



Red Hat OpenStack Platform 15

ロギング、モニタリング、およびトラブルシューティングガイド

OpenStack のロギング、モニタリング、およびトラブルシューティングの詳細ガイド

Red Hat OpenStack Platform 15 ログイン、モニタリング、およびトラブルシューティングガイド

OpenStack のログイン、モニタリング、およびトラブルシューティングの詳細ガイド

Enter your first name here. Enter your surname here.

Enter your organisation's name here. Enter your organisational division here.

Enter your email address here.

法律上の通知

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Logging_Monitoring_and_Troubleshooting_Guide.ent file |.

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

本ガイドは、Red Hat OpenStack Platform 環境のロギングおよびモニタリング、ならびに問題の解決方法について詳しく説明します。

目次

第1章 本ガイドについて	4
第2章 ロギング	5
2.1. OPENSTACK サービスのログファイル	5
2.1.1. Bare Metal Provisioning (ironic) のログファイル	5
2.1.2. Block Storage (cinder) のログファイル	5
2.1.3. Compute (nova) のログファイル	5
2.1.4. Dashboard (horizon) のログファイル	6
2.1.5. Data Processing (sahara) のログファイル	7
2.1.6. Database as a Service (trove) のログファイル	7
2.1.7. Identity サービス (keystone) のログファイル	7
2.1.8. Image サービス (glance) のログファイル	8
2.1.9. Networking (neutron) のログファイル	8
2.1.10. Object Storage (swift) のログファイル	8
2.1.11. Orchestration (heat) のログファイル	9
2.1.12. Shared File Systems サービス (manila) のログファイル	9
2.1.13. Telemetry (ceilometer) のログファイル	10
2.1.14. サポートサービスのログファイル	10
2.2. ロギングオプションの設定	11
第3章 リモートロギングのインストールと設定	12
第4章 TELEMETRY 用時系列データベース (GNOCCHI) の設定	13
4.1. 時系列データベースについて	13
4.2. メトリック	13
4.3. 時系列データベースのコンポーネント	14
4.4. 時系列データベースの実行	14
4.5. WSGI アプリケーションとしての実行	15
4.6. METRICD ワーカー	15
4.7. 時系列データベースのモニタリング	15
4.8. 時系列データベースのバックアップおよびリストア	15
4.9. GNOCCHI からの古いリソースのバッチ削除	15
第5章 TELEMETRY サービスを使用した容量のメータリング	17
5.1. 計測値の表示	17
5.2. 計測値の作成	17
5.3. 例: クラウドの使用状況に関する計測値の表示	17
5.4. 例: L3 キャッシュの使用状況の表示	17
5.5. 既存のアラームの表示	19
5.6. アラームの作成	19
5.7. アラームの無効化または削除	20
5.8. 例: インスタンスのディスク動作の監視	20
5.9. 例: CPU 使用率の監視	21
5.10. リソース種別の管理	25
第6章 トラブルシューティング	26
6.1. サポート	26
6.2. IDENTITY クライアント (KEYSTONE) の接続性に関する問題のトラブルシューティング	26
6.3. OPENSTACK NETWORKING に関する問題のトラブルシューティング	27
6.4. ダッシュボードのネットワークまたはルータータブの表示に関する問題のトラブルシューティング	28
6.5. DASHBOARD でのインスタンス起動エラーに関するトラブルシューティング	28
6.6. DASHBOARD の KEYSTONE V3 認証に関するトラブルシューティング	29

6.7. OPENSTACK DASHBOARD: RED HAT ACCESS タブ	30
6.7.1. 検索	32
6.7.2. ログ	33
6.7.3. サポート	34

第1章 本ガイドについて



警告

現在 Red Hat では、本リリース用の本ガイドに記載されている情報および手順の見直しを行っています。

本書は、製品ドキュメント から利用可能な Red Hat OpenStack Platform 12 のドキュメントをベースにしています。

現在の Red Hat OpenStack Platform リリース用にサポートが必要な場合は、Red Hat サポートにお問い合わせください。

本書では、Red Hat OpenStack Platform 環境で利用可能なロギングおよびモニタリング機能の概要、ならびに発生する可能性のある問題のトラブルシューティング方法について説明します。

第2章 ログイン

Red Hat OpenStack Platform は、情報メッセージを特定のログファイルに書き込みます。これらのメッセージをトラブルシューティングおよびシステムイベントの監視に使用できます。



注記

個々のログファイルをサポートケースに手動で添付する必要はありません。必要な情報はすべて、**sosreport** ユーティリティーにより自動的に収集されます。これは、[6章 トラブルシューティング](#)で説明されています。

2.1. OPENSTACK サービスのログファイル

それぞれの OpenStack コンポーネントには、実行中のサービス固有のファイルが含まれる個別のログディレクトリーがあります。

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

Service	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

2.1.2. Block Storage (cinder) のログファイル

Service	サービス名	ログのパス
Block Storage API	openstack-cinder-api.service	/var/log/containers/cinder/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

2.1.3. Compute (nova) のログファイル

Service	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
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

2.1.4. Dashboard (horizon) のログファイル

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

Apache HTTP サーバーは、Dashboard Web インターフェース用にさまざまな追加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、nova) を使用してアクセスすることができます。以下のログファイルは、Dashboard の使用状況のトラッキングおよび障害の診断に役立ちます。

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

目的	ログのパス
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

また、**/var/log/containers/httpd/default_error.log** も用意されています。これは、同じホストで実行している他の Web サービスが報告するエラーを保存します。

2.1.5. Data Processing (sahara) のログファイル

Service	サービス名	ログのパス
Sahara API サーバー	openstack-sahara-all.service openstack-sahara-api.service	/var/log/containers/sahara/sahara-all.log /var/log/containers/messages
Sahara Engine サーバー	openstack-sahara-engine.service	/var/log/containers/messages

2.1.6. Database as a Service (trove) のログファイル

Service	サービス名	ログのパス
OpenStack Trove API サービス	openstack-trove-api.service	/var/log/containers/trove/trove-api.log
OpenStack Trove Conductor サービス	openstack-trove-conductor.service	/var/log/containers/trove/trove-conductor.log
OpenStack Trove ゲストエージェント サービス	openstack-trove-guestagent.service	/var/log/containers/trove/logfile.txt
OpenStack Trove タスクマネージャー サービス	openstack-trove-taskmanager.service	/var/log/containers/trove/trove-taskmanager.log

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

Service	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

2.1.8. Image サービス (glance) のログファイル

Service	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/a pi.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/r egistry.log

2.1.9. Networking (neutron) のログファイル

Service	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/l 3-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	該当せず	/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

2.1.10. Object Storage (swift) のログファイル

OpenStack Object Storage は、システムのログイン機能にのみ、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 のログメッセージは、REST API サービスによるものとバックグラウンドデーモンによるものの2つのカテゴリに大別されます。API サービスのメッセージには、一般的な HTTP サーバーと同様に、API リクエストごとに1つの行が含まれます。フロントエンド (プロキシ) とバックエンド (アカウント、コンテナ、オブジェクト) サービスの両方がこのメッセージの POST を行います。デーモンメッセージは構造化されておらず、通常、定期的なタスクを実行するデーモンに関する人間が判読できる情報が含まれます。ただし、メッセージを生成する Object Storage の部分に関係なく、ソースの ID は常に行の先頭に置かれます。

プロキシメッセージの例:

```
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
```

バックグラウンドデーモンからのアドホックメッセージの例:

```
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
```

2.1.11. Orchestration (heat) のログファイル

Service	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	該当せず	/var/log/containers/heat/heat-manage.log

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

Service	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

Service	サービス名	ログのパス
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



注記

Manila Python ライブラリーの一部の情報は、**/var/log/containers/manila/manila-manage.log** に記録することもできます。

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

Service	サービス名	ログのパス
OpenStack ceilometer 通知エージェント	openstack-ceilometer-notification.service	/var/log/containers/ceilometer/agent-notification.log
OpenStack ceilometer アラーム評価	openstack-ceilometer-alarm-evaluator.service	/var/log/containers/ceilometer/alarm-evaluator.log
OpenStack ceilometer アラーム通知	openstack-ceilometer-alarm-notifier.service	/var/log/containers/ceilometer/alarm-notifier.log
OpenStack ceilometer API	httpd.service	/var/log/containers/ceilometer/api.log
情報メッセージ	MongoDB インテグレーション	/var/log/containers/ceilometer/ceilometer-dbsync.log
OpenStack ceilometer 中央エージェント	openstack-ceilometer-central.service	/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	openstack-ceilometer-compute.service	/var/log/containers/ceilometer/compute.log

2.1.14. サポートサービスのログファイル

以下のサービスは OpenStack のコアコンポーネントにより使用され、独自のログディレクトリーおよびファイルを持ちます。

Service	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
ドキュメント指向データベース (MongoDB)	mongod.service	/var/log/mongodb/mongodb.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

2.2. ログインオプションの設定

各コンポーネントは、それぞれの設定ファイルに個別のログイン設定を維持します。たとえば、Compute では、以下のオプションは **/etc/nova/nova.conf** で設定されます。

- デバッグを有効にすることで、情報ログインのレベルを増やします。このオプションを使用するとキャプチャーされた情報の量が大幅に増大するため、一時的に使用すること、または最初にログローテーションの設定を確認することを検討することもできます。

```
debug=True
```

- 詳細ログインを有効化します。

```
verbose=True
```

- ログファイルのパスを変更します。

```
log_dir=/var/log/containers/nova
```

- ログを中央の syslog サーバーに送信します。

```
use_syslog=True
syslog_log_facility=LOG_USER
```



注記

タイムスタンプの設定やログのフォーマットなどのオプションも利用できます。追加のログインオプションについては、コンポーネントの設定ファイルを確認してください。

第3章 リモートロギングのインストールと設定

すべての OpenStack サービスはログファイルを生成および更新します。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。OpenStack のような分散環境では、これらのログを一元的な場所に収集することで、デバッグおよび管理が簡素化されます。

集中ロギングの詳細は、『[監視ツール設定ガイド](#)』を参照してください。

第4章 TELEMETRY 用時系列データベース (GNOCCHI) の設定

時系列データベース(gnocchi)はマルチテナント、メトリクス、およびリソースデータベースです。大規模なメトリックを保管する一方で、オペレーターやユーザーにメトリックおよびリソースの情報へのアクセスを提供します。

4.1. 時系列データベースについて

本セクションでは、時系列データベース (gnocchi) の機能に関して一般的に使用される用語を定義します。

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** 集約メソッドは、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。

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構成されます。

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構成されます。

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。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。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一覧

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。

4.2. メトリック

時系列データベース (gnocchi) は Telemetry からの **メトリック** を保管します。これには、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ー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が含まれます。

メトリックには以下の属性が含まれます。

- メトリックを識別するための UUID
- メトリック名

- 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

時系列データベースは、**etc/ceilometer/polling.yaml** ファイルで定義されているように、デフォルトで以下のメトリックを保存します。

```
[root@controller-0 ~]# podman exec -ti ceilometer_agent_central cat /etc/ceilometer/polling.yaml
---
sources:
  - name: some_pollsters
    interval: 300
    meters:
      - cpu
      - memory.usage
      - network.incoming.bytes
      - network.incoming.packets
      - network.outgoing.bytes
      - network.outgoing.packets
      - disk.read.bytes
      - disk.read.requests
      - disk.write.bytes
      - disk.write.requests
      - hardware.cpu.util
      - hardware.memory.used
      - hardware.memory.total
      - hardware.memory.buffer
      - hardware.memory.cached
      - hardware.memory.swap.avail
      - hardware.memory.swap.total
      - hardware.system_stats.io.outgoing.blocks
      - hardware.system_stats.io.incoming.blocks
      - hardware.network.ip.incoming.datagrams
      - hardware.network.ip.outgoing.datagrams
```

polling.yaml ファイルでは、デフォルトのポーリング間隔 300 秒（5 分）も指定します。

4.3. 時系列データベースのコンポーネント

現在、gnocchi は認証に Identity サービスを、受信測定値のストレージに Redis を、それぞれ使用します。集約した計測値を保管するのに、gnocchi は Swift または Ceph (オブジェクトストレージ) のいずれかに依存します。gnocchi は MySQL も活用して、リソースおよびメトリックのインデックスを保管します。

時系列データベースは、**statsd** プロトコルと互換性のある **statsd** デーモン(**gnocchi-statsd**)を提供し、ネットワークに送信されるメトリックをリッスンできます。Gnocchi で **statsd** のサポートを有効にするには、設定ファイルで **[statsd]** オプションを設定します。リソース ID パラメーターは、全メトリックがアタッチされる主要な一般リソース、リソースとメトリックに関連付けられるユーザーとプロジェクト ID、メトリックの作成に使用するアーカイブポリシー名として使用されます。

メトリックは**gnocchi-statsd** に送信され、指定した名前を設定したリソース ID にアタッチされるので、すべてのメトリックは動的に作成されます。

4.4. 時系列データベースの実行

HTTP サーバーおよびメトリックデーモンを実行して、時系列データベースを実行します。

■

```
# gnocchi-api
# gnocchi-metricd
```

4.5. WSGI アプリケーションとしての実行

mod_wsgi などの WSGI サービスや他の WSGI アプリケーションを使用して Gnocchi を実行することができます。Gnocchi で提供される **gnocchi/rest/app.wsgi** ファイルを使用して、Gnocchi を WSGI アプリケーションとして有効にできます。

gnocchi API 層は、WSGI を使用して実行します。つまり、Apache **httpd** および **mod_wsgi** や **uwsgi** などの別の HTTP デーモンを使用して実行できます。所有している CPU の数に応じて、プロセスおよびスレッドの数を設定します（通常はおおよそ **CPU の数の1.5 倍** です）。1 台のサーバーで十分ではない場合、必要なだけ新規 API サーバーを増設し、異なるマシン上にでも gnocchi をスケールアウトすることができます。

4.6. METRICD ワーカー

メトリックの集約を処理する際、デフォルトでは **gnocchi-metricd** デーモンは、CPU の使用率を最大化するために CPU の全能力を活用します。**gnocchi status** コマンドを使用して HTTP API にクエリーを行い、メトリック処理のクラスターステータスを取得することができます。このコマンドにより、**gnocchi-metricd** の処理のバックログである処理するメトリックの数が表示されます。このバックログが継続的に増加していない限り、**gnocchi-metricd** が送信されるメトリックの量に対応できることを意味します。処理する計測値の数が継続的に増加している場合は、**gnocchi-metricd** デーモンの数を一時的に増やさなければならぬ場合があります。任意の数のサーバー上で、metricd デーモンをいくつでも実行することができます。

director ベースのデプロイメントの場合、環境ファイルで特定のメトリック処理パラメーターを調整することができます。

- **MetricProcessingDelay**: メトリック処理の反復間の遅延時間を調整します。
- **GnocchiMetricdWorkers**: **metricd** ワーカーの数を設定します。

4.7. 時系列データベースのモニタリング

HTTP API の **/v1/status** エンドポイントは、処理する計測値の数（計測値のバックログ）など、さまざまな情報を返すので、簡単に監視することができます。システム全体の正常性を検証するには、HTTP サーバーおよび **gnocchi-metricd** デーモンが動作中で、ログファイルにエラーを書き込んでいないことを確認します。

4.8. 時系列データベースのバックアップおよびリストア

問題のある状況から回復するために、インデックスとストレージの両方をバックアップします。データベースのダンプを作成し (PostgreSQL または MySQL)、データストレージのスナップショットまたはコピーを作成する (Ceph、Swift、またはファイルシステム) 必要があります。復元の手順としては、インデックスおよびストレージのバックアップを復元し、必要に応じて gnocchi を再インストールし、再起動します。

4.9. GNOCCHI からの古いリソースのバッチ削除

古い計測値を削除するには、要件に合ったアーカイブポリシーを作成します。リソース、メトリック、および計測値をバッチで削除するには、CLI または REST API を使用します。たとえば、30 日経過したリソースおよび関連するすべてのメトリックを削除するには、以下のコマンドを実行します。

■ openstack metric resource batch delete "ended_at < '-30days'"

第5章 TELEMETRY サービスを使用した容量のメータリング

OpenStack Telemetry サービスは、請求、課金、およびショーバックの目的に使用できる使用状況のメトリックを提供します。このようなメトリックデータは、サードパーティアプリケーションでクラスターの容量を計画するためにも使用でき、OpenStack heat を使用した仮想インスタンスの自動スケーリングに活用することもできます。詳細は、「[Auto Scaling for Instances](#)」を参照してください。

モニタリングおよびアラーム用に、ceilometer と gnocchi の組み合わせを使用することができます。この構成は小規模のクラスターでサポートされ、既知の制限が存在します。リアルタイムモニタリング用に、Red Hat OpenStack Platform にはメトリックデータを提供するエージェントが同梱されており、このデータを別のモニタリングインフラストラクチャーおよびアプリケーションで処理することができます。詳細は、「[監視ツールの設定](#)」を参照してください。

5.1. 計測値の表示

特定のリソースの測定をすべて表示するには、以下のコマンドを使用します。

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

タイムスタンプの範囲内における特定のリソースの測定のみを表示するには以下のコマンドを使用します。

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

START_TIME および END_TIME の形式は iso-dateThh:mm:ss です。

5.2. 計測値の作成

計測値を使用して、Telemetry サービスにデータを送信することができます。以前に定義した計測値と一致する必要はありません。以下に例を示します。

```
# gnocchi measures add -m 2015-01-12T17:56:23@42 --resource-id UUID METER_NAME
```

5.3. 例: クラウドの使用状況に関する計測値の表示

以下の例では、各プロジェクトの全インスタンスの平均メモリ使用量を表示します。

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

5.4. 例: L3 キャッシュの使用状況の表示

お使いの Intel ハードウェアおよび libvirt のバージョンが **Cache Monitoring Technology (CMT)**に対応している場合は、**cpu_l3_cache** メーターを使用して、インスタンスが使用する L3 キャッシュの量を監視することができます。

L3 キャッシュを監視するには、以下の項目が必要です。

- **LibvirtEnabledPerfEvents** パラメーターの **cmt**。
- **gnocchi_resources.yaml** ファイルの **cpu_l3_cache**。

- Ceilometer **polling.yaml** ファイルの **cpu_l3_cache**。

L3 キャッシュモニタリングの有効化

L3 キャッシュのモニタリングを有効にするには、以下の手順を実施します。

1. Telemetry の YAML ファイルを作成し（**ceilometer-environment.yaml** など）、**LibvirtEnabledPerfEvents** パラメーターに **cmt** を追加します。

```
parameter_defaults:
  LibvirtEnabledPerfEvents: cmt
```

2. この YAML ファイルを使用してオーバークラウドをデプロイします。

```
#!/bin/bash

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

3. コンピュートノード上の Gnocchi で **cpu_l3_cache** が有効であることを確認します。

```
$ sudo -i
# podman exec -ti ceilometer_agent_compute cat /etc/ceilometer/gnocchi_resources.yaml |
grep cpu_l3_cache
```

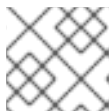
4. Telemetry のポーリングについて **cpu_l3_cache** が有効になっていることを確認します。

```
# podman exec -ti ceilometer_agent_compute cat /etc/ceilometer/polling.yaml | grep
cpu_l3_cache
```

5. Telemetry について **cpu_l3_cache** が有効にされていない場合は、これを有効にしてサービスを再起動します。

```
# podman exec -ti ceilometer_agent_compute echo "      - cpu_l3_cache" >>
/etc/ceilometer/polling.yaml

# podman exec -ti ceilometer_agent_compute pkill -HUP -f "ceilometer.*master process"
```



注記

この podman の変更は、リブート後に維持されません。

このコンピュートノードでゲストインスタンスを起動したら、**gnocchi measures show** コマンドを使用して CMT メトリックを監視することができます。

```
(overcloud) [stack@undercloud-0 ~]$ gnocchi 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 |
+-----+-----+-----+
```

5.5. 既存のアラームの表示

既存の Telemetry アラームを一覧表示するには、**aodh** コマンドを使用します。以下に例を示します。

```
# aodh 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   |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

リソースに割り当てた計測値を一覧表示するには、リソース (インスタンス、イメージ、ボリューム等) の **UUID** を指定します。以下に例を示します。

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

5.6. アラームの作成

aodh を使用して、しきい値に達した時にアクティブになるアラームを作成することができます。以下の例では、個々のインスタンスの平均 CPU 使用率が 80% を超えると、アラームがアクティブになりログエントリーが追加されます。監視目的で、クエリーを使用して特定インスタンスの ID(**94619081-abf5-4f1f-81c7-9cedaa872403**)を分離します。

```
# aodh 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 |
+-----+-----+

```

既存のアラームのしきい値を編集するには、**aodh alarm update** コマンドを使用します。たとえば、アラームのしきい値を 75% に増やすには、以下のコマンドを実行します。

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

5.7. アラームの無効化または削除

アラームを無効にするには、以下のコマンドを実行します。

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

アラームを削除するには、以下のコマンドを実行します。

```
# aodh alarm delete --name cpu_usage_high
```

5.8. 例: インスタンスのディスク動作の監視

aodh アラームを使用して、特定のプロジェクトに含まれるすべてのインスタンスの漸増するディスク動作を監視する方法を、以下の例で説明します。

1. 既存のプロジェクトを確認し、監視するプロジェクトの適切な UUID を選択します。以下の例では、**admin** テナントを使用します。

```

$ openstack project list
+-----+-----+
| ID          | Name    |
+-----+-----+
| 745d33000ac74d30a77539f8920555e7 | admin   |

```



```
| 983739bb834a42ddb48124a38def8538 | services |
| be9e767afd4c4b7ead1417c6dfedde2b | demo    |
+-----+-----+
```

2. プロジェクトの UUID を使用して、**管理** テナントのインスタンスによって生成されたすべての読み取りリクエストの **sum ()** を解析するアラームを作成します (**--query** パラメーターを使用して、クエリーをさらに絞り込むことができます)。

```
# aodh 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                                    |
+-----+-----+
```

5.9. 例：CPU 使用率の監視

インスタンスのパフォーマンスを監視する場合には、Gnocchi データベースを調べ、メモリーや CPU の使用状況などのモニタリング可能なメトリックを特定します。たとえば、インスタンスに対して **gnocchi resource show** を実行して、モニタリング可能なメトリックを特定します。

1. 特定のインスタンスの UUID に対して利用できるメトリックのクエリーを行います。

```
$ gnocchi 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-daae-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             |
+-----+-----+

```

この出力結果の **metrics** の値に、Aodh アラームを使用して監視可能なコンポーネントが一覧表示されます (例: **cpu_util**)。

2. CPU の使用状況を監視するには、**cpu_util** メトリックが必要です。このメトリックの詳細を表示するには、以下のコマンドを実行します。

```
$ gnocchi metric show --resource 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** の値を計算する際の集約間隔を定義します。

3. Aodh を使用して、**cpu_util** をクエリーするモニタリングタスクを作成します。このタスクは、指定した設定に基づいてイベントをトリガーします。たとえば、インスタンスの CPU 使用率が上昇し一定期間 80% を超える場合にログエントリを生成するには、以下のコマンドを実行します。

```
aodh 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** 演算子は、CPU 使用率が 80% 以上の場合にアラームがトリガーされることを定義します。
- **granularity** : メトリックにはアーカイブポリシーが関連付けられます。ポリシーには、さまざまな粒度を設定することができます（例：5 分間隔の集約を 1 時間、および 1 時間間隔の集約を 1 カ月）。**granularity** の値は、アーカイブポリシーで指定された期間と一致する必要があります。
- **evaluation-periods**: アラームがトリガーされる前に満たさなければならない **granularity** 期間の数。たとえば、この値を **2** に設定すると、アラームがトリガーされる前に、2 つのポーリング期間において CPU の使用率が 80% を超える必要があります。
- **[u'log://']**: この値は、イベントを Aodh ログファイルに記録します。



注記

アラームがトリガーされた時(**alarm_actions**)と通常の状態に戻る時(**ok_actions**)で、異なるアクションが実行されるように設定できます (Webhook URL など)。

4. アラームがトリガーされているかどうかを確認するには、アラーム履歴にクエリーを行います。

■

```
aodh 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		

5.10. リソース種別の管理

従来ハードコードされていた Telemetry リソース種別が、**gnocchi** クライアントによって管理できるようになりました。Gnocchi クライアントを使用して、リソース種別を作成、表示、削除することができ、Gnocchi API を使用して属性を更新または削除することができます。

1. 新しい **リソース種別** を作成します。

```
$ gnocchi 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. **リソース種別** の設定を確認します。

```
$ gnocchi resource-type show testResource01
```

Field	Value
attributes/bla	max_length=255, min_length=123, required=True, type=string
name	testResource01
state	active

3. **リソース種別** を削除します。

```
$ gnocchi resource-type delete testResource01
```



注記

リソースが使用中のリソース種別を削除することはできません。

第6章 トラブルシューティング

本章では、Red Hat OpenStack Platform デプロイメントのトラブルシューティングに役立つログ記録およびサポート情報について記載します。

6.1. サポート

クライアントコマンドが失敗したり、他の問題が発生した場合は、発生した問題、完全なコンソール出力、コンソール出力で参照されているすべてのログファイル、問題のある（またはその可能性がある）ノードの **sosreport** と共に、Red Hat テクニカルサポートまでお問い合わせください。たとえば、コンピュートレベルで問題が発生した場合、Nova ノードで **sosreport** を実行します。また、ネットワークの問題の場合は、Neutron ノードでユーティリティを実行します。一般的なデプロイメントの問題については、クラウドコントローラー上で **sosreport** を実行すると良いでしょう。

sosreport コマンド (**sos** パッケージ) の詳細は、「[What is a sosreport and how to create one in Red Hat Enterprise Linux 4.6 and later](#)」を参照してください。

`/var/log/messages` ファイルでヒントがないか確認もしてください。

6.2. IDENTITY クライアント (KEYSTONE) の接続性に関する問題のトラブルシューティング

Identity クライアント (**keystone**) が Identity サービスにコンタクトできない場合には、次のようなエラーが返されます。

```
Unable to communicate with identity service: [Errno 113] No route to host. (HTTP 400)
```

この問題をデバッグするには、以下に挙げる一般的な原因を確認してください。

Identity サービスが稼働していない

Identity サービスは `httpd.service` 内で実行されるようになりましたIdentity サービスをホストしているシステム上で、サービスのステータスを確認します。

```
# systemctl status httpd.service
```

サービスがアクティブでない場合には、root ユーザーとしてログインしてサービスを起動します。

```
# systemctl start httpd.service
```

ファイアウォールが適切に設定されていない

ポート **5000** および **35357** の TCP トラフィックを許可するようにファイアウォールが設定されていない可能性があります。その場合は、『オーバークラウドの高度なカスタマイズ』の「**オーバークラウドのファイアウォールの管理**」に記載の手順を参照して、ファイアウォール設定の確認およびカスタムルールの定義を行います。

サービスエンドポイントが正しく定義されていない

Identity サービスをホストするシステムで、エンドポイントが正しく定義されていることを確認します。

1. 管理トークンを取得します。

```
# grep admin_token /etc/keystone/keystone.conf
admin_token = 91f0866234a64fc299db8f26f8729488
```

- 2. Identity サービスの正しい管理エンドポイントを決定します。

```
http://IP:35357/VERSION
```

IP を、Identity サービスをホストしているシステムの IP アドレスまたはホスト名に置き換えます。VERSION を、使用中の API バージョン（**v2.0** または **v3**）に置き換えます。

- 3. 事前に定義されている Identity サービス関連の環境変数の設定を解除します。

```
# unset OS_USERNAME OS_TENANT_NAME OS_PASSWORD OS_AUTH_URL
```

- 4. 管理トークンとエンドポイントを使用して、Identity サービスとの認証を行います。Identity サービスのエンドポイントが正しいことを確認してください。以下に例を示します。

```
# openstack endpoint list --os-token=91f0556234a64fc299db8f26f8729488 --os-url=https://osp.lab.local:35357/v3/ --os-identity-api-version 3
```

一覧表示された Identity サービスの **publicurl**、**internalurl**、および **adminurl** が正しいことを確認してください。特に、各エンドポイント内にリストされている IP アドレスとポート番号が正しく、ネットワーク上で到達可能であるようにしてください。

これらの値が正しくない場合は、正しいエンドポイントを追加し、**openstack** コマンドの **endpoint delete** アクションを使用して正しくないエンドポイントを削除します。以下に例を示します。

```
# openstack endpoint delete 2d32fa6feccc49aab5de538bdf7aa018 --os-token=91f0866234a64fc299db8f26f8729488 --os-url=https://osp.lab.local:35357/v3/ --os-identity-api-version 3
```

TOKEN および ENDPOINT を前のステップで特定した値に置き換えます。ID を、**endpoint-list** アクションで一覧表示される削除するエンドポイントのIDに置き換えます。

6.3. OPENSTACK NETWORKING に関する問題のトラブルシューティング

本項では、OpenStack Networking サービスに関する問題のトラブルシューティングに使用することができるさまざまなコマンドと手順について説明します。

ネットワークデバイスのデバッグ

- **ip a** コマンドで、すべての物理デバイスおよび仮想デバイスを表示します。
- **ovs-vsctl show** コマンドで、仮想スイッチ内のインターフェースとブリッジを表示します。
- **ovs-dpctl show** コマンドで、スイッチ上のデータパスを表示します。

ネットワークパケットのトラッキング

- **tcpdump** コマンドで、パケットが通過しない場所を確認します。

```
# tcpdump -n -i INTERFACE -e -w FILENAME
```

INTERFACE を、パケットが通過できない箇所を確認するネットワークインターフェースの名前に置き換えます。このインターフェース名には、ブリッジまたはホストのイーサネットデバイスの名前を使用することができます。

-e フラグにより、リンクレベルのヘッダー (**vlan** タグが表示される) がダンプされます。

-w フラグはオプションです。このフラグは、出力をファイルに書き込む場合にのみ使用することができます。そうでない場合には、出力は標準出力 (**stdout**) に書き込まれます。

tcpdump についての詳細は、**man tcpdump** コマンドで **man** ページを開いて参照してください。

ネットワーク名前空間のデバッグ

- **ip netns list** コマンドで、既知のネットワーク名前空間をすべて一覧表示します。
- **ip netns exec** コマンドで、特定の名前空間内のルーティングテーブルを表示します。

```
# ip netns exec NAMESPACE_ID bash
# route -n
```

bash シェルで **ip netns exec** コマンドを起動し、それ以降に実行するコマンドが **ip netns exec** コマンドを実行しなくても呼び出されるようにします。

6.4. ダッシュボードのネットワークまたはルータータブの表示に関する問題のトラブルシューティング

OpenStack Networking を使用するように環境が設定されている場合にのみ、Dashboard に **ネットワーク** および **ルーター** タブが表示されます。現在、デフォルトでは、Packstack ユーティリティによって nova ネットワークがデプロイされるため、この方法でデプロイされた環境には、これらのタブは表示されない点に特に注意してください。

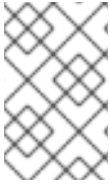
OpenStack Networking が環境にデプロイされているにもかかわらずタブが表示されない場合には、Identity サービスでサービスエンドポイントが正しく定義されて、ファイアウォールがそのエンドポイントへのアクセスを許可し、サービスが稼働していることを確認してください。

6.5. DASHBOARD でのインスタンス起動エラーに関するトラブルシューティング

ダッシュボードを使用してインスタンスを起動する際、操作が失敗すると、一般的な **ERROR** メッセージが表示されます。実際の原因を究明するには、コマンドラインツールを使用する必要があります。

nova list コマンドを使用して、インスタンスの一意識別子を確認します。続いて、この識別子を **nova show** コマンドの引数として使用します。返される項目のいずれかがエラー条件になります。もっとも一般的な値は **NoValidHost** です。

このエラーは、インスタンスをホストするのに十分なリソースが利用できる有効なホストがないことを示しています。この問題を回避するには、より小さなインスタンスサイズを選択するか、その環境のオーバーコミットの上限を高くする方法を検討してください。



注記

インスタンスをホストするには、コンピュータノードで CPU および RAM リソースが使用可能なだけでなく、インスタンスに関連付けられる一時ストレージ用に十分なディスク領域を持つ必要もあります。

6.6. DASHBOARD の KEYSTONE V3 認証に関するトラブルシューティング

django_openstack_auth は、Django の contrib.auth フレームワークと連携する、プラグ可能な Django 認証バックエンドで、OpenStack Identity サービス API に対してユーザー認証を行います。

Django_openstack_auth は、トークンオブジェクトを使用して、ユーザーおよび Keystone 関連の情報をカプセル化します。Dashboard は、トークンオブジェクトを使用して Django ユーザーオブジェクトを再構築します。

現在、トークンオブジェクトは以下を保管します。

- keystone トークン
- ユーザー情報
- 範囲
- ロール
- サービスカタログ

Dashboard は、ユーザーセッションデータの処理に Django のセッションフレームワークを使用します。以下は、利用可能な各種セッションバックエンド一覧です。これらは、local_settings.py ファイルの SESSION_ENGINE 設定で制御されます。

- ローカルメモリーキャッシュ
- Memcached
- データベース
- キャッシュされたデータベース
- クッキー

特に署名付きクッキーのセッションバックエンドが使用されている場合、多数またはすべてのサービスが一度に有効化された場合など、クッキーのサイズが制限に到達して、Dashboard へのログインに失敗する可能性があります。クッキーサイズが増加する理由の1つとして、サービスカタログが挙げられます。多くのサービスが登録されるにつれ、サービスカタログのサイズも増加します。

このようなシナリオでは (特に keystone v3 認証を使用している場合)、セッショントークン管理を向上させるため、Dashboard へログインするための以下の設定を含めてください。

1. /usr/share/openstack-dashboard/openstack_dashboard/settings.py では、以下の設定を追加します。

```
DATABASES =
{
    'default':
    {
```

```
'ENGINE': 'django.db.backends.mysql',
'NAME': 'horizondb',
'USER': 'User Name',
'PASSWORD': 'Password',
'HOST': 'localhost',
}
```

2. 同じファイルで、SESSION_ENGINE を以下に変更します。

```
SESSION_ENGINE = 'django.contrib.sessions.backends.cached_db'
```

3. mysql コマンドを使用してデータベースサービスに接続します。USER は、接続に使用するユーザー名に置き換えます。また、USER は root ユーザー (あるいは、少なくとも正しい権限「create db」を持つユーザー) でなければなりません。

```
# mysql -u USER -p
```

4. Horizon データベースを作成します。

```
mysql > create database horizondb;
```

5. mysql クライアントを終了します。

```
mysql > exit
```

6. 以下のコマンドで、openstack_dashboard ディレクトリーに移動して、データベースを同期します。

```
# cd /usr/share/openstack-dashboard/openstack_dashboard
$ ./manage.py syncdb
```

スーパーユーザーを作成する必要はないため、質問には「n」と回答します。

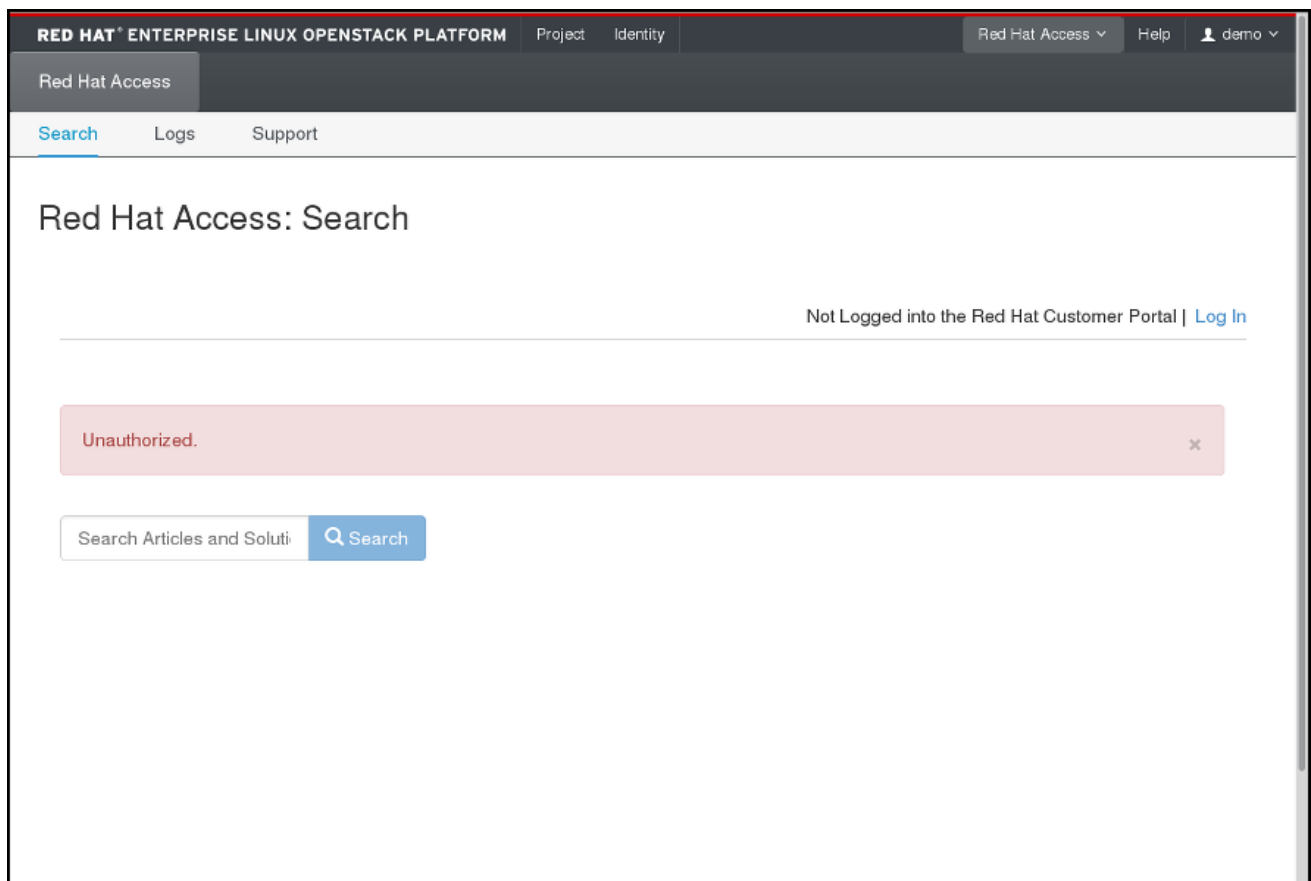
7. Apache http サーバーを再起動します。Red Hat Enterprise Linux の場合は以下のコマンドを実行します。

```
# systemctl restart httpd
```

6.7. OPENSTACK DASHBOARD: RED HAT ACCESS タブ

Red Hat Access タブ (OpenStack Dashboard の一部) では、Red Hat カスタマーポータルの記事やソリューションの検索、表示、インスタンスからのログの表示や診断、カスタマーサポートケースの操作を行うことができます。

図6.1 Red Hat Access タブ



重要

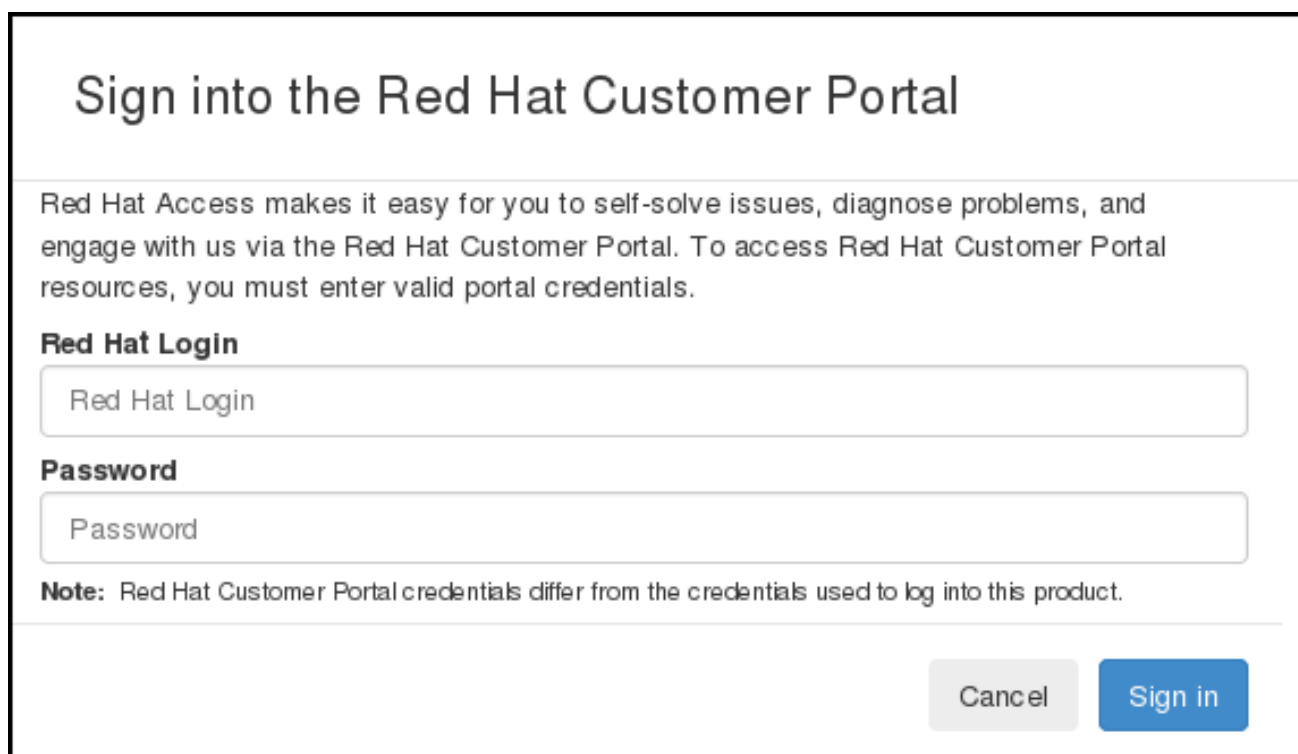
Red Hat Access タブの機能を使用するには、ブラウザーで Red Hat カスタマーポータルにログインする必要があります。

ログインされていない場合には、以下の手順でログインしてください。

1. **ログイン** をクリックします。
2. Red Hat のログイン情報を入力します。
3. Red Hat パスワードを入力します。
4. **サインイン** をクリックします。

フォームは以下のとおりです。

図6.2 Red Hat カスタマーポータルへのログイン



Sign into the Red Hat Customer Portal

Red Hat Access makes it easy for you to self-solve issues, diagnose problems, and engage with us via the Red Hat Customer Portal. To access Red Hat Customer Portal resources, you must enter valid portal credentials.

Red Hat Login

Password

Note: Red Hat Customer Portal credentials differ from the credentials used to log into this product.

[Cancel](#) [Sign in](#)

この時点でログインしないと、認証が必要な機能の1つを使用する際に Red Hat ログインとパスワードが要求されます。

6.7.1. 検索

1つまたは複数の検索キーワードを入力して、Red Hat カスタマーポータルからの記事やソリューションを検索できます。関連の記事やソリューションのタイトルが表示されます。タイトルをクリックして、指定の記事またはソリューションを表示します。

図6.3 Red Hat Access タブの検索結果の例

The screenshot shows the Red Hat Access Search interface. The top navigation bar includes 'RED HAT ENTERPRISE LINUX OPENSTACK PLATFORM', 'Project', 'Identity', 'Red Hat Access', 'Help', and a user profile 'demo'. Below the navigation bar, there are tabs for 'Search', 'Logs', and 'Support'. The main heading is 'Red Hat Access: Search'. A search bar contains the text 'POODLE' and a 'Search' button. To the right, it says 'Logged into the Red Hat Customer Portal as [user] | Log Out'.

On the left, under 'Recommendations', there are four items:

- Poodle TLS vulnerability CVE-2014-8730
- EAP 6.2.1 JBossWeb native and POODLE
- Disabling SSLv3 For POODLE vulnerability produces errors
- Resolution for POODLE SSLv3.0 vulnerability (CVE-2014-3566) in

On the right, under 'Environment', there are four entries:

- Red Hat Enterprise Linux (RHEL) 7
- Red Hat Enterprise Linux (RHEL) 6
- Red Hat Enterprise Linux (RHEL) 5
- Red Hat Enterprise Linux (RHEL) 4

Below 'Environment' is the 'Issue' section, which states: 'Recent media publications are publishing articles indicating that in some cases, TLS is now also impacted by the POODLE flaw and has been tracked by Red Hat as CVE-2014-8730 at Bugzilla-CVE-2014-8730 TLS: incorrect check of padding bytes when using CBC cipher suites.'

6.7.2. ログ

ここから、OpenStack インスタンスからのログを確認することができます。

図6.4 Red Hat Access タブでのインスタンスのログ

The screenshot shows the Red Hat Access Logs interface. The top navigation bar is the same as in Figure 6.3. Below the navigation bar, there are tabs for 'Search', 'Logs', and 'Support'. The main heading is 'Red Hat Access: Logs'. Under 'Instances', there is a filter section with 'Instance Name' and 'Filter' dropdowns, and a 'Filter' button.

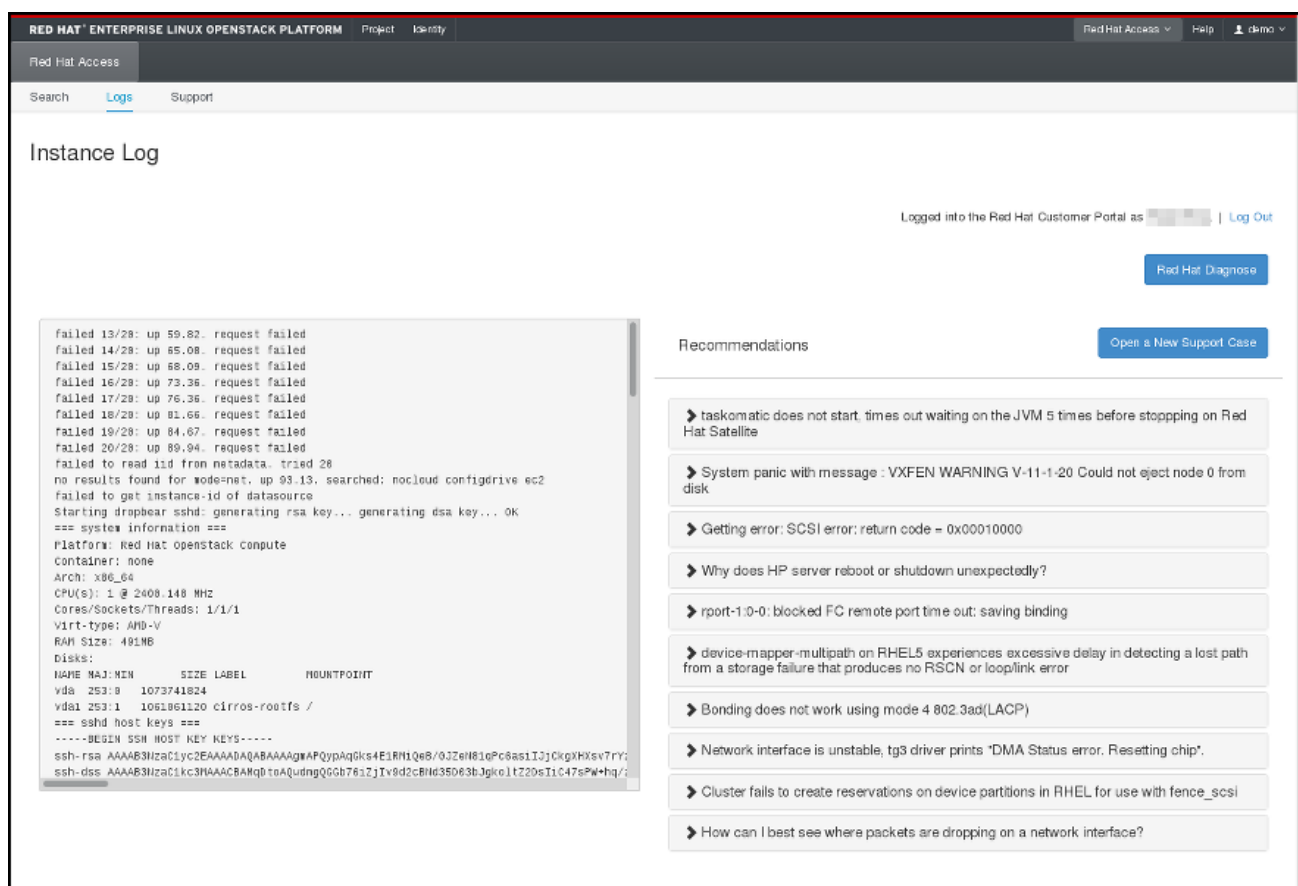
	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
<input type="checkbox"/>	testinstance2	cirros	192.168.0.7	m1.small	OS-Key	Error	nova	None	No State	1 week, 6 days	View Log
<input type="checkbox"/>	testinstance	cirros	192.168.0.2	m1.tiny	-	Shutoff	nova	None	Shut Down	3 weeks, 2 days	View Log

At the bottom, it says 'Displaying 2 items'.

表で必要なインスタンスを探します。インスタンスが多数ある場合は、名前、ステータス、イメージ ID、またはフレーバー ID で絞り込むことができます。確認するインスタンスの **アクション** 欄で、**ログの参照** をクリックします。

インスタンスのログが表示されたら、**Red Hat 診断** をクリックして、コンテンツに関連した提案ソリューションを取得することができます。

図6.5 Red Hat Access タブでのインスタンスのログ

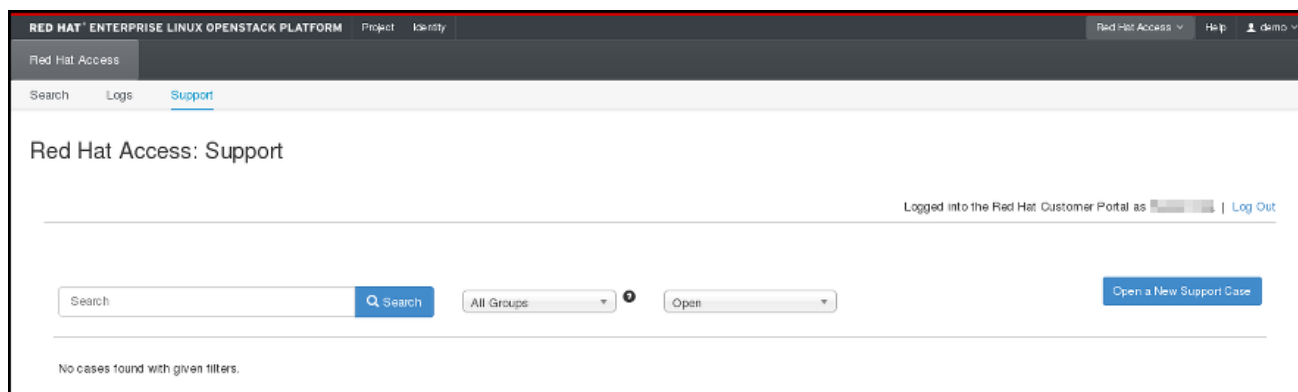


提案されたソリューションで役に立つものがない場合や、問題が正しくロギングされている場合には、**サポートケースを新規作成** をクリックして、問題を Red Hat サポートに報告してください。

6.7.3. サポート

Red Hat Access タブの最後のオプションでは、Red Hat カスタマーポータルをサポートケースを検索することができます。

図6.6 サポートケースの検索



また、適切なボタンをクリックして、以下のページでフォームに入力して新規サポートケースを開くことも可能です。

図6.7 サポートケースの新規作成

RED HAT® ENTERPRISE LINUX OPENSTACK PLATFORM Project Identity Red Hat Access Help demo

Red Hat Access Search Logs Support

Red Hat Access: Support

Logged into the Red Hat Customer Portal as [user] | Log Out

Account: My Account

Owner:

Product:

Product Version:

Summary:

Description:

Next

Recommendations

- ▶ The Production Support Scope of Coverage and Production Support Service Level Agreement
- ▶ What Is The Red Hat Satellite 6 Managed Design Program (MDP) and will there be a Beta?
- ▶ Error message from subscription-manager when attempting to auto-attach shows No installed products on system. No need to attach subscriptions.