Red Hat OpenStack Platform 13

운영 측정

물리적 리소스 및 가상 리소스 추적 및 메트릭 수집

Last Updated: 2023-10-24

Red Hat OpenStack Platform 13 운영 측정

물리적 리소스 및 가상 리소스 추적 및 메트릭 수집

OpenStack Team
rhos-docs@redhat.com

법적 공지

초록

운영 도구를 사용하여 Red Hat OpenStack Platform 환경을 측정하고 유지 관리할 수 있습니다.

차례

1장. 운영 측정 소개

Red Hat OpenStack Platform 환경에서 Telemetry 서비스의 구성 요소를 사용하면 물리적 및 가상 리소스를 추적하고 배포의 CPU 사용량 및 리소스 가용성과 같은 지표를 Gnocchi 백엔드에 저장하는 데이터 수집 데몬으로 수집할 수 있습니다.

1.1. 작동 측정 이해

운영 도구를 사용하여 Red Hat OpenStack Platform 환경을 측정하고 유지 관리할 수 있습니다. 이러한 측정 도구는 다음과 같은 기능을 수행합니다.

- **가용성 모니터링**: RHOSP(Red Hat OpenStack Platform) 환경의 모든 구성 요소를 모니터링하고 구성 요소에 현재 정전이 있거나 작동하지 않는지 확인합니다. 또한 문제가 식별될 때 경고하도록 시스템을 구성할 수도 있습니다.

- **성능 모니터링**: 시스템 정보를 주기적으로 수집하고 데이터 수집 데몬을 사용하여 다양한 방법으로 값을 저장하고 모니터링하는 메커니즘을 제공합니다. 이 데몬은 수집한 데이터를 (예: 운영 체제 및 로그 파일) 저장하거나 네트워크를 통해 데이터를 사용할 수 있도록 합니다. 데이터에서 수집된 통계를 사용하여 시스템을 모니터링하고, 성능 병목 현상을 찾고, 향후 시스템 부하를 예측할 수 있습니다.

1.2. TELEMETRY 아키텍처

RHOSP(Red Hat OpenStack Platform) Telemetry는 OpenStack 기반 클라우드에 대한 사용자 수준 사용 데이터를 제공합니다. 고객 청구, 시스템 모니터링 또는 경고에 데이터를 사용할 수 있습니다. Compute usage events와 같은 기존 RHOSP 구성 요소에서 보낸 알림에서 데이터를 수집하거나 libvirt와 같은 RHOSP 인프라 리소스를 폴링하여 Telemetry 구성 요소를 구성할 수 있습니다. Telemetry에서는 데이터 저장소 및 메시지 큐를 포함하여 다양한 대상에 수집된 데이터를 게시합니다.

Telemetry는 다음 구성 요소로 구성됩니다.

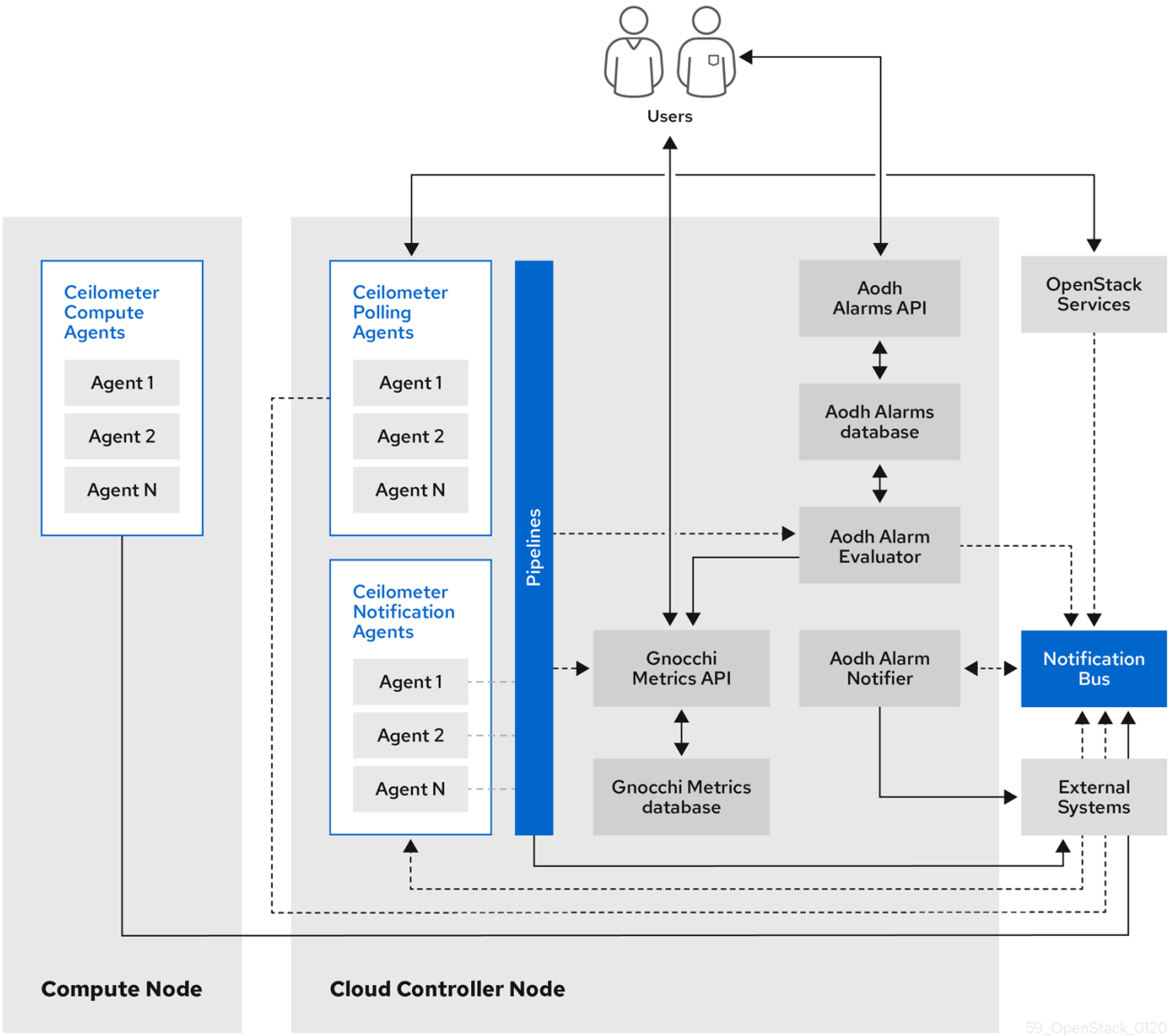
- **데이터 수집**: Telemetry는 Ceilometer를 사용하여 메트릭 및 이벤트 데이터를 수집합니다. 자세한 내용은 1.3.1절. “Ceilometer”의 내용을 참조하십시오.

- **Storage**: Telemetry는 Gnocchi에 지표 데이터를 저장하고 Panko에 이벤트 데이터를 저장합니다. 자세한 내용은 1.4절. “Gnocchi가 있는 스토리지” 의 내용을 참조하십시오.

- 알 **람 서비스**: Telemetry는 Aodh를 사용하여 지표 또는 Ceilometer에서 수집한 이벤트 데이터에 대해 정의된 규칙에 따라 작업을 트리거합니다.

데이터를 수집한 경우 타사 툴(예: Red Hat Cloudforms)을 사용하여 지표 데이터를 표시 및 분석하고, 알람 서비스 Aodh를 사용하여 이벤트에 대한 알람을 구성할 수 있습니다.

그림 1.1. Telemetry 아키텍처



1.3. 데이터 수집

RHOSP(Red Hat OpenStack Platform)는 다음 두 가지 유형의 데이터 수집을 지원합니다.

- 인프라 모니터링을 위한 collectd. 자세한 내용은 1.3.2절. “collectd”의 내용을 참조하십시오.
- OpenStack 구성 요소 수준 모니터링용 Ceilometer. 자세한 내용은 1.3.1절. “Ceilometer”의 내용을 참조하십시오.

1.3.1. Ceilometer

Ceilometer는 현재 OpenStack 핵심 구성 요소에서 데이터를 정규화하고 변환하는 기능을 제공하는 OpenStack Telemetry 서비스의 기본 데이터 수집 구성 요소입니다. Ceilometer는 OpenStack 서비스와 관련된 미터링 및 이벤트 데이터를 수집합니다. 수집된 데이터는 배포 구성에 따라 사용자가 액세스할 수 있습니다.

Ceilometer 서비스는 세 개의 에이전트를 사용하여 RHOSP(Red Hat OpenStack Platform) 구성 요소에서 데이터를 수집합니다.

- **컴퓨팅 에이전트(ceilometer-agent-compute)**: 각 컴퓨팅 노드에서 실행되고 리소스 사용률 통계에 대해 폴링합니다. 이 에이전트는 **--polling namespace-compute** 매개 변수를 사용하여 실행되는 폴링 에이전트 **ceilometer-polling** 과 동일합니다.

- **중앙 에이전트(ceilometer-agent-central)**: 중앙 관리 서버에서 를 실행하여 인스턴스 또는 컴퓨팅 노드에 연결되지 않은 리소스의 리소스 사용률 통계를 폴링합니다. 여러 에이전트를 시작하여 서비스를 수평으로 확장할 수 있습니다. 이는 **--polling namespace-central** 매개 변수를 사용하여 실행되는 폴링 에이전트 **ceilometer-polling** 과 동일합니다.

- **알림 에이전트(ceilometer-agent-notification)**: 중앙 관리 서버에서 실행되며 메시지 대기열의 메시지를 사용하여 이벤트 및 미터링 데이터를 빌드합니다. 그런 다음 정의된 대상에 데이터가 게시됩니다. 기본적으로 데이터는 Gnocchi로 푸시됩니다. 이러한 서비스는 RHOSP 알림 버스를 사용하여 통신합니다.

Ceilometer 에이전트는 게시자를 사용하여 해당 엔드포인트(예: Gnocchi)로 데이터를 전송합니다. 이 정보는 **pipeline.yaml** 파일에서 구성할 수 있습니다.

추가 리소스

- 게시자에 대한 자세한 내용은 1.3.1.1절. "게시자" 을 참조하십시오.

1.3.1.1. 게시자

원격 분석 서비스에서는 수집된 데이터를 외부 시스템으로 전송하는 여러 전송 방법을 제공합니다. 이러한 데이터의 소비자는 예를 들어 데이터 손실이 허용되는 시스템 모니터링 및 안정적인 데이터 운송이 필요한 청구 시스템 등 다양합니다. Telemetry는 두 시스템 유형의 요구 사항을 이행하는 방법을 제공합니다. 서비스의 게시자 구성 요소를 사용하여 메시지 버스를 통해 데이터를 영구 스토리지에 저장하거나 하나 이상의 외부 소비자에게 보낼 수 있습니다. 하나의 체인에는 여러 게시자가 포함될 수 있습니다.

다음과 같은 게시자 유형이 지원됩니다.

- **Gnocchi(기본값):** Gnocchi 게시자가 활성화되면 시계열 최적화된 스토리지의 경우 측정 및 리소스 정보가 Gnocchi로 푸시됩니다. Ceilometer로 ID 서비스에 Gnocchi를 등록하고 ID 서비스를 통해 정확한 경로를 검색할 수 있는지 확인합니다.

- Panko: Ceilometer의 이벤트 데이터를 **panko** 에 저장하면 HTTP REST 인터페이스를 통해 Red Hat OpenStack Platform에서 시스템 이벤트를 쿼리할 수 있습니다. 데이터를 panko로 푸시하려면 게시자를 **direct://?dispatcher=panko** 로 설정합니다.

1.3.1.1.1. 게시자 매개변수 구성

원격 분석 서비스 내의 각 데이터 포인트에 대해 멀티 게시자를 구성할 수 있으므로 동일한 기술 미터 또는 이벤트를 여러 대상에 게시할 수 있으며, 각각 다른 전송 방법을 사용할 수 있습니다.

절차

1. YAML 파일을 생성하여 가능한 게시자 매개변수 및 기본값(예: **ceilometer-publisher.yaml**)을 설명합니다. **parameter_defaults** 에 다음 매개변수를 삽입합니다.

```
parameter_defaults:

ManagePipeline: true
ManageEventPipeline: true
EventPipelinePublishers:
```

```
- gnocchi://?archive_policy=high
PipelinePublishers:
- gnocchi://?archive_policy=high
```

2. 사용자 정의 오버클라우드를 배포합니다. 오버클라우드를 배포하는 방법은 다음 두 가지가 있습니다.

   - **openstack overcloud deploy** 명령에 수정된 YAML 파일을 포함하여 게시자를 정의합니다. 다음 예제에서는 &lt **;environment-files** >를 배포에 포함할 다른 YAML 파일로 바꿉니다.

     ```
     $ openstack overcloud  deploy
     --templates \
     -e /home/custom/ceilometer-publisher.yaml
     -e <environment-files>
     ```

   - 모든 로컬 수정(예: **local_modifications.yaml**)을 포함하도록 YAML 파일을 생성합니다. 다음 예와 같이 스크립트를 사용하여 배포를 실행할 수 있습니다.

     ```
     $ sh deploy.sh:

     #!/bin/bash
     openstack overcloud deploy
     -e <environment-files> \
     -e local_modifications.yaml \
     ....
     ```

추가 리소스

- 매개변수에 대한 자세한 내용은 *Advanced Overcloud Customization* 가이드의 *Overcloud Parameters* 가이드 및 Parameters 에서 Telemetry 매개변수 를 참조하십시오.

1.3.2. collectd

성능 모니터링은 시스템 정보를 주기적으로 수집하고 데이터 수집 에이전트를 사용하여 다양한 방법으로 값을 저장하고 모니터링하는 메커니즘을 제공합니다. Red Hat은 collectd 데몬을 데이터 수집 에이전트로 지원합니다. 이 데몬은 시계열 데이터베이스에 데이터를 저장합니다. Red Hat 지원 데이터베이스 중 하나는 Gnocchi라고 합니다. 이 저장된 데이터를 사용하여 시스템을 모니터링하고, 성능 병목 현상을 찾고, 향후 시스템 부하를 예측할 수 있습니다.

추가 리소스

- Gnocchi에 대한 자세한 내용은 1.4절. “Gnocchi가 있는 스토리지” 을 참조하십시오.
- collectd에 대한 자세한 내용은 3.1절. “collectd 설치” 을 참조하십시오.

1.4. GNOCCHI가 있는 스토리지

Gnocchi는 오픈 소스 시계열 데이터베이스입니다. 매우 큰 규모의 지표를 저장하고 운영자 및 사용자에게 메트릭 및 리소스에 대한 액세스를 제공합니다. Gnocchi는 아카이브 정책을 사용하여 계산할 집계와 유지할 집계 수를 정의합니다. 인덱서 드라이버를 사용하여 모든 리소스, 보관 정책 및 지표의 인덱스를 저장합니다.

1.4.1. 보관 정책: 시계열 데이터베이스에 단기 및 장기 데이터 저장

보관 정책은 계산에 필요한 집계와 유지할 집계 수를 정의합니다. Gnocchi는 최소, 최대, 평균, Nth percentile 및 표준 편차와 같은 다양한 집계 방법을 지원합니다. 이러한 집계는 세분성이라는 기간 동안 계산되고 특정 시간 동안 유지됩니다.

보관 정책은 지표를 집계하는 방법과 저장 기간에 대해 정의합니다. 각 아카이브 정책은 시간별 포인트 수로 정의됩니다.

예를 들어 아카이브 정책이 1초 단위로 10개의 포인트 정책을 정의하면 시계열 아카이브는 각각 1초 이상의 집계를 나타내는 최대 10초를 유지합니다. 즉, 시계열은 최대 10초의 데이터가 더 최근 지점과 이전 지점 사이에 10초의 데이터를 유지합니다.

보관 정책은 사용되는 집계 방법도 정의합니다. 기본값은 기본적으로 mean, min, max. sum, std, count로 설정된 parameter **default_aggregation_methods** 로 설정됩니다. 따라서 사용 사례에 따라 아카이브 정책과 세분화가 달라집니다.

추가 리소스

- 보관 정책에 대한 자세한 내용은 보관 정책 계획 및 관리를 참조하십시오.

1.4.2. 인덱서 드라이버

인덱서는 모든 리소스, 보관 정책 및 메트릭의 인덱스를 정의, 유형 및 속성과 함께 저장합니다. 또한 리소스를 메트릭과 연결할 책임이 있습니다. Red Hat OpenStack Platform director는 기본적으로 인덱서 드라이버를 설치합니다. Gnocchi에서 처리하는 모든 리소스와 지표를 인덱싱하는 데이터베이스가 필요합니다. 지원되는 드라이버는 MySQL입니다.

1.4.3. Gnocchi Metric-as-a-Service 용어

이 표에는 Metric-as-a-Service 기능에 일반적으로 사용되는 용어 정의가 포함되어 있습니다.

표 1.1. metric-as-a-Service 용어

용어	정의
집계 방법	여러 개의 측정값을 집계하는 데 사용되는 함수입니다.A function used to aggregate multiple measures into an aggregate. 예를 들어 min 집계 메서드는 시간 범위의 모든 측정값의 최소 값에 대해 다른 측정값의 값을 집계합니다.For example, the min aggregation method aggregates the values of different measures to the minimum value of all the measures in the time range.
집계	데이터 지점 튜플 보관 정책에 따라 여러 측정에서 생성됩니다.A data point tuple generated from several measures according to the archive policy. 집계는 타임 스탬프 및 값으로 구성됩니다.
보관 정책	지표에 연결된 집계 스토리지 정책입니다. 아카이브 정책은 지표에 집계되는 기간과 집계(집계 방법)를 결정합니다.
세분성	집계된 시계열에 두 집계 사이의 시간입니다.
measure	API에서 시계열 데이터베이스로 전송된 들어오는 데이터 지점 튜플입니다. 측정값은 타임스탬프 및 값으로 구성됩니다.

용어	정의
메트릭	UUID로 식별된 집계를 저장하는 엔터티입니다. 이름을 사용하여 리소스에 지표를 연결할 수 있습니다. 지표에서 해당 집계를 저장하는 방법은 지표가 연결된 보관 정책에 의해 정의됩니다.
리소스	메트릭을 연결하는 인프라의 모든 항목을 나타내는 엔터티입니다. 리소스는 고유한 ID로 식별되며 속성을 포함할 수 있습니다.
시계열	시간별로 정렬된 집계 목록입니다.
TimeSpan	지표가 집계를 유지하는 시간입니다. 이는 보관 정책의 컨텍스트에서 사용됩니다.

1.5. 메트릭 데이터 표시

다음 도구를 사용하여 메트릭 데이터를 표시하고 분석할 수 있습니다.

- **Grafana**: 오픈 소스 지표 분석 및 시각화 모음입니다. Grafana는 일반적으로 인프라 및 애플리케이션 분석에 대한 시계열 데이터를 시각화하는 데 사용됩니다.
- **Red Hat CloudForms**: IT 부서에서 가상 시스템 및 프라이빗 클라우드 간의 규정 준수를 프로비저닝, 관리 및 보장하기 위해 사용자의 셀프 서비스 기능을 제어하는 데 사용하는 인프라 관리 플랫폼입니다.

추가 리소스

- Grafana에 대한 자세한 내용은 1.5.1절. “Grafana를 사용하고 연결하여 데이터 표시” 을 참조하십시오.
- Red Hat Cloudforms에 대한 자세한 내용은 제품 설명서 를 참조하십시오.

1.5.1. Grafana를 사용하고 연결하여 데이터 표시

타사 소프트웨어(예: Grafana)를 사용하여 수집 및 저장된 지표의 그래픽 표현을 볼 수 있습니다.

Grafana는 오픈 소스 지표 분석, 모니터링 및 시각화 모음입니다. Grafana를 설치하고 구성하려면 공식 Grafana 설명서 를 참조하십시오.

2장. 작동 측정 계획

모니터링하는 리소스는 비즈니스 요구 사항에 따라 다릅니다. Ceilometer 또는 collectd를 사용하여 리소스를 모니터링할 수 있습니다.

- collectd 측정에 대한 자세한 내용은 2.2절. "collectd 측정" 을 참조하십시오.
- Ceilometer 측정에 대한 자세한 내용은 2.1절. "Ceilometer 측정" 을 참조하십시오.

2.1. CEILOMETER 측정

Ceilometer 측정의 전체 목록은 https://docs.openstack.org/ceilometer/queens/admin/telemetry-measurements.html을 참조하십시오.

2.2. COLLECTD 측정

다음 측정은 가장 일반적으로 사용되는 collectd 지표입니다.

- disk
- 인터페이스
- load
- memory
- 프로세스
- tcpconns

전체 측정 목록은 collectd 지표 및 이벤트 를 참조하십시오.

2.3. GNOCCHI 및 CEILOMETER 성능 모니터링

openstack metric 명령을 사용하여 배포에서 보관 정책, 벤치마크, 측정, 메트릭 및 리소스를 관리할 수 있습니다.

절차

- 명령줄에서 **openstack metric status** 를 입력하여 배포에서 Gnocchi 설치를 모니터링하고 측정 상태를 확인합니다.

```
(overcloud) [stack@undercloud-0 ~]$ openstack metric status
+-----------------------------------------------------+-------+
| Field                                               | Value |
+-----------------------------------------------------+-------+
| storage/number of metric having measures to process | 0     |
| storage/total number of measures to process         | 0     |
+-----------------------------------------------------+-------+
```

2.4. 데이터 스토리지 계획

Gnocchi는 각 데이터 포인트가 집계인 데이터 포인트 컬렉션을 저장합니다. 스토리지 형식은 다른 기술을 사용하여 압축됩니다. 결과적으로 시계열 데이터베이스의 크기를 계산하기 위해 최악의 시나리오를 기반으로 크기를 추정합니다.

절차

1. 데이터 요소 수를 계산합니다.
   포인트 수 = 시간 간격 / 세분 단위

   예를 들어 1분 해상도가 있는 데이터의 연도를 유지하려면 수식을 사용합니다.For example, if you want to retain a year of data with one-minute resolution, use the formula:

   데이터 포인트 수 = (365일 X 24 시간 X 60 분) / 1 분 수 = 525600

2. 시계열 데이터베이스의 크기를 계산합니다.
   크기(바이트) = X 8 바이트의 데이터 지점 수

   이 수식을 예제에 적용하면 결과가 4.1MB입니다.

   크기(바이트 단위) = 525600은 X 8바이트 = 4204800바이트 = 4.1MB

   이 값은 통합된 시계열 데이터베이스에 대한 예상 스토리지 요구 사항입니다. 아카이브 정책에서 여러 집계 방법(최소, max, mean, sum, std, count)을 사용하는 경우 이 값을 사용하는 집계 방법 수로 곱합니다.

추가 리소스

- 자세한 내용은 1.4.1절. "보관 정책: 시계열 데이터베이스에 단기 및 장기 데이터 저장" 의 내용을 참조하십시오.

2.5. 보관 정책 계획 및 관리

아카이브 정책은 지표를 집계하는 방법과 시계열 데이터베이스에 지표를 저장하는 기간을 정의합니다. 보관 정책은 일정 기간 동안 포인트 수로 정의됩니다.

아카이브 정책이 1초 단위로 10개의 포인트 정책을 정의하면 시계열 아카이브는 최대 10초 동안 유지되며 각각 1초 이상 집계를 나타냅니다. 즉, 시계열은 가장 최근 지점과 이전 지점 사이에 최대 10초의 데이터가 유지됩니다. 보관 정책은 사용할 집계 방법도 정의합니다. 기본값은 parameter **default_aggregation_methods** 로 설정됩니다. 여기서 기본값은 **mean,min,max** 로 설정됩니다. **합계,std,count**. 따라서 사용 사례에 따라 보관 정책과 세분화가 다를 수 있습니다.

보관 정책을 계획하려면 다음 개념을 잘 알고 있어야 합니다.

- 지표. 자세한 내용은 2.5.1절. "지표"의 내용을 참조하십시오.
- 측정값. 자세한 내용은 2.5.2절. "사용자 정의 측정값 생성"의 내용을 참조하십시오.
- 집계. 자세한 내용은 2.5.4절. "시계열 집계의 크기 계산"의 내용을 참조하십시오.
- 지표 작업자. 자세한 내용은 2.5.5절. "지표 작업자"의 내용을 참조하십시오.

보관 정책을 생성하고 관리하려면 다음 작업을 완료합니다.

1. 아카이브 정책을 생성합니다. 자세한 내용은 2.5.6절. "아카이브 정책 생성"의 내용을 참조하십시오.

2. 보관 정책을 관리합니다. 자세한 내용은 2.5.7절. "아카이브 정책 관리"의 내용을 참조하십시오.

3. 보관 정책 규칙을 생성합니다. 자세한 내용은 2.5.8절. "아카이브 정책 규칙 생성"의 내용을 참조하십시오.

2.5.1. 지표

Gnocchi는 *metric* 이라는 오브젝트 유형을 제공합니다. 메트릭은 서버의 CPU 사용량, 방의 온도 또는 네트워크 인터페이스에서 전송하는 바이트 수를 측정할 수 있는 모든 것입니다. 메트릭에는 다음 속성이 있습니다.

- 식별할 UUID
- 이름
- 측정값을 저장하고 집계하는 데 사용되는 보관 정책

추가 리소스

- 용어 정의의 경우 Gnocchi Metric-as-a-Service 용어를 참조하십시오.

2.5.1.1. 메트릭 생성

절차

1. 리소스를 생성합니다. <resource_name>을 리소스 이름으로 바꿉니다.

```
$ openstack metric resource create <resource_name>
```

2. 지표를 생성합니다. <resource_name>을 리소스 이름으로 바꾸고 <metric_name>을 지표 이름으로 바꿉니다.

```
$ openstack metric metric create -r <resource_name> <metric_name>
```

지표를 생성할 때 보관 정책 속성이 고정되고 변경할 수 없습니다. **archive_policy** 엔드포인트를 통해 아카이브 정책의 정의를 변경할 수 있습니다.

2.5.2. 사용자 정의 측정값 생성

측정값은 API가 Gnocchi로 보내는 들어오는 데이터 포인트 튜플입니다. 타임 스탬프 및 값으로 구성됩니다. 사용자 고유의 사용자 지정 측정값을 만들 수 있습니다.You can create your own custom measures.

절차

- 사용자 지정 측정을 생성합니다.

```
$ openstack metric measures add -m <MEASURE1> -m <MEASURE2> .. -r
<RESOURCE_NAME> <METRIC_NAME>
```

2.5.3. 기본 보관 정책

기본적으로 Gnocchi에는 다음과 같은 아카이브 정책이 있습니다.

- 낮음 (LOW)
  - 30일 동안의 세분성 5분
  - 사용된 집계 방법 **default_aggregation_methods**
  - 지표당 최대 예상 크기: 406KiB
- 중간
  - 7일 동안의 1분 단위
  - 365일 1시간 동안의 세분성
  - 사용된 집계 방법 **default_aggregation_methods**
  - 메트릭당 최대 예상 크기: 887 KiB
- 높음
  - 1시간 동안의 두 번째 세분
  - 1분 단위 1주
  - 1년 동안의 세분성 1시간
  - 사용된 집계 방법 **default_aggregation_methods**
  - 지표당 최대 예상 크기: 1 057 KiB
- bool
  - 1년 동안의 두 번째 세분
  - 사용된 집계 방법: last
  - 메트릭당 최대 최적화 크기: 1539 KiB
  - 메트릭당 최대 배관 크기: 277 172 KiB

2.5.4. 시계열 집계의 크기 계산

Gnocchi는 각 포인트가 집계되는 데이터 포인트 컬렉션을 저장합니다. 스토리지 형식은 다른 기술을 사용하여 압축됩니다. 결과적으로 다음 예제와 같이 시계열 크기 계산은 최악의 시나리오를 기반으로 합니다.

절차

1. 이 수식을 사용하여 포인트 수를 계산합니다.
   포인트 수 = 시간 간격 / 세분 단위

   예를 들어 1분 단위로 해결된 데이터를 1년 단위로 유지하려면 다음을 수행하십시오.

   포인트 수 = (365 일 X 24 시간 X 60 분) / 1 분

   포인트 수 = 525600

2. 바이트 단위를 계산하려면 다음 수식을 사용합니다.To calculate the point size in bytes, use this formula:

크기(바이트) = X 8 바이트 수

크기(바이트 단위) = 525600은 X 8바이트 = 4204800바이트 = 4.1MB

이 값은 집계된 단일 시계열에 대한 예상 스토리지 요구 사항입니다. 아카이브 정책에 여러 집계 방법 - min, max, mean, sum, std, count를 사용하는 집계 방법 수로 곱합니다.

2.5.5. 지표 작업자

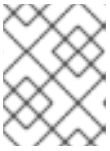
지표 데몬을 사용하여 측정값을 처리하고 집계를 생성하며 집계 스토리지에 측정값을 저장하고 지표를 삭제할 수 있습니다. 지표d 데몬은 Gnocchi의 대부분의 CPU 사용량 및 I/O 작업을 담당합니다. 각 지표의 보관 정책에 따라 지표 데몬이 수행하는 속도가 결정됩니다. 지표는 들어오는 스토리지에서 새 측정값을 주기적으로 확인합니다. 각 검사 간의 지연을 구성하려면 **[metricd]metric_processing_delay 구성** 옵션을 사용하면 됩니다.

2.5.6. 아카이브 정책 생성

절차

- 아카이브 정책을 생성합니다. <archive-policy-name>을 정책 이름으로 바꾸고 <aggregation-method>를 집계 방법으로 바꿉니다.

```
# openstack metric archive policy create <archive-policy-name> --definition <definition> \
--aggregation-method <aggregation-method>
```



참고

<definition>은 정책 정의입니다. 여러 속성을 쉼표(,)로 구분합니다. 다음과 같이 아카이브 정책 정의의 이름과 값을 콜론(:)과 분리합니다.

2.5.7. 아카이브 정책 관리

- 보관 정책을 삭제하려면 다음을 수행합니다.

```
openstack metric archive policy delete <archive-policy-name>
```

- 모든 보관 정책을 보려면 다음을 수행합니다.

```
# openstack metric archive policy list
```

- 아카이브 정책의 세부 정보를 보려면 다음을 수행합니다.

```
# openstack metric archive-policy show <archive-policy-name>
```

2.5.8. 아카이브 정책 규칙 생성

보관 정책 규칙은 지표와 보관 정책 간의 매핑을 정의합니다. 이렇게 하면 사용자가 규칙을 사전 정의할 수 있으므로 일치하는 패턴을 기반으로 보관 정책이 메트릭에 할당됩니다.

절차

- 보관 정책 규칙을 생성합니다. <rule-name>을 규칙 이름으로 바꾸고 <archive-policy-name>을 보관 정책 이름으로 바꿉니다.

```
# openstack metric archive-policy-rule create <rule-name> /
--archive-policy-name  <archive-policy-name>
```

3장. 운영 측정 툴 설치 및 구성

데이터 수집 에이전트, collectd 및 시계열 데이터베이스 Gnocchi를 설치해야 합니다.

3.1. COLLECTD 설치

collectd를 설치할 때 환경에 맞게 여러 collectd 플러그인을 구성할 수 있습니다.

절차

1. **/usr/share/openstack-tripleo-heat-templates/environments/collectd-environment.yaml** 파일을 로컬 디렉터리에 복사합니다.
2. **collectd-environment.yaml** 을 열고 **CollectdExtraPlugins** 에서 원하는 플러그인을 나열합니다. **ExtraConfig** 섹션에서 매개변수를 제공할 수도 있습니다.

```
parameter_defaults:
  CollectdExtraPlugins:
    - disk
    - df
    - virt

  ExtraConfig:
    collectd::plugin::virt::connection: "qemu:///system"
    collectd::plugin::virt::hostname_format: "hostname uuid"
```

   기본적으로 collectd는 **디스크**,**인터페이스**,**로드**,**메모리**,**프로세스**, **tcpconns** 플러그인과 함께 제공됩니다. **CollectdExtraPlugins** 매개변수를 사용하여 추가 플러그인을 추가할 수 있습니다. 표시된 대로 **ExtraConfig** 옵션을 사용하여 CollectdExtraPlugins에 대한 추가 구성 정보를 제공할 수도 있습니다. 이 예제에서는 **virt** 플러그인을 추가하고 연결 문자열과 호스트 이름 형식을 구성합니다.

3. **openstack overcloud deploy** 명령에 수정된 YAML 파일을 포함하여 모든 오버클라우드 노드에 collectd 데몬을 설치합니다.

```
$ openstack overcloud deploy
--templates \/home/templates/environments/collectd.yaml \
-e /path-to-copied/collectd-environment.yaml
```

추가 리소스

- collectd에 대한 자세한 내용은 1.3.2절. “collectd” 을 참조하십시오.
- collectd 플러그인 및 구성을 보려면 *Service Telemetry Framework* 가이드의 collectd 플러그인을 참조하십시오.

3.2. GNOCCHI 설치

기본적으로 Gnocchi는 언더클라우드에서 활성화되어 있지 않습니다. Red Hat은 제한된 리소스 및 단일 장애 조치로 인해 언더클라우드가 처리할 수 없는 많은 데이터를 생성하여 언더클라우드에서 Telemetry를 활성화하는 것을 권장하지 않습니다.

기본적으로 Telemetry 및 Gnocchi는 컨트롤러 및 컴퓨팅 노드에 설치됩니다. Gnocchi의 기본 스토리지 백엔드는 file입니다.

다음 두 가지 방법 중 하나로 오버클라우드에 Gnocchi를 배포할 수 있습니다.

- 내부. 자세한 내용은 3.2.1절. "내부적으로 Gnocchi 배포"의 내용을 참조하십시오.
- 외부에서. 자세한 내용은 3.2.2절. "외부적으로 Gnocchi 배포"의 내용을 참조하십시오.

3.2.1. 내부적으로 Gnocchi 배포

기본 배포는 internal입니다.

절차

- collectd를 배포하여 지표 데이터를 내부 Gnocchi로 보내려면 **/usr/share/openstack-tripleo-heat-templates/environments/services/collectd.yaml** 을 **overcloud deploy** 명령에 추가합니다.

추가 리소스

- 자세한 내용은 3.1절. "collectd 설치"의 내용을 참조하십시오.

3.2.2. 외부적으로 Gnocchi 배포

절차

1. 로컬 디렉터리(예: **ExternalGnocchi.yaml**)에 사용자 지정 YAML 파일을 생성하고 다음 세부 정보를 포함해야 합니다.

```
CollectdGnocchiServer: <IPofExternalServer>
CollectdGnocchiUser: admin
CollectdGnocchiAuth: basic
```

2. Gnocchi를 배포하려면 사용자 지정 YAML 파일을 **overcloud deploy** 명령에 추가합니다. 을 기존 배포의 일부인 환경 파일 목록으로 바꿉니다 **<existing_overcloud_environment_files>**.

```
openstack overcloud deploy \
-e <existing_overcloud_environment_files> \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/collectd.yaml \
-e /home/templates/environments/ExternalGnocchi.yaml \
...
```

참고

다음 YAML 파일에서 모든 Gnocchi 매개변수를 찾을 수 있습니다. **/usr/share/openstack-tripleo-heat-templates/puppet/services/metrics/collectd.yaml**

3.2.3. Gnocchi 배포 확인

절차

- 새 리소스 및 지표를 나열합니다.

```
$ (overcloud) [stack@undercloud-0 ~]$ openstack metric  metric list |more
+--------------------------------------+---------------------+-------------------------------------------
-+------+--------------------------------------+
| id                                   | archive_policy/name | name                                      | unit | resource_id
|
+--------------------------------------+---------------------+-------------------------------------------
-+------+--------------------------------------+
| 001715fe-8ad1-4f21-b0fa-1dee2b1c8201 | low    | interface-eth4@if_packets-rx  | None |
e1357192-e563-5524-b0f0-b30fd11ea8df |
| 003057df-0271-4c59-b793-f885fe09668a | low    | pconns-6000-local@tcp_connections-
LISTEN   | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 0030a5ba-0e4a-4cce-a48b-4122bfbc4e7a | low    | tcpconns-8004-local@tcp_connections-
CLOSING   | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 0030e0a7-2cea-42db-860c-fa9ab175c8e1 | low    | tcpconns-25-local@tcp_connections-
CLOSING     | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 007ec848-f16a-48b8-8706-742a28e2efcf | low    | memcached-
local@memcached_command-get         | None | 8900c37d-501a-565d-b38c-85e5a07a0463 |
| 009d08c7-7483-4e0a-b1bd-0b1451a3b2ab | low | tcpconns-8041-local@tcp_connections-
TIME_WAIT | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 00a8183c-910e-4849-8d08-9401fbe82029 | low    | tcpconns-11211-local@tcp_connections-
CLOSING  | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 010af93c-62be-442a-94da-5cb426053fe8 | low    | tcpconns-4567-local@tcp_connections-
FIN_WAIT2 | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 0113b4f3-786d-4a0a-bb4b-59417d63b60f | low    | tcpconns-38857-local@tcp_connections-
LAST_ACK | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 0128dac7-b237-4e4c-8d5f-83f6e53771b4 | low    | tcpconns-37871-local@tcp_connections-
SYN_SENT | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 014ca1f7-565d-4ce7-b35f-52fd916a8b06 | low    | tcpconns-43752-local@tcp_connections-
TIME_WAIT   | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 0154f901-18c2-4dd9-a593-db26d356611a | low  | tcpconns-1993-local@tcp_connections-
CLOSED    | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 0158d618-d3ba-45e8-bce8-33411d6f35e3 | low    | tcpconns-111-local@tcp_connections-
CLOSED     | None | e1357192-e563-5524-b0f0-b30fd11ea8df |
| 016fe93f-2794-490e-8057-fd5ab75eb4ec | low      | tcpconns-6001-local@tcp_connections-
SYN_RECV  | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 01ce3b82-98ad-4f61-88d0-ba46b9862688 | low    | interface-br-tenant@if_dropped-rx
| None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 01d4d186-cf26-4264-87a0-9f5deb8872b5 | low    | tcpconns-8774-local@tcp_connections-
ESTABLISHED  | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 01f19617-3ef6-43e8-9ad8-8c84b923f13f | low      | tcpconns-43394-
local@tcp_connections-FIN_WAIT2   | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 020646e9-2d50-4126-9a63-a5f36180cc0b | low    | tcpconns-6000-local@tcp_connections-
CLOSE_WAIT   | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 02155fd3-0d68-4eec-8cd5-8b158f5f03a4 | low      | tcpconns-6633-
local@tcp_connections-LISTEN    | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 0236355e-3415-4a72-99f2-38ada7b3db68 | low      | tcpconns-43806-
local@tcp_connections-FIN_WAIT2   | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 024e89d2-aa77-49c0-ae6e-65a665db01b3 | low      | tcpconns-35357-
local@tcp_connections-FIN_WAIT2   | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 027adb62-272f-4331-8167-28c3489d3b44 | low      | tcpconns-9292-
local@tcp_connections-LISTEN    | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 0290d15e-7687-4683-bd25-4c030cad12cf | low      | tcpconns-37378-
local@tcp_connections-CLOSE_WAIT  | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 02a5383f-b061-4422-9570-bfd3b2532832 | low      | processes@ps_state-zombies
| None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 02c959d5-8ae2-4d14-a140-189530b4e8f6 | low      | disk-vda2@disk_merged-write
```

```
| None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 02d174b5-6783-4db5-bfe7-8d45931aa0b0 | low | interface-br-tun@if_octets-rx
| None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 02d8c001-90da-4997-bfa5-e5d3afe599fa | low | tcpconns-25672-
local@tcp_connections-CLOSING | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 02d932f0-5745-4694-86d9-84e365789a7d | low | tcpconns-9292-
local@tcp_connections-CLOSING | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 02e1a5e2-194d-4e49-8b36-a843b5dbdc3d | low | tcpconns-45228-
local@tcp_connections-CLOSE_WAIT | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 02e5dcec-f5b7-41e9-9f3c-714ade502836 | low | load@load-5min |
None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 02fe05ed-e4a5-4a18-ad6c-64f8787225c9 | low | tcpconns-8774-
local@tcp_connections-SYN_SENT | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 03129089-7bbd-47f3-ab14-9cd583d775ae | low | tcpconns-6379-
local@tcp_connections-LAST_ACK | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 032b1771-93c4-4051-bbec-9115c9d329c4 | low | tcpconns-8042-
local@tcp_connections-TIME_WAIT | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 033516d5-ac15-4767-b005-cff52276badd | low | tcpconns-22-local@tcp_connections-
CLOSED | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 03589183-efa7-43ed-aea6-9d3c8accf97a | low | interface-br-tun@if_errors-tx
| None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 0360077f-729d-4591-a9fc-d96fbb7e0326 | low | tcpconns-6080-
local@tcp_connections-CLOSING | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 0365f5a1-4a98-435e-a809-c8706b0671bd | low | tcpconns-34296-
local@tcp_connections-LAST_ACK | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 039c57b7-ae5b-4f13-846e-8e5f184cdc4d | low | tcpconns-111-
local@tcp_connections-CLOSE_WAIT | None | 926d87fe-b388-501d-afa6-dc6af55ce4ad |
| 04169046-80c4-478e-b263-b9529b5ca739 | low | interface-br-tenant@if_packets-tx
| None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 0432314d-eb8b-4bf9-ab8f-5ecc5b30592d | low | tcpconns-37415-
local@tcp_connections-CLOSE_WAIT | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 043d9d6e-5db0-40e4-83b1-b8120506e9ec | low | tcpconns-6379-
local@tcp_connections-ESTABLISHED | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 04639fe9-6dc1-46eb-8b86-9b39e8fd782d | low | tcpconns-35357-
local@tcp_connections-SYN_RECV | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 04871c86-b176-45ed-9f37-bb6fae27e930 | low | tcpconns-8042-
local@tcp_connections-CLOSE_WAIT | None | caac8e14-0a2f-5e31-b4bd-e07a14361d81 |
| 049c50c4-bf5e-4098-988a-fb867649ee17 | low | tcpconns-11211-
local@tcp_connections-SYN_SENT | None | 54b1ecb3-5ccf-533f-84ab-e86eb426606f |
| 04e37efb-fc67-418e-b53b-6b8055967a2a | low | tcpconns-8004-
local@tcp_connections-CLOSING | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
| 04ef0d0f-db22-4501-ae4a-4bc9ee719075 | low | tcpconns-9200-
local@tcp_connections-SYN_RECV | None | 33ac72d9-ab4d-54f1-9fcf-56e02f770b70 |
```

4장. 운영 측정 관리

4.1. 배포를 기반으로 환경 변수 수정

절차

1. **/usr/share/openstack-tripleo-heat-templates/environments/gnocchi-environment.yaml** 파일을 홈 디렉터리에 복사합니다.

2. 환경에 맞게 매개변수를 수정합니다. YAML 파일에서 다음 기본 매개변수를 수정할 수 있습니다.

   - GnocchiIndexerBackend: 사용할 데이터베이스 인덱서 백엔드(예: **mysql**)입니다. 자세한 내용은 https://github.com/openstack/tripleo-heat-templates/blob/stable/queens/puppet/services/gnocchi-base.yaml#L33

   - GnocchiBackend: 임시 스토리지의 유형입니다. 값은 **rbd**,**swift** 또는 **파일** (ceph)일 수 있습니다. 자세한 내용은 https://github.com/openstack/tripleo-heat-templates/blob/stable/queens/environments/storage-environment.yaml#L29-L30에서 참조하십시오.

   - NumberOfStorageSacks: 스토리지 스패치 수입니다. 자세한 내용은 4.1.2절. "Sacks 수"의 내용을 참조하십시오.

3. 사용자 환경과 관련된 기타 환경 파일과 함께 **overcloud deploy** 명령에 **gnocchi-environment.yaml** 을 추가하고 배포합니다. 기존 배포의 일부인 환경 파일 목록으로 바꿉니다 **<existing_overcloud_environment_files>**.

```
$ openstack overcloud deploy \
<existing_overcloud_environment_files> \
-e ~gnocchi-environment.yaml \
...
```

4.1.1. 지표 작업자 실행

기본적으로 **gnocchi-metricd** 데몬은 컴퓨팅 메트릭 집계 시 CPU 사용을 최대화하기 위해 CPU 전력을 확장합니다.

절차

- **openstack metric status** 명령을 사용하여 HTTP API를 쿼리하고 지표 처리 상태를 검색합니다.

```
# openstack metric status
```

  명령 출력에서는 **gnocchi-metrid** 데몬의 백로그 처리를 보여줍니다. 이 백로그가 지속적으로 증가하지 않는 한 **gnocchi-metricd** 는 수집 중인 지표 수에 대처할 수 있음을 의미합니다. 처리할 측정값 수가 지속적으로 증가하는 경우 **gnocchi-metricd** 데몬 수를 늘립니다. 여러 서버에서 지표 데몬을 실행할 수 있습니다.

4.1.2. Sacks 수

Gnocchi에 수신되는 지표 데이터는 서로 다른 스랙으로 푸시되며 각 스패스는 처리를 위해 하나 이상의 **gnocchi-metricd** 데몬에 할당됩니다. 스패치 수는 시스템이 캡처하는 활성 메트릭에 따라 다릅니다.

Red Hat은 **gnocchi-metricd** 작업자의 총 수보다 많은 스패치 수를 권장합니다.

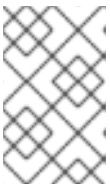
4.1.3. Sack 크기 변경

원래 예상했던 것보다 더 많은 메트릭을 수집하려는 경우 스패치 크기를 변경할 수 있습니다.

Gnocchi로 푸시된 측정 데이터는 더 나은 배포를 위해 스패치로 나뉩니다. 들어오는 메트릭은 특정 스ack에 푸시되고 각 스ack은 처리를 위해 하나 이상의 **gnocchi-metricd** 데몬에 할당됩니다. 스패치 수를 설정하려면 시스템에서 캡처하는 활성 메트릭 수를 사용합니다. 스패킹 수는 활성 **gnocchi-metricd** 작업자의 총 수보다 커야 합니다.

절차

- 설정할 적절한 sacks 값을 결정하려면 다음 식을 사용합니다.
  sacks value = **활성** 메트릭 수 / 300



참고

예상 지표 수가 절대 최대값인 경우 값을 500으로 나눕니다. 예상 활성 메트릭 수가 보수적이고 증가할 것으로 예상되는 경우 해당 값을 100으로 나누어 성장을 수용합니다.

4.2. 시계열 데이터베이스 서비스 모니터링

HTTP API의 **/v1/status** 끝점에서 처리할 작업 수(백로그)와 같은 정보를 반환합니다. 다음 조건은 정상 시스템을 나타냅니다.

- HTTP 서버와 **gnocchi-metricd** 가 실행 중입니다.
- HTTP 서버와 **gnocchi-metricd** 는 로그 파일에 오류 메시지를 쓰지 않습니다.

절차

- 시계열 데이터베이스의 상태를 확인합니다.

```
# openstack metric status
```

출력에는 시계열 데이터베이스의 상태와 처리할 지표 수가 표시됩니다. 더 낮은 값은 여기에서 더 좋습니다. 이상적으로는 0에 가깝습니다.

4.3. 시계열 데이터베이스 백업 및 복원

불행한 이벤트에서 복구할 수 있으려면 인덱스와 스토리지를 백업하십시오. PostgreSQL 또는 MySQL을 사용하여 데이터베이스 덤프를 생성하고 Ceph, Swift 또는 파일 시스템을 사용하여 데이터 스토리지의 스냅샷 또는 사본을 가져와야 합니다.

절차

1. 인덱스 및 스토리지 백업을 복원합니다.
2. 필요한 경우 Gnocchi를 다시 설치하십시오.
3. Gnocchi를 다시 시작합니다.

4.4. 작업 보기

특정 리소스에 대한 측정값 목록을 볼 수 있습니다.

절차

1. **메트릭 측정** 값 명령을 사용합니다.

```
# openstack metric measures show --resource-id UUID <METER_NAME>
```

2. 타임스탬프 범위 내에서 특정 리소스에 대한 측정값을 나열합니다.

```
# openstack metric measures show --aggregation mean --start <START_TIME> --stop
<STOP_TIME> --resource-id UUID <METER_NAME>
```

타임스탬프 변수 <START_TIME> 및 <END_TIME> 형식은 iso-dateTh:mm:ss를 사용합니다.

4.5. 리소스 유형 관리

리소스 유형을 생성, 보기 및 삭제할 수 있습니다. 기본 리소스 유형은 일반이지만 추가 특성을 사용하여 고유한 리소스 유형을 생성할 수 있습니다.

절차

1. 새 리소스 유형을 생성합니다.

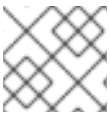
```
$ openstack metric resource-type create testResource01 -a bla:string:True:min_length=123
+-----------------------------------------------------------------------------+
| Field          | Value                                                      |
+-----------------------------------------------------------------------------+
| attributes/bla | max_length=255, min_length=123, required=True, type=string |
| name           | testResource01                                             |
| state          | active                                                     |
+-----------------------------------------------------------------------------+
```

2. 리소스 유형의 구성을 검토합니다.

```
$ openstack metric resource-typegnocchi resource-type show testResource01
+-----------------------------------------------------------------------------+
| Field          | Value                                                      |
+-----------------------------------------------------------------------------+
| attributes/bla | max_length=255, min_length=123, required=True, type=string |
| name           | testResource01                                             |
| state          | active                                                     |
+-----------------------------------------------------------------------------+
```

3. 리소스 유형을 삭제합니다.

```
$ openstack metric resource-type delete testResource01
```



참고

리소스가 사용 중인 경우 리소스 유형을 삭제할 수 없습니다.

4.6. 클라우드 사용량 측정 보기

절차

- 각 프로젝트의 모든 인스턴스의 평균 메모리 사용량을 확인합니다.

```
openstack metrics measures aggregation --resource-type instance --groupby project_id -m
“memoryView L3” --resource-id UUID
```

4.6.1. L3 캐시 모니터링 활성화

Intel 하드웨어 및 **libvirt** 버전이 Cache Monitoring Technology(CMT)를 지원하는 경우 **cpu_l3_cache** 미터를 사용하여 인스턴스에서 사용하는 L3 캐시 양을 모니터링할 수 있습니다.

L3 캐시를 모니터링하려면 다음 매개변수와 파일이 있어야 합니다.

- **LibvirtEnabledPerfEvents** 매개변수의 CMT.
- **gnocchi_resources.yaml 파일의 cpu_l3_cache**
- **Ceilometer 폴링.yaml 파일의 cpu_l3_cache**

절차

1. **Telemetry용 YAML 파일을 생성합니다(예: ceilometer-environment.yaml).**
2. **ceilometer-environment.yaml 파일에서 LibvirtEnabledPerfEvents 매개변수에 cmt 를 추가합니다. 자세한 내용은 /usr/share/openstack-triple-heat-templates/puppet/services/nova_libvirt.yaml 을 참조하십시오.**
3. **이 YAML 파일을 사용하여 오버클라우드를 배포합니다. 기존 배포의 일부인 환경 파일 목록으로 바꿉니다 <existing_overcloud_environment_files>.**

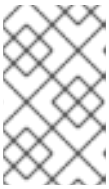
```
#!/bin/bash

openstack overcloud deploy \
--templates \
<existing_overcloud_environment_files> \
-e  /home/stack/ceilometer-environment.yaml \
...
```

4.

컴퓨팅 노드의 **Gnocchi**에서 **cpu_l3_cache** 가 활성화되었는지 확인합니다.

```
$ sudo -i
# docker exec -ti ceilometer_agent_compute cat /etc/ceilometer/gnocchi_resources.yaml |
grep cpu_l3_cache
//Verify that cpu_l3_cache is enabled for Telemetry polling.
# docker exec -ti ceilometer_agent_compute cat /etc/ceilometer/polling.yaml  | grep
cpu_l3_cache
//If cpu_l3_cache is not enabled for Telemetry, enable it and restart the service.
# docker exec -ti ceilometer_agent_compute echo "        - cpu_l3_cache" >>
/etc/ceilometer/polling.yaml
# docker exec -ti ceilometer_agent_compute pkill -HUP -f "ceilometer.*master process"
```



참고

컨테이너 이미지의 설정을 변경해도 재부팅은 유지되지 않습니다.

5.

이 컴퓨팅 노드에서 게스트 인스턴스를 시작한 후 **CMT** 메트릭을 모니터링합니다.

```
(overcloud) [stack@undercloud-0 ~]$ openstack metric measures show --resource-id
a6491d92-b2c8-4f6d-94ba-edc9dfde23ac cpu_l3_cache
+---------------------------+-------------+-----------+
| timestamp                 | granularity |     value |
+---------------------------+-------------+-----------+
| 2017-10-25T09:40:00+00:00 |       300.0 | 1966080.0 |
| 2017-10-25T09:45:00+00:00 |       300.0 | 1933312.0 |
| 2017-10-25T09:50:00+00:00 |       300.0 | 2129920.0 |
| 2017-10-25T09:55:00+00:00 |       300.0 | 1966080.0 |
| 2017-10-25T10:00:00+00:00 |       300.0 | 1933312.0 |
| 2017-10-25T10:05:00+00:00 |       300.0 | 2195456.0 |
| 2017-10-25T10:10:00+00:00 |       300.0 | 1933312.0 |
| 2017-10-25T10:15:00+00:00 |       300.0 | 1998848.0 |
| 2017-10-25T10:20:00+00:00 |       300.0 | 2097152.0 |
| 2017-10-25T10:25:00+00:00 |       300.0 | 1933312.0 |
| 2017-10-25T10:30:00+00:00 |       300.0 | 1966080.0 |
| 2017-10-25T10:35:00+00:00 |       300.0 | 1933312.0 |
| 2017-10-25T10:40:00+00:00 |       300.0 | 1933312.0 |
| 2017-10-25T10:45:00+00:00 |       300.0 | 1933312.0 |
| 2017-10-25T10:50:00+00:00 |       300.0 | 2850816.0 |
| 2017-10-25T10:55:00+00:00 |       300.0 | 2359296.0 |
| 2017-10-25T11:00:00+00:00 |       300.0 | 2293760.0 |
+---------------------------+-------------+-----------+
```

4.7. GNOCCHI 업그레이드

기본적으로 **Red Hat OpenStack Platform director**를 사용하여 배포를 업그레이드합니다. 배포 업그레이드에 대한 자세한 내용은 **Red Hat OpenStack Platform** 업그레이드 를 참조하십시오. **Red Hat**

OpenStack Platform 10을 사용하고 Red Hat OpenStack Platform 13으로 업그레이드하려면 Fast Forward Upgrades 를 참조하십시오.

5장. 알람 관리

aodh라는 알람 서비스를 사용하여 지표 또는 Ceilometer 또는 Gnocchi에서 수집한 이벤트 데이터에 대해 정의된 규칙에 따라 작업을 트리거할 수 있습니다.

5.1. 기존 알람 보기

절차

1. 기존 Telemetry 알람을 나열합니다.

```
# openstack alarm list
+--------------------------------------+---------------------------------------+-----------------------
--+------------------+----------+---------+
| alarm_id                             | type                                  | name                  | state
| severity | enabled |
+--------------------------------------+---------------------------------------+-----------------------
--+------------------+----------+---------+
| 922f899c-27c8-4c7d-a2cf-107be51ca90a |
gnocchi_aggregation_by_resources_threshold | iops-monitor-read-requests |
insufficient data | low      | True    |
+--------------------------------------+---------------------------------------+-----------------------
--+------------------+----------+---------+
```

2. 리소스에 할당된 미터를 나열하려면 리소스의 UUID를 지정합니다. 예를 들면 다음과 같습니다.

```
# openstack resource show 5e3fcbe2-7aab-475d-b42c-a440aa42e5ad
```

5.2. 알람 생성

aodh를 사용하여 임계값에 도달할 때 활성화되는 알람을 생성할 수 있습니다. 이 예에서 알람은 개별 인스턴스의 평균 CPU 사용률이 80%를 초과하면 로그 항목을 활성화하고 추가합니다.

절차

1. 알람을 생성하고 쿼리를 사용하여 모니터링을 위해 인스턴스의 특정 id(94619081-abf5-4f1f-81c7-9cedaa872403)를 격리합니다.

```
# openstack alarm create --type gnocchi_aggregation_by_resources_threshold --name
cpu_usage_high --metric cpu_util --threshold 80 --aggregation-method sum --
resource-type instance --query '{"=": {"id": "94619081-abf5-4f1f-81c7-9cedaa872403"}}'
```

```
--alarm-action 'log://'
+---------------------------+-------------------------------------------------------------+
| Field                     | Value                                                       |
+---------------------------+-------------------------------------------------------------+
| aggregation_method        | sum                                                         |
| alarm_actions             | [u'log://']                                                 |
| alarm_id                  | b794adc7-ed4f-4edb-ace4-88cbe4674a94                        |
| comparison_operator       | eq                                                          |
| description               | gnocchi_aggregation_by_resources_threshold alarm rule       |
| enabled                   | True                                                        |
| evaluation_periods        | 1                                                           |
| granularity               | 60                                                          |
| insufficient_data_actions | []                                                          |
| metric                    | cpu_util                                                    |
| name                      | cpu_usage_high                                              |
| ok_actions                | []                                                          |
| project_id                | 13c52c41e0e543d9841a3e761f981c20                            |
| query                     | {"=": {"id": "94619081-abf5-4f1f-81c7-9cedaa872403"}}       |
| repeat_actions            | False                                                       |
| resource_type             | instance                                                    |
| severity                  | low                                                         |
| state                     | insufficient data                                           |
| state_timestamp           | 2016-12-09T05:18:53.326000                                  |
| threshold                 | 80.0                                                        |
| time_constraints          | []                                                          |
| timestamp                 | 2016-12-09T05:18:53.326000                                  |
| type                      | gnocchi_aggregation_by_resources_threshold                  |
| user_id                   | 32d3f2c9a234423cb52fb69d3741dbbc                            |
+---------------------------+-------------------------------------------------------------+
```

2. 기존 임계값 알람을 편집하려면 **aodh alarm update** 명령을 사용합니다. 예를 들어 알람 임계값을 **75%**로 늘리려면 다음 명령을 사용합니다.

```
# openstack alarm update --name cpu_usage_high --threshold 75
```

5.3. 알람 비활성화

절차

- 알람을 비활성화하려면 다음 명령을 입력합니다.

```
# openstack alarm update --name cpu_usage_high --enabled=false
```

5.4. 알람 삭제

절차

- 알람을 삭제하려면 다음 명령을 입력합니다.

```
# openstack alarm delete --name cpu_usage_high
```

5.5. 예: 인스턴스의 디스크 활동 모니터링

다음 예제에서는 **aodh** 알람을 사용하여 특정 프로젝트에 포함된 모든 인스턴스에 대한 누적 디스크 활동을 모니터링하는 방법을 보여줍니다.

절차

1. 기존 프로젝트를 검토하고 모니터링할 프로젝트의 적절한 **UUID**를 선택합니다. 이 예에서는 **admin** 테넌트를 사용합니다.

```
$ openstack project list
+----------------------------------+----------+
| ID                               | Name     |
+----------------------------------+----------+
| 745d33000ac74d30a77539f8920555e7 | admin    |
| 983739bb834a42ddb48124a38def8538 | services |
| be9e767afd4c4b7ead1417c6dfedde2b | demo     |
+----------------------------------+----------+
```

2. 프로젝트 **UUID**를 사용하여 **admin** 테넌트의 인스턴스에서 생성한 모든 읽기 요청의 **sum()**을 분석하는 알람을 생성합니다. **--query** 매개변수를 사용하여 쿼리를 추가로 **restrain**할 수 있습니다.

```
# openstack alarm create --type gnocchi_aggregation_by_resources_threshold --name
iops-monitor-read-requests --metric disk.read.requests.rate --threshold 42000 --
aggregation-method sum --resource-type instance --query '{"=": {"project_id":
"745d33000ac74d30a77539f8920555e7"}}'
+---------------------------+-------------------------------------------------------------+
| Field                     | Value                                                       |
+---------------------------+-------------------------------------------------------------+
| aggregation_method        | sum                                                         |
| alarm_actions             | []                                                          |
| alarm_id                  | 192aba27-d823-4ede-a404-7f6b3cc12469                        |
| comparison_operator       | eq                                                          |
| description               | gnocchi_aggregation_by_resources_threshold alarm rule       |
| enabled                   | True                                                        |
| evaluation_periods        | 1                                                           |
| granularity               | 60                                                          |
| insufficient_data_actions | []                                                          |
| metric                    | disk.read.requests.rate                                     |
| name                      | iops-monitor-read-requests                                  |
| ok_actions                | []                                                          |
```

```
| project_id              | 745d33000ac74d30a77539f8920555e7                          |
| query                   | {"=": {"project_id": "745d33000ac74d30a77539f8920555e7"}} |
| repeat_actions          | False                                                     |
| resource_type           | instance                                                  |
| severity                | low                                                       |
| state                   | insufficient data                                         |
| state_timestamp         | 2016-11-08T23:41:22.919000                                |
| threshold               | 42000.0                                                   |
| time_constraints        | []                                                        |
| timestamp               | 2016-11-08T23:41:22.919000                                |
| type                    | gnocchi_aggregation_by_resources_threshold                |
| user_id                 | 8c4aea738d774967b4ef388eb41fef5e                          |
+-------------------------+-----------------------------------------------------------+
```

5.6. 예: CPU 사용 모니터링

인스턴스의 성능을 모니터링하려면 Gnocchi 데이터베이스를 검사하여 메모리 또는 CPU 사용량과 같이 모니터링할 수 있는 지표를 확인합니다.

절차

1. 인스턴스 UUID와 함께 openstack metric resource show 명령을 입력하여 모니터링할 수 있는 지표를 확인합니다.

```
$ openstack metric resource show --type instance d71cdf9a-51dc-4bba-8170-
9cd95edd3f66
+-----------------------+--------------------------------------------------------------------------+
| Field                 | Value                                                                    |
+-----------------------+--------------------------------------------------------------------------+
| created_by_project_id | 44adccdc32614688ae765ed4e484f389                                         |
| created_by_user_id    | c24fa60e46d14f8d847fca90531b43db                                         |
| creator               |
c24fa60e46d14f8d847fca90531b43db:44adccdc32614688ae765ed4e484f389   |
| display_name          | test-instance                                                            |
| ended_at              | None                                                                     |
| flavor_id             | 14c7c918-df24-481c-b498-0d3ec57d2e51                                     |
| flavor_name           | m1.tiny                                                                  |
| host                  | overcloud-compute-0                                                      |
| id                    | d71cdf9a-51dc-4bba-8170-9cd95edd3f66                                     |
| image_ref             | e75dff7b-3408-45c2-9a02-61fbfbf054d7                                     |
| metrics               | compute.instance.booting.time: c739a70d-2d1e-45c1-8c1b-
4d28ff2403ac |
|                       | cpu.delta: 700ceb7c-4cff-4d92-be2f-6526321548d6                          |
|                       | cpu: 716d6128-1ea6-430d-aa9c-ceaff2a6bf32                                |
|                       | cpu_l3_cache: 3410955e-c724-48a5-ab77-c3050b8cbe6e                       |
|                       | cpu_util: b148c392-37d6-4c8f-8609-e15fc15a4728                           |
|                       | disk.allocation: 9dd464a3-acf8-40fe-bd7e-3cb5fb12d7cc                    |
|                       | disk.capacity: c183d0da-e5eb-4223-a42e-855675dd1ec6                      |
|                       | disk.ephemeral.size: 15d1d828-fbb4-4448-b0f2-2392dcfed5b6                |
|                       | disk.iops: b8009e70-daee-403f-94ed-73853359a087                          |
|                       | disk.latency: 1c648176-18a6-4198-ac7f-33ee628b82a9                       |
```

```
|                     | disk.read.bytes.rate: eb35828f-312f-41ce-b0bc-cb6505e14ab7         |
|                     | disk.read.bytes: de463be7-769b-433d-9f22-f3265e146ec8              |
|                     | disk.read.requests.rate: 588ca440-bd73-4fa9-a00c-8af67262f4fd      |
|                     | disk.read.requests: 53e5d599-6cad-47de-b814-5cb23e8aaf24           |
|                     | disk.root.size: cee9d8b1-181e-4974-9427-aa7adb3b96d9               |
|                     | disk.usage: 4d724c99-7947-4c6d-9816-abbbc166f6f3                   |
|                     | disk.write.bytes.rate: 45b8da6e-0c89-4a6c-9cce-c95d49d9cc8b        |
|                     | disk.write.bytes: c7734f1b-b43a-48ee-8fe4-8a31b641b565             |
|                     | disk.write.requests.rate: 96ba2f22-8dd6-4b89-b313-1e0882c4d0d6     |
|                     | disk.write.requests: 553b7254-be2d-481b-9d31-b04c93dbb168          |
|                     | memory.bandwidth.local: 187f29d4-7c70-4ae2-86d1-191d11490aad       |
|                     | memory.bandwidth.total: eb09a4fc-c202-4bc3-8c94-aa2076df7e39       |
|                     | memory.resident: 97cfb849-2316-45a6-9545-21b1d48b0052              |
|                     | memory.swap.in: f0378d8f-6927-4b76-8d34-a5931799a301               |
|                     | memory.swap.out: c5fba193-1a1b-44c8-82e3-9fdc9ef21f69              |
|                     | memory.usage: 7958d06d-7894-4ca1-8c7e-72ba572c1260                 |
|                     | memory: a35c7eab-f714-4582-aa6f-48c92d4b79cd                       |
|                     | perf.cache.misses: da69636d-d210-4b7b-bea5-18d4959e95c1            |
|                     | perf.cache.references: e1955a37-d7e4-4b12-8a2a-51de4ec59efd        |
|                     | perf.cpu.cycles: 5d325d44-b297-407a-b7db-cc9105549193              |
|                     | perf.instructions: 973d6c6b-bbeb-4a13-96c2-390a63596bfc            |
|                     | vcpus: 646b53d0-0168-4851-b297-05d96cc03ab2                        |
| original_resource_id  | d71cdf9a-51dc-4bba-8170-9cd95edd3f66                             |
| project_id          | 3cee262b907b4040b26b678d7180566b                                   |
| revision_end        | None                                                               |
| revision_start      | 2017-11-16T04:00:27.081865+00:00                                   |
| server_group        | None                                                               |
| started_at          | 2017-11-16T01:09:20.668344+00:00                                   |
| type                | instance                                                           |
| user_id             | 1dbf5787b2ee46cf9fa6a1dfea9c9996                                   |
+---------------------+--------------------------------------------------------------------+
```

결과적으로 **metrics** 값에는 **aodh** 알람을 사용하여 모니터링할 수 있는 구성 요소가 나열됩니다(예: **cpu_util**).

2. **CPU** 사용량을 모니터링하려면 **cpu_util** 지표를 사용합니다.

```
$ openstack metric show --resource-id d71cdf9a-51dc-4bba-8170-9cd95edd3f66
cpu_util
+-----------------------------------+--------------------------------------------------------------------------+
| Field                             | Value                                                                    |
+-----------------------------------+--------------------------------------------------------------------------+
| archive_policy/aggregation_methods | std, count, min, max, sum, mean
|
| archive_policy/back_window        | 0                                                                        |
| archive_policy/definition         | - points: 8640, granularity: 0:05:00, timespan: 30
days, 0:00:00  |
| archive_policy/name               | low                                                                      |
| created_by_project_id             | 44adccdc32614688ae765ed4e484f389
|
| created_by_user_id                | c24fa60e46d14f8d847fca90531b43db
|
```

```
| creator                         |
c24fa60e46d14f8d847fca90531b43db:44adccdc32614688ae765ed4e484f389 |
| id                              | b148c392-37d6-4c8f-8609-e15fc15a4728                          |
| name                            | cpu_util                                                      |
| resource/created_by_project_id  | 44adccdc32614688ae765ed4e484f389
|
| resource/created_by_user_id     | c24fa60e46d14f8d847fca90531b43db
|
| resource/creator                |
c24fa60e46d14f8d847fca90531b43db:44adccdc32614688ae765ed4e484f389 |
| resource/ended_at               | None                                                          |
| resource/id                     | d71cdf9a-51dc-4bba-8170-9cd95edd3f66                          |
| resource/original_resource_id   | d71cdf9a-51dc-4bba-8170-9cd95edd3f66
|
| resource/project_id             | 3cee262b907b4040b26b678d7180566b
|
| resource/revision_end           | None                                                          |
| resource/revision_start         | 2017-11-17T00:05:27.516421+00:00
|
| resource/started_at             | 2017-11-16T01:09:20.668344+00:00                              |
| resource/type                   | instance                                                      |
| resource/user_id                | 1dbf5787b2ee46cf9fa6a1dfea9c9996                              |
| unit                            | None                                                          |
+---------------------------------+---------------------------------------------------------------+
```

- **archive_policy: std, count, min, max, sum, mean** 값을 계산하는 집계 간격을 정의합니다.

3. **aodh**를 사용하여 **cpu_util** 을 쿼리하는 모니터링 작업을 생성합니다. 이 작업은 지정한 설정에 따라 이벤트를 트리거합니다. 예를 들어, 인스턴스의 **CPU**가 확장된 기간 동안 **80%** 이상을 급증할 때 로그 항목을 생성하려면 다음 명령을 사용합니다.

```
$ openstack alarm create \
  --project-id 3cee262b907b4040b26b678d7180566b \
  --name high-cpu \
  --type gnocchi_resources_threshold \
  --description 'High CPU usage' \
  --metric cpu_util \
  --threshold 80.0 \
  --comparison-operator ge \
  --aggregation-method mean \
  --granularity 300 \
  --evaluation-periods 1 \
  --alarm-action 'log://' \
  --ok-action 'log://' \
  --resource-type instance \
  --resource-id d71cdf9a-51dc-4bba-8170-9cd95edd3f66
+---------------------------+-------------------------------------+
| Field                     | Value                               |
+---------------------------+-------------------------------------+
| aggregation_method        | mean                                |
| alarm_actions             | [u'log://']                         |
```

```
| alarm_id              | 1625015c-49b8-4e3f-9427-3c312a8615dd |
| comparison_operator   | ge                                   |
| description           | High CPU usage                       |
| enabled               | True                                 |
| evaluation_periods    | 1                                    |
| granularity           | 300                                  |
| insufficient_data_actions | []                                   |
| metric                | cpu_util                             |
| name                  | high-cpu                             |
| ok_actions            | [u'log://']                          |
| project_id            | 3cee262b907b4040b26b678d7180566b     |
| repeat_actions        | False                                |
| resource_id           | d71cdf9a-51dc-4bba-8170-9cd95edd3f66 |
| resource_type         | instance                             |
| severity              | low                                  |
| state                 | insufficient data                    |
| state_reason          | Not evaluated yet                    |
| state_timestamp       | 2017-11-16T05:20:48.891365           |
| threshold             | 80.0                                 |
| time_constraints      | []                                   |
| timestamp             | 2017-11-16T05:20:48.891365           |
| type                  | gnocchi_resources_threshold          |
| user_id               | 1dbf5787b2ee46cf9fa6a1dfea9c9996     |
+-----------------------+--------------------------------------+
```

- **comparison-operator: ge** Operator는 **CPU** 사용량이 **80%**보다 크거나 같은 경우 알람이 트리거되도록 정의합니다.

- 세분성: 메트릭에 보관 정책이 연결되어 있습니다. 정책에는 다양한 세분화된 기능이 있을 수 있습니다. 예를 들어 한 달에 **1**시간 **+ 1**시간 집계를 위한 **5**분 집계입니다. 세분성 값은 보관 정책에 설명된 기간과 일치해야 합니다.

- **evaluation-periods:** 알람이 트리거되기 전에 경과해야 하는 세분 기간입니다. 예를 들어 이 값을 **2**로 설정하면 알람이 트리거되기 전에 두 폴링 기간 동안 **CPU** 사용량이 **80%** 이상이어야 합니다.

- **[U'log://']: alarm_actions** 또는 **ok_actions** 를 **[u'log://']]**로 설정하면 알람이 트리거되거나 정상 상태로 되돌아가며 **aodh** 로그 파일에 기록됩니다.

참고

알람이 트리거될 때 실행할 다양한 작업을 정의하고(**alerts_actions**) 일반 상태(**ok_actions**)로 돌아갈 때 웹 후크 **URL**과 같은 다양한 작업을 정의할 수 있습니다.

5.7. 알람 기록 보기

알람이 트리거되었는지 확인하려면 알람 기록을 쿼리할 수 있습니다.

절차

- openstack alarm-history show 명령을 사용합니다.

```
openstack alarm-history show 1625015c-49b8-4e3f-9427-3c312a8615dd --fit-width
+---------------------------+-----------------+-------------------------------------------------------------------
-------------------------------------------------------------------------------+----------------------
-----------+
| timestamp                 | type            | detail
| event_id                            |
+---------------------------+-----------------+-------------------------------------------------------------------
-------------------------------------------------------------------------------+----------------------
-----------+
| 2017-11-16T05:21:47.850094 | state transition | {"transition_reason": "Transition to ok
due to 1 samples inside threshold, most recent: 0.0366665763", "state": "ok"}
| 3b51f09d-ded1-4807-b6bb-65fdc87669e4 |
+---------------------------+-----------------+-------------------------------------------------------------------
-------------------------------------------------------------------------------+----------------------
-----------+
```

6장. 로그

RHOSP(Red Hat OpenStack Platform)는 특정 로그 파일에 정보 메시지를 씁니다. 이러한 메시지를 사용하여 시스템 이벤트 문제 해결 및 모니터링이 가능합니다.

참고

개별 로그 파일을 지원 케이스에 수동으로 연결할 필요는 없습니다. **sosreport** 유틸리티는 필요한 로그를 자동으로 수집합니다.

6.1. OPENSTACK 서비스의 로그 파일 위치

각 **OpenStack** 구성 요소에는 실행 중인 서비스 고유의 파일이 포함된 별도의 로깅 디렉터리가 있습니다.

6.1.1. Bare Metal Provisioning(ironic) 로그 파일

서비스	서비스 이름	로그 경로
OpenStack Ironic API	openstack-ironic-api.service	/var/log/containers/ironic/ironic-api.log
OpenStack Ironic Conductor	openstack-ironic-conductor.service	/var/log/containers/ironic/ironic-conductor.log

6.1.2. Block Storage(cinder) 로그 파일

서비스	서비스 이름	로그 경로
Block Storage API	openstack-cinder-api.service	/var/log/containers/cinder-api.log
블록 스토리지 백업	openstack-cinder-backup.service	/var/log/containers/cinder/backup.log
정보 메시지	cinder-manage 명령	/var/log/containers/cinder/cinder-manage.log
블록 스토리지 스케줄러	openstack-cinder-scheduler.service	/var/log/containers/cinder/scheduler.log

서비스	서비스 이름	로그 경로
블록 스토리지 볼륨	openstack-cinder-volume.service	/var/log/containers/cinder/volume.log

6.1.3. Compute(nova) 로그 파일

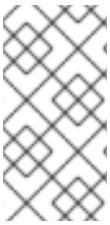
서비스	서비스 이름	로그 경로
OpenStack Compute API 서비스	openstack-nova-api.service	/var/log/containers/nova/nova-api.log
OpenStack Compute 인증서 서버	openstack-nova-cert.service	/var/log/containers/nova/nova-cert.log
OpenStack Compute 서비스	openstack-nova-compute.service	/var/log/containers/nova/nova-compute.log
OpenStack Compute Conductor 서비스	openstack-nova-conductor.service	/var/log/containers/nova/nova-conductor.log
OpenStack Compute VNC 콘솔 인증 서버	openstack-nova-consoleauth.service	/var/log/containers/nova/nova-consoleauth.log
정보 메시지	Nova-manage 명령	/var/log/containers/nova/nova-manage.log
OpenStack Compute NoVNC Proxy 서비스	openstack-nova-novncproxy.service	/var/log/containers/nova/nova-novncproxy.log
OpenStack Compute Scheduler 서비스	openstack-nova-scheduler.service	/var/log/containers/nova/nova-scheduler.log

6.1.4. 대시보드(horizon) 로그 파일

서비스	서비스 이름	로그 경로
특정 사용자 상호 작용의 로그	대시보드 인터페이스	/var/log/containers/horizon/horizon.log

Apache HTTP 서버는 웹 브라우저 또는 명령줄 클라이언트(예: keystone 및 nova)를 사용하여 액세스할 수 있는 대시보드 웹 인터페이스에 대해 몇 가지 추가 로그 파일을 사용합니다. 다음 로그 파일은 대시보드 사용을 추적하고 오류를 진단하는 데 유용할 수 있습니다.

목적	로그 경로
처리된 모든 HTTP 요청	/var/log/containers/httpd/horizon_access.log
HTTP 오류	/var/log/containers/httpd/horizon_error.log
admin-role API 요청	/var/log/containers/httpd/keystone_wsgi_admin_access.log
admin-role API 오류	/var/log/containers/httpd/keystone_wsgi_admin_error.log
member-role API 요청	/var/log/containers/httpd/keystone_wsgi_main_access.log
member-role API 오류	/var/log/containers/httpd/keystone_wsgi_main_error.log



참고

동일한 호스트에서 실행 중인 다른 웹 서비스에서 보고한 오류를 저장하는 /var/log/containers/httpd/default_error.log 도 있습니다.

6.1.5. 데이터 처리(sahara) 로그 파일

서비스	서비스 이름	로그 경로
Sahara API Server	openstack-sahara-all.service openstack-sahara-api.service	/var/log/containers/sahara/sahara-all.log /var/log/containers/messages
Sahara Engine Server	openstack-sahara-engine.service	/var/log/containers/messages

6.1.6. Database as a Service (trove) 로그 파일

서비스	서비스 이름	로그 경로
OpenStack Trove API Service	openstack-trove-api.service	/var/log/containers/trove/trove-api.log
OpenStack Trove Conductor Service	openstack-trove-conductor.service	/var/log/containers/trove/trove-conductor.log

서비스	서비스 이름	로그 경로
OpenStack Trove guestagent Service	openstack-trove-guestagent.service	/var/log/containers/trove/logfile.txt
OpenStack Trove taskmanager Service	openstack-trove-taskmanager.service	/var/log/containers/trove/trove-taskmanager.log

6.1.7. Identity 서비스(keystone) 로그 파일

서비스	서비스 이름	로그 경로
OpenStack ID 서비스	openstack-keystone.service	/var/log/containers/keystone/keystone.log

6.1.8. Image 서비스(glance) 로그 파일

서비스	서비스 이름	로그 경로
OpenStack Image 서비스 API 서버	openstack-glance-api.service	/var/log/containers/glance/api.log
OpenStack Image 서비스 레지스트리 서버	openstack-glance-registry.service	/var/log/containers/glance/registry.log

6.1.9. Networking(neutron) 로그 파일

서비스	서비스 이름	로그 경로
OpenStack Neutron DHCP 에이전트	neutron-dhcp-agent.service	/var/log/containers/neutron/dhcp-agent.log
OpenStack Networking 계층 3 에이전트	neutron-l3-agent.service	/var/log/containers/neutron/l3-agent.log
메타데이터 에이전트 서비스	neutron-metadata-agent.service	/var/log/containers/neutron/metadata-agent.log
메타데이터 네임스페이스 프록시	해당 없음	/var/log/containers/neutron/neutron-ns-metadata-proxy-*UUID*.log
Open vSwitch 에이전트	neutron-openvswitch-agent.service	/var/log/containers/neutron/openvswitch-agent.log

서비스	서비스 이름	로그 경로
OpenStack Networking 서비스	neutron-server.service	/var/log/containers/neutron/server.log

6.1.10. Object Storage(swift) 로그 파일

OpenStack Object Storage는 시스템 로깅 기능에만 로그를 보냅니다.

참고

기본적으로 모든 **Object Storage** 로그 파일은 **local0, local1** 및 **local2 syslog** 기능을 사용하여 **/var/log/containers/swift/swift.log** 로 이동합니다.

Object Storage의 로그 메시지는 **REST API** 서비스 및 백그라운드 데몬의 두 가지 광범위한 범주로 분류됩니다. **API** 서비스 메시지에는 널리 사용되는 **HTTP** 서버와 유사한 방식으로 **API** 요청당 하나의 행이 포함되어 있습니다. **frontend(Proxy)** 및 **backend(Account, Container, Object)** 서비스에서 이러한 메시지를 게시해야 합니다. 데몬 메시지는 덜 구조화되지 않으며 일반적으로 주기적인 작업을 수행하는 데몬에 대한 사람이 읽을 수 있는 정보를 포함합니다. 그러나 **Object Storage**의 어느 부분이 메시지를 생성하든 소스 **ID**는 항상 행의 시작 부분에 있습니다.

다음은 프록시 메시지의 예입니다.

```
Apr 20 15:20:34 rhev-a24c-01 proxy-server: 127.0.0.1 127.0.0.1 20/Apr/2015/19/20/34 GET
/v1/AUTH_zaitcev%3Fformat%3Djson%26marker%3Dtestcont HTTP/1.0 200 - python-swiftclient-
2.1.0 AUTH_tk737d6... - 2 - txc454fa8ea4844d909820a-0055355182 - 0.0162 - -
1429557634.806570053 1429557634.822791100
```

다음은 백그라운드 데몬의 **ad-hoc** 메시지의 예입니다.

```
Apr 27 17:08:15 rhev-a24c-02 object-auditor: Object audit (ZBF). Since Mon Apr 27 21:08:15 2015:
Locally: 1 passed, 0 quarantined, 0 errors files/sec: 4.34 , bytes/sec: 0.00, Total time: 0.23, Auditing
time: 0.00, Rate: 0.00
Apr 27 17:08:16 rhev-a24c-02 object-auditor: Object audit (ZBF) "forever" mode completed: 0.56s.
Total quarantined: 0, Total errors: 0, Total files/sec: 14.31, Total bytes/sec: 0.00, Auditing time: 0.02,
Rate: 0.04
Apr 27 17:08:16 rhev-a24c-02 account-replicator: Beginning replication run
Apr 27 17:08:16 rhev-a24c-02 account-replicator: Replication run OVER
Apr 27 17:08:16 rhev-a24c-02 account-replicator: Attempted to replicate 5 dbs in 0.12589 seconds
```

```
(39.71876/s)
Apr 27 17:08:16 rhev-a24c-02 account-replicator: Removed 0 dbs
Apr 27 17:08:16 rhev-a24c-02 account-replicator: 10 successes, 0 failures
```

6.1.11. 오케스트레이션(heat) 로그 파일

서비스	서비스 이름	로그 경로
OpenStack Heat API 서비스	openstack-heat-api.service	/var/log/containers/heat/heat-api.log
OpenStack Heat Engine Service	openstack-heat-engine.service	/var/log/containers/heat/heat-engine.log
오케스트레이션 서비스 이벤트	해당 없음	/var/log/containers/heat/heat-manage.log

6.1.12. Shared Filesystem Service(manila) 로그 파일

서비스	서비스 이름	로그 경로
OpenStack Manila API Server	openstack-manila-api.service	/var/log/containers/manila/api.log
OpenStack Manila Scheduler	openstack-manila-scheduler.service	/var/log/containers/manila/scheduler.log
OpenStack Manila Share Service	openstack-manila-share.service	/var/log/containers/manila/share.log

참고

Manila Python 라이브러리의 일부 정보는 /var/log/containers/manila/manila-manage.log 에 로그인할 수도 있습니다.

6.1.13. Telemetry(ceilometer) 로그 파일

서비스	서비스 이름	로그 경로
OpenStack ceilometer 알림 에이전트	openstack-ceilometer-notification.service	/var/log/containers/ceilometer/agent-notification.log
OpenStack ceilometer 알람 평가	openstack-ceilometer-alarm-evaluator.service	/var/log/containers/ceilometer/alarm-evaluator.log

서비스	서비스 이름	로그 경로
OpenStack ceilometer 알람 알림	openstack-ceilometer-alarm-notifier.service	/var/log/containers/ceilometer/alarm-notifier.log
OpenStack ceilometer API	httpd.service	/var/log/containers/ceilometer/api.log
정보 메시지	MongoDB 통합	/var/log/containers/ceilometer/ceilometer-dbsync.log
OpenStack ceilometer 중앙 에이전트	openstack-ceilometer-central.service	/var/log/containers/ceilometer/central.log
OpenStack ceilometer 컬렉션	openstack-ceilometer-collector.service	/var/log/containers/ceilometer/collector.log
OpenStack ceilometer 컴퓨팅 에이전트	openstack-ceilometer-compute.service	/var/log/containers/ceilometer/compute.log

6.1.14. 지원 서비스를 위한 로그 파일

다음 서비스는 핵심 OpenStack 구성 요소에서 사용하며 자체 로그 디렉터리 및 파일이 있습니다.

서비스	서비스 이름	로그 경로
메시지 브로커 (RabbitMQ)	rabbitmq-server.service	/var/log/rabbitmq/rabbit@*short_hostname*.log /var/log/rabbitmq/rabbit@*short_hostname*-sasl.log (Simple Authentication and Security Layer 관련 로그 메시지의 경우)
데이터베이스 서버 (MariaDB)	mariadb.service	/var/log/mariadb/mariadb.log
문서 지향 데이터베이스 (MongoDB)	mongod.service	/var/log/mongodb/mongodb.log
가상 네트워크 스위치 (Open vSwitch)	openvswitch-nonetwork.service	/var/log/openvswitch/ovsdb-server.log /var/log/openvswitch/ovs-vswitchd.log

6.2. 중앙 집중식 로그 시스템 아키텍처 및 구성 요소

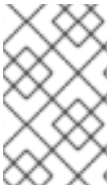
모니터링 툴은 RHOSP(Red Hat OpenStack Platform) 오버클라우드 노드에 배포된 클라이언트와 함께 클라이언트-서버 모델을 사용합니다. Fluentd 서비스는 클라이언트 쪽의 중앙 집중식 로깅(CL)을 제공합니다. 모든 RHOSP 서비스는 로그 파일을 생성하고 업데이트합니다. 이러한 로그 파일은 작업, 오류, 경

고 및 기타 이벤트를 기록합니다. OpenStack과 같은 분산 환경에서는 중앙 위치에서 이러한 로그를 수집하면 디버깅 및 관리가 간소화됩니다. 중앙 집중식 로깅을 사용하면 전체 OpenStack 환경에서 로그를 볼 수 있는 중앙 집중식 위치를 사용할 수 있습니다. 이러한 로그는 syslog 및 감사 로그 파일, RabbitMQ 및 MariaDB와 같은 인프라 구성 요소, ID, 계산 등의 OpenStack 서비스와 같은 운영 체제에서 가져옵니다. 중앙 집중식 로깅 툴체인은 다음 구성 요소로 구성됩니다.

- 로그 수집 에이전트(Fluentd)
- Log Relay/Transformer (Fluentd)
- 데이터 저장소(ElasticSearch)
- API/Presentation Layer (Kibana)

참고

Red Hat OpenStack Platform director는 중앙 집중식 로깅을 위해 서버 쪽 구성 요소를 배포하지 않습니다. Red Hat은 로그 집계기로 실행되는 플러그인이 포함된 ElasticSearch 데이터베이스, Kibana 및 Fluentd를 포함하여 서버 측 구성 요소를 지원하지 않습니다. 중앙 집중식 로깅 구성 요소 및 상호 작용은 다음 다이어그램에 설명되어 있습니다.



참고

파란색으로 표시된 항목은 Red Hat에서 지원하는 구성 요소를 나타냅니다.

그림 6.1. Red Hat OpenStack Platform용 단일 HA 배포

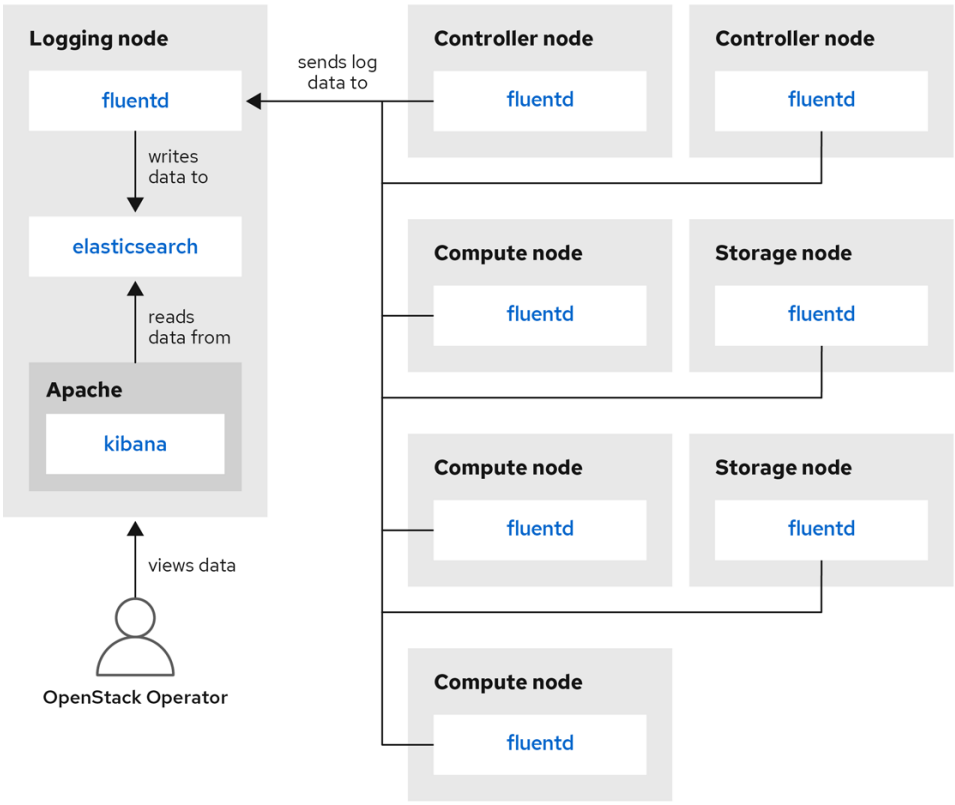
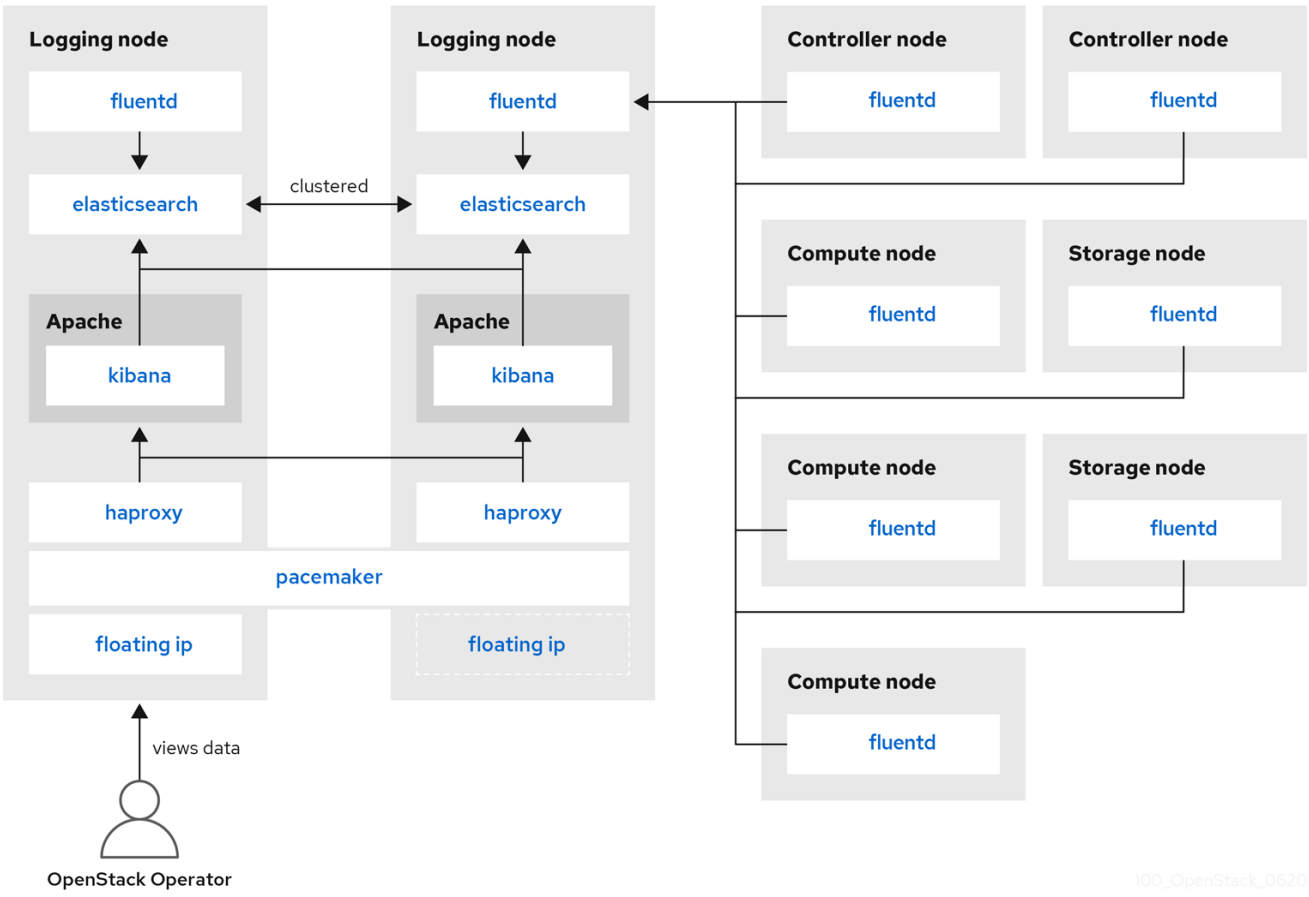


그림 6.2. Red Hat OpenStack Platform용 HA 배포



6.3. 로그 서비스 설치 개요

로그 수집 에이전트 **Fluentd**는 클라이언트 측에서 로그를 수집하고 이러한 로그를 서버 측에서 실행되는 **Fluentd** 인스턴스로 보냅니다. 이 **Fluentd** 인스턴스는 저장을 위해 로그 레코드를 **Elasticsearch**로 리디렉션합니다.

6.4. 모든 머신에 FLUENTD 배포

Fluentd는 로그 수집 에이전트이며 중앙 집중식 로깅 툴체인의 일부입니다. 모든 머신에 **Fluentd**를 배포하려면 **logging-environment.yaml** 파일에서 **LoggingServers** 매개변수를 수정해야 합니다.

사전 요구 사항

- **Elasticsearch** 및 **Fluentd** 릴레이가 서버 측에 설치되어 있는지 확인합니다. 자세한 내용은 클라이언트 측 통합이 호환되는 **opstools-ansible** 프로젝트 의 배포 예를 참조하십시오.

절차

1. **tripleo-heat-templates/environments/logging-environment.yaml** 파일을 홈 디렉터리에 복사합니다.

2. 복사한 파일에서 환경에 맞게 **LoggingServers** 매개변수에 항목을 생성합니다. 다음 스니펫은 **LoggingServers** 매개변수 구성의 예입니다.

```
parameter_defaults:

Simple configuration

LoggingServers:
 - host: log0.example.com
   port: 24224
 - host: log1.example.com
   port: 24224
```

3. 사용자 환경과 관련된 기타 환경 파일과 함께 수정된 환경 파일을 **openstack overcloud deploy** 명령에 포함하고 배포합니다. 기존 배포의 일부인 환경 파일 목록으로 바꿉니다 **<existing_overcloud_environment_files>.**

```
$ openstack overcloud deploy \
<existing_overcloud_environment_files> \
-e /home/templates/environments/logging-environment.yaml \
...
```

s . additional resources

- **자세한 내용은 6.5절. “구성 가능한 로깅 매개변수”의 내용을 참조하십시오.**

6.5. 구성 가능한 로깅 매개변수

이 표에는 구성할 수 있는 로깅 매개변수에 대한 설명이 포함되어 있습니다. tripleo-heat-templates/puppet/services/logging/fluentd-config.yaml 파일에서 이러한 매개변수를 찾을 수 있습니다.

매개변수	설명
LoggingDefaultFormat	로그 파일에서 메시지를 구문 분석하는 데 사용되는 기본 형식입니다.
LoggingPosFilePath	Fluentd **pos_file** 파일을 배치할 디렉터리입니다.**tail** 입력 유형의 파일 위치를 추적하는 데 사용됩니다.
LoggingDefaultGroups	Fluentd 사용자를 이러한 그룹에 추가합니다. 기본 그룹 목록을 수정하려면 이 매개변수를 재정의합니다. **LoggingExtraGroups** 매개변수를 사용하여 Fluentd 사용자를 추가 그룹에 추가합니다.
LoggingExtraGroups	**LoggingDefaultGroups** 매개변수 외에도 개별 구성 가능 서비스에서 제공하는 그룹 외에도 Fluentd 사용자를 이러한 그룹에 추가합니다.
LoggingDefaultFilters	Fluentd 기본 필터 목록입니다. 이 목록은 **fluentd::config** 리소스의 **필터** 키에 동사atim으로 전달됩니다. 기본 필터 집합이 필요하지 않은 경우 이 값을 재정의합니다. 서버를 추가하려면 **LoggingExtraFilters** 매개변수를 사용합니다.
LoggingExtraFilters	추가 Fluentd 필터 목록입니다. 이 목록은 **fluentd::config** 리소스의 **필터** 키에 동사atim으로 전달됩니다.
LoggingUsesSSL	**secure_forward** 플러그인을 사용하여 로그 메시지를 전달할지 여부를 나타내는 부울 값입니다.
LoggingSSLKey	Fluentd CA 인증서에 대한 PEM 인코딩 키입니다. **in_secure_forward** 매개변수는 **LoggingSSLKey** 매개변수의 값을 사용합니다.
LoggingSSLCertificate	Fluentd용 PEM으로 인코딩된 SSL CA 인증서입니다.
LoggingSSLKeyPassphrase	**LoggingSSLKey** 매개변수의 암호입니다.**in_secure_forward** 매개변수는 **LoggingSSLKeyPassphrase** 매개변수의 값을 사용합니다.
LoggingSharedKey	Fluentd **secure-forward** 플러그인의 공유 시크릿.

매개변수	설명
LoggingDefaultSources	Fluentd의 기본 로깅 소스 목록입니다. 기본 로깅 소스를 비활성화하려면 이 매개변수를 재정의합니다. **LoggingExtraSources** 매개변수를 사용하여 추가 소스 구성을 정의합니다.
LoggingExtraSources	이 목록은 **LoggingDefaultSources** 매개변수 및 구성 가능 서비스에서 정의한 로깅 소스와 결합합니다.

6.6. 로그 파일의 기본 경로 덮어쓰기

기본 컨테이너를 수정하고 수정에 서비스 로그 파일의 경로가 포함된 경우 기본 로그 파일 경로도 수정해야 합니다. 모든 구성 가능 서비스에는 < service_name>LoggingSource 매개변수가 있습니다. 예를 들어 nova-compute 서비스의 경우 매개 변수는 NovaComputeLoggingSource 입니다.

절차

1. **nova-compute 서비스의 기본 경로를 재정의하려면 구성 파일의 NovaComputeLoggingSource 매개변수에 경로를 추가합니다.**

```
NovaComputeLoggingSource:
    tag: openstack.nova.compute
    path: /some/other/path/nova-compute.log
```

태그 및 경로 속성은 < service_name>LoggingSource 매개변수의 필수 요소입니다. 각 서비스에서 태그와 경로가 정의되고 나머지 값은 기본적으로 파생됩니다.

2. **특정 서비스의 형식을 수정할 수 있습니다. 이렇게 하면 Fluentd 구성에 직접 전달됩니다. LoggingDefaultFormat 매개변수의 기본 형식은 /(? <time>\d\d{4}-\d{2}\d{2}:\d{2}:\d{2}.\d+)(? <pid>\d+)(<priority>\d+) 구문(<priority>\d{2}\d+)입니다.**

```
<service_name>LoggingSource:
  tag: <service_name>.tag
  path: <service_name>.path
  format: <service_name>.format
```

다음 코드 조각은 더 복잡한 변환의 예입니다.

```
ServiceLoggingSource:
  tag: openstack.Service
  path: /var/log/containers/service/service.log
  format: multiline
```

```
    format_firstline: '/^\d{4}-\d{2}-\d{2} \d{2}:\d{2}:\d{2}.\d{3} \d+ \S+ \S+ \[(req-\S+ \S+ \S+ \S+
\S+ \S+|-)\]/'
    format1: '/^(?<Timestamp>\S+ \S+) (?<Pid>\d+) (?<log_level>\S+) (?
<python_module>\S+) (\[(req-(?<request_id>\S+) (?<user_id>\S+) (?<tenant_id>\S+) (?
<domain_id>\S+) (?<user_domain>\S+) (?<project_domain>\S+)|-)\])? (?<Payload>.*)?$/'
```

6.7. 배포 성공 확인

중앙 집중식 로깅이 성공적으로 배포되었는지 확인하려면 로그를 보고 출력이 기대치와 일치하는지 확인합니다. 타사 시각화 소프트웨어(예: Kibana)를 사용할 수 있습니다.