Red Hat OpenStack Platform 17.1

オーバークラウドの可観測性の管理

物理リソースおよび仮想リソースのトラッキングおよびメトリックの収集

Last Updated: 2024-05-23

Red Hat OpenStack Platform 17.1 オーバークラウドの可観測性の管理

物理リソースおよび仮想リソースのトラッキングおよびメトリックの収集

OpenStack Team
rhos-docs@redhat.com

法律上の通知

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

運用ツールを使用して、Red Hat OpenStack Platform 環境の計測と維持に役立てます。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに対するご意見をお聞かせください。ドキュメントの改善点があればお知らせください。

Jira でドキュメントのフィードバックを提供する

ドキュメントに関するフィードバックを提供するには、Create Issue フォームを使用します。Red Hat OpenStack Platform Jira プロジェクトで Jira Issue が作成され、フィードバックの進行状況を追跡できます。

1. Jira にログインしていることを確認してください。Jira アカウントをお持ちでない場合は、アカウントを作成してフィードバックを送信してください。
2. Create Issue をクリックして、**Create Issue** ページを開きます。
3. **Summary** フィールドと **Description** フィールドに入力します。**Description** フィールドに、ドキュメントの URL、章またはセクション番号、および問題の詳しい説明を入力します。フォーム内の他のフィールドは変更しないでください。
4. **Create** をクリックします。

第1章 運用データ計測の概要

ceilometer、collectd、ロギングサービスなどの可観測性コンポーネントを使用して、Red Hat OpenStack Platform (RHOSP) 環境からデータを収集できます。自動スケーリングのユースケースのために収集したデータを Gnocchi に保存することも、**metrics_qdr** を使用してデータを Service Telemetry Framework (STF) に転送することもできます。

自動スケーリングの詳細は、インスタンスの自動スケーリング を参照してください。

STF の詳細は、Service Telemetry Framework 1.5 を参照してください。

1.1. 可観測性アーキテクチャー

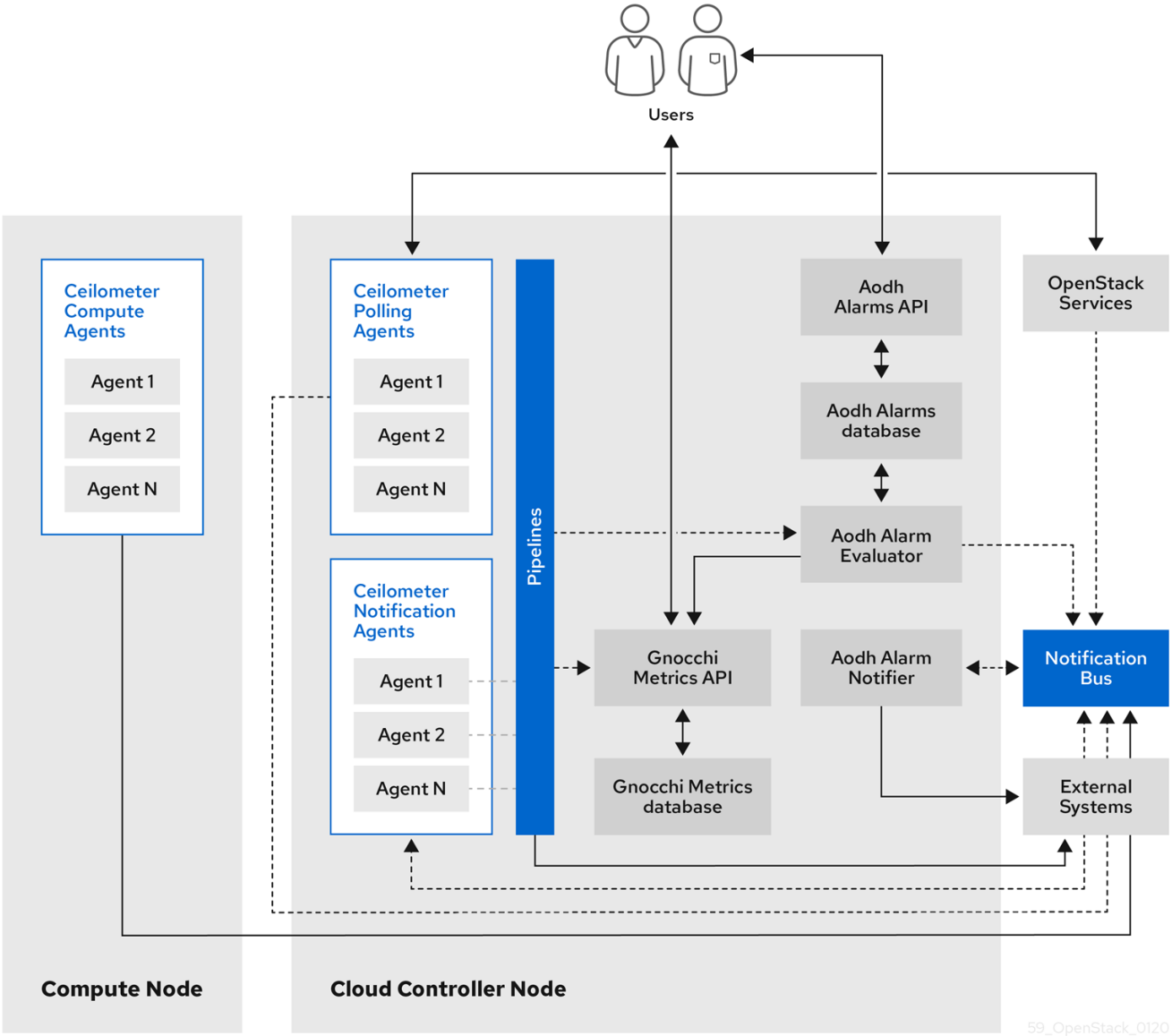
Red Hat OpenStack Platform (RHOSP) Observability は、OpenStack ベースのクラウドのユーザーレベルの使用状況データを提供します。可観測性コンポーネントを設定し、既存の OpenStack コンポーネントにより送信される通知から (例: Compute の使用状況イベント)、または RHOSP インフラストラクチャーリソースへのポーリングにより (例: libvirt)、データを収集することができます。Ceilometer は、収集したデータを、Service Telemetry Framework (STF) を含むデータストアやメッセージキューなどのさまざまなターゲットに公開します。

可観測性は、以下のコンポーネントで設定されます。

- **データ収集**: Observability は Ceilometer を使用してメトリックおよびイベントデータを収集します。詳細は、「Ceilometer」 を参照してください。
- **ストレージ**: 可観測性は、メトリックデータを Gnocchi に保存します。詳細は、「Gnocchi を使用したストレージ」 を参照してください。
- **Alarm service**: 可観測性は、Alarming サービス Aodh を使用してアクションをトリガーします。アクションのトリガーは、Ceilometer の収集するメトリックデータまたはイベントデータに対して定義されたルールに基づきます。

データを収集したら、サードパーティーのツールを使用してメトリックデータを表示および解析し、Alarming サービスを使用してイベントのアラームを設定できます。

図1.1 可観測性アーキテクチャー



1.1.1. 監視コンポーネントのサポート状況

以下の表に、Red Hat OpenStack Platform (RHOSP) の監視用コンポーネントに対するサポート状況を示します。

表1.1 サポート状況

コンポーネント	以降でフルサポート	非推奨化	以降で削除	注記
Aodh	RHOSP 9	RHOSP 15		自動スケーリングのユースケースでサポートされています。

コンポーネント	以降でフルサポート	非推奨化	以降で削除	注記
Ceilometer	RHOSP 4			自動スケーリングおよびサービステレメトリーフレームワーク (STF) のユースケースで RHOSP のメトリクスとイベントの収集がサポートされています。
Collectd	RHOSP 11	RHOSP 17.1		STF のインフラストラクチャーメトリクスの収集がサポートされています。
Gnocchi	RHOSP 9	RHOSP 15		自動スケーリングのユースケースのメトリックのストレージがサポートされています。
Panko	RHOSP 11	RHOSP 12、RHOSP 14 以降、デフォルトではインストールされていません	RHOSP 17.0	
QDR	RHOSP 13	RHOSP 17.1		RHOSP から STF へのメトリクスおよびイベントデータの送信がサポートされています。

1.2. RED HAT OPENSTACK PLATFORM におけるデータ収集

Red Hat OpenStack Platform (RHOSP) は、2 種類のデータ収集をサポートします。

- RHOSP コンポーネントレベルのモニタリング用の Ceilometer。詳細は、「Ceilometer」 を参照してください。
- インフラストラクチャーモニタリング用の collectd。詳細は、「collectd」 を参照してください。

1.2.1. Ceilometer

Ceilometer は、Red Hat OpenStack Platform (RHOSP) のデフォルトのデータ収集コンポーネントであり、現在のすべての RHOSP コアコンポーネントにわたってデータを正規化および変換する機能を提供します。Ceilometer は、RHOSP サービスに関連する計測データとイベントデータを収集します。

Ceilometer サービスは、3 つのエージェントを使用して Red Hat OpenStack Platform (RHOSP) コンポーネントからデータを収集します。

- **コンピュートエージェント (ceilometer-agent-compute)**: 各コンピュートノードで実行され、リソースの使用状況の統計値をポーリングします。このエージェントは、パラメーター **--polling namespace-compute** を使用して実行しているポーリングエージェント **ceilometer-polling** と同じです。

- **中央エージェント (ceilometer-agent-central)**: 中央の管理サーバーで実行され、インスタンスまたは Compute ノードに関連付けられないリソースの使用状況の統計値をポーリングします。複数のエージェントを起動して、サービスをスケーリングすることができます。これは、パラメーター **--polling namespace-central** を使用して実行しているポーリングエージェント **ceilometer-polling** と同じです。

- **通知エージェント (ceilometer-agent-notification)**: 中央の管理サーバーで実行され、メッセージキューからのメッセージを処理してイベントデータおよび計測データをビルドします。定義されたターゲットにデータを公開します。デフォルトのターゲットは gnocchi です。これらのサービスは、RHOSP の通知バスを使用して通信します。

Ceilometer エージェントは、パブリッシャーを使用して、対応するエンドポイント (Gnocchi や AMQP バージョン 1 (QDR) など) にデータを送信します。

1.2.2. collectd

Collectd は、インフラストラクチャーメトリックを提供するために使用できる別のデータ収集エージェントです。設定されたソースからデータを繰り返し取得します。メトリックを Service Telemetry Framework (STF) に転送して、データを保存および視覚化できます。

1.3. GNOCCHI を使用したストレージ

Gnocchi はオープンソースの時系列データベースです。gnocchi を使用すると、メトリックとリソースを保存し、Operator とユーザーにそれらへのアクセスを提供できます。Gnocchi は、アーカイブポリシーを使用して処理する集約および保持する集約値の数を定義します。インデクサードライバーは、すべてのリソース、アーカイブポリシー、およびメトリックのインデックスを保管します。

Red Hat OpenStack Platform (RHOSP) での Gnocchi の使用は、自動スケーリングのユースケースでサポートされています。自動スケーリングの詳細は、インスタンスの自動スケーリング を参照してください。

1.3.1. アーカイブポリシー: 時系列データベースへの短期および長期両データの保管

アーカイブポリシーにより、処理する集約および保持する集約値の数を定義します。Gnocchi は、最小値、最大値、平均値、N 番目パーセンタイル、標準偏差などのさまざまな集約メソッドをサポートします。これらの集約は粒度と呼ばれる期間にわたって処理され、特定のタイムスパンの間保持されます。

アーカイブポリシーは、メトリックの集約方法および保管期間を定義します。それぞれのアーカイブポリシーは、タイムスパンにおけるポイント数として定義されます。

たとえば、アーカイブポリシーで 1 秒の粒度および 10 ポイントのポリシーを定義すると、時系列アーカイブは最大 10 秒間保持し、それぞれが 1 秒間の集約を表します。つまり、時系列は最大で、より新しいポイントと古いポイント間の 10 秒間のデータを保持します。

アーカイブポリシーは、使用する集約メソッドも定義します。デフォルトはパラメーター **default_aggregation_methods** で設定し、そのデフォルト値は mean、min、max、sum、std、count に設定されています。したがって、ユースケースによってアーカイブポリシーおよび粒度は異なりま

す。

関連情報

- アーカイブポリシーの詳細は、アーカイブポリシーのプランニングおよび管理を参照してください。

1.3.2. インデクサードライバー

インデクサーは、すべてのリソース、アーカイブポリシー、およびメトリックのインデックス、ならびにそれらの定義、種別、および属性を保管するロールを担います。また、リソースとメトリックをリンクさせる機能も果たします。Red Hat OpenStack Platform director は、デフォルトでインデクサードライバーをインストールします。Gnocchi が処理するすべてのリソースおよびメトリックをインデックス化するデータベースが必要です。サポートされるドライバーは MySQL です。

1.3.3. Gnocchi の用語

この表には、Gnocchi 機能で一般的に使用される用語の定義が含まれています。

表1.2 Gnocchi の用語

用語	定義
集約メソッド	複数の計測値から 1 つの集約値を生成するのに使用される関数。たとえば、min 集約メソッドであれば、さまざまな計測値を、特定期間内の全計測値の最小値に集約します。
集約値 (Aggregate)	アーカイブポリシーに従って複数の計測値から生成されたデータポイントタプル。集約値はタイムスタンプおよび値で設定されます。
アーカイブポリシー	メトリックに割り当てられた集約値の保管ポリシー。アーカイブポリシーにより、集約値がメトリックに保持される期間および集約値の生成方法 (集約メソッド) が決定されます。
粒度 (Granularity)	メトリックの集約時系列における 2 つの集約値の時間間隔
計測値 (Measure)	API によって時系列データベースに送信される受信データポイントタプル。計測値はタイムスタンプおよび値で設定されます。
メトリック	UUID で識別される集約値を保管するエンティティー。名前を使用して、メトリックをリソースに割り当てることができます。メトリックがその集約値をどのように保管するかは、メトリックが関連付けられたアーカイブポリシーで定義されます。
リソース	メトリックを関連付ける、インフラストラクチャー内の任意の項目を表すエンティティー。リソースは一意の ID で識別され、属性を含めることができます。
時系列 (Time series)	集約値を時刻順に並べたリスト
タイムスパン	メトリックがその集約値を保持する期間。アーカイブポリシーを定義する際に使用されます。

第2章 運用データ計測のプランニング

Ceilometer または collectd を使用して、自動スケーリングまたは Service Telemetry Framework (STF) 用の Telemetry データを収集できます。

2.1. COLLECTD による計測

以下は、デフォルトの collectd による計測です。

- cpu
- 空きディスク容量
- ディスク使用量
- hugepages
- interface
- load
- memory
- unixsock
- uptime

2.2. データストレージのプランニング

Gnocchi は、データポイントのコレクションを保管します。この場合、それぞれのデータポイントが集約値です。ストレージの形式は、異なる技術を使用して圧縮されます。したがって、時系列データベースのサイズを計算する場合、ワーストケースのシナリオに基づいてサイズを見積もる必要があります。

警告

時系列データベース (Gnocchi) ストレージ用の Red Hat OpenStack Platform (RHOSP) Object Storage (swift) の使用は、小規模な非実稼働環境でのみサポートされています。

手順

1. データポイントの数を計算します。
   ポイント数 = タイムスパン / 粒度

   たとえば、1 分間の解像度で 1 年分のデータを保持する場合は、以下の式を使用します。

   データポイント数 = (365 日 X 24 時間 X 60 分) / 1 分 = 525600

2. 時系列データベースのサイズを計算します。
   サイズ (バイト単位) = データポイント数 X 8 バイト

この式を例に当てはめると、結果は 4.1 MB になります。

サイズ (バイト単位) = 525600 ポイント X 8 バイト = 4204800 バイト = 4.1 MB

この値は、単一の集約時系列データベースの推定ストレージ要件です。アーカイブポリシーで複数の集約メソッド (min、max、mean、sum、std、および count) が使用される場合は、使用する集約メソッドの数をこの値に掛けます。

関連情報

- 「アーカイブポリシー: 時系列データベースへの短期および長期両データの保管」
- 「アーカイブポリシーのプランニングおよび管理」

2.3. アーカイブポリシーのプランニングおよび管理

アーカイブポリシーを使用して、メトリックを集計する方法と、時系列データベースにメトリックを保存する期間を設定できます。アーカイブポリシーは、タイムスパンにおけるポイント数として定義されます。

アーカイブポリシーで 1 秒の粒度および 10 ポイントのポリシーを定義すると、時系列アーカイブは最大 10 秒間保持し、それぞれが 1 秒間の集約を表します。つまり、時系列は最大で、より新しいポイントと古いポイント間の 10 秒間のデータを保持します。アーカイブポリシーは、使用する集約メソッドも定義します。デフォルトはパラメーター **default_aggregation_methods** に設定され、デフォルト値は **mean**、**min**、**max**、**sum**、**std**、**count** に設定されます。したがって、ユースケースによってアーカイブポリシーおよび粒度は異なる場合があります。

アーカイブポリシーをプランニングするには、以下の概念に精通している必要があります。

- メトリック:詳細は、「メトリック」 を参照してください。
- 計測値:詳細は、「カスタム計測値の作成」 を参照してください。

2.3.1. メトリック

Gnocchi は、**メトリック** と呼ばれるオブジェクトタイプを提供します。メトリックとは、サーバーの CPU 使用状況、部屋の温度、ネットワークインターフェイスによって送信されるバイト数など、計測することのできる任意の項目を指します。メトリックには以下の属性が含まれます。

- 識別用の UUID
- 名前
- 計測値を保管および集約するのに使用されるアーカイブポリシー

関連情報

- 用語の定義は、Gnocchi Metric-as-a-Service の用語 を参照してください。

2.3.2. カスタム計測値の作成

計測値とは、API が Gnocchi に送信する受信タプルを指します。タイムスタンプと値で構成されます。独自のカスタム計測値を作成できます。

手順

- カスタム計測値を作成します。

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

2.3.3. メトリクスステータスの確認

openstack metric コマンドを使用して、デプロイメントが成功したことを確認できます。

手順

- デプロイメントを確認します。

```
(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  |
+-----------------------------------------------------+-------+
```

エラーメッセージが表示されなければ、デプロイメントは成功しています。

2.3.4. アーカイブポリシーの作成

アーカイブポリシーを作成して、メトリックを集計する方法と、時系列データベースにメトリックを保存する期間を定義できます。

手順

- アーカイブポリシーを作成します。<archive-policy-name> をポリシーの名前に、<aggregation-method> を集約メソッドに、それぞれ置き換えます。

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

注記

<definition> はポリシー定義です。コンマ (,) を使用して、複数の属性を区切ります。コロン (:) を使用して、アーカイブポリシー定義の名前と値を区切ります。

2.3.5. アーカイブポリシーの表示

アーカイブポリシーを確認するには、次の手順を使用します。

手順

1. アーカイブポリシーを一覧表示します。

```
$ openstack metric archive policy list
```

2. アーカイブポリシーの詳細を表示します。

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

2.3.6. アーカイブポリシーの削除

アーカイブポリシーを削除する場合は、次の手順を使用します。

手順

- アーカイブポリシーを削除します。<archive-policy-name> は、削除するポリシーの名前に置き換えます。

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

検証

- 削除したアーカイブポリシーがアーカイブポリシーのリストに存在しないことを確認します。

```
$ openstack metric archive policy list
```

2.3.7. アーカイブポリシールールの作成

アーカイブポリシールールを使用して、メトリックとアーカイブポリシー間のマッピングを設定できます。

手順

- アーカイブポリシールールを作成します。<rule-name> はルールの名前に、<archive-policy-name> はアーカイブポリシーの名前に、それぞれ置き換えます。

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

第3章 ログサービスのインストールおよび設定

システムイベントのトラブルシューティングおよびモニタリングには、ログメッセージを使用できます。ログ収集エージェント Rsyslog は、クライアント側でログを収集し、これらのログレコードを、サポートされている Red Hat OpenStack Platform (RHOSP) 環境とは別のリモート Elasticsearch ストレージシステムに送信します。

3.1. ログシステムのアーキテクチャーおよびコンポーネント

モニタリングツールは、クライアントが Red Hat OpenStack Platform (RHOSP) オーバークラウドノードにデプロイされる、クライアント/サーバーモデルを使用します。Rsyslog サービスはクライアント側のロギングを提供します。

RHOSP のログの例には、以下が含まれます。

- syslog や監査ログファイルなどのオペレーティングシステムのログ。
- RabbitMQ や MariaDB などのインフラストラクチャーコンポーネントからのログ。
- Identity (keystone)や Compute (nova)などの RHOSP サービスからのログ。

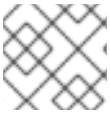
これらのログファイルは、アクション、エラー、アラート、およびその他のイベントを記録します。分散環境では、さまざまなログを 1 か所に収集すると、デバッグや管理に役立ちます。

注記

RHOSP Director は、ログ記録用のサーバー側コンポーネントをデプロイしません。

3.2. ELASTICSEARCH によるロギングの有効化

Elasticsearch はログの保存に使用できるサーバー側のデータベースです。Elasticsearch のログサービスを有効にするには、Elasticsearch のログサービスを認証する必要があります。



注記

Rsyslog サービスは、ロギングのデータストアとして Elasticsearch のみを使用します。

前提条件

- Elasticsearch をデプロイしている。
- サーバーのユーザー名、パスワード、および URL がある。

手順

1. **$HOME/custom_templates/logging-connector.yaml** などのカスタムテンプレートディレクトリーにファイルを作成します。このファイルを編集して、環境に合わせて **RsyslogElasticsearchSetting** パラメーターを設定できます。以下はその例です。

```
parameter_defaults:
  RsyslogElasticsearchSetting:
    uid: "elastic"
    pwd: "yourownpassword"
    skipverifyhost: "on"
```

```
allowunsignedcerts: "on"
server: "https://openstack-log-storage.elasticsearch.tld"
serverport: 443
```

2. **overcloud deployment** コマンドに、**logging-environment-rsyslog.yaml** および **logging-connector.yaml** 環境ファイルのファイルパスを追加します。

```
$ openstack overcloud deploy \
<overcloud_environment_files> \
-e <filepath>/logging-environment-rsyslog.yaml
-e $HOME/custom_templates/logging-connector.yaml
```

- **<overcloud_environment_files>** は、既存のデプロイメント内の環境ファイルのリストに置き換えます。
- **<filepath>** を、**logging-environment-rsyslog.yaml** ファイルへのファイルパス (例: **/usr/share/openstack-tripleo-heat-templates/environments/**) に置き換えます。

3.3. 設定可能なロギングパラメーター

以下の表で、Red Hat OpenStack Platform (RHOSP) のロギング機能の設定に使用するロギングパラメーターについて説明します。これらのパラメーターは **/usr/share/openstack-tripleo-heat-templates/deployment/logging/rsyslog-container-puppet.yaml** ファイルにあります。

表3.1 設定可能なロギングパラメーター

パラメーター	説明
RsyslogElasticsearchSetting	**rsyslog-elasticsearch** プラグインの設定。
RsyslogElasticsearchTlsCACert	Elasticsearch サーバーの証明書を発行した CA の CA 証明書の内容が含まれます。
RsyslogElasticsearchTlsClientCert	Elasticsearch に対してクライアント証明書の認可を行うためのクライアント証明書の内容が含まれます。
RsyslogElasticsearchTlsClientKey	証明書 **RsyslogElasticsearchTlsClientCert** に対応する秘密鍵の内容が含まれます。
RsyslogReopenOnTruncate	ディスク上のファイルサイズがメモリー内の現在のオフセットより小さい場合、入力ファイルを再度開くように rsyslog に要求します。
RsyslogMaxMessageSize	ログメッセージのサイズ制限を設定します。

3.4. デフォルトのログファイルパスのオーバーライド

サービスログファイルへのパスを含めるようにデフォルトのコンテナーを変更する場合は、デフォルトのログファイルパスも変更する必要があります。すべてのコンポーザブルサービスには **<service_name>LoggingSource** パラメーターがあります。たとえば、**nova-compute** サービスの場合、パラメーターは **NovaComputeLoggingSource** です。

手順

1. nova-compute サービスのデフォルトパスをオーバーライドするには、設定ファイルの **NovaComputeLoggingSource** パラメーターにパスを追加します。

```
NovaComputeLoggingSource:
    tag: openstack.nova.compute
    file: <filepath>/nova-compute.log
```

   - **<filepath>** を、**nova-compute.log** ファイルへのファイルパスに置き換えます。
   - 必ずサービスの **tag** および **file** パラメーター値を定義してください。他のパラメーターのデフォルト値を使用できます。

2. 特定のサービスの形式を変更することができます。形式は Rsyslog 設定に渡されます。以下の例は、基本的な構文を示しています。

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

   次の例は、より複雑な変換を示しています。

```
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>.*)?$/'
```

3. 集中型ロギングを有効にした場合は、カスタムテンプレートで次の定義を使用して、追加のログファイル (/var/log/messages など) を転送できます。

```
parameter_defaults:
  ExtraConfig:
    tripleo_logging_sources_messages:
      - tag: openstack.host.messages
        file: /var/log/host/messages
        startmsg.regex: "^[a-zA-Z]{3} [ 1-9][0-9] [:0-9]{8}"
```

3.5. ログレコードの形式の変更

特定のサービスについて、ログレコードの開始の形式を変更することができます。これは Rsyslog 設定に直接渡します。

Red Hat OpenStack Platform (RHOSP) のログレコードのデフォルト形式は ('^[0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}:[0-9]{2}:[0-9]{2}(.[0-9]+ [0-9]+)?(DEBUG|INFO|WARNING|ERROR) ') です。

手順

- ログレコード開始の解析に異なる正規表現を追加するには、設定に **startmsg.regex** を追加します。

```
NovaComputeLoggingSource:
      tag: openstack.nova.compute
      file: /some/other/path/nova-compute.log
      startmsg.regex: `^[0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}:[0-9]{2}:[0-9]{2}(.[0-9]+ \\+[0-9]+)? [A-Z]+ \\([a-z]+\\)`
```

3.6. RSYSLOG と ELASTICSEARCH 間の接続の確認

クライアント側では、Rsyslog と Elasticsearch 間の通信をテストおよび検証できます。

手順

- Elasticsearch 接続ログファイル (Rsyslog コンテナーの **/var/log/rsyslog/omelasticsearch.log**) またはホスト上の **/var/log/containers/rsyslog/omelasticsearch.log** に移動します。このログファイルが存在しない場合や、ログファイルは存在するがログが含まれていない場合、接続の問題はありません。ログファイルが存在しログが含まれている場合は、Rsyslog は正常に接続されていません。

注記

サーバー側から接続をテストするには、Elasticsearch ログを表示して接続に問題を確認します。

3.7. トレースバック

問題のトラブルシューティングを行う場合は、トレースバックログを使用して問題を診断できます。ログファイル中、通常トレースバックとしてすべて同じ問題に関連する複数の情報行が表示されます。

Rsyslog は、ログレコードの開始を定義する正規表現を提供します。通常、各ログレコードはタイムスタンプで始まります。トレースバックの最初の行は、この情報だけが含まれる行です。Rsyslog は最初の行と共に該当するレコードをバンドルし、1 つのログレコードとして送信します。

この動作設定オプションには、<Service>LoggingSource の **startmsg.regex** が使用されます。以下の正規表現が、director のすべての <service>LoggingSource パラメーターのデフォルト値です。

```
startmsg.regex='^[0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}:[0-9]{2}:[0-9]{2}(.[0-9]+ [0-9]+)? (DEBUG|INFO|WARNING|ERROR) '
```

このデフォルトが追加または変更した **LoggingSource** のログレコードと一致しない場合は、それに応じて **startmsg.regex** を変更する必要があります。

3.8. RED HAT OPENSTACK PLATFORM サービスのログファイルの場所

各 Red Hat OpenStack Platform (RHOSP) コンポーネントには、実行中のサービスに固有のファイルを含む個別のログディレクトリーがあります。

3.8.1. Bare Metal Provisioning (ironic) のログファイル

サービス	サービス名	ログのパス
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

3.8.2. Block Storage (cinder) のログファイル

サービス	サービス名	ログのパス
Block Storage API	openstack-cinder-api.service	/var/log/containers/cinder-api.log
Block Storage Backup	openstack-cinder-backup.service	/var/log/containers/cinder/backup.log
情報メッセージ	cinder-manage コマンド	/var/log/containers/cinder/cinder-manage.log
Block Storage Scheduler	openstack-cinder-scheduler.service	/var/log/containers/cinder/scheduler.log
Block Storage Volume	openstack-cinder-volume.service	/var/log/containers/cinder/volume.log

3.8.3. Compute (nova) のログファイル

サービス	サービス名	ログのパス
OpenStack Compute API サービス	openstack-nova-api.service	/var/log/containers/nova/nova-api.log
OpenStack Compute 証明書サーバー	openstack-nova-cert.service	/var/log/containers/nova/nova-cert.log
OpenStack Compute サービス	openstack-nova-compute.service	/var/log/containers/nova/nova-compute.log
OpenStack Compute Conductor サービス	openstack-nova-conductor.service	/var/log/containers/nova/nova-conductor.log
OpenStack Compute VNC コンソール認証サーバー	openstack-nova-consoleauth.service	/var/log/containers/nova/nova-consoleauth.log

サービス	サービス名	ログのパス
情報メッセージ	nova-manage コマンド	/var/log/containers/nova/nova-manage.log
OpenStack Compute NoVNC Proxy サービス	openstack-nova-novncproxy.service	/var/log/containers/nova/nova-novncproxy.log
OpenStack Compute Scheduler サービス	openstack-nova-scheduler.service	/var/log/containers/nova/nova-scheduler.log

3.8.4. Dashboard (horizon) のログファイル

サービス	サービス名	ログのパス
特定ユーザーの対話のログ	Dashboard インターフェイス	/var/log/containers/horizon/horizon.log

Apache HTTP サーバーは、ダッシュボード Web インターフェイス用に追加のログファイルを使用します。このログファイルには、Web ブラウザーまたはコマンドラインクライアント (keystone や nova など) を使用してアクセスできます。以下のログファイルを使用して、ダッシュボードの使用状況を追跡し、障害を診断できます。

目的	ログのパス
すべての処理された HTTP リクエスト	/var/log/containers/httpd/horizon_access.log
HTTP エラー	/var/log/containers/httpd/horizon_error.log
管理者ロールの API リクエスト	/var/log/containers/httpd/keystone_wsgi_admin_access.log
管理者ロールの API エラー	/var/log/containers/httpd/keystone_wsgi_admin_error.log
メンバーロールの API リクエスト	/var/log/containers/httpd/keystone_wsgi_main_access.log
メンバーロールの API エラー	/var/log/containers/httpd/keystone_wsgi_main_error.log

注記

同じホストで実行中の他の Web サービスが報告するエラーを保管するログファイル **/var/log/containers/httpd/default_error.log** もあります。

3.8.5. Identity サービス (keystone) のログファイル

サービス	サービス名	ログのパス
OpenStack Identity サービス	openstack-keystone.service	/var/log/containers/keystone/keystone.log

3.8.6. Image サービス (glance) のログファイル

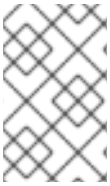
サービス	サービス名	ログのパス
OpenStack Image サービス API サーバー	openstack-glance-api.service	/var/log/containers/glance/api.log
OpenStack Image サービスレジストリーサーバー	openstack-glance-registry.service	/var/log/containers/glance/registry.log

3.8.7. Networking (neutron) のログファイル

サービス	サービス名	ログのパス
OpenStack Neutron DHCP エージェント	neutron-dhcp-agent.service	/var/log/containers/neutron/dhcp-agent.log
OpenStack Networking レイヤー 3 エージェント	neutron-l3-agent.service	/var/log/containers/neutron/l3-agent.log
メタデータエージェントサービス	neutron-metadata-agent.service	/var/log/containers/neutron/metadata-agent.log
メタデータ名前空間プロキシー	該当なし	/var/log/containers/neutron/neutron-ns-metadata-proxy-**UUID**.log
Open vSwitch エージェント	neutron-openvswitch-agent.service	/var/log/containers/neutron/openvswitch-agent.log
OpenStack Networking サービス	neutron-server.service	/var/log/containers/neutron/server.log

3.8.8. Object Storage (swift) のログファイル

OpenStack Object Storage は、システムのロギング機能にのみ、ログを送信します。



注記

デフォルトでは、すべての Object Storage ログファイルは、local0、local1、および local2 syslog ファシリティーを使用して **/var/log/containers/swift/swift.log** に保存されます。

Object Storage ログメッセージは、REST API サービスまたはバックグラウンドデーモンからのものです。

- API サービスメッセージには、API リクエストごとに 1 行含まれます。フロントエンドおよびバックエンドサービスはどちらもメッセージを Post します。
- デーモンメッセージには、デーモンタスクに関する人間が判読できる情報が含まれています。ソース ID は常に行頭にあります。

プロキシーメッセージの例を以下に示します。

```
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
```

デーモンメッセージの例を以下に示します。

```
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
```

3.8.9. Orchestration (heat) のログファイル

サービス	サービス名	ログのパス
OpenStack Heat API サービス	openstack-heat-api.service	/var/log/containers/heat/heat-api.log
OpenStack Heat エンジンサービス	openstack-heat-engine.service	/var/log/containers/heat/heat-engine.log
Orchestration サービスイベント	該当なし	/var/log/containers/heat/heat-manage.log

3.8.10. Shared File Systems サービス (manila) のログファイル

サービス	サービス名	ログのパス
OpenStack Manila API サーバー	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 ファイル共有サービス	openstack-manila-share.service	/var/log/containers/manila/share.log

Manila Python ライブラリーからの情報を **/var/log/containers/manila/manila-manage.log** に記録することもできます。

3.8.11. Telemetry (ceilometer) のログファイル

サービス	サービス名	ログのパス
OpenStack ceilometer 通知エージェント	ceilometer_agent_notification	/var/log/containers/ceilometer/agent-notification.log
OpenStack ceilometer 中央エージェント	ceilometer_agent_central	/var/log/containers/ceilometer/central.log
OpenStack ceilometer コレクション	openstack-ceilometer-collector.service	/var/log/containers/ceilometer/collector.log
OpenStack ceilometer コンピュートエージェント	ceilometer_agent_compute	/var/log/containers/ceilometer/compute.log

3.8.12. サポートサービスのログファイル

次のサービスはコア RHOSP コンポーネントによって使用され、独自のログディレクトリーとファイルがあります。

サービス	サービス名	ログのパス
メッセージブローカー (RabbitMQ)	rabbitmq-server.service	/var/log/rabbitmq/rabbit@**short_hostname**.log /var/log/rabbitmq/rabbit@**short_hostname**-sasl.log (簡易認証およびセキュリティーレイヤーに関するログメッセージ用)
データベースサーバー (MariaDB)	mariadb.service	/var/log/mariadb/mariadb.log
仮想ネットワークスイッチ (Open vSwitch)	openvswitch-nonetwork.service	/var/log/openvswitch/ovsdb-server.log /var/log/openvswitch/ovs-vswitchd.log

3.8.13. aodh (アラームサービス) のログファイル

サービス	コンテナー名	ログのパス
アラーム用 API	aodh_api	/var/log/containers/httpd/aodh-api/aodh_wsgi_access.log
アラームエバリュエーターログ	aodh_evaluator	/var/log/containers/aodh/aodh-evaluator.log
アラームリスナー	aodh_listener	/var/log/containers/aodh/aodh-listener.log
アラーム通知	aodh_notifier	/var/log/containers/aodh/aodh-notifier.log

3.8.14. gnocchi (メトリックストレージ) のログファイル

サービス	コンテナー名	ログのパス
gnocchi API	gnocchi_api	/var/log/containers/httpd/gnocchi-api/gnocchi_wsgi_access.log
gnocchi metricd	gnocchi_metricd	/var/log/containers/gnocchi/gnocchi-metricd.log
gnocchi statsd	gnocchi_statsd	/var/log/containers/gnocchi/gnocchi-statsd.log

第4章 COLLECTD プラグイン

Red Hat OpenStack Platform (RHOSP) 環境に応じて、複数の collectd プラグインを設定できます。

次のプラグインのリストは、デフォルト値をオーバーライドするのに設定できる使用可能なヒートテンプレート **ExtraConfig** パラメーターを示しています。各セクションには、**ExtraConfig** オプションの一般的な設定名が記載されています。たとえば、**example_plugin** という collectd プラグインがある場合、プラグインタイトルの形式は **collectd::plugin::example_plugin** です。

以下の例のように、特定のプラグインで利用可能なパラメーターの表を参照してください。

```
ExtraConfig:
  collectd::plugin::example_plugin::<parameter>: <value>
```

Prometheus または Grafana クエリーの特定プラグインのメトリックテーブルを参照します。

4.1. COLLECTD::PLUGIN::AGGREGATION

複数の値を **aggregation** プラグインで集約できます。メトリックを算出するには、**sum**、**average**、**min**、**max** などの集約関数を使用します (例: 平均および合計の CPU 統計)。

表4.1 集約パラメーター

パラメーター	型
ホスト	文字列
プラグイン	文字列
plugininstance	Integer
agg_type	文字列
typeinstance	文字列
sethost	文字列
setplugin	文字列
setplugininstance	Integer
settypeinstance	文字列
groupBy	文字列の配列
calculatesum	Boolean
calculatenum	Boolean
calculateaverage	Boolean

パラメーター	型
calculateminimum	Boolean
calculatemaximum	Boolean
calculatestddev	Boolean

設定例:

以下のファイルを作成するために、3 つのアグリゲート設定をデプロイします。

1. **aggregator-calcCpuLoadAvg.conf**: ホストおよび状態に分類されるすべての CPU コアの平均 CPU 負荷
2. **aggregator-calcCpuLoadMinMax.conf**: ホストおよび状態による CPU ロードグループの最小および最大数
3. **aggregator-calcMemoryTotalMaxAvg.conf**: タイプ別にグループ化されたメモリーの最大、平均、および合計

集約設定は、デフォルトの **cpu** および **memory** プラグイン設定を使用します。

```
parameter_defaults:
  CollectdExtraPlugins:
    - aggregation

  ExtraConfig:
    collectd::plugin::aggregation::aggregators:
      calcCpuLoadAvg:
        plugin: "cpu"
        agg_type: "cpu"
        groupby:
          - "Host"
          - "TypeInstance"
        calculateaverage: True
      calcCpuLoadMinMax:
        plugin: "cpu"
        agg_type: "cpu"
        groupby:
          - "Host"
          - "TypeInstance"
        calculatemaximum: True
        calculateminimum: True
      calcMemoryTotalMaxAvg:
        plugin: "memory"
        agg_type: "memory"
        groupby:
          - "TypeInstance"
```

```
    calculatemaximum: True
    calculateaverage: True
    calculatesum: True
```

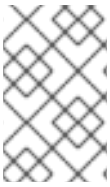
4.2. COLLECTD::PLUGIN::AMQP1

amqp1 プラグインを使用して、AMQ Interconnect などの amqp1 メッセージバスに値を書き込みます。

表4.2 amqp1 パラメーター

パラメーター	タイプ
manage_package	ブール値
transport	文字列
ホスト	文字列
port	Integer
user	文字列
password	文字列
address	文字列
instances	ハッシュ
retry_delay	Integer
send_queue_limit	Integer
interval	Integer

send_queue_limit パラメーターを使用して、送信メトリックキューの長さを制限します。



注記

AMQP1 接続がない場合、プラグインは送信するメッセージをキューに入れ続けます。これにより、バインドされていないメモリー消費が生じる可能性があります。デフォルト値は 0 で、発信メトリックキューを無効にします。

メトリックが見つからない場合は、**send_queue_limit** パラメーターの値を増やします。

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
```

```
    - amqp1

  ExtraConfig:
    collectd::plugin::amqp1::send_queue_limit: 5000
```

4.3. COLLECTD::PLUGIN::APACHE

apache プラグインを使用して、Apache Web サーバーによって提供される **mod_status** プラグインから Apache データを収集します。提供される各インスタンスには **interval** ごとの値 (秒単位) を指定します。インスタンスの **timeout** interval パラメーターを指定すると、値はミリ秒単位です。

表4.3 Apache パラメーター

パラメーター	型
instances	Hash
interval	Integer
manage-package	ブール値
package_install_options	List

表4.4 Apache インスタンスパラメーター

パラメーター	型
url	HTTP URL
user	文字列
password	文字列
verifypeer	Boolean
verifyhost	Boolean
cacert	AbsolutePath
sslciphers	文字列
timeout	Integer

設定例:

この例では、インスタンス名は **localhost** で、Apache Web サーバー (http://10.0.0.111/mod_status?auto) に接続します。プラグインと互換性のないタイプとしてステータスページが返すのを防ぐために、URL の末尾に **?auto** を追加する必要があります。

```
parameter_defaults:
  CollectdExtraPlugins:
  - apache

  ExtraConfig:
    collectd::plugin::apache::instances:
      localhost:
        url: "http://10.0.0.111/mod_status?auto"
```

関連情報

apache プラグインの設定の詳細は、apache を参照してください。

4.4. COLLECTD::PLUGIN::BATTERY

battery プラグインを使用して、ラップトップのバッテリーの残量、電源、または電圧を報告します。

表4.5 バッテリーパラメーター

パラメーター	型
values_percentage	ブール値
report_degraded	ブール値
query_state_fs	ブール値
interval	Integer

関連情報

battery プラグインの設定の詳細は、バッテリー を参照してください。

4.5. COLLECTD::PLUGIN::BIND

bind プラグインを使用して、DNS サーバーからクエリーと応答に関するエンコードされた統計を取得し、それらの値を collectd に送信します。

表4.6 バインドパラメーター

パラメーター	型
url	HTTP URL
memorystats	Boolean
opcodes	Boolean
parsetime	Boolean

パラメーター	型
qtypes	Boolean
resolverstats	Boolean
serverstats	Boolean
zonemaintstats	Boolean
views	Array
interval	Integer

表4.7 バインドビューパラメーター

パラメーター	型
name	String
qtypes	Boolean
resolverstats	Boolean
cacherrsets	Boolean
zones	String のリスト

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
  - bind

  ExtraConfig:
    collectd::plugins::bind:
      url: http://localhost:8053/
      memorystats: true
      opcodes: true
      parsetime: false
      qtypes: true
      resolverstats: true
      serverstats: true
      zonemaintstats: true
      views:
      - name: internal
        qtypes: true
        resolverstats: true
        cacherrsets: true
```

```
    - name: external
      qtypes: true
      resolverstats: true
      cacherrsets: true
      zones:
      - "example.com/IN"
```

4.6. COLLECTD::PLUGIN::CEPH

ceph プラグインを使用して、ceph デーモンからデータを収集します。

表4.8 Ceph パラメーター

パラメーター	型
daemons	Array
longrunavglatency	Boolean
convertspecialmetrictypes	Boolean
package_name	String

設定例:

```
parameter_defaults:
  ExtraConfig:
    collectd::plugin::ceph::daemons:
      - ceph-osd.0
      - ceph-osd.1
      - ceph-osd.2
      - ceph-osd.3
      - ceph-osd.4
```

注記

Object Storage Daemon (OSD) がすべてのノードにない場合には、OSD をリスト表示する必要があります。

collectd をデプロイする時に、**ceph** プラグインを Ceph ノードに追加します。デプロイメントが失敗するので、Ceph ノードの **ceph** プラグインを **CollectdExtraPlugins** に追加しないでください。

関連情報

ceph プラグインの設定の詳細は、ceph を参照してください。

4.7. COLLECTD::PLUGINS::CGROUPS

cgroups プラグインを使用して、cgroup 内のプロセスの情報を収集します。

表4.9 cgroups パラメーター

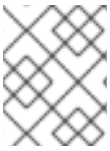
パラメーター	型
ignore_selected	Boolean
interval	Integer
cgroups	List

関連情報

cgroups プラグインの設定の詳細は、cgroups を参照してください。

4.8. COLLECTD::PLUGIN::CONNECTIVITY

connectivity プラグインを使用して、ネットワークインターフェイスの状態を監視します。



注記

インターフェイスがリストにない場合は、すべてのインターフェイスがデフォルトで監視されます。

表4.10 接続性のパラメーター

パラメーター	型
interfaces	Array

設定例:

```
parameter_defaults:
    ExtraConfig:
        collectd::plugin::connectivity::interfaces:
        - eth0
        - eth1
```

関連情報

connectivity プラグインの設定の詳細は、接続性 を参照してください。

4.9. COLLECTD::PLUGIN::CONNTRACK

conntrack プラグインを使用して、Linux 接続追跡テーブルのエントリー数を追跡します。このプラグインのパラメーターはありません。

4.10. COLLECTD::PLUGIN::CONTEXTSWITCH

ContextSwitch プラグインを使用して、システムが処理するコンテキストスイッチの数を収集します。使用できるパラメーターは **interval** (秒単位) のみです。

関連情報

contextswitch プラグインの設定の詳細は、contextswitch を参照してください。

4.11. COLLECTD::PLUGIN::CPU

cpu プラグインを使用して、CPU がさまざまな状態に費やした時間 (例: idle、ユーザーコードの実行中、システムコードの実行中、IO 操作の待機中、その他の状態など) を監視します。

cpu プラグインは、パーセンテージの値ではなく、**jiffies** を収集します。jiffy の値は、ハードウェアプラットフォームのクロック周波数により異なるため、絶対的な間隔単位ではありません。

パーセンテージの値を報告するには、ブール値パラメーター **reportbycpu** および **reportbystate** を **true** に設定し、ブール値のパラメーター値 **percentage** を true に設定します。

このプラグインはデフォルトで有効です。

表4.11 CPU メトリック

名前	説明	クエリー
idle	アイドル時間	**collectd_cpu_total{...,type_instance='idle'}**
interrupt	割り込みでブロックされる CPU	**collectd_cpu_total{...,type_instance='interrupt'}**
nice	優先度の低いプロセスを実行する時間	**collectd_cpu_total{...,type_instance='nice'}**
softirq	割り込み要求の処理に費やされたサイクル数	**collectd_cpu_total{...,type_instance='waitirq'}**
steal	ハイパーバイザーが別の仮想プロセッサーに対応している間、仮想 CPU が実際の CPU を待機する時間の割合	**collectd_cpu_total{...,type_instance='steal'}**
システム	システムレベル (カーネル) で費やした時間	**collectd_cpu_total{...,type_instance='system'}**
user	ユーザープロセスが使用する Jiffies	**collectd_cpu_total{...,type_instance='user'}**
wait	未処理の I/O 要求で待機中の CPU	**collectd_cpu_total{...,type_instance='wait'}**

表4.12 CPU パラメーター

パラメーター	型	デフォルト
reportbystate	Boolean	true

パラメーター	型	デフォルト
valuespercentage	Boolean	true
reportbycpu	Boolean	true
reportnumcpu	Boolean	false
reportgueststate	Boolean	false
subtractgueststate	Boolean	true
interval	Integer	120

設定例:

```
parameter_defaults:
    CollectdExtraPlugins:
      - cpu
    ExtraConfig:
        collectd::plugin::cpu::reportbystate: true
```

関連情報

cpu プラグインの設定の詳細は、cpu を参照してください。

4.12. COLLECTD::PLUGIN::CPUFREQ

cpufreq プラグインを使用して現在の CPU 周波数を収集します。このプラグインのパラメーターはありません。

4.13. COLLECTD::PLUGIN::CSV

csv プラグインを使用して、CSV 形式のローカルファイルに値を書き込みます。

表4.13 csv parameters

パラメーター	型
datadir	String
storerates	Boolean
interval	Integer

4.14. COLLECTD::PLUGIN::DF

df プラグインを使用して、ファイルシステムのディスク領域の使用状況に関する情報を収集します。

このプラグインはデフォルトで有効です。

表4.14 df メトリック

名前	説明	Query
free	空きディスク容量	**collectd_df_df_complex{...,type_instance="free"}**
reserved	予約済みディスク容量	**collectd_df_df_complex{...,type_instance="reserved"}**
used	使用済みディスク容量	**collectd_df_df_complex{...,type_instance="used"}**

表4.15 df パラメーター

パラメーター	型	デフォルト
devices	Array	**[]**
fstypes	Array	**['xfs']**
ignoreselected	Boolean	true
mountpoints	Array	**[]**
reportbydevice	Boolean	true
reportinodes	Boolean	true
reportreserved	Boolean	true
valuesabsolute	Boolean	true
valuespercentage	Boolean	false

設定例:

```
parameter_defaults:
  ExtraConfig:
    collectd::plugin::df::fstypes: ['tmpfs','xfs']
```

関連情報

df プラグインの設定の詳細は、df を参照してください。

4.15. COLLECTD::PLUGIN::DISK

disk プラグインを使用してハードディスクのパフォーマンス統計と (サポートされている場合には) パーティションの情報を収集します。

注記

disk プラグインは、デフォルトですべてのディスクをモニターします。**ignoreselected** パラメーターを使用して、ディスクのリストを無視できます。設定例では、**sda**、**sdb**、および **sdc** ディスクを無視し、リストに含まれていないすべてのディスクをモニターします。

このプラグインはデフォルトで有効です。

表4.16 ディスクパラメーター

パラメーター	型	デフォルト
disks	Array	**[]**
ignoreselected	Boolean	false
udevnameattr	String	**<undefined>**

表4.17 ディスクメトリック

名前	説明
merged	結合可能なキューに置かれた操作の数。たとえば、1 つの物理ディスクアクセスで 2 つ以上の論理操作が提供されます。
time	I/O 操作が完了するまでの平均時間。値は正確ではない場合があります。
io_time	I/O (ms) の処理に費やした時間。このメトリックは、デバイスの負荷率として使用できます。1 秒の値は、負荷の 100% に一致します。
weighted_io_time	I/O の完了時間と、累積する可能性のあるバックログを測定します。
pending_operations	保留中の I/O 操作のキューサイズを表示します。

設定例:

```
parameter_defaults:
  ExtraConfig:
    collectd::plugin::disk::disks: ['sda', 'sdb', 'sdc']
    collectd::plugin::disk::ignoreselected: true
```

関連情報

disk プラグインの設定の詳細は、disk を参照してください。

4.16. COLLECTD::PLUGIN::HUGEPAGES

hugepages プラグインを使用して hugepages 情報を収集します。

> This plugin is enabled by default.

表4.18 hugepages パラメーター

パラメーター	型	デフォルト
report_per_node_hp	Boolean	true
report_root_hp	Boolean	true
values_pages	Boolean	true
values_bytes	Boolean	false
values_percentage	Boolean	false

設定例:

```
parameter_defaults:
  ExtraConfig:
    collectd::plugin::hugepages::values_percentage: true
```

関連情報

- **hugepages** プラグインの設定の詳細は、ヒュージページ を参照してください。

4.17. COLLECTD::PLUGIN::INTERFACE

interface プラグインを使用して、オクテットごとのパケット数、秒ごとのパケットレート、およびエラーレートでインターフェイストラフィックを測定します。

> This plugin is enabled by default.

表4.19 インターフェイスパラメーター

パラメーター	型	デフォルト
interfaces	Array	[]
ignoreselected	Boolean	false
reportinactive	Boolean	true

設定例:

```
parameter_defaults:
  ExtraConfig:
    collectd::plugin::interface::interfaces:
      - lo
    collectd::plugin::interface::ignoreselected: true
```

関連情報

- **interfaces** プラグインの設定の詳細は、インターフェイス を参照してください。

4.18. COLLECTD::PLUGIN::LOAD

load プラグインを使用して、システムロードとシステム使用の概要を収集します。

This plugin is enabled by default.

表4.20 プラグインパラメーター

パラメーター	型	デフォルト
report_relative	Boolean	true

設定例:

```
parameter_defaults:
  ExtraConfig:
    collectd::plugin::load::report_relative: false
```

関連情報

- **load** プラグインの設定の詳細は、ロード を参照してください。

4.19. COLLECTD::PLUGIN::MCELOG

mcelog プラグインを使用して、マシンチェック例外 (MCE) の発生時に関連する通知および統計を送信します。デーモンモードで実行するように **mcelog** を設定し、ログ機能を有効にします。

表4.21 mcelog パラメーター

パラメーター	型
Mcelogfile	String
Memory	Hash **{ mcelogclientsocket[string], persistentnotification[boolean] }**

設定例:

```
parameter_defaults:
    CollectdExtraPlugins: mcelog
    CollectdEnableMcelog: true
```

関連情報

- **mcelog** プラグインの設定の詳細は、celog を参照してください。

4.20. COLLECTD::PLUGIN::MEMCACHED

memcached プラグインを使用して、memcached キャッシュの使用状況、メモリー、およびその他の関連情報に関する情報を取得します。

表4.22 Memcached パラメーター

パラメーター	型
instances	Hash
interval	Integer

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
  - memcached

  ExtraConfig:
    collectd::plugin::memcached::instances:
      local:
        host: "%{hiera('fqdn_canonical')}"
        port: 11211
```

関連情報

- **memcached** プラグインの設定に関する詳細は、memcached を参照してください。

4.21. COLLECTD::PLUGIN::MEMORY

memory プラグインを使用して、システムのメモリーに関する情報を取得します。

> This plugin is enabled by default.

表4.23 メモリーパラメーター

パラメーター	型
デフォルト	valuesabsolute

パラメーター	型
Boolean	true
valuespercentage	Boolean

設定例:

```
parameter_defaults:
  ExtraConfig:
    collectd::plugin::memory::valuesabsolute: true
    collectd::plugin::memory::valuespercentage: false
```

関連情報

- **memory** プラグインの設定の詳細は、メモリー を参照してください。

4.22. COLLECTD::PLUGIN::NTPD

ntpd プラグインを使用して、統計へのアクセスを許可するように設定されているローカル NTP サーバーにクエリーを実行し、設定されたパラメーターと時刻同期ステータスに関する情報を取得します。

表4.24 ntpd パラメーター

パラメーター	型
host	ホスト名
port	Port number (Integer)
reverselookups	Boolean
includeunitid	Boolean
interval	Integer

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
  - ntpd

  ExtraConfig:
    collectd::plugin::ntpd::host: localhost
    collectd::plugin::ntpd::port: 123
    collectd::plugin::ntpd::reverselookups: false
    collectd::plugin::ntpd::includeunitid: false
```

関連情報

- **ntpd** プラグインの設定に関する詳細は、ntpd を参照してください。

4.23. COLLECTD::PLUGIN::OVS_STATS

OVS に接続されたインターフェイスの統計値を収集するには、**ovs_stats** プラグインを使用します。**ovs_stats** プラグインは、OVSDB 管理プロトコル (RFC7047) モニターメカニズムを使用して OVSDB から統計値を取得します。

表4.25 ovs_stats パラメーター

パラメーター	型
address	String
bridges	List
port	Integer
socket	String

設定例:

以下の例は、**ovs_stats** プラグインを有効にする方法を示しています。オーバークラウドを OVS でデプロイする場合には、**ovs_stats** プラグインを有効にする必要はありません。

```
parameter_defaults:
  CollectdExtraPlugins:
    - ovs_stats
  ExtraConfig:
    collectd::plugin::ovs_stats::socket: '/run/openvswitch/db.sock'
```

関連情報

- **ovs_stats** プラグインの設定の詳細は、ovs_stats を参照してください。

4.24. COLLECTD::PLUGIN::PROCESSES

processes プラグインは、システムプロセスに関する情報を提供します。カスタムプロセスマッチングを指定しない場合、プラグインは状態ごとのプロセス数とプロセスフォークレートのみを収集します。

特定のプロセスに関する詳細を収集するには、**process** パラメーターを使用してプロセス名を指定するか、**process_match** オプションを使用して正規表現に一致するプロセス名を指定します。**process_match** 出力の統計は、プロセス名ごとにグループ化されています。

表4.26 プラグインパラメーター

パラメーター	型	デフォルト
processes	Array	**<undefined>**

パラメーター	型	デフォルト
process_matches	Array	**<undefined>**
collect_context_switch	Boolean	**<undefined>**
collect_file_descriptor	Boolean	**<undefined>**
collect_memory_maps	Boolean	**<undefined>**

関連情報

- **processes** プラグインの設定の詳細は プロセス を参照してください。

4.25. COLLECTD::PLUGIN::SMART

smart プラグインを使用して、ノード上の物理ディスクから SMART (自己監視、分析、およびレポートテクノロジー) 情報を収集します。**smart** プラグインが SMART テレメトリーを読み取れるようにするには、パラメーター **CollectdContainerAdditionalCapAdd** を **CAP_SYS_RAWIO** に設定する必要もあります。**CollectdContainerAdditionalCapAdd** パラメーターを設定しないと、以下のメッセージが collectd エラーログに書き込まれます。

Smart プラグイン: collectd を root として実行しますが、CAP_SYS_RAWIO 機能がありません。プラグインの読み取り機能は失敗する可能性があります。init システムで機能がドロップされましたか？

表4.27 スマートパラメーター

パラメーター	型
disks	Array
ignoreselected	Boolean
interval	Integer

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
  - smart

  CollectdContainerAdditionalCapAdd: "CAP_SYS_RAWIO"
```

関連情報

- **smart** プラグインの設定の詳細については、smart を参照してください。

4.26. COLLECTD::PLUGIN::SWAP

swap プラグインを使用して、利用可能なスワップ領域および使用されているスワップ領域に関する情報を収集します。

表4.28 スワッププパラメーター

パラメーター	型
reportbydevice	Boolean
reportbytes	Boolean
valuesabsolute	Boolean
valuespercentage	Boolean
reportio	Boolean

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
  - swap

  ExtraConfig:
    collectd::plugin::swap::reportbydevice: false
    collectd::plugin::swap::reportbytes: true
    collectd::plugin::swap::valuesabsolute: true
    collectd::plugin::swap::valuespercentage: false
    collectd::plugin::swap::reportio: true
```

4.27. COLLECTD::PLUGIN::TCPCONNS

tcpconns プラグインを使用して、設定されたポートからのインバウンドまたはアウトバウンドの TCP 接続の数に関する情報を収集します。ローカルポート設定は、入力接続を表します。リモートポート設定は、出力接続を表します。

表4.29 tcpconns パラメーター

パラメーター	型
localports	Port (Array)
remoteports	Port (Array)
listening	Boolean
allportssummary	Boolean

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
  - tcpconns

  ExtraConfig:
    collectd::plugin::tcpconns::listening: false
    collectd::plugin::tcpconns::localports:
    - 22
    collectd::plugin::tcpconns::remoteports:
    - 22
```

4.28. COLLECTD::PLUGIN::THERMAL

thermal プラグインを使用して、ACPI の通常のゾーン情報を取得します。

表4.30 thermal パラメーター

パラメーター	型
devices	Array
ignoreselected	Boolean
interval	Integer

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
  - thermal
```

4.29. COLLECTD::PLUGIN::UPTIME

uptime プラグインを使用して、システムの稼働時間に関する情報を収集します。

> This plugin is enabled by default.

表4.31 時刻に関するパラメーター

パラメーター	型
interval	Integer

4.30. COLLECTD::PLUGIN::VIRT

virt プラグインを使用して、ホスト上の仮想マシンの **libvirt** API で CPU、ディスク、ネットワーク負荷、およびその他のメトリックを収集します。

このプラグインはコンピュートホストでデフォルトで有効になっています。

表4.32 virt パラメーター

パラメーター	型
connection	String
refresh_interval	Hash
domain	String
block_device	String
interface_device	String
ignore_selected	Boolean
plugin_instance_format	String
hostname_format	String
interface_format	String
extra_stats	String

設定例:

```
ExtraConfig:
  collectd::plugin::virt::hostname_format: "name uuid hostname"
  collectd::plugin::virt::plugin_instance_format: metadata
```

関連情報

virt プラグインの設定の詳細は、virt を参照してください。

4.31. COLLECTD::PLUGIN::VMEM

vmem プラグインを使用して、カーネルサブシステムから仮想メモリーに関する情報を収集します。

表4.33 vmem パラメーター

パラメーター	型
verbose	Boolean
interval	Integer

設定例:

```
parameter_defaults:
  CollectdExtraPlugins:
  - vmem

  ExtraConfig:
    collectd::plugin::vmem::verbose: true
```

4.32. COLLECTD::PLUGIN::WRITE_HTTP

write_http 出力プラグインを使用して、POST リクエストを使用し JSON でメトリックをエンコードして、または **PUTVAL** コマンドを使用して、HTTP サーバーに値を送信します。

表4.34 write_http パラメーター

パラメーター	型
ensure	Enum[**present**, **absent**]
nodes	Hash[String, Hash[String, Scalar]]
urls	Hash[String, Hash[String, Scalar]]
manage_package	Boolean

設定例:

```
parameter_defaults:
    CollectdExtraPlugins:
      - write_http
    ExtraConfig:
        collectd::plugin::write_http::nodes:
            collectd:
                url: “http://collectd.tld.org/collectd”
                metrics: true
                header: “X-Custom-Header: custom_value"
```

関連情報

- **write_http** プラグインの設定に関する詳細は、write_http を参照してください。

4.33. COLLECTD::PLUGIN::WRITE_KAFKA

write_kafka プラグインを使用して、値を Kafka トピックに送信します。 **write_kafka** プラグインを 1 つ以上のトピックブロックで設定します。トピックブロックごとに、一意の名前と 1 つの Kafka プロデューサーを指定する必要があります。topic ブロックでは、以下の per-topic パラメーターを使用できます。

表4.35 write_kafka パラメーター

パラメーター	型
kafka_hosts	Array[String]
topics	Hash
properties	Hash
meta	Hash

設定例:

```
parameter_defaults:
    CollectdExtraPlugins:
      - write_kafka
    ExtraConfig:
      collectd::plugin::write_kafka::kafka_hosts:
        - remote.tld:9092
      collectd::plugin::write_kafka::topics:
        mytopic:
          format: JSON
```

関連情報:

write_kafka プラグインの設定方法は write_kafka を参照してください。