l'utilisateur. Lorsque les données sont supprimées, vous pouvez modifier le temps de conservation de l'instance Prometheus. De plus, vous pouvez définir la quantité maximale d'espace

disque utilisée par les données métriques conservées.

NOTE

Le compactage des données se produit toutes les deux heures. Ainsi, un volume persistant (PV) pourrait se remplir avant le compactage, dépassant potentiellement la limite de taille de rétention. Dans de tels cas, l'alerte KubePersistentVolumeFillingUp s'allume jusqu'à ce que l'espace sur un PV soit inférieur à la limite de taille de rétention.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajoutez la configuration du temps de rétention et de la taille sous data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      retention: <time_specification> 1
      retentionSize: <size_specification> 2
```

1. Le temps de rétention: un nombre directement suivi de ms (millisecondes), s (secondes), m (minutes), h (heures), d (jours), w (semaines) ou y (années). Il est également possible de combiner des valeurs de temps pour des temps spécifiques, tels que 1h30m15s.

2. La taille de rétention: un nombre directement suivi de B (octets), KB (kilooctets), MB (mégaoctets), GB (gigaoctets), TB (teraoctets), PB (pétaoctets) et EB (exabytes).

L'exemple suivant définit le temps de rétention à 24 heures et la taille de rétention à 10 gigaoctets pour l'instance Prometheus:

Exemple de réglage du temps de rétention pour Prometheus

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
```

```
data:
  config.yaml: |
    prometheus:
      retention: 24h
      retentionSize: 10GB
```

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.

3.7.5. La modification du temps de conservation des données métriques Thanos Ruler

Dans le cas des projets définis par l'utilisateur, Thanos Ruler conserve automatiquement les données métriques pendant 24 heures. Il est possible de modifier le temps de conservation pour modifier la durée de conservation de ces données en spécifiant une valeur de temps dans la carte de configuration de configuration de la charge de travail de l'utilisateur dans l'espace de noms openshift-user-workload-monitoring.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditer l'objet ConfigMap de l'utilisateur-workload-monitoring-config dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajouter la configuration du temps de rétention sous data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    thanosRuler:
      retention: <time_specification> 1
```

1 Indiquez le temps de rétention dans le format suivant: un nombre directement suivi de ms (millisecondes), s (secondes), m (minutes), h (heures), d (jours), w (semaines) ou y (années). Il est également possible de combiner des valeurs de temps pour des temps spécifiques, tels que 1h30m15s. La valeur par défaut est 24h.

L'exemple suivant définit le temps de conservation à 10 jours pour les données Thanos Ruler:

```
apiVersion: v1
```

```
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    thanosRuler:
      retention: 10d
```

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.

Ressources supplémentaires

- Comprendre le stockage persistant

3.8. CONFIGURATION DU STOCKAGE D'ÉCRITURE À DISTANCE

Il est possible de configurer le stockage d'écriture à distance pour permettre à Prometheus d'envoyer des métriques ingérées à des systèmes distants pour un stockage à long terme. Cela n'a aucun impact sur la façon dont ou pendant combien de temps Prometheus stocke les métriques.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.
- Il y a un point de terminaison compatible avec l'écriture à distance (tel que Thanos) et vous connaissez l'URL du point de terminaison. Consultez les points de terminaison distants Prometheus et la documentation de stockage pour obtenir des informations sur les points de terminaison compatibles avec la fonction d'écriture à distance.

IMPORTANT

Le Red Hat fournit uniquement des informations pour la configuration des expéditeurs d'écriture à distance et n'offre pas de conseils sur la configuration des points de terminaison du récepteur. Les clients sont responsables de la mise en place de leurs propres points de terminaison compatibles avec l'écriture à distance. Les problèmes avec les configurations des récepteurs de point de terminaison ne sont pas inclus dans le support de production de Red Hat.

- Dans un objet Secret, vous avez configuré des identifiants d'authentification pour le point de terminaison de l'écriture à distance. Il faut créer le secret dans l'espace de noms openshift-user-workload-monitoring.

AVERTISSEMENT

Afin de réduire les risques de sécurité, utilisez HTTPS et l'authentification pour envoyer des métriques à un point de terminaison.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajouter une section RemoteWrite: sous data/config.yaml/prometheus, comme indiqué dans l'exemple suivant:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://remote-write-endpoint.example.com" 1
        <endpoint_authentication_credentials> 2
```

1. L'URL du point de terminaison de l'écriture distante.

2. La méthode d'authentification et les informations d'identification pour le point de terminaison. Les méthodes d'authentification actuellement prises en charge sont AWS Signature Version 4, l'authentification à l'aide de HTTP dans un en-tête de demande d'autorisation, l'authentification de base, OAuth 2.0 et le client TLS. Consultez les paramètres d'authentification d'écriture à distance pris en charge pour les configurations d'échantillons de méthodes d'authentification prises en charge.

3. Ajouter des valeurs de configuration de relabel d'écriture après les identifiants d'authentification:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://remote-write-endpoint.example.com"
```

```
    <endpoint_authentication_credentials>
    writeRelabelConfigs:
    - <your_write_relabel_configs> 1
```

1 Ajoutez une configuration pour les métriques que vous souhaitez envoyer au point de terminaison distant.

Exemple de transfert d'une seule métrique appelée my_metric

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://remote-write-endpoint.example.com"
        writeRelabelConfigs:
        - sourceLabels: [__name__]
          regex: 'my_metric'
          action: keep
```

Exemple de mesures de transmission appelées my_metric_1 et my_metric_2 dans my_namespace namespace

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://remote-write-endpoint.example.com"
        writeRelabelConfigs:
        - sourceLabels: [__name__,namespace]
          regex: '(my_metric_1|my_metric_2);my_namespace'
          action: keep
```

4. Enregistrez le fichier pour appliquer les modifications. La nouvelle configuration est appliquée automatiquement.

3.8.1. Paramètres d'authentification d'écriture à distance pris en charge

Il est possible d'utiliser différentes méthodes pour s'authentifier avec un point de terminaison d'écriture à distance. Les méthodes d'authentification actuellement prises en charge sont AWS Signature Version 4, l'authentification de base, l'autorisation, OAuth 2.0 et le client TLS. Le tableau suivant fournit des détails sur les méthodes d'authentification prises en charge pour une utilisation avec l'écriture à distance.

La méthode d'authentification	Configurer le champ map	Description
AWS Signature Version 4	**à propos de Sigv4**	Cette méthode utilise l'authentification AWS Signature Version 4 pour signer des requêtes. Cette méthode ne peut pas être utilisée simultanément avec l'autorisation, l'authentification OAuth 2.0 ou l'authentification Basic.
Authentification de base	**Basicauth**	L'authentification de base définit l'en-tête d'autorisation sur chaque demande d'écriture à distance avec le nom d'utilisateur et le mot de passe configurés.
autorisation	**autorisation**	L'autorisation définit l'en-tête d'autorisation sur chaque demande d'écriture à distance à l'aide du jeton configuré.
À propos de OAuth 2.0	**à proximité de Oauth2**	La configuration OAuth 2.0 utilise le type de subvention d'identification client. Le Prometheus obtient un jeton d'accès à partir de tokenUrl avec l'ID client spécifié et le secret client pour accéder au point de terminaison d'écriture distante. Cette méthode ne peut pas être utilisée simultanément avec l'autorisation, l'authentification AWS Signature 4 ou l'authentification Basic.
Client TLS	**à propos de TlsConfig**	La configuration du client TLS spécifie le certificat CA, le certificat client et les informations de fichier clé du client utilisées pour authentifier avec le serveur de point de terminaison d'écriture à distance à l'aide de TLS. La configuration de l'échantillon suppose que vous avez déjà créé un fichier de certificat CA, un fichier de certificat client et un fichier clé client.

3.8.2. Exemple de paramètres d'authentification d'écriture à distance

Les échantillons suivants montrent différents paramètres d'authentification que vous pouvez utiliser

pour vous connecter à un point de terminaison d'écriture à distance. Chaque échantillon montre également comment configurer un objet secret correspondant qui contient des informations d'authentification et d'autres paramètres pertinents. Chaque échantillon configure l'authentification pour une utilisation avec la surveillance des projets définis par l'utilisateur dans l'espace de noms openshift-user-workload-monitoring.

3.8.2.1. Exemple d'authentification YAML pour AWS Signature Version 4

Ce qui suit montre les paramètres d'un secret sigv4 nommé sigv4-credentials dans l'espace de noms openshift-user-workload-monitoring.

```
apiVersion: v1
kind: Secret
metadata:
  name: sigv4-credentials
  namespace: openshift-user-workload-monitoring
stringData:
  accessKey: <AWS_access_key> 1
  secretKey: <AWS_secret_key> 2
type: Opaque
```

1 La clé d'accès AWS API.

2 La clé secrète AWS API.

Ce qui suit montre les paramètres d'authentification d'écriture à distance AWS Signature Version 4 qui utilisent un objet secret nommé sigv4-credentials dans l'espace de noms openshift-user-workload-monitoring:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://authorization.example.com/api/write"
        sigv4:
          region: <AWS_region> 1
          accessKey:
            name: sigv4-credentials 2
            key: accessKey 3
          secretKey:
            name: sigv4-credentials 4
            key: secretKey 5
          profile: <AWS_profile_name> 6
          roleArn: <AWS_role_arn> 7
```

1 La région AWS.

2 4 Le nom de l'objet Secret contenant les identifiants d'accès AWS API.

3 La clé qui contient la clé d'accès AWS API dans l'objet Secret spécifié.

5 La clé qui contient la clé secrète AWS API dans l'objet secret spécifié.

6 Le nom du profil AWS qui est utilisé pour authentifier.

7 L'identifiant unique pour le nom de ressource Amazon (ARN) attribué à votre rôle.

3.8.2.2. Échantillon YAML pour l'authentification de base

Ce qui suit montre des paramètres d'authentification de base pour un objet secret nommé rw-basic-auth dans l'espace de noms openshift-user-workload-monitoring:

```
apiVersion: v1
kind: Secret
metadata:
  name: rw-basic-auth
  namespace: openshift-user-workload-monitoring
stringData:
  user: <basic_username> 1
  password: <basic_password> 2
type: Opaque
```

1 Le nom d'utilisateur.

2 Le mot de passe.

L'échantillon suivant montre une configuration d'écriture à distance de baseAuth qui utilise un objet secret nommé rw-basic-auth dans l'espace de noms openshift-user-workload-monitoring. Il suppose que vous avez déjà configuré des informations d'authentification pour le point de terminaison.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://basicauth.example.com/api/write"
        basicAuth:
          username:
            name: rw-basic-auth 1
            key: user 2
          password:
            name: rw-basic-auth 3
            key: password 4
```

1 3 Le nom de l'objet Secret qui contient les identifiants d'authentification.

2 La clé qui contient le nom d'utilisateur dans l'objet secret spécifié.

4 La clé qui contient le mot de passe dans l'objet secret spécifié.

3.8.2.3. Échantillon YAML pour l'authentification avec un jeton porteur à l'aide d'un objet secret

Ce qui suit montre les paramètres de jeton porteur pour un objet secret nommé rw-porter-auth dans l'espace de noms openshift-user-workload-monitoring:

```
apiVersion: v1
kind: Secret
metadata:
  name: rw-bearer-auth
  namespace: openshift-user-workload-monitoring
stringData:
  token: <authentication_token> 1
type: Opaque
```

1 Le jeton d'authentification.

Ce qui suit montre les paramètres de configuration de jeton de porteur d'échantillons qui utilisent un objet secret nommé rw-bearer-auth dans l'espace de noms openshift-user-workload-monitoring:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    enableUserWorkload: true
    prometheus:
      remoteWrite:
      - url: "https://authorization.example.com/api/write"
        authorization:
          type: Bearer 1
          credentials:
            name: rw-bearer-auth 2
            key: token 3
```

1 Le type d'authentification de la demande. La valeur par défaut est Bearer.

2 Le nom de l'objet Secret qui contient les identifiants d'authentification.

3 La clé qui contient le jeton d'authentification dans l'objet secret spécifié.

3.8.2.4. Échantillon YAML pour l'authentification OAuth 2.0

Ce qui suit montre les paramètres OAuth 2.0 pour un objet secret nommé oauth2-credentials dans l'espace de noms openshift-user-workload-monitoring:

```
apiVersion: v1
```

```
kind: Secret
metadata:
  name: oauth2-credentials
  namespace: openshift-user-workload-monitoring
stringData:
  id: <oauth2_id> 1
  secret: <oauth2_secret> 2
type: Opaque
```

1 L'identifiant Oauth 2.0.

2 Le secret OAuth 2.0.

Ce qui suit montre une configuration d'échantillon d'authentification d'écriture à distance oauth2 qui utilise un objet secret nommé oauth2-credentials dans l'espace de noms openshift-user-workload-monitoring:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://test.example.com/api/write"
        oauth2:
          clientId:
            secret:
              name: oauth2-credentials 1
              key: id 2
          clientSecret:
            name: oauth2-credentials 3
            key: secret 4
          tokenUrl: https://example.com/oauth2/token 5
          scopes: 6
          - <scope_1>
          - <scope_2>
          endpointParams: 7
            param1: <parameter_1>
            param2: <parameter_2>
```

1 3 Le nom de l'objet secret correspondant. A noter que ClientId peut se référer alternativement à un objet ConfigMap, bien que clientSecret doive se référer à un objet secret.

2 4 La clé qui contient les informations d'identification OAuth 2.0 dans l'objet secret spécifié.

5 L'URL utilisée pour récupérer un jeton avec le clientId et clientSecret spécifiés.

6 La portée OAuth 2.0 de la demande d'autorisation. Ces portées limitent les données auxquelles les jetons peuvent accéder.

7 Les paramètres de demande d'autorisation OAuth 2.0 requis pour le serveur d'autorisation.

3.8.2.5. Échantillon YAML pour l'authentification client TLS

Ce qui suit montre les paramètres du client TLS pour un objet secret nommé mtls-grouple dans l'espace de noms openshift-user-workload-monitoring.

```
apiVersion: v1
kind: Secret
metadata:
  name: mtls-bundle
  namespace: openshift-user-workload-monitoring
data:
  ca.crt: <ca_cert> 1
  client.crt: <client_cert> 2
  client.key: <client_key> 3
type: tls
```

1. Le certificat CA dans le conteneur Prometheus avec lequel valider le certificat serveur.
2. Le certificat client pour l'authentification avec le serveur.
3. La clé cliente.

L'échantillon suivant montre une configuration d'authentification d'écriture à distance tlsConfig qui utilise un objet TLS Secret nommé mtls-bundle.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://remote-write-endpoint.example.com"
        tlsConfig:
          ca:
            secret:
              name: mtls-bundle 1
              key: ca.crt 2
          cert:
            secret:
              name: mtls-bundle 3
              key: client.crt 4
          keySecret:
            name: mtls-bundle 5
            key: client.key 6
```

1 3 5 Le nom de l'objet Secret correspondant qui contient les identifiants d'authentification TLS. A noter que ca et cert peuvent faire référence alternativement à un objet ConfigMap, bien que keySecret doive se référer à un objet secret.

2. La clé dans l'objet secret spécifié qui contient le certificat CA pour le point de terminaison.

4 La clé de l'objet Secret spécifié qui contient le certificat client pour le point de terminaison.

6 La clé dans l'objet secret spécifié qui contient le secret de clé client.

3.8.3. Exemple de configuration de file d'attente d'écriture à distance

Il est possible d'utiliser l'objet queueConfig pour l'écriture à distance afin d'ajuster les paramètres de file d'écriture à distance. L'exemple suivant montre les paramètres de file d'attente avec leurs valeurs par défaut pour la surveillance des projets définis par l'utilisateur dans l'espace de noms openshift-user-workload-monitoring.

Exemple de configuration des paramètres d'écriture à distance avec des valeurs par défaut

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://remote-write-endpoint.example.com"
        <endpoint_authentication_credentials>
        queueConfig:
          capacity: 10000 1
          minShards: 1 2
          maxShards: 50 3
          maxSamplesPerSend: 2000 4
          batchSendDeadline: 5s 5
          minBackoff: 30ms 6
          maxBackoff: 5s 7
          retryOnRateLimit: false 8
          sampleAgeLimit: 0s 9
```

1 Le nombre d'échantillons à tampon par fragment avant qu'ils ne soient supprimés de la file d'attente.

2 Le nombre minimum de fragments.

3 Le nombre maximum de fragments.

4 Le nombre maximal d'échantillons par envoi.

5 Le temps maximum pour un échantillon d'attendre en tampon.

6 L'heure initiale d'attendre avant de réessayer une demande échouée. Le temps est doublé pour chaque réessayer jusqu'au temps maxbackoff.

7 Le temps maximum d'attendre avant de réessayer une demande échouée.

8 Définissez ce paramètre sur true pour réessayer une requête après avoir reçu un code d'état 429 à partir du stockage d'écriture à distance.

9 Les échantillons qui sont plus anciens que la limite de l'échantillonAgeLimit sont supprimés de la file d'attente. Lorsque la valeur n'est pas définie ou définie sur 0s, le paramètre est ignoré.

Ressources supplémentaires

- Configuration de points de terminaison compatibles avec l'écriture à distance (document Prometheus)
- Réglage des paramètres d'écriture à distance (document Prometheus)

3.9. AJOUT D'ÉTIQUETTES DE CLUSTER ID AUX MÉTRIQUES

Lorsque vous gérez plusieurs clusters dédiés OpenShift et utilisez la fonction d'écriture à distance pour envoyer des données métriques de ces clusters à un emplacement de stockage externe, vous pouvez ajouter des étiquettes ID de cluster pour identifier les données métriques provenant de différents clusters. Ensuite, vous pouvez interroger ces étiquettes pour identifier le cluster source d'une métrique et distinguer ces données des données de mesures similaires envoyées par d'autres clusters.

De cette façon, si vous gérez de nombreux clusters pour plusieurs clients et envoyez des données métriques à un seul système de stockage centralisé, vous pouvez utiliser des étiquettes d'identification de cluster pour interroger des métriques pour un cluster ou un client particulier.

La création et l'utilisation d'étiquettes de cluster ID comporte trois étapes générales:

- Configuration des paramètres de relabel d'écriture pour le stockage d'écriture à distance.
- Ajout d'étiquettes de cluster ID aux métriques.
- Interroger ces étiquettes pour identifier le cluster source ou le client pour une métrique.

3.9.1. Création d'étiquettes d'ID de cluster pour les métriques

Il est possible de créer des étiquettes ID de cluster pour les métriques en ajoutant les paramètres write_relabel pour le stockage d'écriture à distance dans la carte de configuration de configuration de la charge de travail de l'utilisateur dans l'espace de noms openshift-user-workload-monitoring.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.
- Le stockage d'écriture à distance a été configuré.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Dans la section writeRelabelConfigs: sous data/config.yaml/prometheus/remoteWrite, ajoutez les valeurs de configuration du cluster ID:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://remote-write-endpoint.example.com"
        <endpoint_authentication_credentials>
        writeRelabelConfigs: (1)
          - <relabel_config> (2)
```

(1) Ajoutez une liste de configurations de relabel d'écriture pour les métriques que vous souhaitez envoyer au point de terminaison distant.

(2) De substituer la configuration de l'étiquette aux métriques envoyées au point de terminaison de l'écriture distante.

L'échantillon suivant montre comment transférer une métrique avec le cluster ID cluster_id:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      remoteWrite:
      - url: "https://remote-write-endpoint.example.com"
        writeRelabelConfigs:
        - sourceLabels:
          - __tmp_openshift_cluster_id__ (1)
          targetLabel: cluster_id (2)
          action: replace (3)
```

(1) Le système applique initialement une étiquette source de cluster temporaire nommée __tmp_openshift_cluster_id__. Cette étiquette temporaire est remplacée par le nom de l'étiquette d'identification du cluster que vous spécifiez.

(2) Indiquez le nom de l'étiquette de cluster ID pour les métriques envoyées au stockage d'écriture à distance. Lorsque vous utilisez un nom d'étiquette qui existe déjà pour une métrique, cette valeur est écrasée avec le nom de cette étiquette ID de cluster. Dans le cas du nom de l'étiquette, n'utilisez pas __tmp_openshift_cluster_id__. La dernière étape de relabeling supprime les étiquettes qui utilisent ce nom.

(3) L'action de relabel d'écriture remplace l'étiquette temporaire par l'étiquette cible pour les mesures sortantes. Cette action est la valeur par défaut et est appliquée si aucune action n'est spécifiée.

3. Enregistrez le fichier pour appliquer les modifications. La nouvelle configuration est appliquée automatiquement.

Ressources supplémentaires

- Configuration du stockage d'écriture à distance

3.10. CONTRÔLE DE L'IMPACT DES ATTRIBUTS MÉTRIQUES NON LIÉS DANS LES PROJETS DÉFINIS PAR L'UTILISATEUR

Les développeurs peuvent créer des étiquettes pour définir des attributs pour les métriques sous la forme de paires clé-valeur. Le nombre de paires de valeurs clés potentielles correspond au nombre de valeurs possibles pour un attribut. L'attribut qui a un nombre illimité de valeurs potentielles s'appelle un attribut non lié. A titre d'exemple, un attribut customer_id est non lié parce qu'il a un nombre infini de valeurs possibles.

Chaque paire clé-valeur assignée a une série chronologique unique. L'utilisation de nombreux attributs non liés dans les étiquettes peut entraîner une augmentation exponentielle du nombre de séries chronologiques créées. Cela peut avoir un impact sur les performances de Prometheus et peut consommer beaucoup d'espace disque.

L'administration dédiée peut utiliser les mesures suivantes pour contrôler l'impact des attributs métriques non liés dans les projets définis par l'utilisateur:

- Limiter le nombre d'échantillons pouvant être acceptés par raclette cible dans les projets définis par l'utilisateur
- Limiter le nombre d'étiquettes grattées, la longueur des noms d'étiquettes et la longueur des valeurs de l'étiquette
- Configurez les intervalles entre les éraflures consécutives et entre les évaluations des règles de Prometheus
- Créer des alertes indiquant que le feu lorsqu'un seuil d'échantillonnage de grattage est atteint ou lorsque la cible ne peut pas être grattée

NOTE

Limiter les échantillons de raclette peut aider à prévenir les problèmes causés par l'ajout de nombreux attributs non liés aux étiquettes. Les développeurs peuvent également empêcher la cause sous-jacente en limitant le nombre d'attributs non liés qu'ils définissent pour les métriques. L'utilisation d'attributs liés à un ensemble limité de valeurs possibles réduit le nombre de combinaisons de paires clé-valeur potentielles.

3.10.1. Fixer des intervalles de raclette, des intervalles d'évaluation et des limites imposées pour les projets définis par l'utilisateur

Les limites de grattage et d'étiquette suivantes peuvent être définies pour les projets définis par l'utilisateur:

- Limiter le nombre d'échantillons qui peuvent être acceptés par raclette cible
- Limiter le nombre d'étiquettes grattées
- Limiter la longueur des noms d'étiquettes et des valeurs d'étiquettes

Il est également possible de définir un intervalle entre les éraflures consécutives et entre les évaluations des règles de Prometheus.

AVERTISSEMENT

Lorsque vous définissez des limites d'échantillons ou d'étiquettes, aucune autre donnée d'échantillon n'est ingérée pour cette éraflure cible une fois la limite atteinte.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditer l'objet ConfigMap de l'utilisateur-workload-monitoring-config dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajouter les configurations de limite et d'intervalle de temps appliquées à data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      enforcedSampleLimit: 50000 (1)
      enforcedLabelLimit: 500 (2)
      enforcedLabelNameLengthLimit: 50 (3)
      enforcedLabelValueLengthLimit: 600 (4)
      scrapeInterval: 1m30s (5)
      evaluationInterval: 1m15s (6)
```

(1) La valeur est requise si ce paramètre est spécifié. Cet exemple appliquéSampleLimit limite le nombre d'échantillons qui peuvent être acceptés par éraflure cible dans les projets définis par l'utilisateur à 50 000.

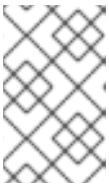
(2) Indique le nombre maximal d'étiquettes par raclette. La valeur par défaut est 0, ce qui ne spécifie aucune limite.

(3) Indique la longueur maximale du caractère d'un nom d'étiquette. La valeur par défaut est 0, ce qui ne spécifie aucune limite.

4 Indique la longueur maximale du caractère pour une valeur d'étiquette. La valeur par défaut est 0, ce qui ne spécifie aucune limite.

5 Indique l'intervalle entre les éraflures consécutives. L'intervalle doit être réglé entre 5 secondes et 5 minutes. La valeur par défaut est 30s.

6 Indique l'intervalle entre les évaluations des règles de Prometheus. L'intervalle doit être réglé entre 5 secondes et 5 minutes. La valeur par défaut pour Prometheus est 30s.



NOTE

Il est également possible de configurer la propriété AssessmentInterval pour Thanos Ruler via le champ data/config.yaml/thanosRuler. La valeur par défaut pour Thanos Ruler est 15s.

3. Enregistrez le fichier pour appliquer les modifications. Les limites sont appliquées automatiquement.

3.11. CONFIGURATION D'INSTANCES EXTERNES ALERTMANAGER

La pile de surveillance OpenShift Dedicated inclut une instance Alertmanager locale qui dirige les alertes de Prometheus.

Il est possible d'ajouter des instances Alertmanager externes pour acheminer les alertes pour les projets définis par l'utilisateur.

Lorsque vous ajoutez la même configuration externe Alertmanager pour plusieurs clusters et que vous désactivez l'instance locale pour chaque cluster, vous pouvez ensuite gérer le routage d'alerte pour plusieurs clusters en utilisant une instance Alertmanager externe unique.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajoutez une section supplémentaireAlertmanagerConfigs avec les détails de configuration sous data/config.yaml/<component>:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
```

```
data:
  config.yaml: |
    <component>: 1
      additionalAlertmanagerConfigs:
      - <alertmanager_specification> 2
```

2 <alertmanager_spécification> par l'authentification et d'autres détails de configuration pour les instances Alertmanager supplémentaires. Les méthodes d'authentification actuellement prises en charge sont le jeton porteur (porteurToken) et le client TLS (tlsConfig).

1 De substituer <composant> à l'un des deux composants de Alertmanager externes pris en charge: prometheus ou thanosRuler.

La carte de configuration de l'échantillon suivant configure un gestionnaire d'alerte supplémentaire pour Thanos Ruler en utilisant un jeton porteur avec l'authentification TLS client:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    thanosRuler:
      additionalAlertmanagerConfigs:
      - scheme: https
        pathPrefix: /
        timeout: "30s"
        apiVersion: v1
        bearerToken:
          name: alertmanager-bearer-token
          key: token
        tlsConfig:
          key:
            name: alertmanager-tls
            key: tls.key
          cert:
            name: alertmanager-tls
            key: tls.crt
          ca:
            name: alertmanager-tls
            key: tls.ca
        staticConfigs:
        - external-alertmanager1-remote.com
        - external-alertmanager1-remote2.com
```

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.

3.12. CONFIGURATION DES SECRETS POUR ALERTMANAGER

La pile de surveillance OpenShift Dedicated inclut Alertmanager, qui dirige les alertes de Prometheus

vers les récepteurs de point de terminaison. Lorsque vous avez besoin d'authentifier avec un récepteur afin que Alertmanager puisse lui envoyer des alertes, vous pouvez configurer Alertmanager pour utiliser un secret qui contient des informations d'authentification pour le récepteur.

À titre d'exemple, vous pouvez configurer Alertmanager pour utiliser un secret pour s'authentifier avec un récepteur de point de terminaison qui nécessite un certificat délivré par une autorité de certification privée (CA). Il est également possible de configurer Alertmanager pour utiliser un secret pour s'authentifier avec un récepteur qui nécessite un fichier mot de passe pour l'authentification HTTP Basic. Dans les deux cas, les détails d'authentification sont contenus dans l'objet Secret plutôt que dans l'objet ConfigMap.

3.12.1. Ajouter un secret à la configuration Alertmanager

Il est possible d'ajouter des secrets à la configuration Alertmanager en modifiant la carte de configuration de configuration utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring.

Après avoir ajouté un secret à la carte de configuration, le secret est monté en volume à /etc/alertmanager/secrets/<secret_name> dans le conteneur Alertmanager pour les pods Alertmanager.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- « vous avez créé le secret à configurer dans Alertmanager dans le projet openshift-user-workload-monitoring.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajouter un secret: section sous data/config.yaml/alertmanager avec la configuration suivante:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    alertmanager:
      secrets: 1
      - <secret_name_1> 2
      - <secret_name_2>
```

1. Cette section contient les secrets à monter dans Alertmanager. Les secrets doivent être situés dans le même espace de noms que l'objet Alertmanager.

2 Le nom de l'objet Secret qui contient des informations d'authentification pour le récepteur. En ajoutant plusieurs secrets, placez chacun sur une nouvelle ligne.

Les paramètres de carte de configuration de l'échantillon suivants configurent Alertmanager pour utiliser deux objets secrets nommés test-secret-basic-auth et test-secret-api-token:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    alertmanager:
      secrets:
      - test-secret-basic-auth
      - test-secret-api-token
```

3. Enregistrez le fichier pour appliquer les modifications. La nouvelle configuration est appliquée automatiquement.

3.13. ATTACHER DES ÉTIQUETTES SUPPLÉMENTAIRES À VOS SÉRIES CHRONOLOGIQUES ET ALERTES

Il est possible d'attacher des étiquettes personnalisées à toutes les séries chronologiques et d'alerter Prometheus en utilisant la fonction d'étiquettes externes de Prometheus.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Définissez les étiquettes que vous souhaitez ajouter pour chaque métrique sous data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
```

```
prometheus:
  externalLabels:
    <key>: <value> 1
```

1 <key>: <value> par des paires clé-valeur où <key> est un nom unique pour la nouvelle étiquette et <value> est sa valeur.

AVERTISSEMENT

- Il ne faut pas utiliser promitheus ou prometheus_replica comme noms clés, parce qu'ils sont réservés et seront écrasés.
- Il ne faut pas utiliser cluster ou Manag_cluster comme noms de clés. Les utiliser peut causer des problèmes où vous ne pouvez pas voir les données dans les tableaux de bord des développeurs.

NOTE

Dans le projet openshift-user-workload-monitoring, Prometheus gère les métriques et Thanos Ruler gère les règles d'alerte et d'enregistrement. La configuration de l'objet ConfigMap pour la configuration de l'objet ConfigMap ne configurera que les étiquettes externes pour les métriques et non pour les règles.

À titre d'exemple, pour ajouter des métadonnées sur la région et l'environnement à toutes les séries chronologiques et aux alertes, utilisez l'exemple suivant:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      externalLabels:
        region: eu
        environment: prod
```

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.

3.14. À PROPOS DES CONTRAINTES DE PROPAGATION DE LA TOPOLOGIE DES POD POUR LA SURVEILLANCE

Il est possible d'utiliser des contraintes de propagation de la topologie des pods pour contrôler comment les gousses de surveillance sont réparties sur une topologie réseau lorsque des gousses dédiées OpenShift sont déployées dans plusieurs zones de disponibilité.

Les contraintes de propagation de la topologie des pod sont appropriées pour contrôler la planification des pods dans les topologies hiérarchiques dans lesquelles les nœuds sont répartis à différents niveaux d'infrastructure, tels que les régions et les zones à l'intérieur de ces régions. De plus, en étant capable de programmer des pods dans différentes zones, vous pouvez améliorer la latence du réseau dans certains scénarios.

Il est possible de configurer les contraintes de propagation de la topologie des pods pour toutes les gousses déployées par l'opérateur de surveillance du cluster afin de contrôler la façon dont les répliques des pod sont programmées en nœuds à travers les zones. Cela garantit que les pods sont très disponibles et fonctionnent plus efficacement, car les charges de travail sont réparties entre les nœuds dans différents centres de données ou zones d'infrastructure hiérarchiques.

Ressources supplémentaires

- Kubernetes Pod Topology Spread Contraints documentation

3.14.1. Configuration des contraintes de propagation de la topologie des pod

Il est possible de configurer les contraintes de propagation de la topologie des pods pour toutes les gousses pour une surveillance définie par l'utilisateur afin de contrôler la façon dont les répliques des pod sont programmées pour les nœuds à travers les zones. Cela garantit que les pods sont très disponibles et fonctionnent plus efficacement, car les charges de travail sont réparties entre les nœuds dans différents centres de données ou zones d'infrastructure hiérarchiques.

Il est possible de configurer les contraintes de propagation de la topologie pod pour la surveillance des pods à l'aide de la carte de configuration de configuration de la charge de travail-surveillance de l'utilisateur.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajoutez les paramètres suivants dans le champ data/config.yaml pour configurer les contraintes de propagation de la topologie pod:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    <component>: 1
      topologySpreadConstraints:
```

```
    - maxSkew: <n> 2
      topologyKey: <key> 3
      whenUnsatisfiable: <value> 4
      labelSelector: 5
        <match_option>
```

1. Indiquez un nom du composant pour lequel vous souhaitez configurer des contraintes de propagation de la topologie de pod.
2. Indiquez une valeur numérique pour maxSkew, qui définit le degré auquel les pods sont autorisés à être distribués de manière inégale.
3. Indiquez une clé des étiquettes des nœuds pour topologyKey. Les nœuds qui ont une étiquette avec cette clé et des valeurs identiques sont considérés comme étant dans la même topologie. Le planificateur essaie de mettre un nombre équilibré de pods dans chaque domaine.
4. Indiquez une valeur pour quand Insatisfiable. Les options disponibles sont DoNotSchedule et ScheduleAnyway. Indiquez DoNotSchedule si vous voulez que la valeur maxSkew définisse la différence maximale autorisée entre le nombre de gousses correspondantes dans la topologie cible et le minimum global. Spécifiez ScheduleAnyway si vous voulez que le programmeur planifie toujours le pod, mais donne une priorité plus élevée aux nœuds qui pourraient réduire l'oscillation.
5. Indiquez le labelSelector pour trouver les gousses correspondantes. Les pods qui correspondent à ce sélecteur d'étiquettes sont comptés pour déterminer le nombre de pods dans leur domaine de topologie correspondant.

Exemple de configuration pour Thanos Ruler

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    thanosRuler:
      topologySpreadConstraints:
      - maxSkew: 1
        topologyKey: monitoring
        whenUnsatisfiable: ScheduleAnyway
        labelSelector:
          matchLabels:
            app.kubernetes.io/name: thanos-ruler
```

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.

3.15. DÉFINITION DES NIVEAUX DE LOG POUR LES COMPOSANTS DE SURVEILLANCE

Le niveau de journal peut être configuré pour Alertmanager, Prometheus Operator, Prometheus et Thanos Ruler.

Les niveaux de journal suivants peuvent être appliqués au composant pertinent de l'objet ConfigMap:

- déboguer. Débogue de journal, messages d'information, d'avertissement et d'erreur.
- info. Logez les messages d'information, d'avertissement et d'erreur.
- avertissez-vous. Enregistrez uniquement les messages d'avertissement et d'erreur.
- erreur. Journal des messages d'erreur seulement.

Le niveau de journal par défaut est info.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajouter logLevel: <log_level> pour un composant sous data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    <component>: 1
      logLevel: <log_level> 2
```

1. Le composant de la pile de surveillance pour lequel vous définissez un niveau de journal. Les valeurs des composants disponibles sont les promémorations, le gestionnaire d'alerte, le prométheusOperator et le thanosRuler.
2. Le niveau de journal à définir pour le composant. Les valeurs disponibles sont l'erreur, l'avertissement, l'information et le débogue. La valeur par défaut est info.

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.
4. Confirmez que le niveau de journal a été appliqué en examinant la configuration de déploiement ou de pod dans le projet connexe. L'exemple suivant vérifie le niveau de journal pour le déploiement protérébeus-operator:

```
$ oc -n openshift-user-workload-monitoring get deploy prometheus-operator -o yaml | grep
"log-level"
```

Exemple de sortie

```
        - --log-level=debug
```

5. Assurez-vous que les pods du composant sont en cours d'exécution. L'exemple suivant répertorie l'état des pods:

```
$ oc -n openshift-user-workload-monitoring get pods
```

NOTE

Lorsqu'une valeur logLevel non reconnue est incluse dans l'objet ConfigMap, les pods du composant peuvent ne pas redémarrer avec succès.

3.16. ACTIVER LE FICHIER JOURNAL DE REQUÊTE POUR PROMETHEUS

Configurez Prometheus pour écrire toutes les requêtes qui ont été exécutées par le moteur dans un fichier journal.

IMPORTANT

Comme la rotation des journaux n'est pas prise en charge, activez cette fonctionnalité temporairement lorsque vous devez résoudre un problème. Après avoir terminé le dépannage, désactivez l'enregistrement des requêtes en retournant les modifications que vous avez apportées à l'objet ConfigMap pour activer la fonctionnalité.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajouter le paramètre QueryLogFile pour Prometheus sous data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
```

```
data:
  config.yaml: |
    prometheus:
      queryLogFile: <path> 1
```

1 Ajoutez le chemin complet au fichier dans lequel les requêtes seront enregistrées.

3. Enregistrez le fichier pour appliquer les modifications. Les pods affectés par la nouvelle configuration sont automatiquement redéployés.

4. Assurez-vous que les pods pour le composant sont en cours d'exécution. La commande d'échantillon suivante répertorie l'état des pods:

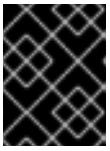
```
$ oc -n openshift-user-workload-monitoring get pods
```

Exemple de sortie

```
...
prometheus-operator-776fcbbd56-2nbfm   2/2     Running   0          132m
prometheus-user-workload-0             5/5     Running   1          132m
prometheus-user-workload-1             5/5     Running   1          132m
thanos-ruler-user-workload-0           3/3     Running   0          132m
thanos-ruler-user-workload-1           3/3     Running   0          132m
...
```

5. Lisez le journal des requêtes:

```
$ oc -n openshift-user-workload-monitoring exec prometheus-user-workload-0 -- cat <path>
```



IMPORTANT

Inversez le paramètre dans la carte de configuration après avoir examiné les informations de requête enregistrée.

CHAPITRE 4. DÉSACTIVATION DU SUIVI DES PROJETS DÉFINIS PAR L'UTILISATEUR

En tant qu'administrateur dédié, vous pouvez désactiver la surveillance des projets définis par l'utilisateur. Il est également possible d'exclure les projets individuels de la surveillance de la charge de travail des utilisateurs.

4.1. DÉSACTIVATION DU SUIVI DES PROJETS DÉFINIS PAR L'UTILISATEUR

La surveillance des projets définis par l'utilisateur est activée par défaut. Lorsque vous ne souhaitez pas utiliser la pile de surveillance intégrée pour surveiller les projets définis par l'utilisateur, vous pouvez la désactiver.

Conditions préalables

- Connectez-vous à OpenShift Cluster Manager.

Procédure

1. Dans la console de cloud hybride OpenShift Cluster Manager, sélectionnez un cluster.
2. Cliquez sur l'onglet Paramètres.
3. Cliquez sur la case à cocher Activer la surveillance de la charge de travail de l'utilisateur pour désélectionner l'option, puis cliquez sur Enregistrer.
   La surveillance de la charge de travail des utilisateurs est désactivée. Les composants Prometheus, Prometheus Operator et Thanos Ruler sont arrêtés dans le projet de surveillance de la charge de travail de l'utilisateur.

4.2. EXCLURE UN PROJET DÉFINI PAR L'UTILISATEUR DE LA SURVEILLANCE

Les projets individuels définis par l'utilisateur peuvent être exclus de la surveillance de la charge de travail des utilisateurs. À cette fin, ajoutez l'étiquette openshift.io/user-monitoring à l'espace de noms du projet avec une valeur de false.

Procédure

1. Ajouter l'étiquette à l'espace de noms du projet:

```
$ oc label namespace my-project 'openshift.io/user-monitoring=false'
```

2. Afin de réactiver la surveillance, retirez l'étiquette de l'espace de noms:

```
$ oc label namespace my-project 'openshift.io/user-monitoring-'
```

NOTE

Lorsqu'il y avait des cibles de surveillance active pour le projet, il peut prendre quelques minutes pour que Prométhée cesse de les gratter après avoir ajouté l'étiquette.

CHAPITRE 5. ACTIVER LE ROUTAGE D'ALERTE POUR LES PROJETS DÉFINIS PAR L'UTILISATEUR

Dans OpenShift Dedicated, un administrateur peut activer le routage d'alerte pour les projets définis par l'utilisateur. Ce processus comprend les étapes suivantes:

- Activer le routage d'alerte pour les projets définis par l'utilisateur pour utiliser une instance Alertmanager séparée.
- Autoriser les utilisateurs à configurer le routage d'alerte pour les projets définis par l'utilisateur.

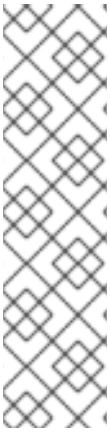
Après avoir terminé ces étapes, les développeurs et les autres utilisateurs peuvent configurer des alertes personnalisées et un routage d'alerte pour leurs projets définis par l'utilisateur.

5.1. COMPRENDRE LE ROUTAGE D'ALERTE POUR LES PROJETS DÉFINIS PAR L'UTILISATEUR

En tant qu'administrateur dédié, vous pouvez activer le routage d'alerte pour les projets définis par l'utilisateur. Avec cette fonctionnalité, vous pouvez permettre aux utilisateurs avec le rôle de cluster de routage d'alerte-édition de configurer le routage de notification d'alerte et les récepteurs pour les projets définis par l'utilisateur. Ces notifications sont acheminées par une instance Alertmanager dédiée à la surveillance définie par l'utilisateur.

Les utilisateurs peuvent ensuite créer et configurer le routage d'alerte défini par l'utilisateur en créant ou en modifiant les objets AlertmanagerConfig pour leurs projets définis par l'utilisateur sans l'aide d'un administrateur.

Après qu'un utilisateur a défini le routage d'alerte pour un projet défini par l'utilisateur, les notifications d'alerte définies par l'utilisateur sont acheminées vers les pods alertmanager-user-workload-load dans l'espace de noms openshift-user-workload-monitoring.



NOTE

Examiner les limites suivantes du routage d'alerte pour les projets définis par l'utilisateur:

- Dans le cas des règles d'alerte définies par l'utilisateur, le routage défini par l'utilisateur est étendu à l'espace de noms dans lequel la ressource est définie. À titre d'exemple, une configuration de routage dans namespace ns1 s'applique uniquement aux ressources PrometheusRules dans le même espace de noms.
- Lorsqu'un espace de noms est exclu de la surveillance définie par l'utilisateur, les ressources AlertmanagerConfig dans l'espace de noms cessent de faire partie de la configuration Alertmanager.

5.2. ACTIVER UNE INSTANCE D'ALERTE SÉPARÉE POUR LE ROUTAGE D'ALERTE DÉFINI PAR L'UTILISATEUR

Dans OpenShift Dedicated, vous voudrez peut-être déployer une instance Alertmanager dédiée pour les projets définis par l'utilisateur, qui fournit des alertes définies par l'utilisateur séparément des alertes de plate-forme par défaut. Dans ces cas, vous pouvez optionnellement activer une instance distincte de Alertmanager pour envoyer des alertes uniquement pour les projets définis par l'utilisateur.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditer l'objet ConfigMap:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Ajouter activé: true et activerAlertmanagerConfig: true dans la section Alertmanager sous data/config.yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    alertmanager:
      enabled: true 1
      enableAlertmanagerConfig: true 2
```

1. Définissez la valeur activée à true pour activer une instance dédiée du gestionnaire d'alerte pour les projets définis par l'utilisateur dans un cluster. Définissez la valeur pour fausser ou omettre complètement la clé pour désactiver le gestionnaire d'alerte pour les projets définis par l'utilisateur. Lorsque vous définissez cette valeur sur false ou si la clé est omise, les alertes définies par l'utilisateur sont acheminées vers l'instance Alertmanager de plate-forme par défaut.

2. Définissez la valeur enableAlertmanagerConfig à true pour permettre aux utilisateurs de définir leurs propres configurations de routage d'alerte avec des objets AlertmanagerConfig.

3. Enregistrez le fichier pour appliquer les modifications. L'instance dédiée de Alertmanager pour les projets définis par l'utilisateur commence automatiquement.

La vérification

- Assurez-vous que les pods d'alerte-manager-utilisateur-téléchargement sont en cours d'exécution:

```
# oc -n openshift-user-workload-monitoring get pods
```

Exemple de sortie

```
NAME                                 READY   STATUS    RESTARTS   AGE
alertmanager-user-workload-0         6/6     Running   0          38s
alertmanager-user-workload-1         6/6     Running   0          38s
...
```

5.3. AUTORISER LES UTILISATEURS À CONFIGURER LE ROUTAGE D'ALERTE POUR LES PROJETS DÉFINIS PAR L'UTILISATEUR

Les utilisateurs peuvent autoriser les utilisateurs à configurer le routage d'alerte pour les projets définis par l'utilisateur.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- Le compte utilisateur auquel vous attribuez le rôle existe déjà.
- L'OpenShift CLI (oc) a été installé.

Procédure

- Assigner le rôle de cluster de routage-alerte à un utilisateur dans le projet défini par l'utilisateur:

```
$ oc -n <namespace> adm policy add-role-to-user alert-routing-edit <user>
```
(1)

(1) Dans <namespace>, remplacer l'espace de noms pour le projet défini par l'utilisateur, tel que ns1. Dans <user>, remplacez le nom d'utilisateur pour le compte auquel vous souhaitez attribuer le rôle.

Ressources supplémentaires

- Configuration du routage d'alerte pour les projets définis par l'utilisateur

CHAPITRE 6. GESTION DES MÉTRIQUES

Il est possible de collecter des métriques pour surveiller la performance des composants du cluster et de vos propres charges de travail.

6.1. COMPRENDRE LES MÉTRIQUES

Dans OpenShift Dedicated, les composants de cluster sont surveillés en grattant les métriques exposées par des points de terminaison de service. Il est également possible de configurer la collecte de métriques pour les projets définis par l'utilisateur. Les métriques vous permettent de surveiller la performance des composants du cluster et de vos propres charges de travail.

Il est possible de définir les métriques que vous souhaitez fournir pour vos propres charges de travail en utilisant les bibliothèques clientes Prometheus au niveau de l'application.

Dans OpenShift Dedicated, les métriques sont exposées à travers un point d'extrémité de service HTTP sous le nom canonique /metrics. Il est possible de répertorier toutes les métriques disponibles pour un service en exécutant une requête curl contre http://<endpoint>/metrics. À titre d'exemple, vous pouvez, par exemple, exposer un itinéraire à l'application d'exemple de prometheus-exemple-app, puis exécuter ce qui suit pour afficher toutes ses métriques disponibles:

```
$ curl http://<example_app_endpoint>/metrics
```

Exemple de sortie

```
# HELP http_requests_total Count of all HTTP requests
# TYPE http_requests_total counter
http_requests_total{code="200",method="get"} 4
http_requests_total{code="404",method="get"} 2
# HELP version Version information about this binary
# TYPE version gauge
version{version="v0.1.0"} 1
```

Ressources supplémentaires

- Documentation de la bibliothèque client Prometheus

6.2. CONFIGURATION DE LA COLLECTION DE MÉTRIQUES POUR LES PROJETS DÉFINIS PAR L'UTILISATEUR

Dans un projet défini par l'utilisateur, vous pouvez créer une ressource ServiceMonitor pour gratter des métriques à partir d'un point de terminaison de service. Cela suppose que votre application utilise une bibliothèque cliente Prometheus pour exposer les métriques au nom canonique /metrics.

Cette section décrit comment déployer un exemple de service dans un projet défini par l'utilisateur, puis créer une ressource ServiceMonitor qui définit la façon dont ce service doit être surveillé.

6.2.1. Déploiement d'un service d'échantillonnage

Afin de tester la surveillance d'un service dans un projet défini par l'utilisateur, vous pouvez déployer un service d'échantillonnage.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle cluster-admin ou en tant qu'utilisateur avec des autorisations administratives pour l'espace de noms.

Procédure

1. Créez un fichier YAML pour la configuration du service. Dans cet exemple, il est appelé promesseheus-example-app.yaml.

2. Ajoutez les détails de configuration de déploiement et de service suivants au fichier:

```
apiVersion: v1
kind: Namespace
metadata:
  name: ns1
---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: prometheus-example-app
  name: prometheus-example-app
  namespace: ns1
spec:
  replicas: 1
  selector:
    matchLabels:
      app: prometheus-example-app
  template:
    metadata:
      labels:
        app: prometheus-example-app
    spec:
      containers:
      - image: ghcr.io/rhobs/prometheus-example-app:0.4.2
        imagePullPolicy: IfNotPresent
        name: prometheus-example-app
---
apiVersion: v1
kind: Service
metadata:
  labels:
    app: prometheus-example-app
  name: prometheus-example-app
  namespace: ns1
spec:
  ports:
  - port: 8080
    protocol: TCP
    targetPort: 8080
    name: web
  selector:
    app: prometheus-example-app
  type: ClusterIP
```

Cette configuration déploie un service nommé prometheus-example-app dans le projet ns1 défini par l'utilisateur. Ce service expose la métrique de version personnalisée.

3. Appliquer la configuration au cluster:

```
$ oc apply -f prometheus-example-app.yaml
```

Il faut du temps pour déployer le service.

4. Il est possible de vérifier que le pod est en cours d'exécution:

```
$ oc -n ns1 get pod
```

Exemple de sortie

```
NAME                                  READY   STATUS    RESTARTS   AGE
prometheus-example-app-7857545cb7-sbgwq   1/1     Running   0          81m
```

6.2.2. Indiquer comment un service est surveillé

Afin d'utiliser les métriques exposées par votre service, vous devez configurer la surveillance OpenShift Dedicated pour gratter les métriques du point de terminaison /metrics. Il est possible de le faire à l'aide d'une définition de ressource personnalisée (CRD) de ServiceMonitor qui spécifie la façon dont un service doit être surveillé, ou un CRD PodMonitor qui spécifie comment un pod doit être surveillé. Le premier nécessite un objet Service, tandis que le second ne le fait pas, permettant à Prometheus de gratter directement les métriques à partir du point de terminaison métrique exposé par un pod.

Cette procédure vous montre comment créer une ressource ServiceMonitor pour un service dans un projet défini par l'utilisateur.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle dédié-admin ou le rôle de suivi-édition.
- Dans cet exemple, vous avez déployé le service d'échantillons d'applications de promesseheus-example dans le projet ns1.

NOTE

Le service d'échantillon d'applications prometheus-example ne prend pas en charge l'authentification TLS.

Procédure

1. Créez un nouveau fichier de configuration YAML nommé example-app-service-monitor.yaml.
2. Ajouter une ressource ServiceMonitor au fichier YAML. L'exemple suivant crée un moniteur de service nommé prometheus-example-monitor pour gratter les métriques exposées par le service prometheus-example-app dans l'espace de noms ns1:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
```

```
  name: prometheus-example-monitor
  namespace: ns1 (1)
spec:
  endpoints:
  - interval: 30s
    port: web (2)
    scheme: http
  selector: (3)
    matchLabels:
      app: prometheus-example-app
```

(1) Indiquez un espace de noms défini par l'utilisateur où votre service s'exécute.

(2) Indiquez les ports de point de terminaison à gratter par Prometheus.

(3) Configurez un sélecteur pour correspondre à votre service en fonction de ses étiquettes de métadonnées.

NOTE

La ressource ServiceMonitor dans un espace de noms défini par l'utilisateur ne peut découvrir que des services dans le même espace de noms. Autrement dit, le champ namespaceSelector de la ressource ServiceMonitor est toujours ignoré.

3. Appliquer la configuration au cluster:

```
$ oc apply -f example-app-service-monitor.yaml
```

Il faut du temps pour déployer la ressource ServiceMonitor.

4. Assurez-vous que la ressource ServiceMonitor est en cours d'exécution:

```
$ oc -n <namespace> get servicemonitor
```

Exemple de sortie

```
NAME                         AGE
prometheus-example-monitor   81m
```

6.2.3. Exemples de paramètres d'authentification du point de terminaison du service

En utilisant les définitions de ressources personnalisées (CRD) de ServiceMonitor et PodMonitor, vous pouvez configurer l'authentification des points de terminaison de service pour la surveillance des projets définis par l'utilisateur.

Les échantillons suivants montrent différents paramètres d'authentification pour une ressource ServiceMonitor. Chaque échantillon montre comment configurer un objet secret correspondant qui contient des informations d'authentification et d'autres paramètres pertinents.

6.2.3.1. Exemple d'authentification YAML avec un jeton porteur

L'échantillon suivant montre les paramètres de jetons au porteur pour un objet secret nommé example-porter-auth dans l'espace de noms ns1:

Exemple de jeton de porteur secret

```
apiVersion: v1
kind: Secret
metadata:
  name: example-bearer-auth
  namespace: ns1
stringData:
  token: <authentication_token> 1
```

1 Indiquez un jeton d'authentification.

L'échantillon suivant montre les paramètres d'authentification de jetons au porteur pour un CRD ServiceMonitor. L'exemple utilise un objet secret nommé example-porter-auth:

Exemple de paramètres d'authentification de jetons au porteur

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: prometheus-example-monitor
  namespace: ns1
spec:
  endpoints:
  - authorization:
      credentials:
        key: token 1
        name: example-bearer-auth 2
    port: web
  selector:
    matchLabels:
      app: prometheus-example-app
```

1 La clé qui contient le jeton d'authentification dans l'objet secret spécifié.

2 Le nom de l'objet Secret qui contient les identifiants d'authentification.

IMPORTANT

Il ne faut pas utiliser BearerTokenFile pour configurer le jeton porteur. Lorsque vous utilisez la configuration BearerTokenFile, la ressource ServiceMonitor est rejetée.

6.2.3.2. Échantillon YAML pour l'authentification de base

L'échantillon suivant montre les paramètres d'authentification de base d'un objet secret nommé example-basic-auth dans l'espace de noms ns1:

Exemple de secret d'authentification de base

```
apiVersion: v1
kind: Secret
metadata:
  name: example-basic-auth
  namespace: ns1
stringData:
  user: <basic_username> 1
  password: <basic_password> 2
```

1 Indiquez un nom d'utilisateur pour l'authentification.

2 Indiquez un mot de passe pour l'authentification.

L'échantillon suivant montre les paramètres d'authentification de base pour un CRD ServiceMonitor. L'exemple utilise un objet secret nommé example-basic-auth:

Exemple de paramètres d'authentification de base

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: prometheus-example-monitor
  namespace: ns1
spec:
  endpoints:
  - basicAuth:
      username:
        key: user 1
        name: example-basic-auth 2
      password:
        key: password 3
        name: example-basic-auth 4
    port: web
  selector:
    matchLabels:
      app: prometheus-example-app
```

1 La clé qui contient le nom d'utilisateur dans l'objet secret spécifié.

2 4 Le nom de l'objet Secret qui contient l'authentification de base.

3 La clé qui contient le mot de passe dans l'objet secret spécifié.

6.2.3.3. Exemple d'authentification YAML avec OAuth 2.0

L'échantillon suivant montre les paramètres OAuth 2.0 pour un objet secret nommé example-oauth2 dans l'espace de noms ns1:

Exemple OAuth 2.0 secret

```
apiVersion: v1
kind: Secret
```

```
metadata:
  name: example-oauth2
  namespace: ns1
stringData:
  id: <oauth2_id> 1
  secret: <oauth2_secret> 2
```

1 Indiquez un identifiant Oauth 2.0.

2 Indiquez un secret Oauth 2.0.

L'échantillon suivant montre les paramètres d'authentification OAuth 2.0 pour un CRD ServiceMonitor. L'exemple utilise un objet secret nommé example-oauth2:

Exemple de paramètres d'authentification OAuth 2.0

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: prometheus-example-monitor
  namespace: ns1
spec:
  endpoints:
  - oauth2:
      clientId:
        secret:
          key: id 1
          name: example-oauth2 2
      clientSecret:
        key: secret 3
        name: example-oauth2 4
      tokenUrl: https://example.com/oauth2/token 5
    port: web
  selector:
    matchLabels:
      app: prometheus-example-app
```

1 La clé qui contient l'identifiant OAuth 2.0 dans l'objet secret spécifié.

2 4 Le nom de l'objet secret qui contient les informations d'identification OAuth 2.0.

3 La clé qui contient le secret OAuth 2.0 dans l'objet secret spécifié.

5 L'URL utilisée pour récupérer un jeton avec le clientId et clientSecret spécifiés.

Ressources supplémentaires

- Comment gratter les métriques en utilisant TLS dans une configuration ServiceMonitor dans un projet défini par l'utilisateur

6.3. INTERROGER DES MÉTRIQUES POUR TOUS LES PROJETS AVEC LA CONSOLE WEB DÉDIÉE OPENSHIFT

Le navigateur de requêtes OpenShift Dedicated métriques vous permet d'exécuter des requêtes Prometheus Query Language (PromQL) pour examiner les métriques visualisées sur une parcelle. Cette fonctionnalité fournit des informations sur l'état d'un cluster et toutes les charges de travail définies par l'utilisateur que vous surveillez.

En tant qu'administrateur dédié ou en tant qu'utilisateur avec des autorisations de vision pour tous les projets, vous pouvez accéder à des métriques pour tous les projets par défaut OpenShift dédiés et définis par l'utilisateur dans l'interface utilisateur de Metrics.

NOTE

Les administrateurs dédiés seuls ont accès aux interfaces utilisateur tierces fournies par OpenShift Dedicated monitoring.

L'interface utilisateur Metrics inclut des requêtes prédéfinies, par exemple, CPU, mémoire, bande passante ou paquet réseau pour tous les projets. Il est également possible d'exécuter des requêtes Prometheus Query Language (PromQL) personnalisées.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié ou avec des autorisations de vision pour tous les projets.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Dans la perspective administrateur de la console Web dédiée OpenShift, cliquez sur Observer et accédez à l'onglet Metrics.
2. Afin d'ajouter une ou plusieurs requêtes, effectuez l'une des actions suivantes:

L'option	Description
Choisissez une requête existante.	Dans la liste déroulante Sélectionner la requête, sélectionnez une requête existante.
Créez une requête personnalisée.	Ajoutez votre requête Prometheus Query Language (PromQL) dans le champ Expression. Lorsque vous tapez une expression PromQL, les suggestions autocomplètes apparaissent dans une liste déroulante. Ces suggestions incluent des fonctions, des métriques, des étiquettes et des jetons de temps. Appuyez sur les flèches du clavier pour sélectionner l'un de ces éléments suggérés, puis appuyez sur Entrée pour ajouter l'élément à votre expression. Déplacez votre pointeur de souris sur un élément suggéré pour afficher une brève description de cet élément.
Ajoutez plusieurs requêtes.	Cliquez sur Ajouter une requête.

L'option	Description
Dupliquer une requête existante.	Cliquez sur le menu des options à côté de la requête, puis choisissez la requête duplicate.
Désactivez une requête d'être exécutée.	Cliquez sur le menu des options à côté de la requête et choisissez Désactiver la requête.

3. Cliquez sur Exécuter des requêtes pour exécuter les requêtes que vous avez créées. Les métriques des requêtes sont visualisées sur l'intrigue. En cas d'invalidation d'une requête, l'interface utilisateur affiche un message d'erreur.

NOTE

- Lorsque vous dessinez des graphiques de séries chronologiques, les requêtes qui fonctionnent sur de grandes quantités de données peuvent sortir ou surcharger le navigateur. Afin d'éviter cela, cliquez sur Masquer le graphique et calibrez votre requête en utilisant uniquement la table des métriques. Ensuite, après avoir trouvé une requête réalisable, activez l'intrigue pour dessiner les graphiques.
- Le tableau de requête affiche par défaut une vue élargie qui répertorie chaque métrique et sa valeur actuelle. Cliquez sur la flèche vers le bas pour minimiser la vue étendue d'une requête.

4. Facultatif: Enregistrez l'URL de la page pour utiliser à nouveau cet ensemble de requêtes à l'avenir.

5. Explorez les métriques visualisées. Initialement, toutes les métriques de toutes les requêtes activées sont affichées sur l'intrigue. Choisissez les métriques affichées en effectuant l'une des actions suivantes:

L'option	Description
Cachez toutes les métriques d'une requête.	Cliquez sur le menu des options pour la requête et cliquez sur Masquer toutes les séries.
Cacher une métrique spécifique.	Allez dans la table de requête et cliquez sur le carré coloré près du nom de la métrique.
Zoomez dans l'intrigue et modifiez la plage de temps.	Effectuez l'une des actions suivantes: • Choisissez visuellement la plage de temps en cliquant et en faisant glisser sur le tracé horizontalement. • Dans le menu, sélectionnez l'intervalle de temps.
De réinitialiser la plage de temps.	Cliquez sur Réinitialiser le zoom.

L'option	Description
Afficher les sorties pour toutes les requêtes à un moment précis.	Flottez au-dessus de l'intrigue au point qui vous intéresse. Les sorties de requête apparaissent dans une fenêtre contextuelle.
Cachez l'intrigue.	Cliquez sur Masquer le graphique.

Ressources supplémentaires

- Documentation de requête Prometheus

6.4. INTERROGER DES MÉTRIQUES POUR DES PROJETS DÉFINIS PAR L'UTILISATEUR AVEC LA CONSOLE WEB DÉDIÉE OPENSHIFT

Le navigateur de requêtes OpenShift Dedicated métriques vous permet d'exécuter des requêtes Prometheus Query Language (PromQL) pour examiner les métriques visualisées sur une parcelle. Cette fonctionnalité fournit des informations sur toutes les charges de travail définies par l'utilisateur que vous surveillez.

En tant que développeur, vous devez spécifier un nom de projet lors de l'interrogation de mesures. Il faut avoir les privilèges requis pour afficher les métriques du projet sélectionné.

L'interface utilisateur Metrics inclut des requêtes prédéfinies, par exemple CPU, mémoire, bande passante ou paquet réseau. Ces requêtes sont limitées au projet sélectionné. Il est également possible d'exécuter des requêtes Prometheus Query Language (PromQL) personnalisées pour le projet.

NOTE

Les développeurs ne peuvent utiliser que la perspective Développeur et non la perspective Administrateur. En tant que développeur, vous ne pouvez interroger que des métriques pour un projet à la fois. Les développeurs ne peuvent pas accéder aux interfaces utilisateur tierces fournies avec OpenShift Dedicated monitoring.

Conditions préalables

- En tant que développeur ou en tant qu'utilisateur, vous avez accès au cluster avec des autorisations de vision pour le projet pour lequel vous consultez des métriques.
- La surveillance des projets définis par l'utilisateur a été activée.
- Dans un projet défini par l'utilisateur, vous avez déployé un service.
- La définition de ressource personnalisée (CRD) de ServiceMonitor permet au service de définir la façon dont le service est surveillé.

Procédure

1. Dans la perspective Développeur de la console Web dédiée OpenShift, cliquez sur Observer et accédez à l'onglet Metrics.

2. Choisissez le projet pour lequel vous souhaitez afficher les métriques de la liste Project:.

3. Afin d'ajouter une ou plusieurs requêtes, effectuez l'une des actions suivantes:

L'option	Description
Choisissez une requête existante.	Dans la liste déroulante Sélectionner la requête, sélectionnez une requête existante.
Créez une requête personnalisée.	Ajoutez votre requête Prometheus Query Language (PromQL) dans le champ Expression. Lorsque vous tapez une expression PromQL, les suggestions autocomplètes apparaissent dans une liste déroulante. Ces suggestions incluent des fonctions, des métriques, des étiquettes et des jetons de temps. Appuyez sur les flèches du clavier pour sélectionner l'un de ces éléments suggérés, puis appuyez sur Entrée pour ajouter l'élément à votre expression. Déplacez votre pointeur de souris sur un élément suggéré pour afficher une brève description de cet élément.
Ajoutez plusieurs requêtes.	Cliquez sur Ajouter une requête.
Dupliquer une requête existante.	Cliquez sur le menu des options à côté de la requête, puis choisissez la requête duplicate.
Désactivez une requête d'être exécutée.	Cliquez sur le menu des options à côté de la requête et choisissez Désactiver la requête.

4. Cliquez sur Exécuter des requêtes pour exécuter les requêtes que vous avez créées. Les métriques des requêtes sont visualisées sur l'intrigue. En cas d'invalidation d'une requête, l'interface utilisateur affiche un message d'erreur.

NOTE

- Lorsque vous dessinez des graphiques de séries chronologiques, les requêtes qui fonctionnent sur de grandes quantités de données peuvent sortir ou surcharger le navigateur. Afin d'éviter cela, cliquez sur Masquer le graphique et calibrez votre requête en utilisant uniquement la table des métriques. Ensuite, après avoir trouvé une requête réalisable, activez l'intrigue pour dessiner les graphiques.

- Le tableau de requête affiche par défaut une vue élargie qui répertorie chaque métrique et sa valeur actuelle. Cliquez sur la flèche vers le bas pour minimiser la vue étendue d'une requête.

5. Facultatif: Enregistrez l'URL de la page pour utiliser à nouveau cet ensemble de requêtes à l'avenir.

6. Explorez les métriques visualisées. Initialement, toutes les métriques de toutes les requêtes activées sont affichées sur l'intrigue. Choisissez les métriques affichées en effectuant l'une des actions suivantes:

L'option	Description
Cachez toutes les métriques d'une requête.	Cliquez sur le menu des options pour la requête et cliquez sur Masquer toutes les séries.
Cacher une métrique spécifique.	Allez dans la table de requête et cliquez sur le carré coloré près du nom de la métrique.
Zoomez dans l'intrigue et modifiez la plage de temps.	Effectuez l'une des actions suivantes: ● Choisissez visuellement la plage de temps en cliquant et en faisant glisser sur le tracé horizontalement. ● Dans le menu, sélectionnez l'intervalle de temps.
De réinitialiser la plage de temps.	Cliquez sur Réinitialiser le zoom.
Afficher les sorties pour toutes les requêtes à un moment précis.	Flottez au-dessus de l'intrigue au point qui vous intéresse. Les sorties de requête apparaissent dans une fenêtre contextuelle.
Cachez l'intrigue.	Cliquez sur Masquer le graphique.

Ressources supplémentaires

- Documentation de requête Prometheus

6.5. L'OBTENTION D'INFORMATIONS DÉTAILLÉES SUR UNE CIBLE DE MESURES

La console Web dédiée OpenShift vous permet d'afficher, de rechercher et de filtrer les points de terminaison actuellement ciblés pour le grattage, ce qui vous aide à identifier et à résoudre les problèmes. À titre d'exemple, vous pouvez voir l'état actuel des points de terminaison ciblés pour voir lorsque la surveillance dédiée OpenShift n'est pas en mesure de gratter des métriques à partir d'un composant ciblé.

La page cible Metrics affiche des cibles pour les projets définis par l'utilisateur.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.

Procédure

1. Dans la perspective de l'administrateur de la console Web dédiée OpenShift, allez à Observer → Targets. La page cible Metrics s'ouvre avec une liste de toutes les cibles de point de terminaison de service qui sont grattées pour les métriques.
   Cette page affiche des détails sur les cibles pour les projets par défaut OpenShift dédiés et définis par l'utilisateur. Cette page répertorie les informations suivantes pour chaque cible:

   - L'URL du point de terminaison du service est grattée
   - La ressource ServiceMonitor est surveillée
   - Le statut haut ou bas de la cible
   - Espace de noms
   - Dernier temps d'éraflure
   - Durée de la dernière éraflure

2. Facultatif: Pour trouver une cible spécifique, effectuez l'une des actions suivantes:

L'option	Description
Filtrer les cibles par statut et source.	Choisissez les filtres dans la liste des filtres. Les options de filtrage suivantes sont disponibles: ● Filtres d'état: ○ En haut. La cible est actuellement en hausse et est activement grattée pour les métriques. ○ En bas. La cible est actuellement en baisse et n'est pas grattée pour les métriques. ● Filtres source: ○ La plate-forme. Les cibles au niveau de la plate-forme se rapportent uniquement à Red Hat OpenShift Service par défaut sur les projets AWS. Ces projets fournissent le service OpenShift Red Hat de base sur les fonctionnalités AWS. ○ L'utilisateur. Les cibles des utilisateurs se rapportent à des projets définis par l'utilisateur. Ces projets sont créés par l'utilisateur et peuvent être personnalisés.
Cherchez une cible par nom ou étiquette.	Entrez un terme de recherche dans le champ Texte ou Label à côté de la zone de recherche.
Triez les cibles.	Cliquez sur un ou plusieurs des en-têtes de colonne État de point final, Namespace, Last Scrape et Scrape Duration.

3. Cliquez sur l'URL dans la colonne Endpoint pour qu'une cible accède à sa page de détails de cible. Cette page fournit des informations sur la cible, y compris les informations suivantes:

    - L'URL endpoint étant grattée pour les métriques
    - Le statut actuel Up or Down de la cible
    - Lien vers l'espace de noms
    - Lien vers les détails de la ressource ServiceMonitor
    - Étiquettes attachées à la cible
    - La dernière fois que la cible a été grattée pour les métriques

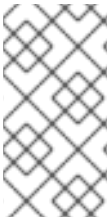
CHAPITRE 7. GÉRER LES ALERTES

Dans OpenShift Dedicated 4, l'interface d'alerte vous permet de gérer les alertes, les silences et les règles d'alerte.

- Des règles d'alerte. Les règles d'alerte contiennent un ensemble de conditions qui définissent un état particulier au sein d'un cluster. Les alertes sont déclenchées lorsque ces conditions sont vraies. Il est possible d'attribuer une règle d'alerte à une sévérité qui définit la manière dont les alertes sont acheminées.

- Des alertes. L'alerte est déclenchée lorsque les conditions définies dans une règle d'alerte sont vraies. Les alertes fournissent une notification indiquant qu'un ensemble de circonstances sont apparentes au sein d'un cluster dédié OpenShift.

- Des silences. Le silence peut être appliqué à une alerte pour empêcher l'envoi de notifications lorsque les conditions d'une alerte sont vraies. Après la notification initiale, vous pouvez réduire une alerte pendant que vous travaillez à résoudre le problème.



NOTE

Les alertes, les silences et les règles d'alerte disponibles dans l'interface utilisateur d'alerte se rapportent aux projets auxquels vous avez accès. Ainsi, si vous êtes connecté en tant qu'utilisateur avec le rôle cluster-admin, vous pouvez accéder à toutes les alertes, silences et règles d'alerte.

7.1. ACCÉDER À L'INTERFACE UTILISATEUR D'ALERTE DU POINT DE VUE DE L'ADMINISTRATEUR

L'interface utilisateur d'alerte est accessible via la perspective de l'administrateur de la console Web dédiée OpenShift.

- Du point de vue de l'administrateur, allez à Observer → Alerter. Les trois pages principales de l'interface utilisateur d'alerte dans cette perspective sont les pages de règles d'alerte, de silence et d'alerte.

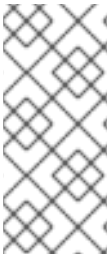
7.2. ACCÉDER À L'INTERFACE UTILISATEUR D'ALERTE DU POINT DE VUE DU DÉVELOPPEUR

L'interface utilisateur d'alerte est accessible par le biais de la perspective Développeur de la console Web dédiée OpenShift.

- Du point de vue Développeur, allez à Observer et allez à l'onglet Alertes.

- Choisissez le projet pour lequel vous souhaitez gérer les alertes dans la liste Projet:

Dans cette perspective, les alertes, les silences et les règles d'alerte sont tous gérés à partir de l'onglet Alertes. Les résultats affichés dans l'onglet Alertes sont spécifiques au projet sélectionné.



NOTE

Dans la perspective Développeur, vous pouvez sélectionner parmi les projets clés OpenShift Dédicés et définis par l'utilisateur auxquels vous avez accès dans la liste Projet: <project_name>. Cependant, les alertes, les silences et les règles d'alerte relatives aux principaux projets dédiés à OpenShift ne sont pas affichés si vous n'êtes pas connecté en tant qu'administrateur de cluster.

7.3. CHERCHER ET FILTRER LES ALERTES, LES SILENCES ET LES RÈGLES D'ALERTE

Il est possible de filtrer les alertes, les silences et les règles d'alerte qui s'affichent dans l'interface utilisateur d'alerte. Cette section fournit une description de chacune des options de filtrage disponibles.

7.3.1. Comprendre les filtres d'alerte

Dans la perspective de l'administrateur, la page Alertes de l'interface d'alerte fournit des détails sur les alertes relatives aux projets par défaut OpenShift dédiés et définis par l'utilisateur. La page contient un résumé de la gravité, de l'état et de la source de chaque alerte. L'heure à laquelle une alerte est entrée dans son état actuel est également affichée.

Il est possible de filtrer par état d'alerte, gravité et source. Par défaut, seules les alertes Plate-forme qui sont Firing sont affichées. Ce qui suit décrit chaque option de filtrage d'alerte:

- Filtres d'état:
    - Des tirs. L'alerte est déclenchée parce que l'état d'alerte est vrai et que l'option pour la durée est passée. L'alerte continue de s'allumer tandis que la condition reste vraie.
    - En attente. L'alerte est active mais attend la durée spécifiée dans la règle d'alerte avant qu'elle ne s'allume.
    - Au silence. L'alerte est maintenant réduite au silence pendant une période définie. Les silences mutent temporairement les alertes basées sur un ensemble de sélecteurs d'étiquettes que vous définissez. Les notifications ne sont pas envoyées pour les alertes qui correspondent à toutes les valeurs listées ou aux expressions régulières.
- Filtres de gravité:
    - C'est critique. L'état qui a déclenché l'alerte pourrait avoir un impact critique. L'alerte nécessite une attention immédiate lorsqu'elle est tirée et est généralement affichée à un individu ou à une équipe d'intervention critique.
    - Avertissement. L'alerte fournit une notification d'avertissement sur quelque chose qui pourrait nécessiter une attention pour éviter qu'un problème ne se produise. Les avertissements sont généralement acheminés vers un système de billetterie pour examen non immédiat.
    - Info. L'alerte est fournie à titre informatif seulement.
    - Aucun. L'alerte n'a pas de sévérité définie.
    - Il est également possible de créer des définitions de gravité personnalisées pour les alertes relatives aux projets définis par l'utilisateur.
- Filtres source:

- La plate-forme. Les alertes au niveau de la plate-forme se rapportent uniquement aux projets par défaut OpenShift dédiés. Ces projets fournissent des fonctionnalités de base OpenShift dédiées.

- L'utilisateur. Les alertes utilisateur se rapportent à des projets définis par l'utilisateur. Ces alertes sont créées par l'utilisateur et sont personnalisables. La surveillance de la charge de travail définie par l'utilisateur peut être activée après l'installation pour fournir l'observabilité de vos propres charges de travail.

7.3.2. Comprendre les filtres de silence

Dans la perspective de l'administrateur, la page Silences de l'interface d'alerte fournit des détails sur les silences appliqués aux alertes dans les projets par défaut OpenShift dédiés et définis par l'utilisateur. La page contient un résumé de l'état de chaque silence et de l'heure à laquelle un silence se termine.

Il est possible de filtrer par état de silence. Les silences actifs et en attente sont affichés par défaut. Ce qui suit décrit chaque option de filtre d'état de silence:

- Filtres d'état:

  - Actif. Le silence est actif et l'alerte sera réduite jusqu'à ce que le silence soit expiré.

  - En attente. Le silence a été programmé et il n'est pas encore actif.

  - Expiré. Le silence a expiré et les notifications seront envoyées si les conditions d'une alerte sont vraies.

7.3.3. Comprendre les filtres de règles d'alerte

Dans la perspective de l'administrateur, la page des règles d'alerte de l'interface d'alerte fournit des détails sur les règles d'alerte relatives aux projets par défaut OpenShift dédiés et définis par l'utilisateur. La page contient un résumé de l'état, de la gravité et de la source de chaque règle d'alerte.

Il est possible de filtrer les règles d'alerte par état d'alerte, gravité et source. Par défaut, seules les règles d'alerte de la Plateforme sont affichées. Ce qui suit décrit chaque option de filtrage des règles d'alerte:

- Filtres d'état d'alerte:

  - Des tirs. L'alerte est déclenchée parce que l'état d'alerte est vrai et que l'option pour la durée est passée. L'alerte continue de s'allumer tandis que la condition reste vraie.

  - En attente. L'alerte est active mais attend la durée spécifiée dans la règle d'alerte avant qu'elle ne s'allume.

  - Au silence. L'alerte est maintenant réduite au silence pendant une période définie. Les silences mutent temporairement les alertes basées sur un ensemble de sélecteurs d'étiquettes que vous définissez. Les notifications ne sont pas envoyées pour les alertes qui correspondent à toutes les valeurs listées ou aux expressions régulières.

  - Il n'y a pas de firing. L'alerte ne tire pas.

- Filtres de gravité:

  - C'est critique. Les conditions définies dans la règle d'alerte pourraient avoir un impact critique. Lorsque c'est vrai, ces conditions nécessitent une attention immédiate. Les alertes relatives à la règle sont généralement affichées à un individu ou à une équipe d'intervention critique.

- Avertissement. Les conditions définies dans la règle d'alerte peuvent nécessiter une attention pour éviter qu'un problème ne se produise. Les alertes relatives à la règle sont généralement acheminées vers un système de billetterie pour examen non immédiat.
- Info. La règle d'alerte fournit uniquement des alertes d'information.
- Aucun. La règle d'alerte n'a pas de sévérité définie.
- Il est également possible de créer des définitions de gravité personnalisées pour les règles d'alerte relatives aux projets définis par l'utilisateur.

- Filtres source:
  - La plate-forme. Les règles d'alerte au niveau de la plate-forme ne concernent que les projets par défaut OpenShift dédiés. Ces projets fournissent des fonctionnalités de base OpenShift dédiées.
  - L'utilisateur. Les règles d'alerte de charge de travail définies par l'utilisateur se rapportent à des projets définis par l'utilisateur. Ces règles d'alerte sont créées par l'utilisateur et sont personnalisables. La surveillance de la charge de travail définie par l'utilisateur peut être activée après l'installation pour fournir l'observabilité de vos propres charges de travail.

7.3.4. Chercher et filtrer les alertes, les silences et les règles d'alerte dans la perspective du développeur

Dans la perspective Développeur, la page Alertes de l'interface d'alerte fournit une vue combinée des alertes et des silences relatifs au projet sélectionné. Le lien vers la règle régissant l'alerte est fourni pour chaque alerte affichée.

Dans cette vue, vous pouvez filtrer par état d'alerte et sévérité. Par défaut, toutes les alertes du projet sélectionné sont affichées si vous avez l'autorisation d'accéder au projet. Ces filtres sont les mêmes que ceux décrits pour la perspective de l'administrateur.

7.4. OBTENIR DES INFORMATIONS SUR LES ALERTES, LES SILENCES ET LES RÈGLES D'ALERTE DU POINT DE VUE DE L'ADMINISTRATEUR

L'interface utilisateur d'alerte fournit des informations détaillées sur les alertes et leurs règles d'alerte et silences.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec les autorisations de vision du projet pour lesquelles vous consultez des alertes.

Procédure

Afin d'obtenir des informations sur les alertes:

1. Du point de vue de l'administrateur de la console Web dédiée OpenShift, accédez à la page Observer → Alertes → Alertes.
2. Facultatif: Rechercher des alertes par nom en utilisant le champ Nom dans la liste de recherche.
3. Facultatif : Filtrer les alertes par état, gravité et source en sélectionnant les filtres dans la liste des filtres.

4. Facultatif: Trier les alertes en cliquant sur une ou plusieurs des colonnes Nom, gravité, état et source.

5. Cliquez sur le nom d'une alerte pour afficher sa page de détails d'alerte. La page comprend un graphique qui illustre les données des séries chronologiques d'alerte. Il fournit également les informations suivantes sur l'alerte:

    - Description de l'alerte
    - Les messages associés à l'alerte
    - Étiquettes attachées à l'alerte
    - Lien vers sa règle d'alerte
    - Des silences pour l'alerte, s'il y en a

Afin d'obtenir des informations sur les silences:

1. Du point de vue administrateur de la console Web dédiée OpenShift, accédez à la page Observer → Alerter → Silences.

2. Facultatif: Filtrer les silences par nom à l'aide du champ Recherche par nom.

3. Facultatif: Filtrer les silences par état en sélectionnant les filtres dans la liste des filtres. Des filtres actifs et en attente sont appliqués par défaut.

4. Facultatif: Trier les silences en cliquant sur un ou plusieurs des en-têtes de colonne Nom, Alertes de Firing, State et Creator.

5. Choisissez le nom d'un silence pour afficher sa page Détails du silence. La page comprend les détails suivants:

    - La spécification d'alerte
    - Heure de début
    - Heure de fin
    - État du silence
    - Le nombre et la liste des alertes de tir

Afin d'obtenir des informations sur les règles d'alerte:

1. Du point de vue de l'administrateur de la console Web dédiée OpenShift, allez à la page Observer → Alerting → Alerting rules.

2. Facultatif: Filtrer les règles d'alerte par état, gravité et source en sélectionnant les filtres dans la liste des filtres.

3. Facultatif: Trier les règles d'alerte en cliquant sur un ou plusieurs des en-têtes de la colonne Nom, gravité, alerte et source.

4. Choisissez le nom d'une règle d'alerte pour afficher sa page de détails des règles d'alerte. La page fournit les détails suivants sur la règle d'alerte:

    - Alerte du nom de la règle, de la gravité et de la description.

- L'expression qui définit la condition pour lancer l'alerte.
- L'heure pour laquelle la condition doit être vraie pour une alerte au feu.
- Graphique pour chaque alerte régie par la règle d'alerte, montrant la valeur avec laquelle l'alerte est déclenchée.
- Le tableau de toutes les alertes régies par la règle d'alerte.

7.5. OBTENIR DES INFORMATIONS SUR LES ALERTES, LES SILENCES ET LES RÈGLES D'ALERTE DU POINT DE VUE DU DÉVELOPPEUR

L'interface utilisateur d'alerte fournit des informations détaillées sur les alertes et leurs règles d'alerte et silences.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec les autorisations de vision du projet pour lesquelles vous consultez des alertes.

Procédure

Afin d'obtenir des informations sur les alertes, les silences et les règles d'alerte:

1. Du point de vue développeur de la console Web dédiée OpenShift, accédez à la page Observer → <project_name> → Alertes.
2. Consultez les détails d'une alerte, d'un silence ou d'une règle d'alerte:
   - Les détails d'alerte peuvent être visualisés en cliquant sur un symbole supérieur (>) à côté d'un nom d'alerte, puis en sélectionnant l'alerte dans la liste.
   - Les détails du silence peuvent être consultés en cliquant sur un silence dans la section Silenced par la page de détails Alerte. La page Détails du Silence comprend les informations suivantes:
     - La spécification d'alerte
     - Heure de début
     - Heure de fin
     - État du silence
     - Le nombre et la liste des alertes de tir
   - Les détails des règles d'alerte peuvent être visualisés en cliquant sur le menu à côté d'une alerte dans la page Alertes, puis en cliquant sur Afficher la règle d'alerte.

NOTE

Les alertes, les silences et les règles d'alerte concernant le projet sélectionné sont uniquement affichés dans la perspective Développeur.

Ressources supplémentaires

- Consultez les runbooks de l'opérateur de surveillance du cluster pour aider à diagnostiquer et à résoudre les problèmes qui déclenchent des alertes de surveillance spécifiques à OpenShift.

7.6. GÉRER LES SILENCES

Dans les perspectives Administrateur et Développeur, vous pouvez créer un silence pour une alerte dans la console Web OpenShift Dedicated. Après avoir créé un silence, vous ne recevrez pas de notifications sur une alerte lorsque l'alerte s'allume.

Créer des silences est utile dans les scénarios où vous avez reçu une notification d'alerte initiale, et vous ne voulez pas recevoir d'autres notifications pendant le temps où vous résolvez le problème sous-jacent causant l'alerte.

Lors de la création d'un silence, vous devez spécifier s'il devient actif immédiatement ou à un moment ultérieur. Il faut également fixer une période de temps après laquelle le silence expire.

Après avoir créé des silences, vous pouvez les afficher, les modifier et les expirer.

NOTE

Lorsque vous créez des silences, ils sont reproduits à travers les pods Alertmanager. Cependant, si vous ne configurez pas le stockage persistant pour Alertmanager, les silences peuvent être perdus. Cela peut se produire, par exemple, si tous les pods Alertmanager redémarrent en même temps.

Ressources supplémentaires

- Configuration du stockage persistant

7.6.1. Alertes silencieuses du point de vue de l'administrateur

Il est possible de réduire au silence une alerte spécifique ou des alertes de silence qui correspondent à une spécification que vous définissez.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle cluster-admin.

Procédure

Faire taire une alerte spécifique:

1. Du point de vue de l'administrateur de la console Web dédiée OpenShift, allez à Observer → Alerte → Alertes.
2. Cliquez sur l'alerte Silence pour ouvrir la page d'alerte Silence avec une configuration par défaut pour l'alerte choisie.
3. Facultatif: Modifiez les détails de configuration par défaut pour le silence.

NOTE

Il faut ajouter un commentaire avant d'enregistrer un silence.

4. Cliquez sur Silence pour sauver le silence.

Faire taire un ensemble d'alertes:

1. Du point de vue de l'administrateur de la console Web dédiée OpenShift, allez à Observer → Alerter → Silences.
2. Cliquez sur Créer le silence.
3. Dans la page Créer le silence, définissez le calendrier, la durée et les détails de l'étiquette pour une alerte.

NOTE

Il faut ajouter un commentaire avant d'enregistrer un silence.

4. Cliquez sur Silence pour créer des silences pour les alertes correspondant aux étiquettes que vous avez saisies.

7.6.2. Alertes silencieuses du point de vue du développeur

Il est possible de réduire au silence une alerte spécifique ou des alertes de silence qui correspondent à une spécification que vous définissez.

Conditions préalables

- En tant qu'administrateur de cluster, vous avez accès au cluster en tant qu'utilisateur avec le rôle d'administrateur dédié.
- En tant qu'utilisateur non administrateur, vous avez accès au cluster en tant qu'utilisateur avec les rôles d'utilisateur suivants:
  - Le rôle de cluster cluster-monitoring-view, qui vous permet d'accéder à Alertmanager.
  - Le rôle monitoring-alertmanager-édition, qui vous permet de créer et de réduire au silence les alertes dans la perspective de l'administrateur dans la console Web.
  - Le rôle de cluster monitoring-règle-édition, qui vous permet de créer et de réduire au silence les alertes dans la perspective Développeur dans la console Web.

Procédure

Faire taire une alerte spécifique:

1. Du point de vue Développeur de la console Web dédiée OpenShift, allez à Observer et allez à l'onglet Alertes.
2. Choisissez le projet pour lequel vous souhaitez réduire au silence une alerte dans la liste Projet:
3. Au besoin, développez les détails de l'alerte en cliquant sur un symbole supérieur (>) à côté du nom de l'alerte.
4. Cliquez sur le message d'alerte dans la vue élargie pour ouvrir la page de détails de l'alerte pour l'alerte.
5. Cliquez sur l'alerte Silence pour ouvrir la page d'alerte Silence avec une configuration par défaut pour l'alerte.

6. Facultatif: Modifiez les détails de configuration par défaut pour le silence.

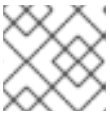
NOTE

Il faut ajouter un commentaire avant d'enregistrer un silence.

7. Cliquez sur Silence pour sauver le silence.

Faire taire un ensemble d'alertes:

1. Du point de vue Développeur de la console Web dédiée OpenShift, allez à Observer et allez à l'onglet Silences.
2. Choisissez le projet pour lequel vous souhaitez réduire au silence les alertes dans la liste Project:.
3. Cliquez sur Créer le silence.
4. Dans la page Créer le silence, définissez la durée et les détails de l'étiquette pour une alerte.



NOTE

Il faut ajouter un commentaire avant d'enregistrer un silence.

5. Cliquez sur Silence pour créer des silences pour les alertes correspondant aux étiquettes que vous avez saisies.

7.6.3. Édition des silences du point de vue de l'administrateur

Il est possible d'éditer un silence qui expire le silence existant et d'en créer un nouveau avec la configuration modifiée.

Conditions préalables

- En tant qu'administrateur de cluster, vous avez accès au cluster en tant qu'utilisateur avec le rôle d'administrateur dédié.
- En tant qu'utilisateur non administrateur, vous avez accès au cluster en tant qu'utilisateur avec les rôles d'utilisateur suivants:
  - Le rôle de cluster cluster-monitoring-view, qui vous permet d'accéder à Alertmanager.
  - Le rôle monitoring-alertmanager-édition, qui vous permet de créer et de réduire au silence les alertes dans la perspective de l'administrateur dans la console Web.

Procédure

1. Du point de vue de l'administrateur de la console Web dédiée OpenShift, allez à Observer → Alerter → Silences.
2. Dans le cas du silence que vous souhaitez modifier, cliquez et sélectionnez Modifier le silence. Alternativement, vous pouvez cliquer sur Actions et sélectionner Modifier le silence sur la page Détails du silence pour un silence.
3. Dans la page Modifier le silence, apportez des modifications et cliquez sur Silence. Ce faisant expire le silence existant et en crée un avec la configuration mise à jour.

7.6.4. Éditer les silences du point de vue du développeur

Il est possible d'éditer un silence qui expire le silence existant et d'en créer un nouveau avec la configuration modifiée.

Conditions préalables

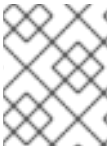
- En tant qu'administrateur de cluster, vous avez accès au cluster en tant qu'utilisateur avec le rôle d'administrateur dédié.
- En tant qu'utilisateur non administrateur, vous avez accès au cluster en tant qu'utilisateur avec les rôles d'utilisateur suivants:
  - Le rôle de cluster cluster-monitoring-view, qui vous permet d'accéder à Alertmanager.
  - Le rôle de cluster monitoring-règle-édition, qui vous permet de créer et de réduire au silence les alertes dans la perspective Développeur dans la console Web.

Procédure

1. Du point de vue Développeur de la console Web dédiée OpenShift, allez à Observer et allez à l'onglet Silences.
2. Choisissez le projet pour lequel vous souhaitez modifier les silences dans la liste Project:.
3. Dans le cas du silence que vous souhaitez modifier, cliquez et sélectionnez Modifier le silence. Alternativement, vous pouvez cliquer sur Actions et sélectionner Modifier le silence sur la page Détails du silence pour un silence.
4. Dans la page Modifier le silence, apportez des modifications et cliquez sur Silence. Ce faisant expire le silence existant et en crée un avec la configuration mise à jour.

7.6.5. Expiration des silences du point de vue de l'administrateur

« vous pouvez expirer un seul silence ou plusieurs silences. L'expiration d'un silence le désactive définitivement.



NOTE

Il est impossible de supprimer les alertes expirées et réduites au silence. Les silences expirés de plus de 120 heures sont collectés.

Conditions préalables

- En tant qu'administrateur de cluster, vous avez accès au cluster en tant qu'utilisateur avec le rôle d'administrateur dédié.
- En tant qu'utilisateur non administrateur, vous avez accès au cluster en tant qu'utilisateur avec les rôles d'utilisateur suivants:
  - Le rôle de cluster cluster-monitoring-view, qui vous permet d'accéder à Alertmanager.
  - Le rôle monitoring-alertmanager-édition, qui vous permet de créer et de réduire au silence les alertes dans la perspective de l'administrateur dans la console Web.

Procédure

1. Allez à Observer → Alerter → Silences.
2. Dans le cas du silence ou des silences que vous souhaitez expirer, sélectionnez la case à cocher dans la ligne correspondante.
3. Cliquez sur Expire 1 silence pour expirer un seul silence sélectionné ou Expire <n> silences pour expirer plusieurs silences sélectionnés, où <n> est le nombre de silences que vous avez sélectionnés.
   Alternativement, pour expirer un seul silence, vous pouvez cliquer sur Actions et sélectionner Expire le silence sur la page Détails du silence pour un silence.

7.6.6. Expiration des silences du point de vue des développeurs

« vous pouvez expirer un seul silence ou plusieurs silences. L'expiration d'un silence le désactive définitivement.

NOTE

Il est impossible de supprimer les alertes expirées et réduites au silence. Les silences expirés de plus de 120 heures sont collectés.

Conditions préalables

- En tant qu'administrateur de cluster, vous avez accès au cluster en tant qu'utilisateur avec le rôle d'administrateur dédié.
- En tant qu'utilisateur non administrateur, vous avez accès au cluster en tant qu'utilisateur avec les rôles d'utilisateur suivants:
  - Le rôle de cluster cluster-monitoring-view, qui vous permet d'accéder à Alertmanager.
  - Le rôle de cluster monitoring-règle-édition, qui vous permet de créer et de réduire au silence les alertes dans la perspective Développeur dans la console Web.

Procédure

1. Du point de vue Développeur de la console Web dédiée OpenShift, allez à Observer et allez à l'onglet Silences.
2. Choisissez le projet pour lequel vous souhaitez expirer un silence dans la liste Projet:
3. Dans le cas du silence ou des silences que vous souhaitez expirer, sélectionnez la case à cocher dans la ligne correspondante.
4. Cliquez sur Expire 1 silence pour expirer un seul silence sélectionné ou Expire <n> silences pour expirer plusieurs silences sélectionnés, où <n> est le nombre de silences que vous avez sélectionnés.
   Alternativement, pour expirer un seul silence, vous pouvez cliquer sur Actions et sélectionner Expire le silence sur la page Détails du silence pour un silence.

7.7. CRÉATION DE RÈGLES D'ALERTE POUR LES PROJETS DÉFINIS PAR L'UTILISATEUR

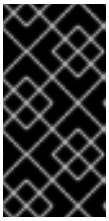
Dans OpenShift Dedicated, vous pouvez créer des règles d'alerte pour les projets définis par l'utilisateur. Ces règles d'alerte déclencheront des alertes basées sur les valeurs des métriques choisies.

Lorsque vous créez des règles d'alerte pour un projet défini par l'utilisateur, considérez les comportements clés et les limites importantes suivants lorsque vous définissez les nouvelles règles:

- La règle d'alerte définie par l'utilisateur peut inclure des métriques exposées par son propre projet en plus des métriques par défaut de la surveillance de la plate-forme de base. Il n'est pas possible d'inclure des métriques d'un autre projet défini par l'utilisateur.
  À titre d'exemple, une règle d'alerte pour le projet défini par l'utilisateur ns1 peut utiliser des métriques exposées par le projet ns1 en plus des métriques de plate-forme de base, telles que les métriques CPU et mémoire. Cependant, la règle ne peut pas inclure des métriques d'un projet défini par l'utilisateur ns2 différent.

- Lorsque vous créez une règle d'alerte, l'étiquette de l'espace de noms est appliquée par défaut, même si une règle avec le même nom existe dans un autre projet. Afin de créer des règles d'alerte qui ne sont pas liées à leur projet d'origine, voir « Créer des règles d'alerte croisée pour les projets définis par l'utilisateur ».

- Afin de réduire la latence et de minimiser la charge sur les composants de surveillance de plate-forme de base, vous pouvez ajouter à une règle l'étiquette openshift.io/prometheus-rule-evaluation-scope: feuille-prometheus. Cette étiquette force uniquement l'instance Prometheus déployée dans le projet openshift-user-workload-monitoring à évaluer la règle d'alerte et empêche l'instance Thanos Ruler de le faire.



IMPORTANT

Lorsqu'une règle d'alerte comporte cette étiquette, votre règle d'alerte ne peut utiliser que les métriques exposées par votre projet défini par l'utilisateur. Les règles d'alerte que vous créez en fonction des métriques de plate-forme par défaut peuvent ne pas déclencher d'alertes.

7.7.1. L'optimisation de l'alerte pour les projets définis par l'utilisateur

Lors de la création de règles d'alerte, vous pouvez optimiser l'alerte pour vos propres projets en tenant compte des recommandations suivantes:

- Minimisez le nombre de règles d'alerte que vous créez pour votre projet. Créez des règles d'alerte qui vous informent des conditions qui vous affectent. Il est plus difficile de remarquer des alertes pertinentes si vous générez de nombreuses alertes pour des conditions qui ne vous impactent pas.

- Créez des règles d'alerte pour les symptômes au lieu des causes. Créez des règles d'alerte qui vous informent des conditions quelle que soit la cause sous-jacente. La cause peut ensuite faire l'objet d'une enquête. Il vous faudra beaucoup plus de règles d'alerte si chacune se rapporte uniquement à une cause spécifique. Certaines causes sont alors susceptibles d'être manquées.

- Planifiez avant d'écrire vos règles d'alerte. Déterminez quels symptômes sont importants pour vous et quelles actions vous voulez prendre s'ils se produisent. Ensuite, construisez une règle d'alerte pour chaque symptôme.

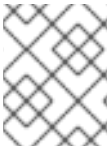
- Fournissez une messagerie d'alerte claire. Indiquez le symptôme et les actions recommandées dans le message d'alerte.

- Incluez les niveaux de gravité dans vos règles d'alerte. La gravité d'une alerte dépend de la façon dont vous devez réagir si le symptôme rapporté se produit. À titre d'exemple, une alerte

critique doit être déclenchée si un symptôme nécessite une attention immédiate de la part d'une personne ou d'une équipe d'intervention critique.

7.7.2. Création de règles d'alerte pour les projets définis par l'utilisateur

Il est possible de créer des règles d'alerte pour les projets définis par l'utilisateur. Ces règles d'alerte déclencheront des alertes basées sur les valeurs des métriques choisies.



NOTE

Afin d'aider les utilisateurs à comprendre l'impact et la cause de l'alerte, assurez-vous que votre règle d'alerte contient un message d'alerte et une valeur de gravité.

Conditions préalables

- La surveillance des projets définis par l'utilisateur a été activée.
- Il est connecté en tant qu'administrateur de cluster ou en tant qu'utilisateur qui a le rôle de cluster monitoring-règle-edit pour le projet où vous souhaitez créer une règle d'alerte.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Créez un fichier YAML pour les règles d'alerte. Dans cet exemple, il est appelé example-app-alerting-rule.yaml.
2. Ajoutez une configuration de règle d'alerte au fichier YAML. L'exemple suivant crée une nouvelle règle d'alerte nommée exemple-alerte. La règle d'alerte déclenche une alerte lorsque la métrique de version exposée par le service d'échantillon devient 0:

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: example-alert
  namespace: ns1
spec:
  groups:
  - name: example
    rules:
    - alert: VersionAlert 1
      for: 1m 2
      expr: version{job="prometheus-example-app"} == 0 3
      labels:
        severity: warning 4
      annotations:
        message: This is an example alert. 5
```

1. Le nom de la règle d'alerte que vous souhaitez créer.
2. La durée pour laquelle la condition doit être vraie avant qu'une alerte ne soit déclenchée.
3. L'expression de requête PromQL qui définit la nouvelle règle.

4 La sévérité que la règle d'alerte attribue à l'alerte.

5 Le message associé à l'alerte.

3. Appliquer le fichier de configuration au cluster:

```
$ oc apply -f example-app-alerting-rule.yaml
```

7.7.3. Création de règles d'alerte croisée pour les projets définis par l'utilisateur

Il est possible de créer des règles d'alerte pour les projets définis par l'utilisateur qui ne sont pas liés à leur projet d'origine en configurant un projet dans la config map. Cela vous permet de créer des règles d'alerte génériques qui s'appliquent à plusieurs projets définis par l'utilisateur au lieu d'avoir des objets PrometheusRule individuels dans chaque projet utilisateur.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.

NOTE

Lorsque vous êtes un utilisateur non-administrateur, vous pouvez toujours créer des règles d'alerte croisée si vous avez le rôle de cluster monitoring-règles-édition pour le projet où vous souhaitez créer une règle d'alerte. Cependant, ce projet doit être configuré dans la carte de configuration de configuration utilisateur-workload-monitoring-config sous la propriété namespaces WithoutLabelEnforcement, qui ne peut être effectuée que par les administrateurs de clusters.

- L'objet ConfigMap existe. Cet objet est créé par défaut lorsque le cluster est créé.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditez la carte de configuration de la configuration de l'utilisateur-workload-monitoring dans le projet openshift-user-workload-monitoring:

```
$ oc -n openshift-user-workload-monitoring edit configmap user-workload-monitoring-config
```

2. Configurez des projets dans lesquels vous souhaitez créer des règles d'alerte qui ne sont pas liées à un projet spécifique:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    namespacesWithoutLabelEnforcement: [ <namespace> ] 1
    # ...
```

1. Indiquez un ou plusieurs projets dans lesquels vous souhaitez créer des règles d'alerte croisée. Les règles Prometheus et Thanos pour la surveillance définie par l'utilisateur n'appliquent pas l'étiquette de l'espace de noms dans les objets PrometheusRule créés dans ces projets.

3. Créez un fichier YAML pour les règles d'alerte. Dans cet exemple, il est appelé exemple-cross-project-alerting-rule.yaml.

4. Ajoutez une configuration de règle d'alerte au fichier YAML. L'exemple suivant crée une nouvelle règle d'alerte croisée appelée exemple-sécurité. La règle d'alerte s'allume lorsqu'un projet utilisateur n'applique pas la politique de sécurité des pods restreints:

Exemple de règle d'alerte croisée de projet

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: example-security
  namespace: ns1 (1)
spec:
  groups:
    - name: pod-security-policy
      rules:
        - alert: "ProjectNotEnforcingRestrictedPolicy" (2)
          for: 5m (3)
          expr: kube_namespace_labels{namespace!~"
(openshift|kube).*|default",label_pod_security_kubernetes_io_enforce!="restricted"} (4)
          annotations:
            message: "Restricted policy not enforced. Project {{ $labels.namespace }} does not
enforce the restricted pod security policy." (5)
          labels:
            severity: warning (6)
```

1. Assurez-vous de spécifier le projet que vous avez défini dans le champ namespaces WithoutLabelEnforcement.
2. Le nom de la règle d'alerte que vous souhaitez créer.
3. La durée pour laquelle la condition doit être vraie avant qu'une alerte ne soit déclenchée.
4. L'expression de requête PromQL qui définit la nouvelle règle.
5. Le message associé à l'alerte.
6. La sévérité que la règle d'alerte attribue à l'alerte.

IMPORTANT

Assurez-vous de créer une règle d'alerte croisée spécifique dans un seul des projets que vous avez spécifiés dans le champ namespaces WithoutLabelEnforcement. Lorsque vous créez la même règle d'alerte croisée dans plusieurs projets, cela se traduit par des alertes répétées.

5. Appliquer le fichier de configuration au cluster:

```
$ oc apply -f example-cross-project-alerting-rule.yaml
```

Ressources supplémentaires

- Documentation d'alerte Prometheus
- Aperçu du suivi

7.8. GESTION DES RÈGLES D'ALERTE POUR LES PROJETS DÉFINIS PAR L'UTILISATEUR

Dans OpenShift Dedicated, vous pouvez afficher, modifier et supprimer les règles d'alerte dans les projets définis par l'utilisateur.

IMPORTANT

La gestion des règles d'alerte pour les projets définis par l'utilisateur n'est disponible que dans OpenShift Dedicated version 4.11 et ultérieure.

Considérations relatives aux règles d'alerte

- Les règles d'alerte par défaut sont utilisées spécifiquement pour le cluster OpenShift Dedicated.
- Certaines règles d'alerte ont intentionnellement des noms identiques. Ils envoient des alertes sur le même événement avec des seuils différents, une gravité différente ou les deux.
- Les règles d'inhibition empêchent les notifications pour les alertes de gravité inférieure qui se déclenchent lorsqu'une alerte de gravité plus élevée est également déclenchée.

7.8.1. Accès aux règles d'alerte pour les projets définis par l'utilisateur

Afin de répertorier les règles d'alerte pour un projet défini par l'utilisateur, vous devez avoir reçu le rôle de cluster monitoring-règle-view pour le projet.

Conditions préalables

- La surveillance des projets définis par l'utilisateur a été activée.
- En tant qu'utilisateur, vous êtes connecté en tant qu'utilisateur qui a le rôle de cluster monitoring-rules-view pour votre projet.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Liste des règles d'alerte dans <projet>:

```
$ oc -n <project> get prometheusrule
```

2. Afin d'énumérer la configuration d'une règle d'alerte, exécutez ce qui suit:

```
$ oc -n <project> get prometheusrule <rule> -o yaml
```

7.8.2. Liste des règles d'alerte pour tous les projets dans une seule vue

En tant qu'administrateur dédié, vous pouvez répertorier ensemble les règles d'alerte pour les principaux projets dédiés à OpenShift et définis par l'utilisateur dans une seule vue.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Du point de vue de l'administrateur de la console Web dédiée OpenShift, allez à Observer → Alerter → Règles d'alerte.
2. Choisissez les sources de la plateforme et de l'utilisateur dans le menu déroulant Filtre.

NOTE

La source de la plateforme est sélectionnée par défaut.

7.8.3. La suppression des règles d'alerte pour les projets définis par l'utilisateur

Il est possible de supprimer les règles d'alerte pour les projets définis par l'utilisateur.

Conditions préalables

- La surveillance des projets définis par l'utilisateur a été activée.
- Il est connecté en tant qu'administrateur de cluster ou en tant qu'utilisateur qui a le rôle de cluster monitoring-règle-edit pour le projet où vous souhaitez créer une règle d'alerte.
- L'OpenShift CLI (oc) a été installé.

Procédure

- Afin de supprimer la règle <foo> dans <namespace>, exécutez ce qui suit:

```
$ oc -n <namespace> delete prometheusrule <foo>
```

7.8.4. Désactivation des règles d'alerte croisée pour les projets définis par l'utilisateur

La création de règles d'alerte croisée pour les projets définis par l'utilisateur est activée par défaut. Les administrateurs de cluster peuvent désactiver la capacité dans la configuration cluster-monitoring-config map pour les raisons suivantes:

- Empêcher la surveillance définie par l'utilisateur de surcharger la pile de surveillance du cluster.

- Afin d'éviter que les règles d'alerte buggy ne soient appliquées au cluster sans avoir à identifier la règle qui cause le problème.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Éditer la carte de configuration cluster-monitoring-config dans le projet openshift-monitoring:

```
$ oc -n openshift-monitoring edit configmap cluster-monitoring-config
```

2. Dans la carte de configuration cluster-monitoring-config, désactivez l'option de créer des règles d'alerte croisée de projet en définissant les règles WithoutLabelEnforcementValeur permise sous data/config.yaml/userWorkload à false:

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: cluster-monitoring-config
  namespace: openshift-monitoring
data:
  config.yaml: |
    userWorkload:
      rulesWithoutLabelEnforcementAllowed: false
    # ...
```

3. Enregistrez le fichier pour appliquer les modifications.

Ressources supplémentaires

- Documentation de Alertmanager

7.9. ENVOI DE NOTIFICATIONS À DES SYSTÈMES EXTERNES

Dans OpenShift Dedicated 4, les alertes de tir peuvent être visualisées dans l'interface utilisateur d'alerte. Les alertes ne sont pas configurées par défaut pour être envoyées à des systèmes de notification. Il est possible de configurer OpenShift Dedicated pour envoyer des alertes aux types de récepteurs suivants:

- À propos de PagerDuty
- À propos de Webhook
- E-mail
- Le slack
- Équipes Microsoft

Les alertes de routage aux récepteurs vous permettent d'envoyer des notifications en temps opportun aux équipes appropriées en cas d'échec. À titre d'exemple, les alertes critiques nécessitent une

attention immédiate et sont généralement redirigées vers une personne ou une équipe d'intervention critique. Les alertes qui fournissent des notifications d'avertissement non critiques pourraient plutôt être acheminées vers un système de billetterie pour un examen non immédiat.

Vérifier que l'alerte est opérationnelle à l'aide de l'alerte du chien de garde

La surveillance dédiée d'OpenShift comprend une alerte de surveillance qui se déclenche en continu. Alertmanager envoie à plusieurs reprises des notifications d'alerte de surveillance aux fournisseurs de notification configurés. Le fournisseur est généralement configuré pour avertir un administrateur lorsqu'il cesse de recevoir l'alerte du chien de garde. Ce mécanisme vous aide à identifier rapidement tout problème de communication entre Alertmanager et le fournisseur de notification.

7.9.1. Configuration de différents récepteurs d'alerte pour les alertes de plate-forme par défaut et les alertes définies par l'utilisateur

Il est possible de configurer différents récepteurs d'alerte pour les alertes de plate-forme par défaut et les alertes définies par l'utilisateur afin d'assurer les résultats suivants:

- Les alertes de plate-forme par défaut sont envoyées à un récepteur appartenant à l'équipe responsable de ces alertes.
- Les alertes définies par l'utilisateur sont envoyées à un autre récepteur afin que l'équipe puisse se concentrer uniquement sur les alertes de plate-forme.

Il est possible d'y parvenir en utilisant l'étiquette openshift_io_alert_source="plateforme" ajoutée par l'opérateur de surveillance du cluster à toutes les alertes de plate-forme:

- Le matcheur openshift_io_alert_source="plateforme" permet de faire correspondre les alertes de plate-forme par défaut.
- Le matcher openshift_io_alert_source!="plateforme" ou 'openshift_io_alert_source="' correspond aux alertes définies par l'utilisateur.

NOTE

Cette configuration ne s'applique pas si vous avez activé une instance distincte de Alertmanager dédiée aux alertes définies par l'utilisateur.

7.9.2. Configuration du routage d'alerte pour les projets définis par l'utilisateur

Lorsque vous êtes un utilisateur non-administrateur qui a reçu le rôle de cluster de routage d'alerte, vous pouvez créer ou modifier le routage d'alerte pour les projets définis par l'utilisateur.

Conditions préalables

- Le routage d'alerte a été activé pour les projets définis par l'utilisateur.
- En tant qu'utilisateur, vous êtes connecté en tant qu'utilisateur qui a le rôle de cluster de routage d'alerte pour le projet pour lequel vous souhaitez créer un routage d'alerte.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Créez un fichier YAML pour le routage d'alerte. L'exemple de cette procédure utilise un fichier appelé example-app-alert-routing.yaml.

2. Ajouter une définition AlertmanagerConfig YAML au fichier. À titre d'exemple:

```
apiVersion: monitoring.coreos.com/v1beta1
kind: AlertmanagerConfig
metadata:
  name: example-routing
  namespace: ns1
spec:
  route:
    receiver: default
    groupBy: [job]
  receivers:
  - name: default
    webhookConfigs:
    - url: https://example.org/post
```

3. Enregistrez le fichier.

4. Appliquer la ressource au cluster:

```
$ oc apply -f example-app-alert-routing.yaml
```

La configuration est automatiquement appliquée aux pods Alertmanager.

7.10. CONFIGURER ALERTMANAGER POUR ENVOYER DES NOTIFICATIONS

Il est possible de configurer Alertmanager pour envoyer des notifications en modifiant le secret Alertmanager-user-workload pour les alertes définies par l'utilisateur.

NOTE

Les fonctionnalités d'une version prise en charge de Alertmanager sont également prises en charge dans une configuration OpenShift Alertmanager. Afin de vérifier toutes les options de configuration d'une version prise en charge de Alertmanager, consultez la configuration Alertmanager.

7.10.1. Configuration du routage d'alerte pour les projets définis par l'utilisateur avec le secret Alertmanager

Lorsque vous avez activé une instance distincte de Alertmanager dédiée au routage d'alerte défini par l'utilisateur, vous pouvez personnaliser où et comment l'instance envoie des notifications en éditant le secret alertmanager-utilisateur-workload dans l'espace de noms openshift-user-workload-monitoring.

NOTE

Les fonctionnalités d'une version prise en charge de Alertmanager en amont sont également prises en charge dans une configuration OpenShift Dedicated Alertmanager. Afin de vérifier toutes les options de configuration d'une version prise en charge de Alertmanager, consultez la configuration Alertmanager (document Prometheus).

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Imprimez la configuration Alertmanager actuellement active dans le fichier alertmanager.yaml:

```
$ oc -n openshift-user-workload-monitoring get secret alertmanager-user-workload --
template='{{ index .data "alertmanager.yaml" }}' | base64 --decode > alertmanager.yaml
```

2. Éditer la configuration dans alertmanager.yaml:

```
global:
  http_config:
    proxy_from_environment: true (1)
route:
  receiver: Default
  group_by:
  - name: Default
  routes:
  - matchers:
    - "service = prometheus-example-monitor" (2)
    receiver: <receiver> (3)
receivers:
- name: Default
- name: <receiver>
  <receiver_configuration> (4)
```

(1) Lorsque vous configurez un proxy HTTP à l'échelle du cluster, définissez le paramètre proxy_from_environnement sur true pour activer le proxying pour tous les récepteurs d'alerte.

(2) Indiquez les étiquettes pour correspondre à vos alertes. Cet exemple cible toutes les alertes portant l'étiquette service="prometheus-example-monitor".

(3) Indiquez le nom du récepteur à utiliser pour le groupe d'alertes.

(4) Indiquez la configuration du récepteur.

3. Appliquer la nouvelle configuration dans le fichier:

```
$ oc -n openshift-user-workload-monitoring create secret generic alertmanager-user-
workload --from-file=alertmanager.yaml --dry-run=client -o=yaml |  oc -n openshift-user-
workload-monitoring replace secret --filename=-
```

7.11. RESSOURCES SUPPLÉMENTAIRES

- Le site officiel de PagerDuty
- Guide d'intégration PagerDuty Prometheus

- La matrice de version de support pour les composants de surveillance

CHAPITRE 8. EXAMEN DES TABLEAUX DE BORD DE SURVEILLANCE

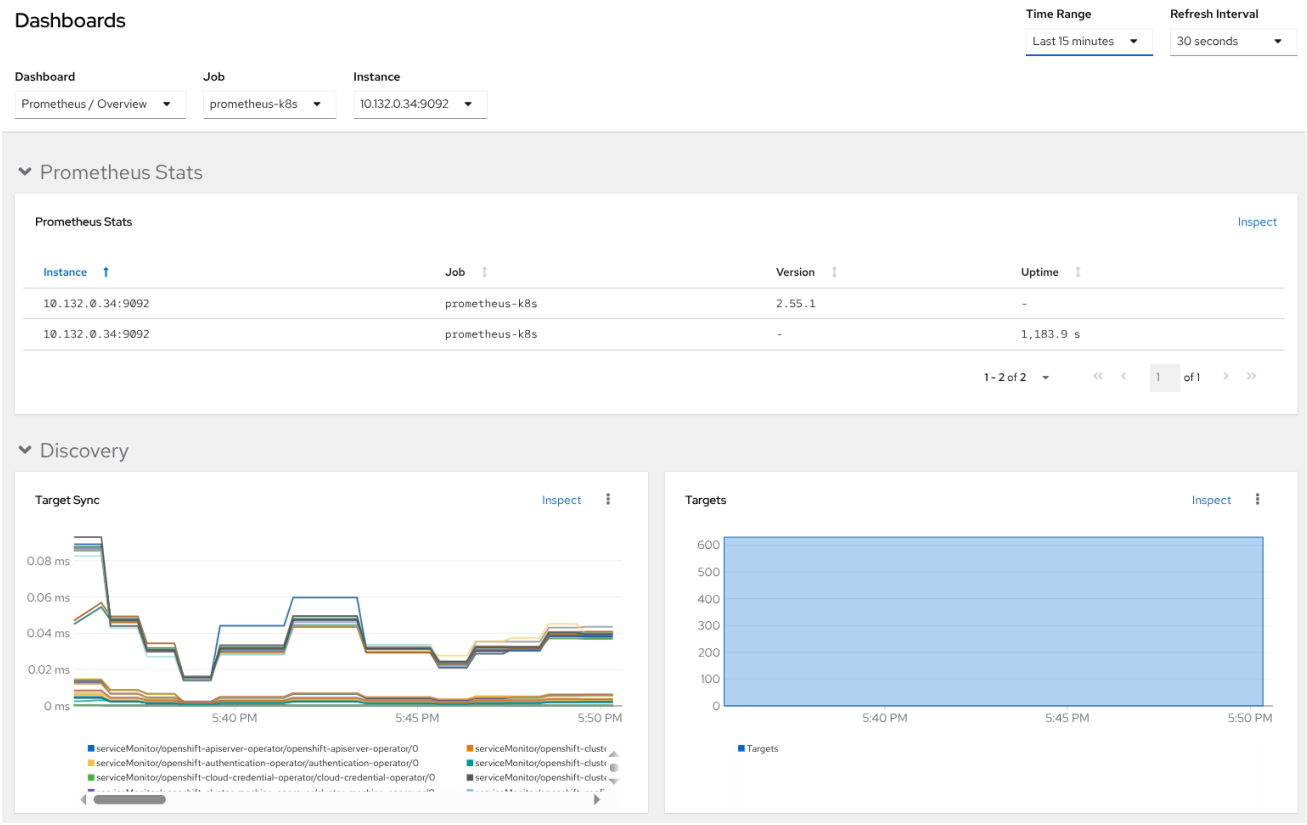
La société OpenShift Dedicated fournit un ensemble de tableaux de bord de surveillance qui vous aident à comprendre l'état des composants du cluster et des charges de travail définies par l'utilisateur.

8.1. LA SURVEILLANCE DES TABLEAUX DE BORD DANS LA PERSPECTIVE DE L'ADMINISTRATEUR

Consultez le point de vue de l'administrateur pour accéder aux tableaux de bord des composants de base OpenShift dédiés, y compris les éléments suivants:

- La performance de l'API
- etc.
- Kubernetes calcule les ressources
- Kubernetes ressources réseau
- Le Prométhée
- Tableaux de bord de la méthode USE relatifs à la performance des clusters et des nœuds
- Indicateurs de performance des nœuds

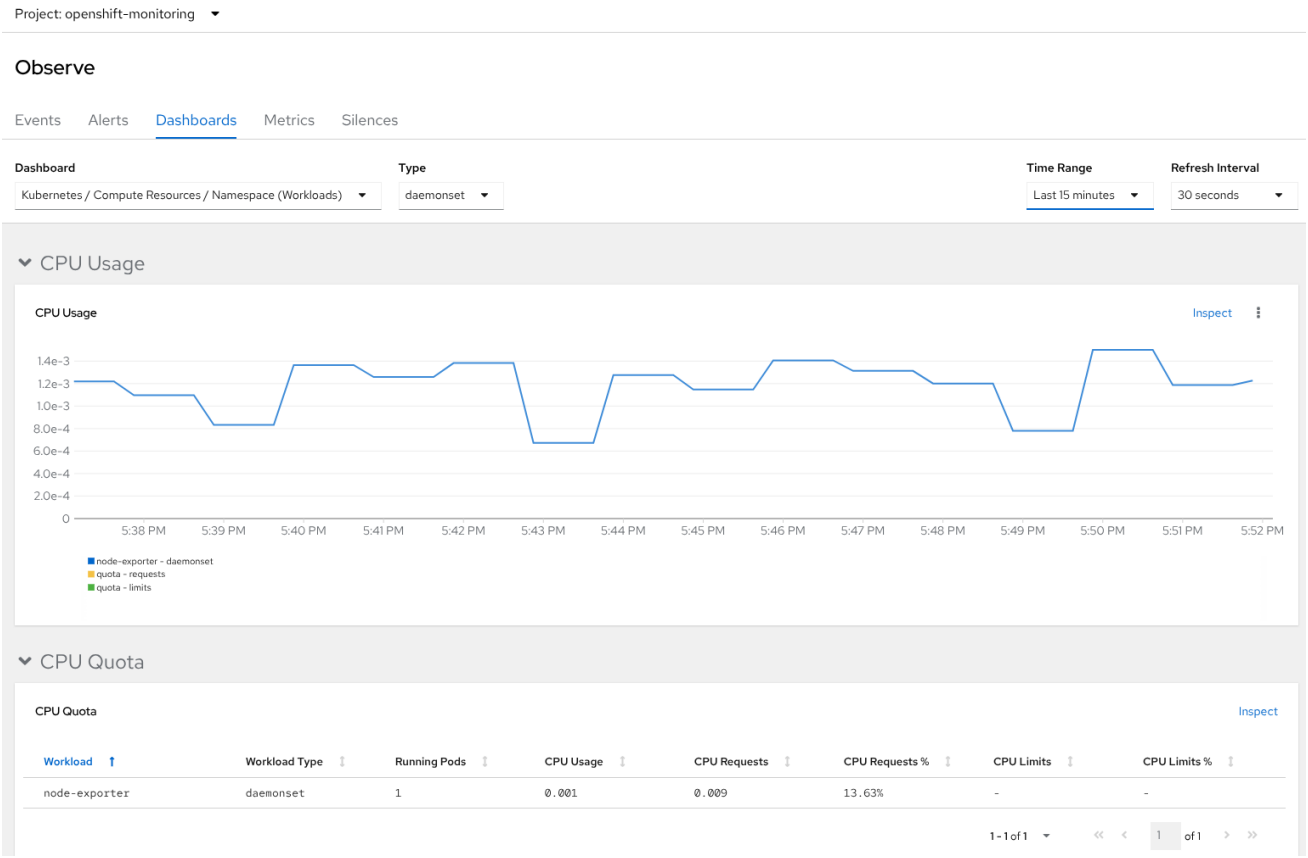
Figure 8.1. Exemple de tableau de bord dans la perspective de l'administrateur



8.2. LA SURVEILLANCE DES TABLEAUX DE BORD DANS LA PERSPECTIVE DES DÉVELOPPEURS

Dans la perspective Développeur, vous pouvez accéder uniquement aux tableaux de bord des ressources de calcul Kubernetes:

Figure 8.2. Exemple de tableau de bord dans la perspective Développeur



8.3. EXAMINER LES TABLEAUX DE BORD DE SURVEILLANCE EN TANT QU'ADMINISTRATEUR DE CLUSTER

Dans la perspective de l'administrateur, vous pouvez afficher les tableaux de bord relatifs aux principaux composants de cluster OpenShift dédiés.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.

Procédure

1. Dans la perspective de l'administrateur de la console Web dédiée OpenShift, allez à Observer → Dashboards.
2. Choisissez un tableau de bord dans la liste Tableau de bord. Certains tableaux de bord, tels que etcd et les tableaux de bord Prometheus, produisent des sous-menus supplémentaires lorsqu'ils sont sélectionnés.
3. Facultatif: Sélectionnez une plage de temps pour les graphiques dans la liste Time Range.
    - Choisissez une période prédéfinie.
    - Définissez une plage de temps personnalisée en cliquant sur la plage de temps personnalisée dans la liste Time Range.

a. Entrez ou sélectionnez les dates et heures de From et jusqu'à.

b. Cliquez sur Enregistrer pour enregistrer la plage de temps personnalisée.

4. Facultatif: Sélectionnez un intervalle de rafraîchissement.

5. Survolez chacun des graphiques d'un tableau de bord pour afficher des informations détaillées sur des éléments spécifiques.

8.4. EXAMINER LES TABLEAUX DE BORD DE SURVEILLANCE EN TANT QUE DÉVELOPPEUR

Dans la perspective Développeur, vous pouvez afficher les tableaux de bord relatifs à un projet sélectionné.

NOTE

Dans la perspective Développeur, vous pouvez afficher les tableaux de bord pour un seul projet à la fois.

Conditions préalables

- En tant que développeur ou utilisateur, vous avez accès au cluster.
- Les autorisations pour le projet que vous consultez le tableau de bord sont affichées.

Procédure

1. Dans la perspective Développeur dans la console Web dédiée OpenShift, cliquez sur Observer et accédez à l'onglet Tableaus de bord.

2. Choisissez un projet dans la liste déroulante Project:

3. Choisissez un tableau de bord dans la liste déroulante du tableau de bord pour voir les métriques filtrées.

NOTE

Les tableaux de bord produisent des sous-menus supplémentaires lorsqu'ils sont sélectionnés, à l'exception de Kubernetes / Compute Resources / Namespace (Pods).

4. Facultatif: Sélectionnez une plage de temps pour les graphiques dans la liste Time Range.
   - Choisissez une période prédéfinie.
   - Définissez une plage de temps personnalisée en cliquant sur la plage de temps personnalisée dans la liste Time Range.
     a. Entrez ou sélectionnez les dates et heures de From et jusqu'à.
     b. Cliquez sur Enregistrer pour enregistrer la plage de temps personnalisée.

5. Facultatif: Sélectionnez un intervalle de rafraîchissement.

6. Survolez chacun des graphiques d'un tableau de bord pour afficher des informations détaillées sur des éléments spécifiques.

CHAPITRE 9. ACCÈS AUX API DE SURVEILLANCE EN UTILISANT LE CLI

Dans OpenShift Dedicated, vous pouvez accéder aux API de service Web pour certains composants de surveillance à partir de l'interface de ligne de commande (CLI).

IMPORTANT

Dans certaines situations, l'accès aux points de terminaison API peut dégrader les performances et l'évolutivité de votre cluster, surtout si vous utilisez des points de terminaison pour récupérer, envoyer ou interroger de grandes quantités de données métriques.

Afin d'éviter ces problèmes, suivez ces recommandations:

- Évitez d'interroger fréquemment les points de terminaison. Limitez les requêtes à un maximum d'une toutes les 30 secondes.
- Il ne faut pas essayer de récupérer toutes les données métriques via le point de terminaison /federate pour Prometheus. Interrogez-le uniquement lorsque vous souhaitez récupérer un ensemble de données agrégé et limité. Ainsi, la récupération de moins de 1 000 échantillons pour chaque demande permet de minimiser le risque de dégradation des performances.

9.1. À PROPOS DE L'ACCÈS AUX API DE SERVICE WEB DE SURVEILLANCE

À partir de la ligne de commande, vous pouvez accéder directement aux points d'extrémité de l'API de service Web pour les composants de la pile de surveillance suivants:

- Le Prométhée
- Alertmanager
- Le chef de Thanos
- À propos de Thanos Querier

IMPORTANT

Afin d'accéder aux API de service Thanos Ruler et Thanos Querier, le compte demandeur doit avoir une autorisation sur la ressource des espaces de noms, qui peut être accordé en liant le rôle de cluster-monitoring-view au compte.

Lorsque vous accédez aux points de terminaison de l'API de service Web pour les composants de surveillance, soyez conscient des limitations suivantes:

- L'authentification de jetons au porteur ne peut être utilisée que pour accéder aux points de terminaison de l'API.
- Il est possible d'accéder uniquement aux points de terminaison dans le chemin /api pour un itinéraire. Lorsque vous essayez d'accéder à un point de terminaison API dans un navigateur Web, une application n'est pas une erreur disponible. Afin d'accéder aux fonctions de

surveillance dans un navigateur Web, utilisez la console Web OpenShift Dedicated pour examiner les tableaux de bord de surveillance.

Ressources supplémentaires

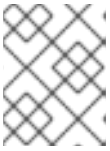
- Examen des tableaux de bord de surveillance

9.2. ACCÈS À UNE API DE SERVICE WEB DE SURVEILLANCE

L'exemple suivant montre comment interroger les récepteurs API de service pour le service Alertmanager utilisé dans la surveillance de la plate-forme de base. Il est possible d'utiliser une méthode similaire pour accéder au service prometheus-k8s pour la plate-forme principale Prometheus et le service thanos-ruler pour Thanos Ruler.

Conditions préalables

- Connectez-vous à un compte lié au rôle de suivi-alertmanager-édition dans l'espace de noms de surveillance openshift.
- Il est connecté à un compte qui a la permission d'obtenir l'itinéraire API Alertmanager.



NOTE

Dans le cas où votre compte n'a pas la permission d'obtenir l'itinéraire API Alertmanager, un administrateur de cluster peut fournir l'URL de l'itinéraire.

Procédure

1. Extraire un jeton d'authentification en exécutant la commande suivante:

```
$ TOKEN=$(oc whoami -t)
```

2. Extrayez l'URL de route API principale d'alerte en exécutant la commande suivante:

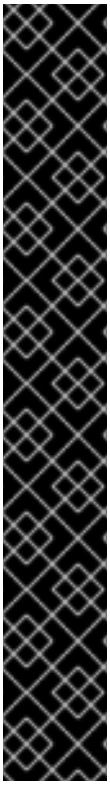
```
$ HOST=$(oc -n openshift-monitoring get route alertmanager-main -ojsonpath=
{.status.ingress[].host})
```

3. Interrogez les récepteurs API de service pour Alertmanager en exécutant la commande suivante:

```
$ curl -H "Authorization: Bearer $TOKEN" -k "https://$HOST/api/v2/receivers"
```

9.3. INTERROGER LES MÉTRIQUES EN UTILISANT LE POINT DE TERMINAISON DE LA FÉDÉRATION POUR PROMETHEUS

Il est possible d'utiliser le point final de la fédération pour Prometheus pour gratter la plate-forme et les métriques définies par l'utilisateur à partir d'un emplacement réseau en dehors du cluster. À cette fin, accédez au point d'extrémité Prometheus /federate pour le cluster via une route OpenShift dédiée.



IMPORTANT

Le retard dans la récupération des données métriques se produit lorsque vous utilisez la fédération. Ce retard peut affecter l'exactitude et la rapidité des mesures grattées.

L'utilisation du point de terminaison de la fédération peut également dégrader les performances et l'évolutivité de votre cluster, surtout si vous utilisez le point de terminaison de la fédération pour récupérer de grandes quantités de données métriques. Afin d'éviter ces problèmes, suivez ces recommandations:

- Il ne faut pas essayer de récupérer toutes les données métriques via le point final de la fédération pour Prometheus. Interrogez-le uniquement lorsque vous souhaitez récupérer un ensemble de données agrégé et limité. Ainsi, la récupération de moins de 1 000 échantillons pour chaque demande permet de minimiser le risque de dégradation des performances.
- Évitez les requêtes fréquentes du point final de la fédération pour Prometheus. Limitez les requêtes à un maximum d'une toutes les 30 secondes.

Lorsque vous devez transférer de grandes quantités de données en dehors du cluster, utilisez plutôt l'écriture à distance. Consultez la section Configuration de l'écriture à distance pour plus d'informations.

Conditions préalables

- L'OpenShift CLI (oc) a été installé.
- En tant qu'utilisateur, vous avez accès au cluster avec le rôle cluster-monitoring-view ou avez obtenu un jeton porteur avec l'autorisation sur la ressource des espaces de noms.

NOTE

Il est possible d'utiliser uniquement l'authentification des jetons porteurs pour accéder au point de terminaison de la fédération Prometheus.

- Connectez-vous à un compte qui a la permission d'obtenir la route de la fédération Prometheus.

NOTE

Dans le cas où votre compte n'a pas la permission d'obtenir l'itinéraire de la fédération Prometheus, un administrateur de cluster peut fournir l'URL de l'itinéraire.

Procédure

1. Il suffit de récupérer le jeton porteur en exécutant la commande suivante:

```
$ TOKEN=$(oc whoami -t)
```

2. Accédez à l'URL de route de la fédération Prometheus en exécutant la commande suivante:

```
$ HOST=$(oc -n openshift-monitoring get route prometheus-k8s-federate -ojsonpath=
{.status.ingress[].host})
```

3. Interrogez les métriques de la route /federate. L'exemple de commande suivant interroge les métriques:

```
$ curl -G -k -H "Authorization: Bearer $TOKEN" https://$HOST/federate --data-urlencode
'match[]=up'
```

Exemple de sortie

```
# TYPE up untyped
up{apiserver="kube-
apiserver",endpoint="https",instance="10.0.143.148:6443",job="apiserver",namespace="default
",service="kubernetes",prometheus="openshift-
monitoring/k8s",prometheus_replica="prometheus-k8s-0"} 1 1657035322214
up{apiserver="kube-
apiserver",endpoint="https",instance="10.0.148.166:6443",job="apiserver",namespace="default
",service="kubernetes",prometheus="openshift-
monitoring/k8s",prometheus_replica="prometheus-k8s-0"} 1 1657035338597
up{apiserver="kube-
apiserver",endpoint="https",instance="10.0.173.16:6443",job="apiserver",namespace="default",
service="kubernetes",prometheus="openshift-
monitoring/k8s",prometheus_replica="prometheus-k8s-0"} 1 1657035343834
...
```

9.4. ACCÉDER AUX MÉTRIQUES DE L'EXTÉRIEUR DU CLUSTER POUR DES APPLICATIONS PERSONNALISÉES

Lors de la surveillance de vos propres services avec des projets définis par l'utilisateur, vous pouvez interroger les métriques Prometheus à partir de l'extérieur du cluster. Accédez à ces données de l'extérieur du cluster en utilisant la route thanos-querier.

Cet accès ne prend en charge que l'utilisation d'un jeton porteur pour l'authentification.

Conditions préalables

- Dans le cadre de la procédure « Activer la surveillance des projets définis par l'utilisateur », vous avez déployé votre propre service.
- Il est connecté à un compte avec le rôle cluster-monitoring-view, qui donne l'autorisation d'accéder à l'API Thanos Querier.
- Connectez-vous à un compte qui a la permission d'obtenir l'itinéraire de l'API Thanos Querier.

NOTE

Dans le cas où votre compte n'a pas la permission d'obtenir l'itinéraire de l'API Thanos Querier, un administrateur de cluster peut fournir l'URL de l'itinéraire.

Procédure

1. Extraire un jeton d'authentification pour se connecter à Prometheus en exécutant la commande suivante:

```
$ TOKEN=$(oc whoami -t)
```

2. Extrayez l'URL route de l'API thanos-querier en exécutant la commande suivante:

```
$ HOST=$(oc -n openshift-monitoring get route thanos-querier -ojsonpath=
{.status.ingress[].host})
```

3. Définissez l'espace de noms sur l'espace de noms dans lequel votre service s'exécute à l'aide de la commande suivante:

```
$ NAMESPACE=ns1
```

4. Interrogez les métriques de vos propres services dans la ligne de commande en exécutant la commande suivante:

```
$ curl -H "Authorization: Bearer $TOKEN" -k "https://$HOST/api/v1/query?" --data-urlencode
"query=up{namespace='$NAMESPACE'}"
```

La sortie montre l'état de chaque pod d'application que Prometheus gratte:

La sortie de l'exemple formaté


```
{
  "status": "success",
  "data": {
    "resultType": "vector",
    "result": [
      {
        "metric": {
          "__name__": "up",
          "endpoint": "web",
          "instance": "10.129.0.46:8080",
          "job": "prometheus-example-app",
          "namespace": "ns1",
          "pod": "prometheus-example-app-68d47c4fb6-jztp2",
          "service": "prometheus-example-app"
        },
        "value": [
          1591881154.748,
          "1"
        ]
      }
    ],
  }
}
```


NOTE

- La sortie d'exemple formatée utilise un outil de filtrage, tel que jq, pour fournir le JSON indenté formaté. Consultez le manuel jq (documentation jq) pour plus d'informations sur l'utilisation de jq.
- La commande demande un point de terminaison de requête instantanée du service Thanos Querier, qui évalue les sélecteurs à un moment donné.

9.5. LA RÉFÉRENCE DES RESSOURCES POUR L'OPÉRATEUR DE SURVEILLANCE DES GRAPPES

Ce document décrit les ressources suivantes déployées et gérées par l'opérateur de surveillance des grappes (CMO):

- Itinéraires
- Les services

Ces informations sont utilisées lorsque vous souhaitez configurer des connexions de point de terminaison API pour récupérer, envoyer ou interroger des données métriques.

IMPORTANT

Dans certaines situations, l'accès aux points de terminaison peut dégrader les performances et l'évolutivité de votre cluster, surtout si vous utilisez des points de terminaison pour récupérer, envoyer ou interroger de grandes quantités de données métriques.

Afin d'éviter ces problèmes, suivez ces recommandations:

- Évitez d'interroger fréquemment les points de terminaison. Limitez les requêtes à un maximum d'une toutes les 30 secondes.
- Il ne faut pas essayer de récupérer toutes les données métriques via le point de terminaison /federate. Interrogez-le uniquement lorsque vous souhaitez récupérer un ensemble de données agrégé et limité. Ainsi, la récupération de moins de 1 000 échantillons pour chaque demande permet de minimiser le risque de dégradation des performances.

9.5.1. L'OCM achemine les ressources

9.5.1.1. contrôle OpenShift/alertmanager-main

Exposez les points de terminaison /api du service Alertmanager-main via un routeur.

9.5.1.2. le système OpenShift-monitoring/prometheus-k8s

Exposez les points de terminaison /api du service prometheus-k8s via un routeur.

9.5.1.3. le logiciel OpenShift-monitoring/prometheus-k8s-federate

Exposez le point de terminaison /federate du service prometheus-k8s via un routeur.

9.5.1.4. accueil > OpenShift-user-workload-monitoring/federate

Exposez le point final /federate du service prometheus-user-load via un routeur.

9.5.1.5. le système OpenShift-monitoring/thanos-querier

Exposez les points de terminaison /api du service thanos-querier via un routeur.

9.5.1.6. accueil > OpenShift-user-workload-monitoring/thanos-ruler

Exposez les points de terminaison /api du service thanos-ruler via un routeur.

9.5.2. Les ressources de services de l'OCM

9.5.2.1. le logiciel OpenShift-monitoring/prometheus-operator-admission-webhook

Exposez le service webhook d'admission qui valide les ressources personnalisées PrometheusRules et AlertmanagerConfig sur le port 8443.

9.5.2.2. le logiciel OpenShift-user-workload-monitoring/alertmanager-user-workload

Exposez le serveur Web Alertmanager défini par l'utilisateur dans le cluster sur les ports suivants:

- Le port 9095 donne accès aux points de terminaison Alertmanager. L'octroi de l'accès nécessite de lier un utilisateur au rôle de suivi-alertmanager-api-lecteur (pour les opérations en lecture seule) ou le rôle de monitoring-alertmanager-api-writer dans le projet de surveillance de la charge de travail ouverte-utilisateur.
- Le port 9092 donne accès aux points de terminaison Alertmanager limités à un projet donné. L'octroi de l'accès nécessite de lier un utilisateur au rôle de cluster de surveillance-règles-édition ou de surveillance-modèle dans le projet.
- Le port 9097 donne accès au point final /metrics uniquement. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.3. contrôle OpenShift/alertmanager-main

Exposez le serveur Web Alertmanager dans le cluster sur les ports suivants:

- Le port 9094 donne accès à tous les points de terminaison Alertmanager. L'octroi d'un accès nécessite de lier un utilisateur au rôle de suivi-alertmanager-view (pour les opérations en lecture seule) ou de surveillance-alertmanager-modifier dans le projet de surveillance ouverte.
- Le port 9092 donne accès aux points de terminaison Alertmanager limités à un projet donné. L'octroi de l'accès nécessite de lier un utilisateur au rôle de cluster de surveillance-règles-édition ou de surveillance-modèle dans le projet.
- Le port 9097 donne accès au point final /metrics uniquement. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.4. caractéristiques OpenShift-monitoring/kube-state-metrics

Exposez les paramètres kube-state-metrics /metrics dans le cluster sur les ports suivants:

- Le port 8443 donne accès aux métriques de ressources Kubernetes. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.
- Le port 9443 donne accès aux métriques internes kube-state-metrics. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.5. le serveur OpenShift-monitoring/metrics-server

Exposez le serveur web métrique-serveur sur le port 443. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.6. le système OpenShift-monitoring/monitoring-plugin

Exposez le service de plugin de surveillance sur le port 9443. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.7. exportateur de nœuds OpenShift-monitoring

Exposez le point final /metrics sur le port 9100. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.8. le système OpenShift-monitoring/openshift-state-metrics

Exposez les points de terminaison openshift-state-metrics /metrics dans le cluster sur les ports suivants:

- Le port 8443 donne accès aux métriques de ressources OpenShift. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.
- Le port 9443 donne accès aux métriques internes à l'état ouvert. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.9. le système OpenShift-monitoring/prometheus-k8s

Exposez le serveur Web Prometheus dans le cluster sur les ports suivants:

- Le port 9091 permet d'accéder à tous les points de terminaison Prometheus. L'octroi d'un accès nécessite de lier un utilisateur au rôle du cluster cluster-monitoring-view.
- Le port 9092 donne accès aux paramètres /metrics et /federate uniquement. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.10. le logiciel OpenShift-user-workload-monitoring/prometheus-operator

Exposez le point final /metrics sur le port 8443. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.11. exploitant OpenShift-monitoring/prometheus-operator

Exposez le point final /metrics sur le port 8443. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.12. le logiciel OpenShift-user-workload-monitoring/prometheus-user-workload

Exposez le serveur Web Prometheus dans le cluster sur les ports suivants:

- Le port 9091 donne accès au point final /metrics uniquement. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.
- Le port 9092 donne accès au point d'extrémité /federate uniquement. L'octroi d'un accès nécessite de lier un utilisateur au rôle du cluster cluster-monitoring-view.

Cela expose également le point de terminaison /metrics du serveur web Thanos sidecar sur le port 10902. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.13. la surveillance OpenShift/télémètre-client

Exposez le point final /metrics sur le port 8443. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.14. le système OpenShift-monitoring/thanos-querier

Exposez le serveur web Thanos Querier dans le cluster sur les ports suivants:

- Le port 9091 permet d'accéder à tous les points de terminaison Thanos Querier. L'octroi d'un accès nécessite de lier un utilisateur au rôle du cluster cluster-monitoring-view.
- Le port 9092 donne accès aux /api/v1/query, /api/v1/query_range/, /api/v1/labels, /api/v1/label/*/values, et /api/v1/series limites à un projet donné. L'octroi d'un accès nécessite de lier un utilisateur au rôle de cluster de vue dans le projet.
- Le port 9093 donne accès aux points de terminaison /api/v1/alerts et /api/v1/rules limités à un projet donné. L'octroi de l'accès nécessite de lier un utilisateur au rôle de cluster de surveillance-règles-édition, de surveillance-edit ou de surveillance-réglementation dans le projet.
- Le port 9094 donne accès au point final /metrics uniquement. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.15. accueil > OpenShift-user-workload-monitoring/thanos-ruler

Exposez le serveur web Thanos Ruler dans le cluster sur les ports suivants:

- Le port 9091 donne accès à tous les points de terminaison Thanos Ruler. L'octroi d'un accès nécessite de lier un utilisateur au rôle du cluster cluster-monitoring-view.
- Le port 9092 donne accès au point final /metrics uniquement. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

Cela expose également les points d'extrémité gRPC sur le port 10901. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.5.2.16. exploitant OpenShift-monitoring/cluster-monitoring

Exposez les paramètres /metrics et /validate-webhook sur le port 8443. Ce port est destiné à un usage interne, et aucune autre utilisation n'est garantie.

9.6. RESSOURCES SUPPLÉMENTAIRES

- Configuration du stockage d'écriture à distance
- Gestion des métriques
- Gérer les alertes

CHAPITRE 10. DÉPANNAGE DES PROBLÈMES DE SURVEILLANCE

Découvrez les étapes de dépannage pour les problèmes courants avec la surveillance de projet définie par l'utilisateur.

10.1. DÉTERMINER POURQUOI LES MÉTRIQUES DE PROJET DÉFINIES PAR L'UTILISATEUR NE SONT PAS DISPONIBLES

Lorsque les métriques ne s'affichent pas lors de la surveillance des projets définis par l'utilisateur, suivez ces étapes pour résoudre le problème.

Procédure

1. Interrogez le nom métrique et vérifiez que le projet est correct:
   a. Dans la perspective Développeur de la console Web, cliquez sur Observer et allez dans l'onglet Metrics.
   b. Choisissez le projet pour lequel vous souhaitez afficher les métriques dans la liste Projet:
   c. Choisissez une requête existante dans la liste Sélectionner la requête, ou exécutez une requête personnalisée en ajoutant une requête PromQL au champ Expression.
   Les métriques sont affichées dans un graphique.

   Les requêtes doivent être faites sur une base par projet. Les métriques affichées concernent le projet que vous avez sélectionné.
2. Assurez-vous que le pod que vous voulez des métriques sert activement les métriques.
   Exécutez la commande oc exec suivante dans un pod pour cibler le podIP, le port et /metrics.

```
$ oc exec <sample_pod> -n <sample_namespace> -- curl <target_pod_IP>:<port>/metrics
```

NOTE

Il faut exécuter la commande sur un pod installé.

L'exemple suivant affiche un résultat avec une métrique de version valide.

Exemple de sortie

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
# HELP version Version information about this binary-- --:--:-- --:--:--     0
# TYPE version gauge
version{version="v0.1.0"} 1
100   102  100   102    0     0  51000      0 --:--:-- --:--:-- --:--:-- 51000
```

La sortie invalide indique qu'il y a un problème avec l'application correspondante.

3. Lorsque vous utilisez un CRD PodMonitor, vérifiez que le CRD PodMonitor est configuré pour pointer vers les bons pods à l'aide de la correspondance d'étiquette. Consultez la documentation de Prometheus Operator pour plus d'informations.

4. Lorsque vous utilisez un CRD ServiceMonitor, et si le point de terminaison /metrics de la pod affiche des données métriques, suivez ces étapes pour vérifier la configuration:

   a. Assurez-vous que le service est pointé vers le point de terminaison correct /metrics. Les étiquettes de service de cette sortie doivent correspondre aux étiquettes de surveillance des services et au point de terminaison /metrics défini par le service dans les étapes suivantes.

   ```
   $ oc get service
   ```

   Exemple de sortie

   ```
   apiVersion: v1
   kind: Service (1)
   metadata:
     labels: (2)
       app: prometheus-example-app
     name: prometheus-example-app
     namespace: ns1
   spec:
     ports:
     - port: 8080
       protocol: TCP
       targetPort: 8080
       name: web
     selector:
       app: prometheus-example-app
     type: ClusterIP
   ```

   (1) Indique qu'il s'agit d'une API de service.

   (2) Indique les étiquettes utilisées pour ce service.

   b. Interrogez les points de terminaison serviceIP, port et /metrics pour voir si les mêmes métriques de la commande curl que vous avez exécutée sur le pod précédemment:

      i. Exécutez la commande suivante pour trouver le service IP:

      ```
      $ oc get service -n <target_namespace>
      ```

      ii. Interrogez le point de terminaison /metrics:

      ```
      $ oc exec <sample_pod> -n <sample_namespace> -- curl <service_IP>:
      <port>/metrics
      ```

      Les métriques valides sont retournées dans l'exemple suivant.

      Exemple de sortie

      ```
      % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                     Dload  Upload   Total   Spent    Left  Speed
      100   102  100   102    0     0  51000      0 --:--:-- --:--:-- --:--:--   99k
      ```

```
# HELP version Version information about this binary
# TYPE version gauge
version{version="v0.1.0"} 1
```

c. Faites correspondre l'étiquette pour vérifier que l'objet ServiceMonitor est configuré pour pointer vers le service souhaité. À cette fin, comparez l'objet Service de l'oc obtient la sortie de service à l'objet ServiceMonitor à partir de la sortie Servicemonitor. Les étiquettes doivent correspondre pour que les métriques soient affichées.
À titre d'exemple, à partir des étapes précédentes, notez comment l'objet Service a l'application: l'étiquette prometheus-example-app et l'objet ServiceMonitor a la même application: prometheus-example-app label.

5. Lorsque tout semble valide et que les métriques sont toujours indisponibles, veuillez contacter l'équipe d'assistance pour obtenir de l'aide.

10.2. DÉTERMINER POURQUOI PROMETHEUS CONSOMME BEAUCOUP D'ESPACE DISQUE

Les développeurs peuvent créer des étiquettes pour définir des attributs pour les métriques sous la forme de paires clé-valeur. Le nombre de paires de valeurs clés potentielles correspond au nombre de valeurs possibles pour un attribut. L'attribut qui a un nombre illimité de valeurs potentielles s'appelle un attribut non lié. A titre d'exemple, un attribut customer_id est non lié parce qu'il a un nombre infini de valeurs possibles.

Chaque paire clé-valeur assignée a une série chronologique unique. L'utilisation de nombreux attributs non liés dans les étiquettes peut entraîner une augmentation exponentielle du nombre de séries chronologiques créées. Cela peut avoir un impact sur les performances de Prometheus et peut consommer beaucoup d'espace disque.

Lorsque Prometheus consomme beaucoup de disque, vous pouvez utiliser les mesures suivantes:

- Consultez l'état de la base de données des séries chronologiques (TSDB) à l'aide de l'API HTTP Prometheus pour plus d'informations sur les étiquettes qui créent le plus de données de séries chronologiques. Cela nécessite des privilèges d'administrateur de cluster.
- Consultez le nombre d'échantillons d'éraflures qui sont collectés.
- Diminuer le nombre de séries chronologiques uniques qui sont créées en réduisant le nombre d'attributs non liés qui sont attribués à des métriques définies par l'utilisateur.

NOTE

L'utilisation d'attributs liés à un ensemble limité de valeurs possibles réduit le nombre de combinaisons de paires clé-valeur potentielles.

- Appliquer des limites sur le nombre d'échantillons pouvant être grattés sur des projets définis par l'utilisateur. Cela nécessite des privilèges d'administrateur de cluster.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'OpenShift CLI (oc) a été installé.

Procédure

Procédure

1. Dans la perspective de l'administrateur, accédez à Observer → Metrics.
2. Entrez une requête Prometheus Query Language (PromQL) dans le champ Expression. Les requêtes d'exemple suivantes aident à identifier les métriques de haute cardinalité qui pourraient entraîner une consommation élevée d'espace disque:
   - En exécutant la requête suivante, vous pouvez identifier les dix emplois qui ont le plus grand nombre d'échantillons de raclette:

     ```
     topk(10, max by(namespace, job) (topk by(namespace, job) (1,
     scrape_samples_post_metric_relabeling)))
     ```

   - En exécutant la requête suivante, vous pouvez identifier les séries chronologiques en identifiant les dix emplois qui ont créé le plus de données de séries chronologiques au cours de la dernière heure:

     ```
     topk(10, sum by(namespace, job) (sum_over_time(scrape_series_added[1h])))
     ```

3. Étudier le nombre de valeurs d'étiquettes non liées attribuées aux mesures dont le nombre d'échantillons de griffes est plus élevé que prévu:
   - Lorsque les métriques se rapportent à un projet défini par l'utilisateur, examinez les paires clés-valeur attribuées à votre charge de travail. Ceux-ci sont mis en œuvre par l'intermédiaire des bibliothèques clientes Prometheus au niveau de l'application. Essayez de limiter le nombre d'attributs non liés référencés dans vos étiquettes.
   - Lorsque les métriques se rapportent à un projet de base OpenShift Dedicated, créez un cas d'assistance Red Hat sur le portail client Red Hat.
4. Examinez le statut TSDB à l'aide de l'API HTTP Prometheus en suivant ces étapes lorsqu'il est connecté en tant qu'administrateur dédié:
   a. Accédez à l'URL de route de l'API Prometheus en exécutant la commande suivante:

      ```
      $ HOST=$(oc -n openshift-monitoring get route prometheus-k8s -ojsonpath=
      {.status.ingress[].host})
      ```

   b. Extraire un jeton d'authentification en exécutant la commande suivante:

      ```
      $ TOKEN=$(oc whoami -t)
      ```

   c. Interrogez le statut TSDB pour Prometheus en exécutant la commande suivante:

      ```
      $ curl -H "Authorization: Bearer $TOKEN" -k "https://$HOST/api/v1/status/tsdb"
      ```

      Exemple de sortie

      ```
      "status": "success","data":{"headStats":{"numSeries":507473,
      "numLabelPairs":19832,"chunkCount":946298,"minTime":1712253600010,
      "maxTime":1712257935346},"seriesCountByMetricName":
      [{"name":"etcd_request_duration_seconds_bucket","value":51840},
      {"name":"apiserver_request_sli_duration_seconds_bucket","value":47718},
      ...
      ```

Ressources supplémentaires

- Accès aux API de surveillance en utilisant le CLI
- Fixer des intervalles de raclette, des intervalles d'évaluation et des limites imposées pour les projets définis par l'utilisateur
- Dépôt d'une affaire de soutien

10.3. LA RÉSOLUTION DU TIR D'ALERTE KUBEPERSISTENTVOLUMEFILLINGUP POUR PROMETHEUS

En tant qu'administrateur de cluster, vous pouvez résoudre l'alerte KubePersistentVolumeFillingUp déclenchée pour Prometheus.

L'alerte critique s'allume lorsqu'un volume persistant (PV) revendiqué par un pod prometheus-k8s-* dans le projet de surveillance à temps ouvert a moins de 3% d'espace total restant. Cela peut causer Prométhée à fonctionner anormalement.

NOTE

Il y a deux alertes KubePersistentVolumeFillingUp:

- Alerte critique: L'alerte avec l'étiquette sévérité="critique" est déclenchée lorsque le PV monté a moins de 3% d'espace total restant.
- Alerte: L'alerte avec l'étiquette sévérité="avertissement" est déclenchée lorsque le PV monté a moins de 15% d'espace total restant et devrait se remplir dans les quatre jours.

Afin de résoudre ce problème, vous pouvez supprimer les blocs Prometheus time-series (TSDB) pour créer plus d'espace pour le PV.

Conditions préalables

- En tant qu'utilisateur, vous avez accès au cluster avec le rôle d'administrateur dédié.
- L'OpenShift CLI (oc) a été installé.

Procédure

1. Énumérez la taille de tous les blocs TSDB, triés du plus ancien au plus récent, en exécutant la commande suivante:

```
$ oc debug <prometheus_k8s_pod_name> -n openshift-monitoring \ 1
-c prometheus --image=$(oc get po -n openshift-monitoring <prometheus_k8s_pod_name> \
2
-o jsonpath='{.spec.containers[?(@.name=="prometheus")].image}') \
-- sh -c 'cd /prometheus/;du -hs $(ls -dt */ | grep -Eo "[0-9|A-Z]{26}")'
```

1 2 <prometheus_k8s_pod_name> par le pod mentionné dans la description de l'alerte KubePersistentVolumeFillingUp.

Exemple de sortie

```
308M    01HVKMPKQWZYWS8WVDAYQHNMW6
52M     01HVK64DTDA81799TBR9QDECEZ
102M    01HVK64DS7TRZRWF2756KHST5X
140M    01HVJS59K11FBVAPVY57K88Z11
90M     01HVH2A5Z58SKT810EM6B9AT50
152M    01HV8ZDVQMX41MKCN84S32RRZ1
354M    01HV6Q2N26BK63G4RYTST71FBF
156M    01HV664H9J9Z1FTZD73RD1563E
216M    01HTHXB60A7F239HN7S2TENPNS
104M    01HTHMGRXGS0WXA3WATRXHR36B
```

2. Identifiez quels blocs et combien de blocs pourraient être supprimés, puis supprimez les blocs. La commande d'exemple suivante supprime les trois plus anciens blocs Prometheus TSDB de la boîte prométhée-k8s-0:

```
$ oc debug prometheus-k8s-0 -n openshift-monitoring \
-c prometheus --image=$(oc get po -n openshift-monitoring prometheus-k8s-0 \
-o jsonpath='{.spec.containers[?(@.name=="prometheus")].image}') \
-- sh -c 'ls -latr /prometheus/ | egrep -o "[0-9|A-Z]{26}" | head -3 | \
while read BLOCK; do rm -r /prometheus/$BLOCK; done'
```

3. Vérifiez l'utilisation du PV monté et assurez-vous qu'il y a suffisamment d'espace disponible en exécutant la commande suivante:

```
$ oc debug <prometheus_k8s_pod_name> -n openshift-monitoring \ (1)
--image=$(oc get po -n openshift-monitoring <prometheus_k8s_pod_name> \ (2)
-o jsonpath='{.spec.containers[?(@.name=="prometheus")].image}') -- df -h /prometheus/
```

(1) (2) <prometheus_k8s_pod_name> par le pod mentionné dans la description de l'alerte KubePersistentVolumeFillingUp.

L'exemple de sortie suivant montre le PV monté revendiqué par le pod prometheus-k8s-0 qui a 63% de l'espace restant:

Exemple de sortie

```
Starting pod/prometheus-k8s-0-debug-j82w4 ...
Filesystem      Size  Used Avail Use% Mounted on
/dev/nvme0n1p4  40G   15G  40G  37% /prometheus

Removing debug pod ...
```

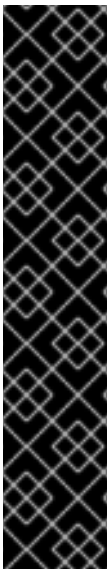
CHAPITRE 11. CONFIGURER LA RÉFÉRENCE DE LA CARTE POUR L'OPÉRATEUR DE SURVEILLANCE DU CLUSTER

11.1. CLUSTER MONITORING RÉFÉRENCE DE CONFIGURATION DE L'OPÉRATEUR

Les parties de la surveillance des clusters d'OpenShift sont configurables. L'API est accessible en définissant les paramètres définis dans diverses cartes de configuration.

- Afin de configurer les composants de surveillance qui surveillent les projets définis par l'utilisateur, modifiez l'objet ConfigMap nommé user-workload-monitoring-config dans l'espace de noms openshift-user-workload-monitoring. Ces configurations sont définies par UserWorkloadConfiguration.

Le fichier de configuration est toujours défini sous la clé config.yaml dans les données cartographiques de configuration.



IMPORTANT

- Les paramètres de configuration de la pile de surveillance ne sont pas tous exposés. Les paramètres et les champs listés dans cette référence sont pris en charge pour la configuration. Consultez Maintenance et support pour la surveillance pour plus d'informations sur les configurations prises en charge.
- La configuration de la surveillance des clusters est facultative.
- Lorsqu'une configuration n'existe pas ou est vide, les valeurs par défaut sont utilisées.
- Lorsque la configuration a des données YAML invalides, ou si elle contient des champs non pris en charge ou dupliqués qui ont contourné la validation précoce, l'opérateur de surveillance du cluster cesse de réconcilier les ressources et signale l'état dégradé=True dans les conditions de statut de l'opérateur.

11.2. COMPLÉMENTALERTMANAGERCONFIG

11.2.1. Description

La ressource SupplementAlertmanagerConfig définit les paramètres de la façon dont un composant communique avec des instances Alertmanager supplémentaires.

11.2.2. A) requis

- **apiVersion**

Apparaît dans: PrometheusK8sConfig, PrometheusRestrictedConfig, ThanosRulerConfig

La propriété	Le type	Description

La propriété	Le type	Description
apiVersion	chaîne de caractères	Définit la version API de Alertmanager. Les valeurs possibles sont v1 ou v2. La valeur par défaut est v2.
BearerToken	* v1.SecretKeySelector	Définit la référence clé secrète contenant le jeton porteur à utiliser lors de l'authentification à Alertmanager.
chemin de Préfixe	chaîne de caractères	Définit le préfixe de chemin à ajouter devant le chemin du point de terminaison push.
le plan d'action	chaîne de caractères	Définit le schéma d'URL à utiliser lors de la communication avec les instances Alertmanager. Les valeurs possibles sont http ou https. La valeur par défaut est http.
les Configs statiques	[]string	Liste des points de terminaison Alertmanager configurés statiquement sous la forme de <hosts>:<port>.
délai d'expiration	*string	Définit la valeur de temps d'attente utilisée lors de l'envoi d'alertes.
à propos de TlsConfig	À propos de TLSConfig	Définit les paramètres TLS à utiliser pour les connexions Alertmanager.

11.3. ALERTMANAGERMAINCONFIG

11.3.1. Description

La ressource AlertmanagerMainConfig définit les paramètres du composant Alertmanager dans l'espace de noms openshift-monitoring.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description

La propriété	Le type	Description
activé	*bool	Drapeau booléen qui active ou désactive l'instance principale Alertmanager dans l'espace de noms de surveillance openshift. La valeur par défaut est true.
enableUserAlertmanagerConfig	bool	Drapeau Boolean qui permet ou désactive les espaces de noms définis par l'utilisateur pour les recherches AlertmanagerConfig. Ce paramètre ne s'applique que si l'instance de surveillance de la charge de travail de l'utilisateur de Alertmanager n'est pas activée. La valeur par défaut est false.
LogLevel	chaîne de caractères	Définit le paramètre de niveau de journal pour Alertmanager. Les valeurs possibles sont: erreur, avertissement, info, débogue. La valeur par défaut est info.
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les Pods sont programmés.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur Alertmanager.
les secrets	[]string	Définit une liste de secrets à monter dans Alertmanager. Les secrets doivent résider dans le même espace de noms que l'objet Alertmanager. Ils sont ajoutés en tant que volumes nommés secret-<secret-name> et montés à /etc/alertmanager/secrets/<secret-name> dans le conteneur alertmanager des pods Alertmanager.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation topologiques d'un pod.

La propriété	Le type	Description
le volumeClaimTemplate	*monv1.EmbeddedPersistentVolumeClaim	Définit le stockage persistant pour Alertmanager. Ce paramètre permet de configurer la revendication de volume persistante, y compris la classe de stockage, la taille du volume et le nom.

11.4. ALERTMANAGERUSERWORKLOADCONFIG

11.4.1. Description

La ressource AlertmanagerUserWorkloadConfig définit les paramètres de l'instance Alertmanager utilisée pour les projets définis par l'utilisateur.

Apparaît dans: UserWorkloadConfiguration

La propriété	Le type	Description
activé	bool	Drapeau Boolean qui active ou désactive une instance dédiée de Alertmanager pour les alertes définies par l'utilisateur dans l'espace de noms openshift-user-workload-monitoring. La valeur par défaut est false.
activerAlertmanagerConfig	bool	Drapeau Boolean pour activer ou désactiver les espaces de noms définis par l'utilisateur à sélectionner pour la recherche AlertmanagerConfig. La valeur par défaut est false.
LogLevel	chaîne de caractères	Définit le paramètre de niveau de journal pour Alertmanager pour la surveillance de la charge de travail des utilisateurs. Les valeurs possibles sont l'erreur, l'avertissement, l'information et le débogue. La valeur par défaut est info.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur Alertmanager.

La propriété	Le type	Description
les secrets	[]string	Définit une liste de secrets à monter dans Alertmanager. Les secrets doivent être situés dans le même espace de noms que l'objet Alertmanager. Ils sont ajoutés en tant que volumes nommés secret-<secret-name> et montés à /etc/alertmanager/secrets/<secret-name> dans le conteneur alertmanager des pods Alertmanager.
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation topologiques d'un pod.
le volumeClaimTemplate	*monv1.EmbeddedPersistentVolumeClaim	Définit le stockage persistant pour Alertmanager. Ce paramètre permet de configurer la revendication de volume persistante, y compris la classe de stockage, la taille du volume et le nom.

11.5. CLUSTERMONITORINGCONFIGURATION

11.5.1. Description

La ressource ClusterMonitoringConfiguration définit les paramètres qui personnalisent la pile de surveillance de plate-forme par défaut via la carte de configuration cluster-monitoring-config dans l'espace de noms openshift-monitoring.

La propriété	Le type	Description
alertmanagerMain	*AlertmanagerMainConfig	AlertmanagerMainConfig définit les paramètres du composant Alertmanager dans l'espace de noms openshift-monitoring.

La propriété	Le type	Description
activeUserWorkload	*bool	UserWorkloadEnabled est un drapeau booléen qui permet la surveillance des projets définis par l'utilisateur.
charge de travail utilisateur	*UserWorkloadConfig	UserWorkload définit les paramètres de surveillance des projets définis par l'utilisateur.
kubeStateMetrics	*KubeStateMetricsConfig	KubeStateMetricsConfig définit les paramètres de l'agent kube-state-metrics.
serveur de métriques	*MetricsServerConfig	Le serveur Metrics définit les paramètres du composant Metrics Server.
♪ prometheusK8s ♪	* PrometheusK8sConfig	Le PrometheusK8sConfig définit les paramètres du composant Prometheus.
accueil > PrometheusOperator	* PrometheusOperatorConfig	Le PrometheusOperatorConfig définit les paramètres du composant Opérateur Prometheus.
accueil > PrometheusOperatorAdmissionWebhook	* PrometheusOperatorAdmissionWebhookConfig	Le logiciel PrometheusOperatorAdmissionWebhookConfig définit les paramètres du composant webhook d'admission de Prometheus Operator.
à propos de OpenshiftStateMetrics	*OpenShiftStateMetricsConfig	L'OpenShiftMetricsConfig définit les paramètres de l'agent openshift-state-metrics.
client de télémètre	*TelemeterClientConfig	Le TelemeterClientConfig définit les paramètres du composant Client Telemeter.
à propos de thanosQuerier	*ThanosQuerierConfig	Le ThanosQuerierConfig définit les paramètres du composant Thanos Querier.
Aéroport de NodeExporter	À propos de NodeExporterConfig	Le NodeExporterConfig définit les paramètres de l'agent node-exporter.

La propriété	Le type	Description
la surveillancePlugin	*MonitoringPluginConfig	La fonction MonitoringPluginConfig définit les paramètres du composant console-plugin de surveillance.

11.6. KUBESTATEMETRICSCONFIG

11.6.1. Description

La ressource KubeStateMetricsConfig définit les paramètres de l'agent kube-state-metrics.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur KubeStateMetrics.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation topologiques d'un pod.

11.7. À PROPOS DE METRICSSERVERCONFIG

11.7.1. Description

La ressource MetricsServerConfig définit les paramètres du composant Metrics Server.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
audit	* Audit	Définit la configuration d'audit utilisée par l'instance Metrics Server. Les valeurs possibles du profil sont les métadonnées, la demande, la réponse à la demande et l'absence. La valeur par défaut est Metadata.

La propriété	Le type	Description
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur Metrics Server.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation topologiques d'un pod.

11.8. ACCUEIL > MONITORINGPLUGINCONFIG

11.8.1. Description

La ressource MonitoringPluginConfig définit les paramètres du composant plugin de console Web dans l'espace de noms de surveillance openshift.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur console-plugin.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation topologiques d'un pod.

11.9. AJOUTER AU PANIERNODEEXPORTERCOLLECTORBUDDYINFOCONFIG

11.9.1. Description

La ressource NodeExporterCollectorBuddyInfoConfig fonctionne comme un interrupteur marche/arrêt pour le collecteur buddyinfo de l'agent de nœud-exportateur. Le collecteur buddyinfo est désactivé par défaut.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collectionneur buddyinfo.

11.10. AJOUTER AU PANIER NODEEXPORTERCOLLECTORCONFIG

11.10.1. Description

La ressource NodeExporterCollectorConfig définit les paramètres pour les collectionneurs individuels de l'agent de nœud-exportateur.

Apparaît dans: NodeExporterConfig

La propriété	Le type	Description
cpufreq	Accueil > NodeExporterCollectorCpufreqConfig	Définit la configuration du collecteur cpufreq, qui collecte les statistiques de fréquence CPU. Désactivé par défaut.
à propos de Tcpstat	Ajouter au panierNodeExporterCollectorTcpStatConfig	Définit la configuration du collecteur tcpstat, qui recueille les statistiques de connexion TCP. Désactivé par défaut.
à propos de Netdev	Ajouter au panierNodeExporterCollectorNetDevConfig	Définit la configuration du collecteur netdev, qui collecte les statistiques des périphériques réseau. Activé par défaut.
classe nette	Ajouter au panierNodeExporterCollectorNetClassConfig	Définit la configuration du collecteur netclass, qui recueille des informations sur les périphériques réseau. Activé par défaut.

La propriété	Le type	Description
Buddyinfo	Ajouter au panierNodeExporterCollectorBuddyInfoConfig	Définit la configuration du collecteur buddyinfo, qui recueille des statistiques sur la fragmentation de la mémoire à partir de la métrique node_buddyinfo_blocks. Cette métrique recueille des données à partir de /proc/buddyinfo. Désactivé par défaut.
les statuts de Mountstats	Ajouter au panierNodeExporterCollectorMountStatsConfig	Définit la configuration du collecteur Mountstats, qui recueille des statistiques sur les activités d'E/S de volume NFS. Désactivé par défaut.
ksmd	Ajouter au panier NodeExporterCollectorKSMDConfig	Définit la configuration du collecteur ksmd, qui recueille des statistiques à partir du démon de fusion de même page du noyau. Désactivé par défaut.
les processus	Ajouter au panierNodeExporterCollectorProcessesConfig	Définit la configuration du collecteur de processus, qui recueille des statistiques à partir de processus et de threads en cours d'exécution dans le système. Désactivé par défaut.
d'un système	Ajouter au panierNodeExporterCollectorSystemdConfig	Définit la configuration du collecteur système, qui recueille des statistiques sur le démon système et ses services gérés. Désactivé par défaut.

11.11. ACCUEIL > NODEEXPORTERCOLLECTORCPUFREQCONFIG

11.11.1. Description

La ressource NodeExporterCollectorCpufreqConfig permet d'activer ou de désactiver le collecteur cpufreq de l'agent node-exporteur. Le collecteur cpufreq est désactivé par défaut. Dans certaines circonstances, l'activation du collecteur cpufreq augmente l'utilisation du CPU sur les machines avec de nombreux cœurs. Lorsque vous activez ce collecteur et disposez de machines avec de nombreux cœurs, surveillez attentivement vos systèmes pour une utilisation excessive du CPU.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collecteur cpufreq.

11.12. AJOUTER AU PANIER NODEEXPORTERCOLLECTORKSMDCONFIG

11.12.1. Description

Utilisez la ressource NodeExporterCollectorKSMDConfig pour activer ou désactiver le collecteur ksmd de l'agent de nœud-exportateur. Le collecteur ksmd est désactivé par défaut.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collecteur ksmd.

11.13. AJOUTER AU PANIERNODEEXPORTERCOLLECTORMOUNTSTATSCONFIG

11.13.1. Description

La ressource NodeExporterCollectorMountStatsConfig permet d'activer ou de désactiver le collecteur Mountstats de l'agent de nœud-exportateur. Le collecteur Mountstats est désactivé par défaut. Lorsque vous activez le collecteur, les métriques suivantes deviennent disponibles : node_mountstats_nfs_read_bytes_total, node_mountstats_nfs_write_bytes_total, et node_mountstats_nfs_operations_requests_total. Attention à ce que ces métriques peuvent avoir une grande cardinalité. Lorsque vous activez ce collecteur, surveillez de près toute augmentation de l'utilisation de la mémoire pour les pods protéheus-k8s.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collectionneur de Mountstats.

11.14. AJOUTER AU PANIERNODEEXPORTERCOLLECTORNETCLASSCONFIG

11.14.1. Description

La ressource NodeExporterCollectorNetClassConfig permet d'activer ou de désactiver le collecteur

netclass de l'agent d'exportation de nœuds. Le collecteur netclass est activé par défaut. Lorsque vous désactivez ce collecteur, ces métriques deviennent indisponibles: node_network_info, node_network_address_type, node_network_carrier_carrier_carrier_changes_total, node_network_carrier_carrier_carrier_changes_total, node_network_carrier_carrier_changes_total, a) Node_network_device_id, node_network_dormant, node_network_flags, node_network_iface_id, node_network_iface_link, node_network_link_mode, node_network_mtu_bytes, la fonction node_network_name_assign_type, node_network_net_dev_group, node_network_speed_bytes, node_network_transmit_queue_length et node_network_protocol_type.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collecteur netclass.
à utiliserNetlink	bool	Drapeau booléen qui active l'implémentation de netlink du collecteur netclass. La valeur par défaut est true, qui active le mode netlink. Cette mise en œuvre améliore les performances du collecteur netclass.

11.15. AJOUTER AU PANIERNODEEXPORTERCOLLECTORNETDEVCONFIG

11.15.1. Description

La ressource NodeExporterCollectorNetDevConfig permet d'activer ou de désactiver le collecteur netdev de l'agent node-exporteur. Le collecteur netdev est activé par défaut. En cas de désactivation, ces métriques deviennent indisponibles: node_network_receive_bytes_total, node_network_receive_compressed_total, node_network_receive_drop_total, node_network_receive_errs_total, node_network_receive_fifo_total, aucun_network_receive_frame_total, node_network_receive_receive_total, node_network_receive_nohandler_total, node_network_receive_packets_total, node_network_bytes_total, node_network_transmit_carrier_total, la fonction node_network_transmit_colls_total, node_network_transmit_compressed_total, node_network_transmit_drop_total, node_network_transmit_errs_total, node_network_transmit_fifo_total et node_network_transmit_packets_total.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collecteur netdev.

11.16. AJOUTER AU PANIERNODEEXPORTERCOLLECTORPROCESSESCONFIG

11.16.1. Description

La ressource NodeExporterCollectorProcessesConfig permet d'activer ou de désactiver le collecteur de processus de l'agent d'exportation de nœuds. Lorsque le collecteur est activé, les métriques suivantes deviennent disponibles : node_processes_max_processes, node_processes_pids, node_processes_state, node_processes_threads, node_processes_threads_state. Les métriques node_processes_state et node_processes_threads_state peuvent avoir jusqu'à cinq séries chacune, en fonction de l'état des processus et des threads. Les états possibles d'un processus ou d'un thread sont: D (UNINTERRUPTABLE_SLEEP), R (RUNNING & RUNNABLE), S (INTERRUPTABLE_SLEEP), T (STOPPED) ou Z (ZOMBIE). Le collecteur de processus est désactivé par défaut.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collecteur de processus.

11.17. AJOUTER AU PANIERNODEEXPORTERCOLLECTORSYSTEMDCONFIG

11.17.1. Description

La ressource NodeExporterCollectorSystemdConfig permet d'activer ou de désactiver le collecteur système de l'agent de nœud-exportateur. Le collecteur système est désactivé par défaut. En cas d'activation, les métriques suivantes deviennent disponibles: node_systemd_running, node_systemd_units, node_systemd_version. Lorsque l'unité utilise une socket, elle génère également les métriques suivantes : node_systemd_socket_acceptd_connections_total, node_systemd_socket_current_connections, node_systemd_socket_refused_connections_total. Il est possible d'utiliser le paramètre unités pour sélectionner les unités système à inclure par le collecteur système. Les unités sélectionnées sont utilisées pour générer la métrique node_systemd_unit_state, qui montre l'état de chaque unité système. Cependant, la cardinalité de cette métrique pourrait être élevée (au moins cinq séries par unité par nœud). Lorsque vous activez ce collecteur à l'aide d'une longue liste d'unités sélectionnées, surveillez de près le déploiement promisheus-k8s pour une utilisation excessive de la mémoire. Il est à noter que la métrique node_systemd_timer_last_trigger_seconds n'est affichée que si vous avez configuré la valeur du paramètre unités en tant que logrotate.timer.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collecteur système.

La propriété	Le type	Description
les unités	[]string	Liste des modèles d'expression régulière (regex) qui correspondent aux unités système à inclure par le collecteur système. La liste par défaut est vide, de sorte que le collecteur n'expose aucune métrique pour les unités système.

11.18. AJOUTER AU PANIERNODEEXPORTERCOLLECTORTCPSTATCONFIG

11.18.1. Description

La ressource NodeExporterCollectorTcpStatConfig fonctionne comme un interrupteur marche/arrêt pour le collecteur tcpstat de l'agent de nœud-exportateur. Le collecteur tcpstat est désactivé par défaut.

Apparaît dans: NodeExporterCollectorConfig

La propriété	Le type	Description
activé	bool	Drapeau booléen qui active ou désactive le collecteur tcpstat.

11.19. À PROPOS DE NODEEXPORTERCONFIG

11.19.1. Description

La ressource NodeExporterConfig définit les paramètres de l'agent node-exporter.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
collectionneurs	Ajouter au panier NodeExporterCollectorConfig	Définit quels collecteurs sont activés et leurs paramètres de configuration supplémentaires.

La propriété	Le type	Description
les maxProcs	à propos de uint32	Le nombre cible de CPU sur lesquels le processus de l'exportateur de nœuds s'exécutera. La valeur par défaut est 0, ce qui signifie que l'exportateur de nœuds s'exécute sur tous les CPU. En cas d'impasse du noyau ou si les performances se dégradent lors de la lecture de sysfs simultanément, vous pouvez changer cette valeur à 1, ce qui limite l'exportation de nœuds à l'exécution sur un CPU. Dans le cas des nœuds avec un nombre élevé de CPU, vous pouvez définir la limite à un nombre faible, ce qui permet d'économiser des ressources en empêchant les routines Go d'être programmées pour s'exécuter sur tous les CPU. Cependant, les performances d'E/S se dégradent si la valeur maxProcs est trop faible et qu'il y a beaucoup de mesures à collecter.
ignoréNetworkDevices	*[]string	Liste des périphériques réseau, définis comme des expressions régulières, que vous souhaitez exclure de la configuration de collecteur pertinente tels que netdev et netclass. En l'absence de liste, l'opérateur de surveillance du cluster utilise une liste prédéfinie d'appareils à exclure pour minimiser l'impact sur l'utilisation de la mémoire. Dans le cas où la liste est vide, aucun appareil n'est exclu. Lorsque vous modifiez ce paramètre, surveillez de près le déploiement de promesseheus-k8s pour une utilisation excessive de la mémoire.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur NodeExporter.

11.20. À PROPOS DE OPENSHIFTSTATEMETRICSCONFIG

11.20.1. Description

La ressource OpenShiftStateMetricsConfig définit les paramètres de l'agent openshift-state-metrics.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur OpenShiftStateMetrics.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation de la topologie du pod.

11.21. À PROPOS DE PROMETHEUSK8SCONFIG

11.21.1. Description

La ressource PrometheusK8sConfig définit les paramètres du composant Prometheus.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
autresAlertmanagerConfigs	[]AdditionalAlertmanagerConfig	Configure les instances Alertmanager supplémentaires qui reçoivent des alertes du composant Prometheus. Aucune instance Alertmanager supplémentaire n'est configurée par défaut.

La propriété	Le type	Description
AppliedBodySizeLimit	chaîne de caractères	Applique une limite de taille de corps pour les métriques grattées de Prometheus. Lorsque la réponse corporelle d'une cible grattée est supérieure à la limite, la raclette échouera. Les valeurs suivantes sont valides: une valeur vide pour spécifier aucune limite, une valeur numérique au format de taille Prometheus (comme 64 Mo), ou la chaîne automatique, qui indique que la limite sera calculée automatiquement en fonction de la capacité du cluster. La valeur par défaut est vide, ce qui n'indique aucune limite.
laboratoires externes	carte[string]string	Définit les étiquettes à ajouter à n'importe quelle série chronologique ou alerte lors de la communication avec des systèmes externes tels que la fédération, le stockage à distance et Alertmanager. Aucune étiquette n'est ajoutée par défaut.
LogLevel	chaîne de caractères	Définit le paramètre de niveau de journal pour Prometheus. Les valeurs possibles sont: erreur, avertissement, info et débogage. La valeur par défaut est info.
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.

La propriété	Le type	Description
à propos de QueryLogFile	chaîne de caractères	Indique le fichier auquel les requêtes PromQL sont enregistrées. Ce paramètre peut être soit un nom de fichier, auquel cas les requêtes sont enregistrées dans un volume videDir à /var/log/prometheus, ou un chemin complet vers un emplacement où un volume videDir sera monté et les requêtes enregistrées. Écrire à /dev/stderr, /dev/stdout ou /dev/null est pris en charge, mais l'écriture sur un autre chemin /dev/ n'est pas prise en charge. Les chemins relatifs ne sont pas non plus pris en charge. Les requêtes PromQL ne sont pas enregistrées par défaut.
à distance d'écriture	[]MoteWriteSpec	Définit la configuration d'écriture à distance, y compris les paramètres d'URL, d'authentification et de relabeling.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur Prometheus.
la rétention	chaîne de caractères	Définit la durée pour laquelle Prometheus conserve les données. Cette définition doit être spécifiée à l'aide du modèle d'expression régulier suivant: [0-9]+(ms|m|m|h|d|w|y) (ms = millisecondes, s = secondes, m = minutes, h = heures, d = jours, w = semaines, y = années). La valeur par défaut est 15d.
conservationSize	chaîne de caractères	Définit la quantité maximale d'espace disque utilisée par les blocs de données plus le journal d'écriture-ahead (WAL). Les valeurs prises en charge sont B, KB, KiB, MB, MiB, GB, GiB, TB, TiB, PB, PiB, EB et EiB. Aucune limite n'est définie par défaut.

La propriété	Le type	Description
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation de la topologie du pod.
collectionProfile	CollectionProfile	Définit le profil de collecte des métriques que Prometheus utilise pour collecter des métriques à partir des composants de la plate-forme. Les valeurs prises en charge sont complètes ou minimales. Dans le profil complet (par défaut), Prometheus recueille toutes les métriques qui sont exposées par les composants de la plate-forme. Dans le profil minimal, Prometheus ne collecte que les métriques nécessaires pour les alertes de plate-forme par défaut, les règles d'enregistrement, la télémétrie et les tableaux de bord de console.
le volumeClaimTemplate	*monv1.EmbeddedPersistentVolumeClaim	Définit le stockage persistant pour Prometheus. Ce paramètre permet de configurer la revendication de volume persistante, y compris la classe de stockage, la taille du volume et le nom.

11.22. CLIQUEZ ICI POUR PROMETHEUSOPERATORCONFIG

11.22.1. Description

La ressource PrometheusOperatorConfig définit les paramètres du composant Opérateur Prometheus.

Apparaît dans: ClusterMonitoringConfiguration, UserWorkloadConfiguration

La propriété	Le type	Description

La propriété	Le type	Description
LogLevel	chaîne de caractères	Définit les paramètres de niveau de journal pour Prometheus Operator. Les valeurs possibles sont l'erreur, l'avertissement, l'information et le débogue. La valeur par défaut est info.
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur PrometheusOperator.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation de la topologie du pod.

11.23. INFORMATIONS SUR PROMETHEUSOPERATORADMISSIONWEBHOOKCONFIG

11.23.1. Description

La ressource PrometheusOperatorAdmissionWebhookConfig définit les paramètres de la charge de travail de webhook d'admission pour Prometheus Operator.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur prometheus-operator-admission-webhook.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation topologiques d'un pod.

11.24. CLIQUEZ ICI POUR PROMETHEUSRESTRICTEDCONFIG

11.24.1. Description

La ressource PrometheusRestrictedConfig définit les paramètres du composant Prometheus qui surveille les projets définis par l'utilisateur.

Apparaît dans: UserWorkloadConfiguration

La propriété	Le type	Description
griffeInterval	chaîne de caractères	Configure l'intervalle par défaut entre les raclettes consécutives dans le cas où la ressource ServiceMonitor ou PodMonitor ne spécifie aucune valeur. L'intervalle doit être réglé entre 5 secondes et 5 minutes. La valeur peut être exprimée en : secondes (par exemple 30s), minutes (par exemple 1m) ou un mélange de minutes et de secondes (par exemple 1m30). La valeur par défaut est 30s.
EvaluationInterval	chaîne de caractères	Configure l'intervalle par défaut entre les évaluations de règles dans le cas où la ressource PrometheusRule ne spécifie aucune valeur. L'intervalle doit être réglé entre 5 secondes et 5 minutes. La valeur peut être exprimée en : secondes (par exemple 30s), minutes (par exemple 1m) ou un mélange de minutes et de secondes (par exemple 1m30). Elle ne s'applique qu'aux ressources PrometheusRule avec l'étiquette openshift.io/prometheus-rule-evaluation-scope=\"leaf-prometheus\". La valeur par défaut est 30s.
autresAlertmanagerConfigs	[]AdditionalAlertmanagerConfig	Configure les instances Alertmanager supplémentaires qui reçoivent des alertes du composant Prometheus. Aucune instance Alertmanager supplémentaire n'est configurée par défaut.

La propriété	Le type	Description
FededLabelLimit	*uint64	Indique une limite par gratte sur le nombre d'étiquettes acceptées pour un échantillon. Lorsque le nombre d'étiquettes dépasse cette limite après le relabelage métrique, l'ensemble de la raclette est traitée comme ayant échoué. La valeur par défaut est 0, ce qui signifie qu'aucune limite n'est définie.
FededLabelNameLengthLimit	*uint64	Indique une limite par gratte sur la longueur d'un nom d'étiquette pour un échantillon. Lorsque la longueur d'un nom d'étiquette dépasse cette limite après le relabelage métrique, la totalité de la raclette est traitée comme ayant échoué. La valeur par défaut est 0, ce qui signifie qu'aucune limite n'est définie.
FededLabelValueLengthLimit	*uint64	Indique une limite par gratte sur la longueur d'une valeur d'étiquette pour un échantillon. Lorsque la longueur d'une valeur d'étiquette dépasse cette limite après relabeling métrique, la totalité de la raclette est traitée comme ayant échoué. La valeur par défaut est 0, ce qui signifie qu'aucune limite n'est définie.
AppliquéeSampleLimit	*uint64	Indique une limite globale du nombre d'échantillons grattés qui seront acceptés. Ce paramètre remplace la valeur SampleLimit définie dans n'importe quel objet ServiceMonitor ou PodMonitor défini par l'utilisateur si la valeur est supérieure à celle appliquéeTargetLimit. Les administrateurs peuvent utiliser ce paramètre pour garder le nombre total d'échantillons sous contrôle. La valeur par défaut est 0, ce qui signifie qu'aucune limite n'est définie.

La propriété	Le type	Description
applicationTargetLimit	*uint64	Indique une limite globale du nombre d'objectifs grattés. Ce paramètre remplace la valeur TargetLimit définie dans n'importe quel objet ServiceMonitor ou PodMonitor défini par l'utilisateur si la valeur est supérieure à celle appliquéeSampleLimit. Les administrateurs peuvent utiliser ce paramètre pour garder le nombre global de cibles sous contrôle. La valeur par défaut est 0.
laboratoires externes	carte[string]string	Définit les étiquettes à ajouter à n'importe quelle série chronologique ou alerte lors de la communication avec des systèmes externes tels que la fédération, le stockage à distance et Alertmanager. Aucune étiquette n'est ajoutée par défaut.
LogLevel	chaîne de caractères	Définit le paramètre de niveau de journal pour Prometheus. Les valeurs possibles sont l'erreur, l'avertissement, l'information et le débogue. Le paramètre par défaut est info.
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.

La propriété	Le type	Description
à propos de QueryLogFile	chaîne de caractères	Indique le fichier auquel les requêtes PromQL sont enregistrées. Ce paramètre peut être soit un nom de fichier, auquel cas les requêtes sont enregistrées dans un volume videDir à /var/log/prometheus, ou un chemin complet vers un emplacement où un volume videDir sera monté et les requêtes enregistrées. Écrire à /dev/stderr, /dev/stdout ou /dev/null est pris en charge, mais l'écriture sur un autre chemin /dev/ n'est pas prise en charge. Les chemins relatifs ne sont pas non plus pris en charge. Les requêtes PromQL ne sont pas enregistrées par défaut.
à distance d'écriture	[]MoteWriteSpec	Définit la configuration d'écriture à distance, y compris les paramètres d'URL, d'authentification et de relabeling.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur Prometheus.
la rétention	chaîne de caractères	Définit la durée pour laquelle Prometheus conserve les données. Cette définition doit être spécifiée à l'aide du modèle d'expression régulier suivant: [0-9]+(ms|m|m|h|d|w|y) (ms = millisecondes, s = secondes, m = minutes, h = heures, d = jours, w = semaines, y = années). La valeur par défaut est 24h.
conservationSize	chaîne de caractères	Définit la quantité maximale d'espace disque utilisée par les blocs de données plus le journal d'écriture-ahead (WAL). Les valeurs prises en charge sont B, KB, KiB, MB, MiB, GB, GiB, TB, TiB, PB, PiB, EB et EiB. La valeur par défaut est nulle.

La propriété	Le type	Description
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation de la topologie du pod.
le volumeClaimTemplate	*monv1.EmbeddedPersistentVolumeClaim	Définit le stockage persistant pour Prometheus. Ce paramètre permet de configurer la classe de stockage et la taille d'un volume.

11.25. CARACTÉRISTIQUES DE REMOTEWRITESPEC

11.25.1. Description

La ressource RemoteWriteSpec définit les paramètres du stockage d'écriture à distance.

11.25.2. A) requis

- **l'URL**

Apparaît dans: PrometheusK8sConfig, PrometheusRestrictedConfig

La propriété	Le type	Description
autorisation	*monv1.SafeAuthorisation	Définit les paramètres d'autorisation pour le stockage d'écriture à distance.
Basicauth	*monv1.BasicAuth	Définit les paramètres d'authentification de base pour l'URL du point de terminaison d'écriture à distance.
BearerTokenFile	chaîne de caractères	Définit le fichier qui contient le jeton porteur pour le point de terminaison d'écriture distante. Cependant, parce que vous ne pouvez pas monter des secrets dans un pod, dans la pratique, vous ne pouvez référencer que le jeton du compte de service.

La propriété	Le type	Description
en-têtes	carte[string]string	Indique les en-têtes HTTP personnalisés à envoyer avec chaque demande d'écriture à distance. Les en-têtes définis par Prometheus ne peuvent pas être écrasés.
les métadonnéesConfig	*monv1.MetadataConfig	Définit les paramètres pour l'envoi de métadonnées de série au stockage d'écriture à distance.
le nom	chaîne de caractères	Définit le nom de la file d'attente d'écriture à distance. Ce nom est utilisé dans les métriques et la journalisation pour différencier les files d'attente. En cas d'indication, ce nom doit être unique.
à proximité de Oauth2	*monv1.OAuth2	Définit les paramètres d'authentification OAuth2 pour le point de terminaison d'écriture distante.
le proxyUrl	chaîne de caractères	Définit une URL proxy optionnelle. Lorsque le proxy à l'échelle du cluster est activé, il remplace le paramètre proxyUrl. Le proxy à l'échelle du cluster prend en charge les proxy HTTP et HTTPS, avec HTTPS ayant préséance.
file d'attenteConfig	*monv1.QueueConfig	Autorise la configuration de réglage pour les paramètres de file d'écriture à distance.
distancetimeout	chaîne de caractères	Définit la valeur de temps d'attente pour les requêtes au point de terminaison de l'écriture distante.

La propriété	Le type	Description
envoyerExemplars	*bool	Active l'envoi d'exemples via l'écriture à distance. Lorsqu'il est activé, ce paramètre configure Prometheus pour stocker un maximum de 100 000 exemplaires en mémoire. Ce paramètre ne s'applique qu'à la surveillance définie par l'utilisateur et ne s'applique pas à la surveillance de base de la plate-forme.
à propos de Sigv4	*monv1.Sigv4	Définit les paramètres d'authentification AWS Signature Version 4.
à propos de TlsConfig	*monv1.SafeTLSConfig	Définit les paramètres d'authentification TLS pour le point de terminaison d'écriture à distance.
l'URL	chaîne de caractères	Définit l'URL du point de terminaison de l'écriture distante à laquelle des échantillons seront envoyés.
écrireRelabelConfigs	[]monv1.RelabelConfig	Définit la liste des configurations de relabel d'écriture à distance.

11.26. À PROPOS DE TLSCONFIG

11.26.1. Description

La ressource TLSConfig configure les paramètres des connexions TLS.

11.26.2. A) requis

- **insecureSkipVerify**

Apparaît dans: ExtraitAlertmanagerConfig

La propriété	Le type	Description

La propriété	Le type	Description
ca	* v1.SecretKeySelector	Définit la référence clé secrète contenant l'autorité de certification (CA) à utiliser pour l'hôte distant.
CERT	* v1.SecretKeySelector	Définit la référence clé secrète contenant le certificat public à utiliser pour l'hôte distant.
la clé	* v1.SecretKeySelector	Définit la référence de clé secrète contenant la clé privée à utiliser pour l'hôte distant.
le nom de serveur	chaîne de caractères	Il est utilisé pour vérifier le nom d'hôte sur le certificat retourné.
insecureSkipVerify	bool	Lorsqu'il est défini sur true, désactive la vérification du certificat et du nom de l'hôte distant.

11.27. FOURNISSEUR DE TELEMETERCLIENTCONFIG

11.27.1. Description

Le TelemeterClientConfig définit les paramètres du composant Client Telemeter.

11.27.2. A) requis

- **le nodeSelector**
- **les tolérances**

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur TelemeterClient.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.

La propriété	Le type	Description
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation de la topologie du pod.

11.28. À PROPOS DE THANOSQUERIERCONFIG

11.28.1. Description

La ressource ThanosQuerierConfig définit les paramètres du composant Thanos Querier.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description
activerRequestLogging	bool	Drapeau booléen qui active ou désactive l'enregistrement des requêtes. La valeur par défaut est false.
LogLevel	chaîne de caractères	Définit le paramètre de niveau de journal pour Thanos Querier. Les valeurs possibles sont l'erreur, l'avertissement, l'information et le débogue. La valeur par défaut est info.
activerCORS	bool	Drapeau booléen qui permet de configurer les en-têtes CORS. Les en-têtes permettent l'accès de n'importe quelle origine. La valeur par défaut est false.
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les pods sont programmés.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur Thanos Querier.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation de la topologie du pod.

11.29. À PROPOS DE THANOSRULERCONFIG

11.29.1. Description

La ressource ThanosRulerConfig définit la configuration de l'instance Thanos Ruler pour les projets définis par l'utilisateur.

Apparaît dans: UserWorkloadConfiguration

La propriété	Le type	Description
autresAlertmanagerConfigs	[]AdditionalAlertmanagerConfig	Configure la façon dont le composant Thanos Ruler communique avec des instances Alertmanager supplémentaires. La valeur par défaut est nulle.
EvaluationInterval	chaîne de caractères	Configure l'intervalle par défaut entre les évaluations des règles Prometheus dans le cas où la ressource PrometheusRule ne spécifie aucune valeur. L'intervalle doit être réglé entre 5 secondes et 5 minutes. La valeur peut être exprimée en : secondes (par exemple 30s), minutes (par exemple 1m) ou un mélange de minutes et de secondes (par exemple 1m30). Il s'applique aux ressources PrometheusRule sans l'étiquette openshift.io/prometheus-rule-évaluation-scope=\"leaf-prometheus\". La valeur par défaut est 15s.
LogLevel	chaîne de caractères	Définit le paramètre de niveau de journal pour Thanos Ruler. Les valeurs possibles sont l'erreur, l'avertissement, l'information et le débogue. La valeur par défaut est info.
le nodeSelector	carte[string]string	Définit les nœuds sur lesquels les Pods sont programmés.
les ressources	* v1.ResourceRequis	Définit les demandes de ressources et les limites pour le conteneur Alertmanager.

La propriété	Le type	Description
la rétention	chaîne de caractères	Définit la durée pour laquelle Prometheus conserve les données. Cette définition doit être spécifiée à l'aide du modèle d'expression régulier suivant: [0-9]+(ms|m|m|h|d|w|y) (ms = millisecondes, s = secondes, m = minutes, h = heures, d = jours, w = semaines, y = années). La valeur par défaut est 15d.
les tolérances	[]v1.Tolérance	Définit les tolérances pour les pods.
la topologieSpreadConstraints	[]v1.TopologySpreadConstraint	Définit les contraintes de propagation de la topologie du pod.
le volumeClaimTemplate	*monv1.EmbeddedPersistentVolumeClaim	Définit le stockage persistant pour Thanos Ruler. Ce paramètre permet de configurer la classe de stockage et la taille d'un volume.

11.30. CONTENU DE USERWORKLOADCONFIG

11.30.1. Description

La ressource UserWorkloadConfig définit les paramètres pour la surveillance des projets définis par l'utilisateur.

Apparaît dans: ClusterMonitoringConfiguration

La propriété	Le type	Description

La propriété	Le type	Description
des règles sans labelEnforcementAllowed	*bool	Drapeau booléen qui active ou désactive la possibilité de déployer des objets PrometheusRules définis par l'utilisateur pour lesquels l'étiquette de l'espace de noms n'est pas appliquée à l'espace de noms de l'objet. Ces objets doivent être créés dans un espace de noms configuré sous la propriété namespaces WithoutLabelEnforcement de la ressource UserWorkloadConfiguration. La valeur par défaut est true.

11.31. CONFIGURATION DE USERWORKLOAD

11.31.1. Description

La ressource UserWorkloadConfiguration définit les paramètres responsables des projets définis par l'utilisateur dans la carte de configuration de configuration de l'utilisateur-workload-monitoring dans l'espace de noms openshift-user-workload-monitoring. La configuration UserWorkload ne peut être activée qu'une fois que vous avez défini l'activationUserWorkload sur true dans la carte de configuration cluster-monitoring-config sous l'espace de noms openshift-monitoring.

La propriété	Le type	Description
Alertmanager	*AlertmanagerUserWorkloadConfig	Définit les paramètres du composant Alertmanager dans la surveillance de la charge de travail de l'utilisateur.
le Prométhée	* PrometheusRestrictedConfig	Définit les paramètres du composant Prometheus dans la surveillance de la charge de travail de l'utilisateur.
accueil > PrometheusOperator	* PrometheusOperatorConfig	Définit les paramètres du composant Prometheus Operator dans la surveillance de la charge de travail de l'utilisateur.
le thanosRuler	*ThanosRulerConfig	Définit les paramètres du composant Thanos Ruler dans la surveillance de la charge de travail de l'utilisateur.

La propriété	Le type	Description
espaces de noms SansbelEnforcement	[]string	Définit la liste des espaces de noms pour lesquels Prometheus et Thanos Ruler dans la surveillance définie par l'utilisateur n'appliquent pas la valeur d'étiquette de l'espace de noms dans les objets PrometheusRule. La propriété namespaces WithoutLabelEnforcement permet aux utilisateurs de définir des règles d'enregistrement et d'alerte qui peuvent interroger plusieurs projets (non limités aux projets définis par l'utilisateur) au lieu de déployer des objets PrometheusRule identiques dans chaque projet utilisateur. Afin de rendre visibles les alertes et les métriques résultantes pour les utilisateurs du projet, les expressions de requête doivent renvoyer une étiquette d'espace de noms avec une valeur non vide.