Red Hat OpenShift Service on AWS 4

사용자 정의 프로젝트 모니터링

AWS에서 Red Hat OpenShift Service의 프로젝트 모니터링

Last Updated: 2023-07-12

Red Hat OpenShift Service on AWS 4 사용자 정의 프로젝트 모니터링

AWS에서 Red Hat OpenShift Service의 프로젝트 모니터링

법적 공지

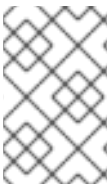
초록

이 문서에서는 ROSA(Red Hat OpenShift Service on AWS)에서 프로젝트 모니터링에 대한 정보를 제공합니다.

차례

1장. 모니터링 스택 이해

AWS의 Red Hat OpenShift Service에서는 Red Hat 사이트 안정성 엔지니어(SRE) 플랫폼 메트릭과는 별도로 자체 프로젝트를 모니터링할 수 있습니다. 추가 모니터링 솔루션 없이도 자체 프로젝트를 모니터링할 수 있습니다.



참고

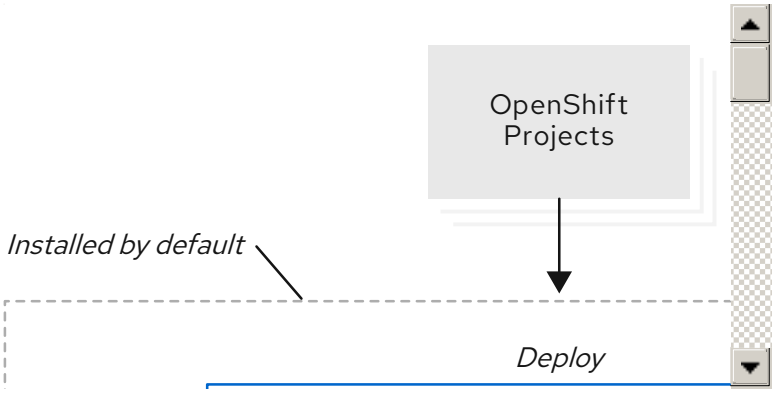
이 문서의 지침에 따라 사용자 정의 프로젝트를 모니터링하기 위해 지원되는 Prometheus 인스턴스를 구성합니다. 사용자 정의 Prometheus 인스턴스는 AWS의 Red Hat OpenShift Service에서 지원되지 않습니다.

1.1. 모니터링 스택 이해

Red Hat OpenShift Service on AWS(ROSA) 모니터링 스택은 Prometheus 오픈 소스 프로젝트와 광범위한 에코시스템을 기반으로 합니다. 모니터링 스택에는 다음이 포함됩니다.

- **기본 플랫폼 모니터링 구성 요소**입니다. 플랫폼 모니터링 구성 요소 세트는 **openshift-monitoring** 프로젝트에 설치되고 ROSA 설치 중에 기본적으로 활성화됩니다. 이를 통해 핵심 클러스터 구성 요소를 모니터링할 수 있습니다. 기본 모니터링 스택은 클러스터에 대한 원격 상태 모니터링도 가능합니다. CPU 및 메모리와 같은 중요한 메트릭은 모든 네임스페이스의 모든 워크로드에서 수집되며 사용 가능합니다.
  이러한 구성 요소는 다음 다이어그램의 **기본적으로 설치됨** 섹션에 설명되어 있습니다.

- **사용자 정의 프로젝트를 모니터링하기 위한 구성 요소**입니다. 이 기능은 기본적으로 활성화되어 있으며 사용자 정의 프로젝트에 대한 모니터링을 제공합니다. 이러한 구성 요소는 다음 다이어그램의 **사용자** 섹션에 설명되어 있습니다.



1.1.1. 사용자 정의 프로젝트를 모니터링하기 위한 구성 요소

AWS의 Red Hat OpenShift Service에는 사용자 정의 프로젝트에서 서비스 및 Pod를 모니터링할 수 있는 모니터링 스택에 선택적 기능이 포함되어 있습니다. 이 기능에는 다음과 같은 구성 요소가 포함됩니다.

표 1.1. 사용자 정의 프로젝트를 모니터링하기 위한 구성 요소

구성 요소	설명
Prometheus Operator	**openshift-user-workload-monitoring** 프로젝트의 Prometheus Operator는 동일한 프로젝트에서 Prometheus 및 Thanos Ruler 인스턴스를 생성, 구성 및 관리합니다.

구성 요소	설명
Prometheus	Prometheus는 사용자 정의 프로젝트에 대한 모니터링이 제공되는 모니터링 시스템입니다. Prometheus는 처리를 위해 Alertmanager에 경고를 보냅니다. 그러나 현재 경고 라우팅은 지원되지 않습니다.
Thanos Ruler	Thanos Ruler는 별도의 프로세스로 배포되는 Prometheus의 규칙 평가 엔진입니다. Red Hat OpenShift Service on AWS 4에서 Thanos Ruler는 사용자 정의 프로젝트의 모니터링에 대한 규칙 및 경고 평가를 제공합니다.

이러한 모든 구성 요소는 스택에서 모니터링되며 AWS의 Red Hat OpenShift Service가 업데이트되면 자동으로 업데이트됩니다.

1.1.2. 사용자 정의 프로젝트의 대상 모니터링

모니터링은 AWS 사용자 정의 프로젝트의 Red Hat OpenShift Service에 기본적으로 활성화됩니다. 다음을 모니터링할 수 있습니다.

- 사용자 정의 프로젝트에서 서비스 끝점을 통해 제공되는 메트릭입니다.
- 사용자 정의 프로젝트에서 실행 중인 Pod.

1.2. 추가 리소스

- 사용자 정의 프로젝트 모니터링에 액세스
- 기본 모니터링 구성 요소
- 기본 모니터링 대상

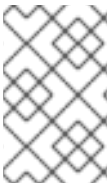
1.3. 다음 단계

- 사용자 정의 프로젝트 모니터링에 액세스

2장. 사용자 정의 프로젝트 모니터링에 액세스

ROLE(Red Hat OpenShift Service on AWS) 클러스터를 설치하면 기본적으로 사용자 정의 프로젝트에 대한 모니터링이 활성화됩니다. 사용자 정의 프로젝트를 모니터링하면 추가 모니터링 솔루션 없이도 자체 ROSA 프로젝트를 모니터링할 수 있습니다.

dedicated-admin 사용자에게는 사용자 정의 프로젝트에 대한 모니터링을 구성하고 액세스할 수 있는 기본 권한이 있습니다.



참고

OLM(Operator Lifecycle Manager)을 통해 설치한 사용자 정의 Prometheus 인스턴스 및 Prometheus Operator는 활성화된 경우 사용자 정의 프로젝트 모니터링에 문제가 발생할 수 있습니다. 사용자 정의 Prometheus 인스턴스는 지원되지 않습니다.

필요한 경우 클러스터 설치 중 또는 설치 후 사용자 정의 프로젝트에 대한 모니터링을 비활성화할 수 있습니다.

2.1. 다음 단계

- 모니터링 스택 구성

3장. 모니터링 스택 구성

모니터링 스택을 구성한 후 일반적인 구성 시나리오를 검토하고 사용자 정의 프로젝트의 모니터링을 구성할 수 있습니다.

3.1. 모니터링의 유지보수 및 지원

지원되는 Red Hat OpenShift Service on AWS Monitoring 구성 방법은 이 문서에 설명된 옵션을 사용하는 것입니다. **다른 구성은 지원되지 않으므로 사용하지 마십시오.**

중요

다른 Prometheus 인스턴스 설치는 Red Hat 사이트 안정성 엔지니어(SREs)에서 지원되지 않습니다.

구성 패러다임은 Prometheus 릴리스마다 변경될 수 있으며 이러한 경우는 모든 구성 가능성이 제어되는 경우에만 정상적으로 처리될 수 있습니다. 이 섹션에 설명된 것과 다른 구성을 사용하는 경우 **cluster-monitoring-operator**가 차이를 조정하므로 변경한 내용이 사라집니다. Operator는 원래 기본적으로 모든 항목이 정의된 상태로 재설정합니다.

3.1.1. 사용자 정의 프로젝트 모니터링에 대한 지원 고려 사항

다음과 같은 수정 사항은 명시적으로 지원되지 않습니다.

- AWS의 Red Hat OpenShift Service에 사용자 정의 Prometheus 인스턴스 설치

3.2. 모니터링 스택 구성

AWS의 Red Hat OpenShift Service에서는 **user-workload-monitoring-config ConfigMap** 오브젝트를 사용하여 사용자 정의 프로젝트의 워크로드를 모니터링하는 스택을 구성할 수 있습니다. 구성 맵은 CCMO(Cluster Monitoring Operator)를 구성한 다음 스택의 구성 요소를 구성합니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **user-workload-monitoring-config ConfigMap** 오브젝트가 생성되어 있습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

프로세스

1. **ConfigMap** 오브젝트를 편집합니다.
   a. **openshift-user-workload-monitoring** 프로젝트에서 **user-workload-monitoring-config ConfigMap** 오브젝트를 편집합니다.

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

   b. **data.config.yaml** 아래의 구성을 키-값 쌍< **component_name>: <component_configuration** >으로 추가합니다.

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

이에 따라 **<component>** 및 **<configuration_for_the_component>**를 바꿉니다.

다음 예제 **ConfigMap** 오브젝트는 Prometheus에 대한 데이터 보존 기간 및 최소 컨테이너 리소스 요청을 구성합니다. 이는 사용자 정의 프로젝트만 모니터링하는 Prometheus 인스턴스와 관련이 있습니다.

```
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. Prometheus 구성 요소를 정의하면 후속 행은 해당 구성을 정의합니다.
2. 사용자 정의 프로젝트를 모니터링하는 Prometheus 인스턴스에 대해 24시간 데이터 보존 기간을 구성합니다.
3. Prometheus 컨테이너에 대한 200밀리코어의 최소 리소스 요청을 정의합니다.
4. Prometheus 컨테이너에 대한 메모리 2GiB의 최소 Pod 리소스 요청을 정의합니다.

2. 파일을 저장하여 **ConfigMap** 오브젝트에 대한 변경 사항을 적용합니다. 새 구성의 영향을 받는 Pod가 자동으로 다시 시작됩니다.

주의

모니터링 구성 맵에 변경 사항이 저장되면 관련 프로젝트의 Pod 및 기타 리소스가 재배포될 수 있습니다. 해당 프로젝트에서 실행 중인 모니터링 프로세스도 다시 시작할 수 있습니다.

3.3. 구성 가능한 모니터링 구성 요소

이 표는 구성할 수 있는 모니터링 구성 요소와 **user-workload-monitoring-config ConfigMap** 오브젝트에서 구성 요소를 지정하는 데 사용되는 키를 보여줍니다.

표 3.1. 구성 가능한 모니터링 구성 요소

구성 요소	user-workload-monitoring-config 구성 맵 키
Prometheus Operator	**prometheusOperator**
Prometheus	**prometheus**
Thanos Ruler	**thanosRuler**

3.4. 다른 노드로 모니터링 구성 요소 이동

사용자 정의 프로젝트의 워크로드를 모니터링하는 구성 요소를 특정 작업자 노드로 이동할 수 있습니다. 구성 요소를 컨트롤 플레인 또는 인프라 노드로 이동할 수 없습니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **user-workload-monitoring-config ConfigMap** 오브젝트가 생성되어 있습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

절차

1. 사용자 정의 프로젝트를 모니터링하는 구성 요소를 이동하려면 **ConfigMap** 오브젝트를 편집합니다.
   a. **openshift-user-workload-monitoring** 프로젝트에서 **user-workload-monitoring-config ConfigMap** 오브젝트를 편집합니다.

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

   b. **data.config.yaml** 에서 구성 요소에 대한 **nodeSelector** 제약 조건을 지정합니다.

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

이에 따라 **<component>**를 바꾸고 **<node_key>: <node_value>**를 대상 노드를 지정하는 키-값 쌍의 맵으로 바꿉니다. 종종 단일 키-값 쌍만 사용됩니다.

구성 요소는 각 지정된 키-값 쌍을 라벨로 갖는 노드에서만 실행할 수 있습니다. 노드에도 추가 라벨이 있을 수 있습니다.

중요

대부분의 모니터링 구성 요소는 클러스터의 다른 노드에서 여러 Pod를 사용하여 고가용성을 유지함으로써 배포됩니다. 모니터링 구성 요소를 라벨된 노드로 이동할 때 구성 요소의 탄력성을 유지하기 위해 일치하는 노드를 사용할 수 있는지 확인합니다. 하나의 라벨만 지정된 경우 별도의 노드 간에 구성 요소에 대한 모든 Pod를 배포할 수 있는 충분한 노드에 해당 라벨이 포함되어 있는지 확인합니다. 또는 개별 노드와 관련된 각 라벨을 여러 개 지정할 수 있습니다.

참고

nodeSelector 제약 조건을 구성한 후 모니터링 구성 요소가 **Pending** 상태인 경우 테인트(Taints) 및 톨러레이션(Tolerations)과 관련된 오류에 대한 Pod 로그를 확인합니다.

예를 들어 사용자 정의 프로젝트의 구성 요소를 **nodename: worker1**, **nodename: worker2** 및 **nodename: worker2**의 라벨이 지정된 특정 작업자 노드로 이동하려면 다음을 사용합니다.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheusOperator:
      nodeSelector:
        nodename: worker1
    prometheus:
      nodeSelector:
        nodename: worker1
        nodename: worker2
    thanosRuler:
      nodeSelector:
        nodename: worker1
        nodename: worker2
```

2. 파일을 저장하여 변경 사항을 적용합니다. 새 구성의 영향을 받는 구성 요소는 새 노드로 자동 이동됩니다.

주의

모니터링 구성 맵에 변경 사항이 저장되면 관련 프로젝트의 Pod 및 기타 리소스가 재배포될 수 있습니다. 해당 프로젝트에서 실행 중인 모니터링 프로세스도 다시 시작할 수 있습니다.

3.5. 사용자 정의 프로젝트를 모니터링하는 구성 요소에 허용 오차 할당

사용자 정의 프로젝트를 모니터링하는 구성 요소에 허용 오차를 할당하여 테인트된 작업자 노드로 이동할 수 있습니다. 컨트롤 플레인 또는 인프라 노드에서는 예약이 허용되지 않습니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **openshift-user-workload-monitoring** 네임스페이스에 **user-workload-monitoring-config ConfigMap** 오브젝트를 생성했습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

절차

1. **ConfigMap** 오브젝트를 편집합니다.
   a. **openshift-user-workload-monitoring** 프로젝트에서 **user-workload-monitoring-config ConfigMap** 오브젝트를 편집합니다.

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

   b. 구성 요소에 대한 **tolerations**를 지정합니다.

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

   이에 따라 **<component>** 및 **<toleration_specification>**을 바꿉니다.

   예를 들어 **oc adm taint nodes node1 key1=value1:NoSchedule**은 **key1**의 키와 **value1**의 값이 있는 **node1**에 테인트를 추가합니다. 이렇게 하면 해당 테인트에 허용 오차가 구성되지 않는 한 **node1**에 모니터링 구성 요소가 배포되지 않습니다. 다음 예제는 예제 테인트를 허용하도록 **thanosRuler** 구성 요소를 구성합니다.

```
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"
```

2. 파일을 저장하여 변경 사항을 적용합니다. 새로운 구성 요소 배치 구성이 자동으로 적용됩니다.

주의

모니터링 구성 맵에 변경 사항이 저장되면 관련 프로젝트의 Pod 및 기타 리소스가 재배포될 수 있습니다. 해당 프로젝트에서 실행 중인 모니터링 프로세스도 다시 시작할 수 있습니다.

추가 리소스

- 테인트(Taints) 및 톨러레이션(Tolerations)에 대한 OpenShift Container Platform 문서를 참조하십시오.
- 테인트(Taints) 및 톨러레이션(Tolerations)에 대한 Kubernetes 문서를 참조하십시오.

3.6. 영구 스토리지 구성

영구 스토리지로 클러스터 모니터링을 실행하면 메트릭이 PV(영구 볼륨)에 저장되며 Pod를 다시 시작하거나 재생성할 수 있습니다. 메트릭 데이터가 데이터 손실에서 보호되어야 하는 경우 이상적입니다. 프로덕션 환경의 경우 영구 스토리지를 구성하는 것이 매우 좋습니다. 높은 IO 요구로 인해 로컬 스토리지를 사용하는 것이 이점이 됩니다.

중요

권장되는 구성 가능한 스토리지 기술을 참조하십시오.

3.6.1. 영구 스토리지 사전 요구 사항

- 스토리지의 블록 유형을 사용합니다.

3.6.2. 로컬 영구 볼륨 클레임 구성

PV(영구 볼륨)를 사용하기 위한 구성 요소를 모니터링하는 경우 PVC(영구 볼륨 클레임)를 구성해야 합니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **user-workload-monitoring-config ConfigMap** 오브젝트가 생성되어 있습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

절차

1. 사용자 정의 프로젝트를 모니터링하는 구성 요소의 PVC를 구성하려면 **ConfigMap** 오브젝트를 편집합니다.

   a. **openshift-user-workload-monitoring** 프로젝트에서 **user-workload-monitoring-config ConfigMap** 오브젝트를 편집합니다.

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

   b. **data.config.yaml** 아래의 구성 요소에 대한 PVC 구성을 추가합니다.

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

   volumeClaimTemplate을 지정하는 방법에 대한 내용은 **PersistentVolumeClaims에 대한 Kubernetes 문서**를 참조하십시오.

   다음 예제는 사용자 정의 프로젝트를 모니터링하는 Prometheus 인스턴스의 로컬 영구 스토리지를 요청하는 PVC를 구성합니다.

   ```
   apiVersion: v1
   kind: ConfigMap
   metadata:
     name: user-workload-monitoring-config
     namespace: openshift-user-workload-monitoring
   data:
     config.yaml: |
       prometheus:
         volumeClaimTemplate:
           spec:
             storageClassName: local-storage
             resources:
               requests:
                 storage: 40Gi
   ```

위의 예에서 Local Storage Operator에 의해 생성된 스토리지 클래스를 **local-storage**라고 합니다.

다음 예제에서는 Thanos Ruler의 로컬 영구 스토리지를 요청하는 PVC를 구성합니다.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    thanosRuler:
      volumeClaimTemplate:
        spec:
          storageClassName: local-storage
          resources:
            requests:
              storage: 40Gi
```

2. 파일을 저장하여 변경 사항을 적용합니다. 새 구성의 영향을 받는 Pod가 자동으로 다시 시작되고 새 스토리지 구성이 적용됩니다.

> **주의**
>
> 모니터링 구성 맵에 변경 사항이 저장되면 관련 프로젝트의 Pod 및 기타 리소스가 재배포될 수 있습니다. 해당 프로젝트에서 실행 중인 모니터링 프로세스도 다시 시작할 수 있습니다.

3.6.3. Prometheus 메트릭 데이터의 보존 시간 수정

기본적으로 AWS 모니터링 스택의 Red Hat OpenShift Service는 Prometheus 데이터의 보존 시간을 15일로 구성합니다. 사용자 정의 프로젝트를 모니터링하는 Prometheus 인스턴스의 보존 시간을 수정하여 데이터 삭제 시간을 변경할 수 있습니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **user-workload-monitoring-config ConfigMap** 오브젝트가 생성되어 있습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

절차

1. 사용자 정의 프로젝트를 모니터링하는 Prometheus 인스턴스의 보존 시간을 수정하려면 **ConfigMap** 오브젝트를 편집합니다.

a. **openshift-user-workload-monitoring** 프로젝트에서 **user-workload-monitoring-config ConfigMap** 오브젝트를 편집합니다.

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

b. **data.config.yaml** 아래에 보존 시간 구성을 추가합니다.

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

<time_specification>을 **ms**(밀리초), **s**(초), **m**(분), **h**(시간), **d**(일), **w**(주) 또는 **y**(년)가 바로 따라오는 숫자로 바꿉니다.

다음 예제에서는 사용자 정의 프로젝트를 모니터링하는 Prometheus 인스턴스의 보존 시간을 24시간으로 설정합니다.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      retention: 24h
```

2. 파일을 저장하여 변경 사항을 적용합니다. 새 구성의 영향을 받는 Pod가 자동으로 다시 시작됩니다.

주의

모니터링 구성 맵에 변경 사항이 저장되면 관련 프로젝트의 Pod 및 기타 리소스가 재배포될 수 있습니다. 해당 프로젝트에서 실행 중인 모니터링 프로세스도 다시 시작할 수 있습니다.

추가 리소스

- 영구저장장치 이해
- 스토리지 최적화

3.7. 사용자 정의 프로젝트에서 바인딩되지 않은 메트릭 속성의 영향 제어

개발자는 라벨을 생성하여 키-값 쌍의 형식으로 메트릭의 속성을 정의할 수 있습니다. 잠재적인 키-값 쌍의 수는 속성에 사용 가능한 값의 수에 해당합니다. 무제한의 잠재적인 값이 있는 속성을 바인딩되지 않은 속성이라고 합니다. 예를 들어, **customer_id** 속성은 무제한 가능한 값이 있기 때문에 바인딩되지 않은 속성입니다.

할당된 모든 키-값 쌍에는 고유한 시계열이 있습니다. 라벨에 있는 바인딩되지 않은 많은 속성을 사용하면 생성되는 시계열 수가 기하급수적으로 증가할 수 있습니다. 이는 Prometheus 성능에 영향을 미칠 수 있으며 많은 디스크 공간을 소비할 수 있습니다.

dedicated-admin 은 다음 방법을 사용하여 사용자 정의 프로젝트에서 바인딩되지 않은 메트릭 속성의 영향을 제어할 수 있습니다.

- 사용자 정의 프로젝트에서 대상 스크랩별로 **허용 가능한 샘플 수 제한**

참고

스크랩 샘플 제한으로 인해 라벨에 많은 바인딩되지 않는 속성을 추가하여 문제가 발생하지 않도록 할 수 있습니다. 개발자는 메트릭에 대해 정의된 바인딩되지 않은 속성 수를 제한하여 기본 원인을 방지할 수도 있습니다. 사용 가능한 값의 제한된 집합에 바인딩되는 속성을 사용하면 가능한 키 - 값 쌍 조합의 수가 줄어듭니다.

3.7.1. 사용자 정의 프로젝트에 대한 스크랩 샘플 제한 설정

사용자 정의 프로젝트에서 대상 스크랩별로 허용할 수 있는 샘플 수를 제한할 수 있습니다.

주의

샘플 제한을 설정하면 제한에 도달한 후 해당 대상 스크랩에 대한 추가 샘플 데이터가 사용되지 않습니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **user-workload-monitoring-config ConfigMap** 오브젝트가 생성되어 있습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

프로세스

1. **openshift-user-workload-monitoring** 프로젝트에서 **user-workload-monitoring-config ConfigMap** 오브젝트를 편집합니다.

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

2. 사용자 정의 프로젝트에서 대상 스크랩별로 허용할 수 있는 샘플 수를 제한하려면 **enforcedSampleLimit** 구성을 **data.config.yaml** 에 추가합니다.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: user-workload-monitoring-config
  namespace: openshift-user-workload-monitoring
data:
  config.yaml: |
    prometheus:
      enforcedSampleLimit: 50000 1
```

1 이 매개변수가 지정된 경우 값이 필요합니다. 이 **enforcedSampleLimit** 예제에서는 사용자 정의 프로젝트의 대상 스크랩별로 허용할 수 있는 샘플 수를 50,000개로 제한합니다.

3. 파일을 저장하여 변경 사항을 적용합니다. 제한이 자동으로 적용됩니다.

주의

user-workload-monitoring-config ConfigMap 오브젝트에 변경 사항이 저장되면 **openshift-user-workload-monitoring** 프로젝트의 Pod 및 기타 리소스가 재배포될 수 있습니다. 해당 프로젝트에서 실행 중인 모니터링 프로세스도 다시 시작할 수 있습니다.

추가 리소스

- 스크랩 샘플이 가장 많은 메트릭을 쿼리하기 위해 Prometheus가 많은 디스크 공간을 소비하는 이유 확인

3.8. 모니터링 구성 요소에 대한 로그 수준 설정

Prometheus Operator, Prometheus 및 Thanos Ruler의 로그 수준을 구성할 수 있습니다.

다음 로그 수준은 **user-workload-monitoring-config ConfigMap** 오브젝트의 각 구성 요소에 적용할 수 있습니다.

- **debug**. 디버그, 정보, 경고 및 오류 메시지를 기록합니다.
- **info**. 정보, 경고 및 오류 메시지를 기록합니다.
- **warn**. 경고 및 오류 메시지만 기록합니다.
- **error**. 오류 메시지만 기록합니다.

기본값 로그 수준은 **info**입니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **user-workload-monitoring-config ConfigMap** 오브젝트가 생성되어 있습니다.

- OpenShift CLI(**oc**)가 설치되어 있습니다.

프로세스

1. **ConfigMap** 오브젝트를 편집합니다.
   a. **openshift-user-workload-monitoring** 프로젝트에서 **user-workload-monitoring-config ConfigMap** 오브젝트를 편집합니다.

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

   b. **data.config.yaml** 아래의 구성 요소에 **logLevel: <log_level** >을 추가합니다.

```
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 로그 수준을 적용하려는 모니터링 구성 요소입니다.

   2 구성 요소에 적용할 로그 수준입니다.

2. 파일을 저장하여 변경 사항을 적용합니다. 로그 수준 변경을 적용하면 구성 요소의 Pod가 자동으로 다시 시작됩니다.

주의

모니터링 구성 맵에 변경 사항이 저장되면 관련 프로젝트의 Pod 및 기타 리소스가 재배포될 수 있습니다. 해당 프로젝트에서 실행 중인 모니터링 프로세스도 다시 시작할 수 있습니다.

3. 관련 프로젝트의 배포 또는 Pod 구성을 검토하여 로그 수준이 적용되었는지 확인합니다. 다음 예제는 **openshift-user-workload-monitoring** 프로젝트의 **prometheus-operator** 배포에서 로그 수준을 확인합니다.

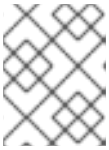
```
$ oc -n openshift-user-workload-monitoring get deploy prometheus-operator -o yaml |  grep "log-level"
```

출력 예

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

4. 구성 요소의 Pod가 실행 중인지 확인합니다. 다음 예제는 **openshift-user-workload-monitoring** 프로젝트의 Pod 상태를 나열합니다.

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



참고

인식할 수 없는 Prometheus Operator **loglevel** 값이 **ConfigMap**에 포함된 경우 Pod 구성 요소가 성공적으로 다시 시작되지 않을 수 있습니다.

3.9. 다음 단계

- 메트릭 관리

4장. 사용자 정의 프로젝트에 대한 경고 라우팅 활성화

AWS의 Red Hat OpenShift Service에서 클러스터 관리자는 사용자 정의 프로젝트에 대한 경고 라우팅을 활성화할 수 있습니다.

중요

사용자 정의 프로젝트에 대한 경고 규칙을 관리하는 것은 AWS 버전 4.11 이상의 Red Hat OpenShift Service에서만 사용할 수 있습니다.

이 프로세스는 두 가지 일반적인 단계로 구성됩니다.

- 사용자 정의 프로젝트에 대한 경고 라우팅을 활성화하여 별도의 Alertmanager 인스턴스를 사용합니다.
- 사용자 정의 프로젝트에 대한 경고 라우팅을 구성할 수 있는 추가 사용자에게 권한을 부여합니다.

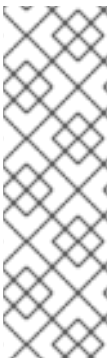
이러한 단계를 완료하면 개발자 및 기타 사용자가 사용자 정의 프로젝트에 대한 사용자 정의 경고 및 경고 라우팅을 구성할 수 있습니다.

4.1. 사용자 정의 프로젝트의 경고 라우팅 이해

클러스터 관리자는 사용자 정의 프로젝트에 대한 경고 라우팅을 활성화할 수 있습니다. 이 기능을 사용하면 **alert-routing-edit** 역할의 사용자가 사용자 정의 프로젝트에 대한 경고 알림 라우팅 및 수신자를 구성하도록 허용할 수 있습니다. 이러한 알림은 사용자 정의 모니터링 전용 Alertmanager 인스턴스에 의해 라우팅됩니다.

그런 다음 관리자는 사용자 정의 프로젝트에 대한 **AlertmanagerConfig** 오브젝트를 생성하거나 편집하여 사용자 정의 경고 라우팅을 생성하고 구성할 수 있습니다.

사용자가 사용자 정의 프로젝트에 대한 경고 라우팅을 정의한 후 사용자 정의 경고 알림이 **openshift-user-workload-monitoring** 네임스페이스의 **alertmanager-user-workload** Pod로 라우팅됩니다.



참고

다음은 사용자 정의 프로젝트에 대한 경고 라우팅의 제한 사항입니다.

- 사용자 정의 경고 규칙의 경우 사용자 정의 라우팅의 범위는 리소스가 정의된 네임스페이스로 지정됩니다. 예를 들어, 네임스페이스 **ns1** 의 라우팅 구성은 동일한 네임스페이스의 **PrometheusRules** 리소스에만 적용됩니다.
- 네임스페이스가 사용자 정의 모니터링에서 제외되면 네임스페이스의 **AlertmanagerConfig** 리소스가 Alertmanager 설정의 일부가 되지 않습니다.

4.2. 사용자 정의 경고 라우팅에 대해 별도의 ALERTMANAGER 인스턴스 활성화

AWS의 Red Hat OpenShift Service에서는 사용자 정의 프로젝트에 전용 Alertmanager 인스턴스를 배포할 수 있습니다. 이 인스턴스는 기본 플랫폼 경고와 별도로 사용자 정의 경고를 제공합니다. 이 경우 선택적으로 Alertmanager의 별도의 인스턴스를 활성화하여 사용자 정의 프로젝트에 대한 경고만 보낼 수 있습니다.

사전 요구 사항

- **cluster-admin** 또는 **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **openshift-monitoring** 네임스페이스에 대해 **cluster-monitoring-config** 구성 맵에서 사용자 정의 프로젝트에 대한 모니터링을 활성화했습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

절차

1. **user-workload-monitoring-config ConfigMap** 오브젝트를 편집합니다.

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

2. **data/config.yaml** 아래의 **alertmanager** 섹션에 **enabled: true** 및 **enableAlertmanagerConfig: true** 를 추가합니다.

```
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 클러스터에서 사용자 정의 프로젝트에 대해 Alertmanager의 전용 인스턴스를 활성화하려면 **enabled** 값을 **true** 로 설정합니다. 값을 **false** 로 설정하거나 키를 완전히 생략하여 사용자 정의 프로젝트에 대한 Alertmanager를 비활성화합니다. 이 값을 **false** 로 설정하거나 키가 생략되면 사용자 정의 경고가 기본 플랫폼 Alertmanager 인스턴스로 라우팅됩니다.

2 사용자가 **AlertmanagerConfig** 오브젝트로 자체 경고 라우팅 구성을 정의할 수 있도록 **enableAlertmanagerConfig** 값을 **true** 로 설정합니다.

3. 파일을 저장하여 변경 사항을 적용합니다. 사용자 정의 프로젝트에 대한 Alertmanager의 전용 인스턴스는 자동으로 시작됩니다.

검증

- **user-workload** Alertmanager 인스턴스가 시작되었는지 확인합니다.

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

출력 예

```
NAME            VERSION   REPLICAS   AGE
user-workload   0.24.0    2          100s
```

4.3. 사용자 정의 프로젝트에 대한 경고 라우팅을 구성할 수 있는 사용자 권한 부여

사용자 정의 프로젝트에 대한 경고 라우팅을 구성할 수 있는 권한을 사용자에게 부여할 수 있습니다.

사전 요구 사항

- **cluster-admin** 또는 **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- 이미 존재하는 역할에 할당 중인 사용자 계정입니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.
- 사용자 정의 프로젝트에 대한 모니터링을 활성화했습니다.

절차

- 사용자 정의 프로젝트의 사용자에게 **alert-routing-edit** 역할을 할당합니다.

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

1 & **lt;namespace** >의 경우 사용자 정의 프로젝트의 네임스페이스(예: **ns** 1)를 대체합니다. & **lt;user** >의 경우 역할을 할당하려는 계정의 사용자 이름을 대체합니다.

추가 리소스

- 사용자 정의 프로젝트 모니터링에 액세스
- 사용자 정의 프로젝트에 대한 경고 라우팅 생성

5장. 메트릭 관리

AWS의 Red Hat OpenShift Service는 클러스터 구성 요소에 대한 지표를 수집하며 Prometheus를 사용하여 쿼리 및 시각화할 수 있습니다.

5.1. 메트릭 이해

AWS의 Red Hat OpenShift Service에서는 서비스 끝점을 통해 노출되는 스크랩 메트릭을 통해 클러스터 구성 요소를 모니터링합니다. 사용자 정의 프로젝트에 대한 메트릭 컬렉션을 구성할 수도 있습니다. 메트릭을 사용하면 클러스터 구성 요소 및 자체 워크로드가 수행하는 방법을 모니터링할 수 있습니다.

애플리케이션 수준에서 Prometheus 클라이언트 라이브러리를 사용하여 자체 워크로드에 대해 제공할 메트릭을 정의할 수 있습니다.

AWS의 Red Hat OpenShift Service에서 **/metrics** 표준 이름 아래에 HTTP 서비스 끝점을 통해 메트릭이 노출됩니다. **http://<endpoint>/metrics**에 대해 **curl** 쿼리를 실행하여 서비스에 사용 가능한 모든 메트릭을 나열할 수 있습니다. 예를 들어 **prometheus-example-app** 예제 애플리케이션에 대한 경로를 노출하고 다음을 실행하여 사용 가능한 모든 메트릭을 확인할 수 있습니다.

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

출력 예

```
# 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
```

추가 리소스

- Prometheus 클라이언트 라이브러리에 대한 자세한 내용은 Prometheus 설명서를 참조하십시오.

5.2. 사용자 정의 프로젝트에 대한 메트릭 컬렉션 설정

ServiceMonitor 리소스를 생성하여 사용자 정의 프로젝트의 서비스 끝점에서 메트릭을 스크랩할 수 있습니다. 애플리케이션은 Prometheus 클라이언트 라이브러리를 사용하여 메트릭을 **/metrics** 표준 이름에 노출한다고 가정합니다.

이 섹션에서는 사용자 정의 프로젝트에 샘플 서비스를 배포한 후 서비스 모니터링 방법을 정의하는 **ServiceMonitor** 리소스를 만드는 방법에 대해 설명합니다.

5.2.1. 샘플 서비스 배포

사용자 정의 프로젝트에서 서비스 모니터링을 테스트하기 위해 샘플 서비스를 배포할 수 있습니다.

프로세스

1. 서비스 구성에 대한 YAML 파일을 생성합니다. 이 예에서는 **prometheus-example-app.yaml**이라고 합니다.

2. 파일에 다음 배포 및 서비스 구성 세부 정보를 추가합니다.

```
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: quay.io/brancz/prometheus-example-app:v0.2.0
        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
```

이 구성은 사용자 정의 **ns1** 프로젝트에 **prometheus-example-app**이라는 서비스를 배포합니다. 이 서비스는 사용자 정의 **version** 메트릭을 노출합니다.

3. 클러스터에 구성을 적용합니다.

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

서비스를 배포하는 데 시간이 다소 걸립니다.

4. Pod가 실행 중인지 확인할 수 있습니다.

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

출력 예

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

5.2.2. 서비스 모니터링 방법 지정

서비스에서 노출하는 메트릭을 사용하려면 **/metrics** 끝점에서 메트릭을 스크랩하도록 AWS 모니터링에서 Red Hat OpenShift Service를 구성해야 합니다. 서비스를 모니터링해야 하는 방법을 지정하는 **ServiceMonitor**(CRD) 또는 Pod를 모니터링해야 하는 방법을 지정하는 **PodMonitor** CRD를 사용하여 이 작업을 수행할 수 있습니다. 전자에는 **Service** 오브젝트가 필요하지만 후자에는 필요하지 않으며 Prometheus가 Pod에서 노출하는 메트릭 끝점에서 메트릭을 직접 스크랩할 수 있습니다.

참고

AWS의 Red Hat OpenShift Service에서는 **ServiceMonitor** 리소스에 **tlsConfig** 속성을 사용하여 끝점에서 메트릭을 스크랩할 때 사용할 TLS 구성을 지정할 수 있습니다. **tlsConfig** 속성은 **PodMonitor** 리소스에서 아직 사용할 수 없습니다. 메트릭을 스크랩할 때 TLS 구성을 사용해야 하는 경우 **ServiceMonitor** 리소스를 사용해야 합니다.

다음 프로세스에서는 사용자 정의 프로젝트에서 서비스에 대한 **ServiceMonitor** 리소스를 생성하는 방법을 보여줍니다.

사전 요구 사항

- **dedicated-admin** 역할 또는 **monitoring-edit** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- 이 예제에서는 **prometheus-example-app** 샘플 서비스를 **ns1** 프로젝트에 배포했습니다.

프로세스

1. **ServiceMonitor** 리소스 구성에 대한 YAML 파일을 생성합니다. 이 예제에서 파일은 **example-app-service-monitor.yaml**이라고 합니다.
2. 다음 **ServiceMonitor** 리소스 구성 세부 정보를 추가합니다.

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    k8s-app: prometheus-example-monitor
  name: prometheus-example-monitor
  namespace: ns1
spec:
  endpoints:
  - interval: 30s
    port: web
    scheme: http
```

```
  selector:
    matchLabels:
      app: prometheus-example-app
```

이는 **버전** 메트릭이 포함된 **prometheus-example-app** 샘플 서비스에서 노출하는 메트릭을 스크랩하는 **ServiceMonitor** 리소스를 정의합니다.

3. 클러스터에 구성을 적용합니다.

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

ServiceMonitor 리소스를 배포하는 데 시간이 다소 걸립니다.

4. **ServiceMonitor** 리소스가 실행 중인지 확인할 수 있습니다.

```
$ oc -n ns1 get servicemonitor
```

출력 예

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

추가 리소스

- 사용자 정의 프로젝트 모니터링에 액세스.

5.3. 메트릭 쿼리

OpenShift 모니터링 대시보드를 사용하면 Prometheus Query Language(PromQL) 쿼리를 실행하여 플롯에서 시각화된 메트릭을 검사할 수 있습니다. 이 기능은 클러스터 상태 및 모니터링 중인 모든 사용자 정의 프로젝트에 대한 정보를 제공합니다.

dedicated-admin 은 사용자 정의 프로젝트에 대한 메트릭을 위해 한 번에 하나 이상의 네임스페이스를 쿼리할 수 있습니다.

개발자는 메트릭을 쿼리할 때 프로젝트 이름을 지정해야 합니다. 선택한 프로젝트의 메트릭을 확인하는 데 필요한 권한이 있어야 합니다.

5.3.1. 관리자로 모든 프로젝트의 메트릭 쿼리

전용 관리자 또는 모든 프로젝트에 대한 보기 권한이 있는 사용자로, AWS의 기본 Red Hat OpenShift Service 및 Metrics UI의 사용자 정의 프로젝트에 대한 메트릭에 액세스할 수 있습니다.

참고

전용 관리자만 AWS 모니터링의 Red Hat OpenShift Service에서 제공되는 타사 UI에 액세스할 수 있습니다.

사전 요구 사항

- **dedicated-admin** 역할의 사용자 또는 모든 프로젝트에 대한 보기 권한이 있는 사용자로 클러스터에 액세스할 수 있습니다.

절차

1. OpenShift 웹 콘솔의 **관리자** 관점에서 **모니터링 → 메트릭 을** 선택합니다.
2. **커서에 표시기 삽입**을 선택하여 사전 정의된 쿼리 목록을 확인합니다.
3. 사용자 정의 쿼리를 생성하려면 **표현식** 필드에 PromQL(Prometheus Query Language) 쿼리를 추가합니다.
4. 여러 쿼리를 추가하려면 **쿼리 추가**를 선택합니다.
5. 쿼리를 삭제하려면 쿼리 옆에 있는 를 선택한 다음 **쿼리 삭제**를 선택합니다.
6. 쿼리 실행을 비활성화하려면 쿼리 옆에 있는 를 선택하고 **쿼리 비활성화**를 선택합니다.
7. 생성된 쿼리를 실행하려면 **쿼리 실행**을 선택합니다. 쿼리의 메트릭은 플롯에 시각화됩니다. 쿼리가 유효하지 않으면 UI에 오류 메시지가 표시됩니다.

참고

대량의 데이터에서 작동하는 쿼리 시간이 초과되거나 시계열 그래프에 있을 때 브라우저가 과부하될 수 있습니다. 이를 방지하려면 **그래프 숨기기**를 선택하고 메트릭 테이블만 사용하여 쿼리를 조정합니다. 그런 다음 실행 가능한 쿼리를 검색한 후 플롯을 활성화하여 그래프를 그립니다.

8. 선택 사항: 이제 페이지 URL에 실행한 쿼리가 포함되어 있습니다. 나중에 이 쿼리 세트를 다시 사용하려면 이 URL을 저장합니다.

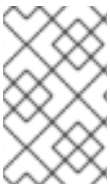
추가 리소스

- PromQL 쿼리 생성에 대한 자세한 내용은 Prometheus 쿼리 문서를 참조하십시오.

5.3.2. 개발자로 사용자 정의 프로젝트의 메트릭 쿼리

사용자 정의 프로젝트의 메트릭에 대해 개발자 또는 프로젝트에 대한 보기 권한이 있는 사용자로 액세스할 수 있습니다.

개발자 관점에서 Metrics UI에는 선택한 프로젝트에 대한 사전 정의된 CPU, 메모리, 대역폭 및 네트워크 패킷 쿼리가 포함되어 있습니다. 프로젝트에 대한 CPU, 메모리, 대역폭, 네트워크 패킷 및 애플리케이션 메트릭에 대해 사용자 정의 Prometheus Query Language(PromQL) 쿼리를 실행할 수도 있습니다.



참고

개발자는 **관리자** 관점이 아닌 **개발자** 관점만 사용할 수 있습니다. 개발자는 한 번에 하나의 프로젝트의 메트릭만 쿼리할 수 있습니다. 개발자는 AWS 모니터링에서 Red Hat OpenShift Service와 함께 제공되는 타사 UI에 액세스할 수 없습니다.

사전 요구 사항

- 개발자로 또는 메트릭을 확인하는 프로젝트에 대한 보기 권한이 있는 사용자로 클러스터에 액세스할 수 있습니다.

- 사용자 정의 프로젝트에 대한 모니터링을 활성화했습니다.
- 사용자 정의 프로젝트에 서비스를 배포했습니다.
- 서비스에서 모니터링 방법을 정의하는 데 사용할 **ServiceMonitor** CRD(사용자 정의 리소스 정의(Custom Resource Definition))가 생성되었습니다.

절차

1. AWS 웹 콘솔의 Red Hat OpenShift Service의 **개발자** 관점에서 **Observe → Metrics** 를 선택합니다.
2. **Project:** 목록에서 메트릭을 보려는 프로젝트를 선택합니다.
3. **쿼리 선택** 목록에서 쿼리를 선택하거나 **PromQL 표시**를 선택하여 사용자 정의 PromQL 쿼리를 실행합니다.

참고

개발자 관점에서는 한 번에 하나의 쿼리만 실행할 수 있습니다.

추가 리소스

- PromQL 쿼리 생성에 대한 자세한 내용은 Prometheus 쿼리 문서를 참조하십시오.
- 개발자 또는 권한 있는 사용자로 비 클러스터 메트릭에 액세스하는 방법에 대한 자세한 내용은 개발자로 사용자 정의 프로젝트 의 메트릭 쿼리를 참조하십시오.

5.3.3. 시각화된 메트릭 살펴보기

쿼리를 실행하면 대화형 플롯에 메트릭이 표시됩니다. 플롯의 X축은 시간을 나타내며, Y축은 메트릭 값을 나타냅니다. 각 메트릭은 그래프에 색상이 지정된 선으로 표시됩니다. 대화형으로 플롯을 조작하고 메트릭을 살펴볼 수 있습니다.

프로세스

관리자 관점에서:

1. 처음에 활성화된 모든 쿼리의 모든 메트릭이 플롯에 표시됩니다. 표시된 메트릭을 선택할 수 있습니다.

참고

기본적으로 쿼리 테이블은 모든 메트릭과 해당 현재 값을 나열하는 확장된 보기를 표시합니다. 쿼리에 대해 확장된 보기를 최소화하려면 ˅를 선택할 수 있습니다.

- 쿼리에서 모든 메트릭을 숨기려면 쿼리에 대해  을 클릭하고 **모든 시리즈 숨기기**를 클릭합니다.
- 특정 메트릭을 숨기려면 쿼리 테이블로 이동하여 메트릭 이름 근처에 있는 색상이 지정된 사각형을 클릭합니다.

2. 플롯을 확장하고 시간 범위를 변경하려면 다음 중 하나를 수행하십시오.

- 수평으로 플롯을 클릭하고 드래그하여 시간 범위를 시각적으로 선택합니다.
- 왼쪽 상단 코너의 메뉴를 사용하여 시간 범위를 선택합니다.

3. 시간 범위를 재설정하려면 **확대/축소 재설정**을 선택합니다.
4. 특정 시점에서 모든 쿼리에 대한 출력을 표시하려면 해당 시점에서 플롯을 마우스로 가리킵니다. 쿼리 출력이 팝업 상자에 나타납니다.
5. 플롯을 숨기려면 **그래프 숨기기**를 선택합니다.

Developer 관점에서:

1. 플롯을 확장하고 시간 범위를 변경하려면 다음 중 하나를 수행하십시오.
   - 수평으로 플롯을 클릭하고 드래그하여 시간 범위를 시각적으로 선택합니다.
   - 왼쪽 상단 코너의 메뉴를 사용하여 시간 범위를 선택합니다.
2. 시간 범위를 재설정하려면 **확대/축소 재설정**을 선택합니다.
3. 특정 시점에서 모든 쿼리에 대한 출력을 표시하려면 해당 시점에서 플롯을 마우스로 가리킵니다. 쿼리 출력이 팝업 상자에 나타납니다.

추가 리소스

- PromQL 인터페이스 사용에 대한 메트릭 쿼리 섹션을 참조하십시오.
- 모니터링 문제 조사

5.4. 다음 단계

- 경고

6장. 경고 관리

AWS 4의 Red Hat OpenShift Service에서 경고 UI를 사용하면 경고, 음소거 및 경고 규칙을 관리할 수 있습니다.

- **경고 규칙**. 경고 규칙에는 클러스터 내에서 특정 상태를 설명하는 일련의 조건이 포함되어 있습니다. 이러한 조건이 true이면 경고가 트리거됩니다. 경고 규칙은 경고의 라우팅 방법을 정의하는 심각도를 할당할 수 있습니다.
- **경고**. 경고 규칙에 정의된 조건이 true이면 경고가 실행됩니다. 경고는 일련의 상황이 AWS 클러스터의 Red Hat OpenShift Service 내에서 발생한다는 통지를 제공합니다.
- **음소거**. 경고 조건이 true일 때 알림이 전송되는 것을 방지하기 위해 경고에 음소거를 적용할 수 있습니다. 기본 문제를 해결하는 동안 초기 알림 후 경고를 음소거할 수 있습니다.

참고

경고 UI에서 사용할 수 있는 경고, 음소거, 경고 규칙은 액세스할 수 있는 프로젝트와 관련이 있습니다. 예를 들어 **cluster-admin** 권한으로 로그인하면 모든 경고, 음소거, 경고 규칙에 액세스할 수 있습니다.

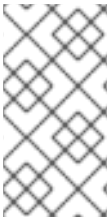
관리자가 아닌 사용자인 경우 다음 사용자 역할이 할당된 경우 경고를 생성하고 음소거할 수 있습니다.

- Alertmanager에 액세스할 수 있는 **cluster-monitoring-view** 역할
- 웹 콘솔의 **관리자** 화면에서 경고를 생성하고 음소거할 수 있는 **monitoring-alertmanager-edit** 역할
- 웹 콘솔의 **개발자** 화면에서 경고를 생성하고 음소거할 수 있는 **monitoring-rules-edit** 역할

6.1. 관리자 및 개발자 관점에서 경고 UI에 액세스

경고 UI는 관리자 관점과 AWS 웹 콘솔의 Red Hat OpenShift Service의 개발자 화면을 통해 액세스할 수 있습니다.

- **관리자** 관점에서 **모니터링 → 경고를** 선택합니다. 이 관점에서 경고 UI의 세 가지 주요 페이지는 **경고**, **음소거** 및 **경고 규칙** 페이지입니다.
- **개발자** 관점에서 **Observe → <project_name> → 경고를** 선택합니다. 이 관점에서 경고, 음소거 및 경고 규칙은 모두 **경고** 페이지에서 관리됩니다. **경고** 페이지에 표시된 결과는 선택한 프로젝트에 특정적입니다.



참고

개발자 관점에서는 AWS의 핵심 Red Hat OpenShift Service 및 **프로젝트:** 목록에서 액세스할 수 있는 사용자 정의 프로젝트에서 선택할 수 있습니다. 그러나 **cluster-admin** 권한이 없는 경우 AWS 프로젝트의 핵심 Red Hat OpenShift Service와 관련된 경고, 음소거, 경고 규칙이 표시되지 않습니다.

6.2. 경고, 음소거, 경고 규칙 검색 및 필터링

경고 UI에 표시되는 경고, 음소거 및 경고 규칙을 필터링할 수 있습니다. 이 섹션에서는 사용 가능한 필터링 옵션 각각에 대해 설명합니다.

경고 필터 이해

관리자 관점에서 경고 UI의 **경고** 페이지는 AWS의 기본 Red Hat OpenShift 서비스 및 사용자 정의 프로젝트와 관련된 경고에 대한 세부 정보를 제공합니다. 페이지에는 각 경고에 대한 심각도, 상태 및 소스가 요약되어 있습니다. 현재 상태로 경고가 표시되는 시간도 표시됩니다.

경고 상태, 심각도 및 소스로 필터링할 수 있습니다. 기본적으로 **실행**되는 **플랫폼**만 표시됩니다. 다음은 각 경고 필터링 옵션을 설명합니다.

- **경고 상태** 필터:
  - **실행**. 경고 조건이 true 이므로 경고가 실행되고 기간 **동안** 선택 사항이 전달됩니다. 상태가 true로 유지되는 동안 경고는 계속 실행됩니다.
  - **보류 중**. 경고는 활성화되어 있지만 실행되기 전에 경고 규칙에 지정된 기간 동안 대기 중입니다.
  - **음소거**. 이제 정의된 기간 동안 경고가 음소거되었습니다. 사용자가 정의한 라벨 선택기 집합에 따라 일시적으로 음소거가 경고를 음소거합니다. 나열된 모든 값 또는 정규식과 일치하는 경고에 대해 알림이 전송되지 않습니다.
- **심각도** 필터:
  - **심각**. 경고를 트리거한 조건으로 심각한 영향을 미칠 수 있습니다. 경고는 실행 시 즉각적인 주의가 필요하며 일반적으로 개인 또는 문제 대응팀으로 호출됩니다.
  - **경고**. 경고는 문제가 발생하지 않도록 주의가 필요할 수 있는 사항에 대한 경고 알림을 제공합니다. 경고는 일반적으로 즉각적이 아닌 검토에 대해 티켓 시스템으로 라우팅됩니다.
  - **정보**. 경고는 정보 목적으로만 제공됩니다.
  - **없음**. 경고에 정의된 심각도가 없습니다.
  - 사용자 정의 프로젝트와 관련된 경고에 대한 사용자 정의 심각도 정의를 생성할 수도 있습니다.
- **소스** 필터:
  - **플랫폼**. 플랫폼 수준 경고는 AWS 프로젝트의 기본 Red Hat OpenShift Service에만 관련이 있습니다. 이러한 프로젝트는 AWS 기능에 대한 핵심 Red Hat OpenShift Service를 제공합니다.
  - **사용자** 사용자 경고는 사용자 정의 프로젝트와 관련되어 있습니다. 이러한 경고는 사용자가 생성하며 사용자 정의할 수 있습니다. 사용자 정의 워크로드 모니터링은 설치 후 활성화하여 자체 워크로드에 관찰성을 제공할 수 있습니다.

음소거 필터 이해

관리자 관점에서 경고 UI **의 음소거** 페이지는 AWS의 기본 Red Hat OpenShift 서비스 및 사용자 정의 프로젝트의 경고에 적용된 음소거에 대한 세부 정보를 제공합니다. 페이지에는 각 음소거의 상태 요약과 음소거가 종료되는 시점이 포함되어 있습니다.

음소거 상태별로 필터링할 수 있습니다. 기본적으로 **활성** 및 **보류 중** 음소거만 표시됩니다. 다음은 각 음소거 상태 필터 옵션을 설명합니다.

- **음소거 상태** 필터:
  - **활성**. 음소거가 활성 상태이며 음소거가 만료될 때까지 경고가 음소거됩니다.
  - **보류 중**. 음소거 예정되어 있고 아직 활성화되지 않았습니다.

- **만료**. 경고 조건이 true이면 음소거가 만료되고 알림이 전송됩니다.

경고 규칙 필터 이해

관리자 관점에서 경고 UI의 **경고 규칙** 페이지는 AWS의 기본 Red Hat OpenShift 서비스 및 사용자 정의 프로젝트와 관련된 경고 규칙에 대한 세부 정보를 제공합니다. 페이지에는 각 경고 규칙의 상태, 심각도 및 소스가 요약되어 있습니다.

경고 상태, 심각도 및 소스에 따라 경고 규칙을 필터링할 수 있습니다. 기본적으로 **플랫폼** 경고 규칙만 표시됩니다. 다음은 각 경고 규칙 필터링 옵션을 설명합니다.

- **경고 상태** 필터:
  - **실행**. 경고 조건이 true 이므로 경고가 실행되고 기간 **동안** 선택 사항이 전달됩니다. 상태가 true로 유지되는 동안 경고는 계속 실행됩니다.
  - **보류 중**. 경고는 활성화되어 있지만 실행되기 전에 경고 규칙에 지정된 기간 동안 대기 중입니다.
  - **음소거**. 이제 정의된 기간 동안 경고가 음소거되었습니다. 사용자가 정의한 라벨 선택기 집합에 따라 일시적으로 음소거가 경고를 음소거합니다. 나열된 모든 값 또는 정규식과 일치하는 경고에 대해 알림이 전송되지 않습니다.
  - **실행하지 않음**. 경고가 실행되지 않습니다.
- **심각도** 필터:
  - **심각**. 경고 규칙에 정의된 조건이 심각한 영향을 미칠 수 있습니다. true인 경우 이러한 조건에는 즉각적인 주의가 필요합니다. 규칙과 관련된 경고는 일반적으로 개인 또는 문제 대응팀으로 호출됩니다.
  - **경고**. 경고 규칙에 정의된 조건은 문제가 발생하지 않도록 주의해야 할 수 있습니다. 규칙과 관련된 경고는 일반적으로 즉각적이 아닌 검토에 대해 티켓 시스템으로 라우팅됩니다.
  - **정보**. 경고 규칙은 정보성 경고만 제공합니다.
  - **없음**. 경고 규칙에는 정의된 심각도가 없습니다.
  - 사용자 정의 프로젝트와 관련된 경고 규칙에 대한 사용자 정의 심각도 정의를 생성할 수도 있습니다.
- **소스** 필터:
  - **플랫폼**. 플랫폼 수준 경고 규칙은 AWS 프로젝트의 기본 Red Hat OpenShift Service에만 관련이 있습니다. 이러한 프로젝트는 AWS 기능에 대한 핵심 Red Hat OpenShift Service를 제공합니다.
  - **사용자** 사용자 정의 워크로드 경고 규칙은 사용자 정의 프로젝트와 관련이 있습니다. 이러한 경고 규칙은 사용자가 생성하며 사용자 정의할 수 있습니다. 사용자 정의 워크로드 모니터링은 설치 후 활성화하여 자체 워크로드에 관찰성을 제공할 수 있습니다.

개발자 관점에서 경고, 음소거, 경고 규칙 검색 및 필터링

개발자 관점에서 경고 UI의 경고 페이지에서는 선택한 프로젝트와 관련된 경고 및 음소거의 결합된 보기를 제공합니다. 표시된 경고마다 관리 경고 규칙에 대한 링크가 제공됩니다.

이 보기에서는 경고 상태 및 심각도로 필터링할 수 있습니다. 기본적으로 프로젝트에 액세스할 수 있는 권한이 있는 경우 선택한 프로젝트의 모든 경고가 표시됩니다. 이러한 필터는 **관리자** 관점에서 설명한 항목과 동일합니다.

6.3. 경고, 음소거 및 경고 규칙에 대한 정보 받기

경고 UI는 경고 및 관리 경고 규칙과 음소거에 대한 자세한 정보를 제공합니다.

사전 요구 사항

- 개발자로 또는 메트릭을 확인하는 프로젝트에 대한 보기 권한이 있는 사용자로 클러스터에 액세스할 수 있습니다.

프로세스

관리자 관점에서 경고에 대한 정보를 얻으려면 다음을 수행합니다.

1. AWS 웹 콘솔에서 Red Hat OpenShift Service를 열고 **모니터링 → 경고 → 경고** 페이지로 이동합니다.
2. 선택 사항: 검색 목록에서 **이름** 필드를 사용하여 이름별로 경고를 검색합니다.
3. 선택 사항: **필터** 목록에서 필터를 선택하여 상태, 심각도 및 소스별로 경고를 필터링합니다.
4. 선택 사항: **이름**, **심각도**, **상태** 및 **소스** 열 헤더 중 하나 이상을 클릭하여 경고를 정렬합니다.
5. **경고 세부 정보** 페이지로 이동하도록 경고 이름을 선택합니다. 페이지에는 경고 시계열 데이터를 설명하는 그래프가 포함되어 있습니다. 또한 다음을 포함하여 경고에 대한 정보를 제공합니다.
   - 경고에 대한 설명
   - 경고와 관련된 메시지
   - 경고에 연결된 라벨
   - 관리 경고 규칙에 대한 링크
   - 존재하는 경우 경고에 대한 음소거

관리자 관점에서 음소거에 대한 정보를 얻으려면 다음을 수행합니다.

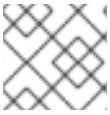
1. **모니터링 → 경고 → 음소거** 페이지로 이동합니다.
2. 선택 사항: **이름으로 검색** 필드를 사용하여 이름으로 음소거를 필터링합니다.
3. 선택 사항: **필터** 목록에서 필터를 선택하여 상태별로 음소거를 필터링합니다. 기본적으로 **활성** 및 **보류 중** 필터가 적용됩니다.
4. 선택 사항: **이름**, **경고 실행** 및 **상태** 열 헤더를 하나 이상 클릭하여 음소거를 정렬합니다.
5. 음소거의 이름을 선택하여 **음소거 상세 정보** 페이지로 이동합니다. 페이지에는 다음과 같은 세부 정보가 포함됩니다.
   - 경고 사양
   - 시작 시간
   - 종료 시간
   - 음소거 상태
   - 실행 경고 수 및 목록

관리자 관점에서 경고 규칙에 대한 정보를 얻으려면다음을 수행합니다.

1. **모니터링 → 경고 → 경고 규칙** 페이지로 이동합니다.
2. 선택 사항: **필터** 목록에서 필터를 선택하여 상태, 심각도 및 소스로 경고 규칙을 필터링합니다.
3. 선택 사항: **이름**, **심각도**, **경고 상태** 및 **소스** 열 헤더 중 하나 이상을 클릭하여 경고 규칙을 정렬합니다.
4. 경고 규칙의 이름을 선택하여 **경고 규칙 세부 정보** 페이지로 이동합니다. 페이지는 경고 규칙에 대한 다음 세부 정보를 제공합니다.
   - 경고 규칙 이름, 심각도 및 설명
   - 경고를 실행하기 위한 조건을 정의하는 표현식
   - 경고가 실행되기 위한 조건이 true여야 하는 시간
   - 경고 규칙에 의해 관리되는 각 경고에 대한 그래프로, 경고가 실행되는 값을 표시
   - 경고 규칙에 의해 관리되는 모든 경고의 테이블

개발자 관점에서 경고, 음소거 및 경고 규칙에 대한 정보를 얻으려면다음을 수행합니다.

1. **모니터링 →** < **project_name**> **→ 경고** 페이지로 이동합니다.
2. 경고, 음소거 또는 경고 규칙에 대한 세부 정보를 표시합니다.
   - **경고 세부 정보**는 경고 이름 왼쪽의 **>**를 선택한 다음 목록에 있는 경고를 선택하여 볼 수 있습니다.
   - **음소거 상세 정보**는 **경고 세부 정보** 페이지의 **음소거 기준** 섹션에서 음소거를 선택하여 볼 수 있습니다. **음소거 상세 정보** 페이지에는 다음 정보가 포함됩니다.
     - 경고 사양
     - 시작 시간
     - 종료 시간
     - 음소거 상태
     - 실행 경고 수 및 목록
   - **경고 규칙 세부 정보**는 **경고** 페이지에 있는 경고 오른쪽의 ⋮ 메뉴에서 **경고 규칙 보기**를 선택하여 볼 수 있습니다.



참고

선택한 프로젝트와 관련된 경고, 음소거, 경고 규칙만 **개발자** 관점에 표시됩니다.

추가 리소스

- AWS 모니터링 경고에서 특정 Red Hat OpenShift Service를 트리거하는 문제를 진단하고 해결하는 데 도움이 되는 Cluster Monitoring Operator runbook 을 참조하십시오.

6.4. 음소거 관리

경고가 실행될 때 경고에 대한 알림을 수신하지 못하도록 음소거를 생성할 수 있습니다. 기본 문제를 해결하는 동안 처음 알림을 받은 후 음소거하는 것이 유용할 수 있습니다.

음소거를 생성할 때 즉시 또는 나중에 활성 상태가 되는지 여부를 지정해야 합니다. 또한 음소거가 만료된 후 기간을 설정해야 합니다.

기존 음소거를 보고 편집하고 만료할 수 있습니다.

6.4.1. 음소거 경고

특정 경고를 음소거하거나 사용자가 정의한 사양과 일치하는 경고를 음소거할 수 있습니다.

사전 요구 사항

- 클러스터 관리자이며 **cluster-admin** 클러스터 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- 관리자가 아닌 사용자이며 다음 사용자 역할을 가진 사용자로 클러스터에 액세스할 수 있습니다.
  - **cluster-monitoring-view** 클러스터 역할로 Alertmanager에 액세스할 수 있습니다.
  - 웹 콘솔의 **관리자** 화면에서 경고를 생성하고 음소거할 수 있는 **monitoring-alertmanager-edit** 역할입니다.
  - 웹 콘솔의 **개발자** 화면에서 경고를 생성하고 음소거할 수 있는 **monitoring-rules-edit** 역할입니다.

절차

특정 경고를 음소거하려면 다음을 수행합니다.

- **관리자** 관점에서:
  1. AWS 웹 콘솔에서 Red Hat OpenShift Service의 **모니터링 → 경고 → 경고** 페이지로 이동합니다.
  2. **음소거할 경고의 경우 오른쪽 열에 있는 를 선택하고 음소거 경고를 선택합니다. 음소거 경고 형식은 선택한 경고에 대해 미리 채워진 사양으로 표시됩니다.**
  3. **선택 사항: 음소거를 수정합니다.**
  4. **음소거를 생성하기 전에 코멘트를 추가해야 합니다.**
  5. **음소거를 생성하려면 음소거를 선택합니다.**
- **개발자 화면에서:**
  1. **AWS 웹 콘솔 의 Red Hat OpenShift Service의 모니터링 → <project_name> →경고 페이지로 이동합니다.**
  2. **경고 이름의 왼쪽에 있는 >를 선택하여 경고에 대한 세부 사항을 확장합니다.경고 세부 정보 페이지를 열어 확장된 보기에서 경고의 이름을 선택합니다.**

3. 음소거 경고를 선택합니다. 음소거 경고 형식은 선택한 경고에 대해 미리 채워진 사양으로 표시됩니다.

4. 선택 사항: 음소거를 수정합니다.

5. 음소거를 생성하기 전에 코멘트를 추가해야 합니다.

6. 음소거를 생성하려면 음소거를 선택합니다.

관리자 관점에서 경고 사양을 생성하여 일련의 경고를 음소거하려면 다음을 수행합니다.

1. AWS 웹 콘솔의 Red Hat OpenShift Service의 모니터링 → 경고 → 음소거 페이지로 이동합니다.

2. 음소거 상태 만들기를 선택합니다.

3. 음소거 상태 만들기 형식에서 경고의 일정, 기간 및 라벨 세부 정보를 설정합니다. 음소거에 대한 코멘트를 추가해야 합니다.

4. 이전 단계에서 입력한 라벨 셀터와 일치하는 경고에 대한 음소거를 생성하려면 음소거를 선택합니다.

6.4.2. 음소거 편집

음소거를 편집하면 기존 음소거가 만료되고 변경된 구성으로 새 파일을 만들 수 있습니다.

프로세스

관리자 관점에서 음소거를 편집하려면 다음을 수행합니다.

1. 모니터링 → 경고 → 음소거 페이지로 이동합니다.

2. 수정하려는 음소거의 경우 마지막 열에서  를 선택하고 음소거 편집을 선택합니다.
또는 음소거에 대한 음소거 상세 정보 페이지에서 동작 → 음소거 편집을 선택할 수 있습니다.

3. 음소거 편지 페이지에서 변경 사항을 입력하고음소거를 선택합니다. 이를 통해 기존 음소거가 만료되고 선택한 구성으로 생성됩니다.

개발자 관점에서 음소거를 편집하려면 다음을 수행합니다.

1. 모니터링 → <project_name> → 경고 페이지로 이동합니다.

2. 경고 이름의 왼쪽에 있는 >를 선택하여 경고에 대한 세부 사항을 확장합니다.경고 세부 정보 페이지를 열어 확장된 보기에서 경고의 이름을 선택합니다.

3. 해당 페이지의 음소거 기준 섹션에서 음소거의 이름을 선택하여 음소거에 대한음소거 상세 정보 페이지로 이동합니다.

4. 음소거의 이름을 선택하여 음소거 상세 정보 페이지로 이동합니다.

5. 음소거에 대한 음소거 상세 정보 페이지에서 동작 → 음소거 편집을 선택합니다.

6. 음소거 편지 페이지에서 변경 사항을 입력하고음소거를 선택합니다. 이를 통해 기존 음소거가 만료되고 선택한 구성으로 생성됩니다.

6.4.3. 음소거 만료

음소거를 만료할 수 있습니다. 음소거를 만료하면 영구적으로 비활성화됩니다.

프로세스

관리자 관점에서 음소거를 만료하려면 다음을 수행합니다.

1. 모니터링 → 경고 → 음소거 페이지로 이동합니다.
2. 수정하려는 음소거의 경우 마지막 열에서  를 선택하고 음소거 만료를 선택합니다.
   또는 음소거에 대한 음소거 상세 정보 페이지에서 동작 → 음소거 만료를 선택할 수 있습니다.

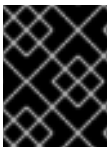
개발자 관점에서의 음소거를 만료하려면 다음을 수행합니다.

1. 모니터링 → <project_name> → 경고 페이지로 이동합니다.
2. 경고 이름의 왼쪽에 있는 >를 선택하여 경고에 대한 세부 사항을 확장합니다.경고 세부 정보 페이지를 열어 확장된 보기에서 경고의 이름을 선택합니다.
3. 해당 페이지의 음소거 기준 섹션에서 음소거의 이름을 선택하여 음소거에 대한음소거 상세 정보 페이지로 이동합니다.
4. 음소거의 이름을 선택하여 음소거 상세 정보 페이지로 이동합니다.
5. 음소거에 대한 음소거 상세 정보 페이지에서 동작 → 음소거 만료를 선택합니다.

6.5. 사용자 정의 프로젝트에 대한 경고 규칙 관리

AWS 모니터링의 Red Hat OpenShift Service는 기본 경고 규칙 세트와 함께 제공됩니다. 클러스터 관리자는 기본 경고 규칙을 볼 수 있습니다.

AWS 4의 Red Hat OpenShift Service에서는 사용자 정의 프로젝트에서 경고 규칙을 생성, 보기, 편집 및 제거할 수 있습니다.



중요

사용자 정의 프로젝트에 대한 경고 규칙을 관리하는 것은 AWS 버전 4.11 이상의 Red Hat OpenShift Service에서만 사용할 수 있습니다.

경고 규칙 고려 사항

- 기본 경고 규칙은 특히 AWS 클러스터의 Red Hat OpenShift Service에 사용됩니다.
- 일부 경고 규칙은 의도적으로 이름이 동일합니다. 임계값, 다른 심각도 또는 둘 다의 경우와 동일한 이벤트에 대한 경고를 보냅니다.
- 억제 규칙은 심각도가 높은 경고가 실행될 때 실행되는 심각도가 낮은 경고에 대한 알림을 방지합니다.

6.5.1. 사용자 정의 프로젝트에 대한 경고 최적화

경고 규칙을 생성할 때 다음 권장 사항을 따라 자체 프로젝트에 대한 경고를 최적화할 수 있습니다.

- 프로젝트에 생성하는 경고 규칙 수를 최소화합니다. 사용자에게 영향을 미치는 조건에 대해 알리는 경고 규칙을 생성합니다. 영향을 주지 않는 조건에 대한 여러 경고를 생성하면 관련 경고를 알리기가 더 어렵습니다.
- 원인 대신 증상에 대한 경고 규칙을 만듭니다. 기본 원인과 관계없이 조건을 알리는 경고 규칙을 만듭니다. 그러면 원인을 조사할 수 있습니다. 각 항목이 특정 원인에만 관련된 경우 더 많은 경고 규칙이 필요합니다. 그러면 일부 원인으로 인해 누락될 가능성이 큽니다.
- 경고 규칙을 작성하기 전에 계획합니다. 어떤 증상이 사용자에게 중요한지, 발생 시 어떤 조치를 수행할지를 결정합니다. 그런 다음 각 증상에 대한 경고 규칙을 구축합니다.
- 명확한 경고 메시지를 제공합니다. 경고 메시지에서 증상과 권장 작업을 설명합니다.
- 경고 규칙에 심각도 수준을 포함합니다. 경고의 심각도는 보고된 증상이 발생하는 경우 어떻게 대응해야 하는지에 따라 다릅니다. 예를 들어 증상이 개인 또는 문제 대응팀에서 즉각적인 주의가 필요한 경우 심각한 경고를 트리거해야 합니다.
- 경고 라우팅을 최적화합니다. 규칙이 AWS 메트릭에서 기본 Red Hat OpenShift Service를 쿼리하지 않는 경우 **openshift-user-workload-monitoring** 프로젝트의 Prometheus 인스턴스에 직접 경고 규칙을 배포합니다. 이렇게 하면 경고 규칙에 대한 대기 시간이 줄어들고 모니터링 구성 요소의 부하를 최소화합니다.

주의

사용자 정의 프로젝트의 AWS 메트릭에 대한 기본 Red Hat OpenShift Service는 CPU 및 메모리 사용량, 대역폭 통계 및 패킷 속도 정보에 대한 정보를 제공합니다. **openshift-user-workload-monitoring** 프로젝트에서 규칙을 직접 Prometheus 인스턴스에 라우팅하면 해당 메트릭을 경고 규칙에 포함할 수 없습니다. 경고 규칙 최적화는 문서를 읽고 모니터링 아키텍처를 포괄적으로 이해하는 경우에만 사용해야 합니다.

추가 리소스

- 경고 최적화에 대한 추가 지침은 Prometheus 경고 문서를 참조하십시오.

6.5.2. 사용자 정의 프로젝트에 대한 경고 규칙 생성

사용자 정의 프로젝트에 대한 경고 규칙을 생성할 수 있습니다. 이러한 경고 규칙은 선택한 메트릭 값을 기반으로 경고가 실행됩니다.

사전 요구 사항

- 사용자 정의 프로젝트에 대한 모니터링을 활성화했습니다.
- 경고 규칙을 생성하려는 프로젝트에 대한 **monitoring-rules-edit** 역할이 있는 사용자로 로그인했습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

프로세스

1. 경고 규칙에 사용할 YAML 파일을 생성합니다. 이 예에서는 **example-app-alerting-rule.yaml**이라고 합니다.

2. YAML 파일에 경고 규칙 구성을 추가합니다. 예를 들면 다음과 같습니다.

참고

경고 규칙을 생성할 때 동일한 이름의 규칙이 다른 프로젝트에 존재하는 경우 프로젝트 라벨이 적용됩니다.

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: example-alert
  namespace: ns1
spec:
  groups:
  - name: example
    rules:
    - alert: VersionAlert
      expr: version{job="prometheus-example-app"} == 0
```

이 구성에서는 **example-alert**라는 경고 규칙이 생성됩니다. 경고 규칙은 샘플 서비스에서 노출된 **version** 메트릭이 **0**이 되면 경고를 실행합니다.

중요

사용자 정의 경고 규칙에는 자체 프로젝트 및 클러스터 메트릭에 대한 메트릭이 포함될 수 있습니다. 다른 사용자 정의 프로젝트에 대한 메트릭은 포함할 수 없습니다.

예를 들어 사용자 정의 프로젝트 **ns1**에 대한 경고 규칙에는 CPU 및 메모리 메트릭과 같은 **ns1** 및 클러스터 메트릭에서 메트릭이 있을 수 있습니다. 그러나 규칙에는 **ns2**의 메트릭을 포함할 수 없습니다.

또한 AWS 프로젝트의 **openshift-*** 코어 Red Hat OpenShift Service에 대한 경고 규칙을 생성할 수 없습니다. 기본적으로 AWS 모니터링의 Red Hat OpenShift Service는 이러한 프로젝트에 대한 일련의 경고 규칙을 제공합니다.

3. 구성 파일을 리클러스터에 적용합니다.

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

경고 규칙을 생성하는 데 시간이 다소 걸립니다.

6.5.3. 플랫폼 메트리을 쿼리하지 않는 경고 규칙에 대한 대기 시간 감소

사용자 정의 프로젝트의 경고 규칙이 기본 클러스터 메트릭을 쿼리하지 않으면 **openshift-user-workload-monitoring** 프로젝트의 Prometheus 인스턴스에서 직접 규칙을 배포할 수 있습니다. 이로 인해 Thanos Ruler를 우회할 필요가 없는 경우 경고 규칙의 대기 시간이 단축됩니다. 또한 모니터링 구성 요소에 대한 전체 부하를 최소화하는 데 도움이 됩니다.

주의

사용자 정의 프로젝트의 AWS 메트릭에 대한 기본 Red Hat OpenShift Service는 CPU 및 메모리 사용량, 대역폭 통계 및 패킷 속도 정보에 대한 정보를 제공합니다. **openshift-user-workload-monitoring** 프로젝트의 Prometheus 인스턴스에 직접 규칙을 배포하는 경우 해당 메트릭을 경고 규칙에 포함할 수 없습니다. 이 섹션에 설명된 프로세스는 문서를 읽고 모니터링 아키텍처를 포괄적으로 이해하는 경우에만 사용해야 합니다.

사전 요구 사항

- 사용자 정의 프로젝트에 대한 모니터링을 활성화했습니다.
- 경고 규칙을 생성하려는 프로젝트에 대한 **monitoring-rules-edit** 역할이 있는 사용자로 로그인했습니다.
- OpenShift CLI(oc)가 설치되어 있습니다.

프로세스

1. 경고 규칙에 사용할 YAML 파일을 생성합니다. 이 예에서는 **example-app-alerting-rule.yaml**이라고 합니다.
2. 키 **openshift.io/prometheus-rule-evaluation-scope** 및 값 **leaf-prometheus**가 있는 라벨을 포함한 YAML 파일에 경고 규칙 구성을 추가합니다. 예를 들면 다음과 같습니다.

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: example-alert
  namespace: ns1
  labels:
    openshift.io/prometheus-rule-evaluation-scope: leaf-prometheus
spec:
  groups:
  - name: example
    rules:
    - alert: VersionAlert
      expr: version{job="prometheus-example-app"} == 0
```

해당 라벨이 있는 경우 경고 규칙이 **openshift-user-workload-monitoring** 프로젝트의 Prometheus 인스턴스에 배포됩니다. 라벨이 없으면 경고 규칙이 Thanos Ruler에 배포됩니다.

1. 구성 파일을 리클러스터에 적용합니다.

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

경고 규칙을 생성하는 데 시간이 다소 걸립니다.

6.5.4. 사용자 정의 프로젝트의 경고 규칙에 액세스

사용자 정의 프로젝트의 경고 규칙을 나열하려면 프로젝트에 대한 **monitoring-rules-view** 역할이 할당되어 있어야 합니다.

사전 요구 사항

- 사용자 정의 프로젝트에 대한 모니터링을 활성화했습니다.
- 프로젝트에 대한 **monitoring-rules-view** 역할이 있는 사용자로 로그인했습니다.
- OpenShift CLI(oc)가 설치되어 있습니다.

프로세스

1. **<project>**에서 경고 규칙을 나열할 수 있습니다.

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

2. 경고 규칙의 구성을 나열하려면 다음을 실행합니다.

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

6.5.5. 단일 보기에서 모든 프로젝트의 경고 규칙 나열

클러스터 관리자는 AWS의 핵심 Red Hat OpenShift Service 및 사용자 정의 프로젝트에 대한 경고 규칙을 단일 보기에서 나열할 수 있습니다.

사전 요구 사항

- **cluster-admin** 또는 **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- OpenShift CLI(oc)가 설치되어 있습니다.

절차

1. 관리자 관점에서 모니터링 → 경고 → 경고 규칙으로 이동합니다.
2. 필터 드롭다운 메뉴에서 플랫폼 및 사용자 소스를 선택합니다.

참고

플랫폼 소스가 기본적으로 선택됩니다.

6.5.6. 사용자 정의 프로젝트에 대한 경고 규칙 제거

사용자 정의 프로젝트에 대한 경고 규칙을 제거할 수 있습니다.

사전 요구 사항

- 사용자 정의 프로젝트에 대한 모니터링을 활성화했습니다.
- 경고 규칙을 생성하려는 프로젝트에 대한 **monitoring-rules-edit** 역할이 있는 사용자로 로그인했습니다.

- OpenShift CLI(oc)가 설치되어 있습니다.

프로세스

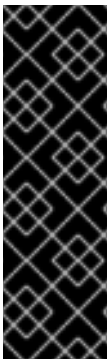
- 규칙 <foo>를 <namespace>에서 제거하려면 다음을 실행합니다.

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

추가 리소스

- Alertmanager 문서 참조

6.6. 코어 플랫폼 모니터링에 대한 경고 규칙 관리



중요

핵심 플랫폼 모니터링에 대한 경고 규칙을 생성하고 수정하는 것은 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 Red Hat 프로덕션 서비스 수준 계약(SLA)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

AWS 4 모니터링의 Red Hat OpenShift Service에는 플랫폼 메트릭에 대한 대규모 기본 경고 규칙 집합이 제공됩니다. 클러스터 관리자는 다음 두 가지 방법으로 이 규칙 세트를 사용자 지정할 수 있습니다.

- 임계값을 조정하거나 라벨을 추가하거나 수정하여 기존 플랫폼 경고 규칙의 설정을 수정합니다. 예를 들어 경고의 심각도 레이블을 경고 에서 **critical** 로 변경하여 경고에 의해 플래그가 지정된 문제를 라우팅 및 분류하는 데 도움이 될 수 있습니다.
- **openshift-monitoring** 네임스페이스의 코어 플랫폼 메트릭을 기반으로 쿼리 표현식을 구성하여 새 사용자 정의 경고 규칙을 정의하고 추가합니다.

핵심 플랫폼 경고 규칙 고려 사항

- 새로운 경고 규칙은 AWS 모니터링 메트릭에 대한 기본 Red Hat OpenShift Service를 기반으로 해야 합니다.
- 경고 규칙만 추가하고 수정할 수 있습니다. 새 레코딩 규칙을 생성하거나 기존 레코딩 규칙을 수정할 수 없습니다.
- **AlertRelabelConfig** 오브젝트를 사용하여 기존 플랫폼 경고 규칙을 수정하면 Prometheus 경고 API에 수정이 반영되지 않습니다. 따라서 삭제된 경고는 더 이상 Alertmanager로 전달되지 않는 경우에도 AWS 웹 콘솔의 Red Hat OpenShift Service에 계속 표시됩니다. 또한 변경된 심각도 레이블과 같은 경고 수정 사항은 웹 콘솔에 표시되지 않습니다.

6.6.1. 코어 플랫폼 경고 규칙 수정

클러스터 관리자는 Alertmanager가 이를 수신자로 라우팅하기 전에 코어 플랫폼 경고를 수정할 수 있습니다. 예를 들어 경고의 심각도 레이블을 변경하거나, 사용자 정의 레이블을 추가하거나, 경고를 Alertmanager로 전송하는 것을 제외할 수 있습니다.

사전 요구 사항

- **cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.
- OpenShift CLI(oc)가 설치되어 있습니다.
- 기술 프리뷰 기능을 활성화했으며 클러스터의 모든 노드가 준비되었습니다.

절차

1. **openshift-monitoring** 네임스페이스에 **example-modified-alerting-rule.yaml** 이라는 새 YAML 구성 파일을 생성합니다.
2. YAML 파일에 **AlertRelabelConfig** 리소스를 추가합니다. 다음 예제에서는 기본 플랫폼워치독 경고 규칙에 대해 심각도 설정을 중요 로 수정합니다.

```
apiVersion: monitoring.openshift.io/v1alpha1
kind: AlertRelabelConfig
metadata:
  name: watchdog
  namespace: openshift-monitoring
spec:
  configs:
  - sourceLabels: [alertname,severity] (1)
    regex: "Watchdog;none" (2)
    targetLabel: severity (3)
    replacement: critical (4)
    action: Replace (5)
```

(1) 수정할 값의 소스 레이블입니다.

(2) **sourceLabels** 값이 일치하는 정규식입니다.

(3) 수정할 값의 대상 레이블입니다.

(4) 대상 라벨을 교체할 새 값입니다.

(5) regex 일치를 기반으로 이전 값을 대체하는 재레이블 작업입니다. 기본 작업은 **Replace** 입니다. 가능한 값은 **Keep,Drop,HashMod,LabelMap,LabelDrop, LabelKeep** 입니다.

3. 구성 파일을 리클러스터에 적용합니다.

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

6.6.2. 새 경고 규칙 생성

클러스터 관리자는 플랫폼 메트릭을 기반으로 새 경고 규칙을 생성할 수 있습니다. 이러한 경고 규칙은 선택한 메트릭의 값에 따라 경고를 트리거합니다.

참고

기존 플랫폼 경고 규칙을 기반으로 사용자 정의 **AlertingRule** 리소스를 생성하는 경우, 충돌하지 않도록 원래 경고를 음소거합니다.

사전 요구 사항

- **cluster-admin** 역할이 있는 사용자로 로그인했습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.
- 기술 프리뷰 기능을 활성화했으며 클러스터의 모든 노드가 준비되었습니다.

절차

1. **openshift-monitoring** 네임스페이스에 **example-alerting-rule.yaml** 이라는 새 YAML 구성 파일을 생성합니다.
2. YAML 파일에 **AlertingRule** 리소스를 추가합니다. 다음 예제에서는 기본워치독 경고와 유사하게 **example** 이라는 새 경고 규칙을 생성합니다.

```
apiVersion: monitoring.openshift.io/v1alpha1
kind: AlertingRule
metadata:
  name: example
  namespace: openshift-monitoring
spec:
  groups:
  - name: example-rules
    rules:
    - alert: ExampleAlert 1
      expr: vector(1) 2
```

1. 생성할 경고 규칙의 이름입니다.
2. 새 규칙을 정의하는 PromQL 쿼리 표현식입니다.

3. 구성 파일을 리클러스터에 적용합니다.

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

추가 리소스

- 경고 규칙에 대한 정보는 Alertmanager 설명서를 참조하십시오.
- 레이블 재지정 작동 방법에 대한 자세한 내용은 Prometheus 재지정 문서를 참조하십시오.
- 경고 최적화에 대한 추가 지침은 Prometheus 경고 문서를 참조하십시오.

6.7. 외부 시스템에 알림 전송

AWS 4의 Red Hat OpenShift Service에서는 알림 UI에서 실행 경고를 볼 수 있습니다. 알림은 기본적으로 모든 알림 시스템으로 전송되지 않습니다. 다음 수신자 유형으로 경고를 전송하도록 AWS에서 Red Hat OpenShift Service를 구성할 수 있습니다.

- PagerDuty
- Webhook

- 이메일
- Slack

알림을 수신기로 라우팅하면 오류가 발생할 때 적절한 팀에게 적절한 알림을 보낼 수 있습니다. 예를 들어, 심각한 경고는 즉각적인 주의가 필요하며 일반적으로 개인 또는 문제 대응팀으로 호출됩니다. 심각하지 않은 경고 알림을 제공하는 경고는 즉각적이지 않은 검토를 위해 티켓팅 시스템으로 라우팅할 수 있습니다.

워치독 경고를 사용하여 해당 경고가 제대로 작동하는지 확인

AWS 모니터링에 대한 Red Hat OpenShift Service에는 지속적으로 실행되는 워치독 경고가 포함되어 있습니다. Alertmanager는 구성된 알림 공급자에게 워치독 경고 알림을 반복적으로 보냅니다. 일반적으로 공급자는 워치독 경고를 수신하지 않을 때 관리자에게 알리도록 구성됩니다. 이 메커니즘을 사용하면 Alertmanager와 알림 공급자 간의 모든 통신 문제를 빠르게 식별할 수 있습니다.

6.7.1. 경고 수신자 구성

클러스터의 중요한 문제를 파악할 수 있도록 경고 수신자를 설정할 수 있습니다.

사전 요구 사항

- **cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.

절차

1. 관리자 관점에서 관리 → 클러스터 설정 → 구성 → Alertmanager 로 이동합니다.

참고

또는 알림 창을 통해 동일한 페이지로 이동할 수 있습니다. AWS 웹 콘솔에서 Red Hat OpenShift Service의 오른쪽 상단에 있는 bell 아이콘을 선택하고 AlertmanagerReceiverNotConfigured 경고에서 구성을 선택합니다.

2. 이 페이지의 수신자 섹션에서 수신자 만들기를 선택합니다.
3. 수신자 만들기에서 수신자 이름을 추가하고 목록에서 수신자 유형을 선택합니다.
4. 수신자 구성을 편집합니다.
   - PagerDuty 수신자의 경우:
     a. 통합 유형을 선택하고 PagerDuty 통합 키를 추가합니다.
     b. PagerDuty 설치의 URL을 추가합니다.
     c. 클라이언트와 인스턴스 세부 정보 또는 심각도 사양을 편집하려면 고급 설정 표시를 선택합니다.
   - Webhook 수신자의 경우:
     a. HTTP POST 요청이 전송되는 끝점을 추가합니다.
     b. 해결된 경보를 수신자에게 보내는 기본 옵션을 편집하려면 고급 설정 표시를 선택합니다.
   - 이메일 수신자의 경우:

a. 알림을 받을 이메일 주소를 추가합니다.

b. 알림을 전송할 주소, 이메일 전송에 사용되는 스마트 호스트 및 포트 번호, SMTP 서버의 호스트 이름, 인증 세부 정보를 포함하여 SMTP 구성 세부 정보를 추가합니다.

c. TLS가 필요한지 여부를 선택합니다.

d. 해결된 경고를 수신자에게 보내지 않도록 기본 옵션을 편집하거나 이메일 알림 구성을 편집하려면 고급 설정 표시를 선택합니다.

- Slack 수신자의 경우:

a. Slack Webhook의 URL을 추가합니다.

b. 알림을 보낼 Slack 채널 또는 사용자 이름을 추가합니다.

c. 해결된 경고를 수신자에게 보내지 않도록 기본 옵션을 편집하거나 아이콘 및 사용자 이름 구성을 편집하려면 고급 설정 표시를 선택합니다. 채널 이름과 사용자 이름을 찾고 연결할지 여부를 선택할 수도 있습니다.

5. 기본적으로 모든 선택 항목과 일치하는 라벨을 사용하여 경고를 수신자에게 보냅니다. 수신자로 전송되기 전에 실행 경고에 대한 라벨 값을 정확히 일치시키려면 다음을 수행하십시오.

a. 양식의 라우팅 라벨 섹션에 라우팅 라벨 이름과 값을 추가합니다.

b. 정규식을 사용하려면 정규식을 선택합니다.

c. 라벨 추가를 선택하여 추가 라우팅 라벨을 추가합니다.

6. 만들기를 선택하여 수신자를 생성합니다.

6.7.2. 사용자 정의 프로젝트에 대한 경고 라우팅 생성

alert-routing-edit 역할이 지정된 관리자가 아닌 사용자인 경우 사용자 정의 프로젝트에 대한 경고 라우팅을 만들거나 편집할 수 있습니다.

사전 요구 사항

- 클러스터 관리자가 사용자 정의 프로젝트에 대한 모니터링을 활성화했습니다.
- 클러스터 관리자가 사용자 정의 프로젝트에 대한 경고 라우팅을 활성화했습니다.
- 경고 라우팅을 생성하려는 프로젝트에 대한 **alert-routing-edit** 역할이 있는 사용자로 로그인했습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.

절차

1. 경고 라우팅을 위한 YAML 파일을 만듭니다. 이 절차의 예제에서는 **example-app-alert-routing.yaml** 이라는 파일을 사용합니다.

2. **AlertmanagerConfig** YAML 정의를 파일에 추가합니다. 예를 들면 다음과 같습니다.

```
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
```

참고

사용자 정의 경고 규칙의 경우 사용자 정의 라우팅의 범위는 리소스가 정의된 네임스페이스로 지정됩니다. 예를 들어 **ns1** 네임스페이스의 **AlertmanagerConfig** 오브젝트에 정의된 라우팅 구성은 동일한 네임스페이스의 **PrometheusRules** 리소스에만 적용됩니다.

3. 파일을 저장합니다.

4. 클러스터에 리소스를 적용합니다.

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

구성은 Alertmanager Pod에 자동으로 적용됩니다.

6.8. 사용자 정의 ALERTMANAGER 설정 적용

Alertmanager의 플랫폼 인스턴스에 대해 **openshift-monitoring** 네임스페이스에서 **alertmanager-main** 시크릿을 편집하여 기본 Alertmanager 구성을 덮어쓸 수 있습니다.

사전 요구 사항

- **cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.

프로세스

CLI에서 Alertmanager 설정을 변경하려면 다음을 수행합니다.

1. 현재 활성화된 Alertmanager 구성을 파일 **alertmanager.yaml**로 출력합니다.

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

2. **alertmanager.yaml**에서 설정을 편집합니다.

```
global:
  resolve_timeout: 5m
route:
  group_wait: 30s (1)
  group_interval: 5m (2)
  repeat_interval: 12h (3)
  receiver: default
  routes:
```

```
  - matchers:
    - "alertname=Watchdog"
    repeat_interval: 2m
    receiver: watchdog
  - matchers:
    - "service=<your_service>" 4
    routes:
    - matchers:
      - <your_matching_rules> 5
      receiver: <receiver> 6
receivers:
- name: default
- name: watchdog
- name: <receiver>
#  <receiver_configuration>
```

1. group_wait 값은 경고 그룹에 대한 초기 알림을 보내기 전에 Alertmanager 대기 기간을 지정합니다. 이 값은 알림을 보내기 전에 동일한 그룹에 대한 초기 경고를 수집하는 동안 Alertmanager가 대기하는 시간을 제어합니다.

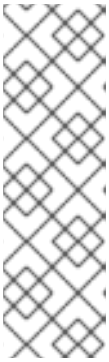
2. group_interval 값은 Alertmanager가 초기 알림이 이미 전송된 경고 그룹에 추가된 새 경고에 대한 알림을 보내기 전에 경과해야 하는 시간을 지정합니다.

3. repeat_interval 값은 경고 알림을 반복하기 전에 전달해야 하는 최소 시간을 지정합니다. 각 그룹 간격마다 알림을 반복하려면 repeat_interval 값을 group_interval 값보다 적도록 설정합니다. 그러나 반복된 알림은 특정 Alertmanager Pod가 재시작되거나 다시 예약되는 경우와 같이 계속 지연될 수 있습니다.

4. 서비스 값은 경고를 실행하는 서비스를 지정합니다.

5. & lt;your_matching_rules& gt; 값은 대상 경고를 지정합니다.

6. 수신자 값은 경고에 사용할 수신자를 지정합니다.



참고

일치자 키 이름을 사용하여 노드와 일치하기 위해 경고가 충족해야 한다는 일치자를 나타냅니다. 더 이상 사용되지 않고 향후 릴리스에서 제거될 예정인 **match** 또는 **match_re** 키 이름을 사용하지 마십시오.

또한 억제 규칙을 정의하는 경우 **target_matchers** 키 이름을 사용하여 대상 일치자 및 **source_matchers** 키 이름을 지정하여 소스 일치자를 나타냅니다. 더 이상 사용되지 않으며 향후 릴리스에서 제거될 예정인 **target_match,target_match_re,source_match _re** 키 이름을 사용하지 마십시오.

다음 Alertmanager 설정 예제에서는 PagerDuty를 경고 수신자로 구성합니다.

```
global:
  resolve_timeout: 5m
route:
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 12h
  receiver: default
```

```
  routes:
  - matchers:
    - "alertname=Watchdog"
    repeat_interval: 2m
    receiver: watchdog
  - matchers:
    - "service=example-app"
    routes:
    - matchers:
      - "severity=critical"
      receiver: team-frontend-page*
receivers:
- name: default
- name: watchdog
- name: team-frontend-page
  pagerduty_configs:
  - service_key: "_your-key_"
```

이 설정을 사용하면 **example-app** 서비스에서 실행되는 **critical** 심각도 경고가 **team-frontend-page** 수신자를 사용하여 전송됩니다. 일반적으로 이러한 유형의 경고는 개인 또는 문제 대응팀으로 호출됩니다.

3. 파일에 새 설정을 적용합니다.

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

AWS의 Red Hat OpenShift Service에서 Alertmanager 설정을 변경하려면 다음을 수행합니다.

1. 웹 콘솔의 관리 → 클러스터 설정 → 구성 → Alertmanager → YAML 페이지로 이동합니다.
2. YAML 설정 파일을 수정합니다.
3. 저장을 선택합니다.

6.9. 사용자 정의 경고 라우팅을 위해 ALERTMANAGER에 사용자 정의 설정 적용

사용자 정의 경고 라우팅 전용 Alertmanager의 별도의 인스턴스를 활성화한 경우 **openshift-user-workload-monitoring** 네임스페이스에서 **alertmanager-user-workload** 시크릿을 편집하여 Alertmanager 인스턴스에 대한 구성을 덮어쓸 수 있습니다.

사전 요구 사항

- **cluster-admin** 또는 **dedicated-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.

절차

1. 현재 활성화된 Alertmanager 설정을 **alertmanager.yaml** 파일에 출력합니다.

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

2. alertmanager.yaml에서 설정을 편집합니다.

```
route:
  receiver: Default
  group_by:
  - name: Default
  routes:
  - matchers:
    - "service = prometheus-example-monitor" 1
    receiver: <receiver> 2
receivers:
- name: Default
- name: <receiver>
#  <receiver_configuration>
```

1 경로와 일치하는 경고를 지정합니다. 이 예제에서는 service="prometheus-example-monitor" 라벨이 있는 모든 경고를 표시합니다.

2 경고 그룹에 사용할 수신자를 지정합니다.

3. 파일에 새 설정을 적용합니다.

```
$ 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=-
```

추가 리소스

- PagerDuty에 대한 자세한 내용은 PagerDuty 공식 사이트를 참조하십시오.
- service_key 검색 방법을 알아보려면 PagerDuty Prometheus 통합 가이드를 참조하십시오.
- 다양한 경고 수신자를 통한 경고 구성은 Alertmanager 설정을 참조하십시오.

7장. 모니터링 대시보드 검토

AWS의 Red Hat OpenShift Service는 사용자 정의 프로젝트의 상태를 이해하는 데 도움이 되는 모니터링 대시보드를 제공합니다.

개발자 관점에서 선택한 프로젝트에 대해 다음 통계를 제공하는 대시보드에 액세스할 수 있습니다.

- CPU 사용량
- 메모리 사용량
- 대역폭 정보
- 패킷 속도 정보

그림 7.1. 개발자 관점의 대시보드 예

참고

개발자 관점에서 한 번에 하나의 프로젝트에 대해서만 대시보드를 볼 수 있습니다.

7.1. 개발자로 모니터링 대시보드 검토

개발자 관점에서 선택한 프로젝트와 관련된 대시보드를 볼 수 있습니다. 대시보드 정보를 보기 위해 프로젝트를 모니터링하려면 액세스할 수 있어야 합니다.

사전 요구 사항

- 전용 관리자로 또는 대시보드를 보고 있는 프로젝트에 대한 보기 권한이 있는 사용자로 클러스터에 액세스할 수 있습니다.

절차

1. AWS 웹 콘솔의 Red Hat OpenShift Service의 개발자 관점에서 Observe → Dashboard 로 이동합니다.
2. Project: 목록에서 프로젝트를 선택합니다.
3. 모든 워크로드 목록에서 워크로드를 선택합니다.
4. 선택 사항: 시간 범위 목록에서 그래프의 시간 범위를 선택합니다.
5. 선택 사항: 새로 고침 간격을 선택합니다.
6. 대시보드 내에서 각각의 그래프로 마우스를 이동하여 특정 항목에 대한 세부 정보를 표시합니다.

7.2. 다음 단계

- 모니터링 문제 조사

8장. 모니터링 문제 조사

사용자 정의 프로젝트의 일반적인 모니터링 문제에 대한 문제 해결 단계를 확인하십시오.

8.1. 사용자 정의 프로젝트 메트릭을 사용할 수 없는 이유 확인

사용자 정의 프로젝트를 모니터링할 때 메트릭이 표시되지 않는 경우 다음 단계에 따라 문제를 해결합니다.

절차

1. 메트릭 이름을 쿼리하고 프로젝트가 올바른지 확인합니다.

   a. OpenShift Container Platform 웹 콘솔의 개발자 관점에서 모니터링 → 메트릭 을 선택합니다.

   b. Project: 목록에서 메트릭을 보려는 프로젝트를 선택합니다.

   c. 쿼리 선택 목록에서 쿼리를 선택하거나PromQL 표시를 선택하여 사용자 정의 PromQL 쿼리를 실행합니다.
   쿼리 선택 창에 지표 이름이 표시됩니다.

   쿼리는 프로젝트별로 수행해야 합니다. 표시된 메트릭은 선택한 프로젝트와 관련이 있습니다.

2. 메트릭을 원하는 Pod가 적극적으로 메트릭을 제공하고 있는지 확인합니다. Pod로 다음 **oc exec** 명령을 실행하여 **podIP**,포트, **/metrics** 를 대상으로 합니다.

```
$ oc exec <sample_pod> -n <sample_namespace> -- curl <target_pod_IP>:
<port>/metrics
```

참고

curl 이 설치된 포드에서 명령을 실행해야 합니다.

다음 예제 출력에서는 유효한 버전 지표가 있는 결과를 보여줍니다.

출력 예

```
  % 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
```

유효하지 않은 출력은 해당 애플리케이션에 문제가 있음을 나타냅니다.

3. **PodMonitor** CRD를 사용하는 경우 라벨 일치를 사용하여**PodMonitor** CRD가 올바른 Pod를 가리키도록 구성되어 있는지 확인합니다. 자세한 내용은 Prometheus Operator 설명서를 참조하십시오.

4. **ServiceMonitor** CRD를 사용하고 Pod의 **/metrics** 끝점에 지표 데이터가 표시되는 경우 다음 단계에 따라 구성을 확인합니다.

   a. 서비스가 올바른 **/metrics** 엔드포인트를 가리키는지 확인합니다. 이 출력의 서비스레이블 은

서비스 모니터 레이블 과 우속 난계의 서비스에서 정의한/metrics 엔드포인트와 일치해야 합니다.

```
$ oc get service
```

출력 예

```
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) 이 API를 서비스 API라고 지정합니다.

(2) 이 서비스에 사용되는 레이블을 지정합니다.

b. serviceIP,port, /metrics 엔드포인트를 쿼리하여 이전에 실행한curl 명령의 동일한 메트릭을 확인합니다.

i. 다음 명령을 실행하여 서비스 IP를 찾습니다.

```
$ oc get service -n <target_namespace>
```

ii. /metrics 엔드포인트를 쿼리합니다.

```
$ oc exec <sample_pod> -n <sample_namespace> -- curl <service_IP>:
<port>/metrics
```

다음 예제에서 유효한 메트릭이 반환됩니다.

출력 예

```
% 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. 레이블 일치를 사용하여 ServiceMonitor 오브젝트가 원하는 서비스를 가리키도록 구성되어 있는지 확인합니다. 이렇게 하려면 oc get service 출력의 Service 오브젝트를 oc get service

monitor 출력의 **ServiceMonitor** 오브젝트와 비교합니다. 표시할 메트릭과 일치해야 하는 레이블입니다.
예를 들어 이전 단계에서 **Service** 오브젝트에 **app: prometheus-example-app** 레이블이 있고 **ServiceMonitor** 오브젝트에 동일한 **app: prometheus-example-app** 일치 라벨이 있는지 확인하십시오.

5. 모든 항목이 유효한 것으로 확인되고 메트릭을 계속 사용할 수 없는 경우 지원 팀에 문의하여 추가 지원을 받으십시오.