Red Hat OpenStack Platform 13

实例的自动扩展

在 Red Hat OpenStack Platform 中配置自动扩展

Last Updated: 2023-04-20

Red Hat OpenStack Platform 13 实例的自动扩展

在 Red Hat OpenStack Platform 中配置自动扩展

OpenStack Team
rhos-docs@redhat.com

法律通告

摘要

自动缩放计算实例以响应系统用量。

目录

第 1 章 为 COMPUTE 实例配置自动扩展

您可以自动横向扩展计算实例，以响应大量系统的使用情况。通过使用考虑 CPU 或内存使用情况等因素的预定义规则，您可以配置 Orchestration (heat)来在需要时自动添加和删除其他实例。

1.1. 自动扩展的架构概述

1.1.1. 编配

提供自动扩展的核心组件是 Orchestration (heat)。您可以使用编排来定义使用人类可读的 YAML 模板的规则。这些规则可根据遥测数据评估系统负载，以确定是否需要添加更多实例到堆栈中。当负载下降时，编配可以再次自动删除未使用的实例。

1.1.2. Telemetry

您可以使用遥测（遥测）来监控 Red Hat OpenStack Platform 环境的性能，收集 CPU、存储和内存用于实例和物理主机的数据。编排模板检查遥测数据，以评估任何预定义的操作是否启动。

1.1.3. 关键术语

- **堆栈** - 堆栈表示操作应用程序所需的所有资源。它可以像一个实例及其资源一样简单，或者作为包含多个实例复杂的，包含多层应用程序的所有资源依赖项。
- **模板** - 为 heat 定义一系列要执行的任务的 YAML 脚本。例如，最好将单独的模板用于某些功能：
  - **模板文件** - 在其中定义 Telemetry 必须响应的阈值，并且定义自动扩展组。
  - 环境文件 - 定义环境的构建信息：要使用的类别和镜像、应如何配置虚拟网络以及如何安装哪些软件。

1.2. 示例：根据 CPU 使用率自动扩展

在本例中，编配会检查遥测数据，并自动增加实例的数量以响应高 CPU 使用量。创建了堆栈模板和环境模板，以定义所需规则和后续配置。本例使用现有资源，如网络，并使用在您自己的环境中可能不同的名称。

1. **创建环境模板，描述实例类别、网络配置和镜像类型，并将它保存到模板 /home/<user>/stacks/example1/cirros.yaml 文件中。将 <user > 变量替换为真实用户名：**

```
heat_template_version: 2016-10-14
description: Template to spawn an cirros instance.

parameters:
  metadata:
    type: json
  image:
    type: string
    description: image used to create instance
    default: cirros
  flavor:
    type: string
    description: instance flavor to be used
    default: m1.tiny
```

```
  key_name:
    type: string
    description: keypair to be used
    default: mykeypair
  network:
    type: string
    description: project network to attach instance to
    default: internal1
  external_network:
    type: string
    description: network used for floating IPs
    default: external_network

resources:
  server:
    type: OS::Nova::Server
    properties:
      block_device_mapping:
        - device_name: vda
          delete_on_termination: true
          volume_id: { get_resource: volume }
      flavor: {get_param: flavor}
      key_name: {get_param: key_name}
      metadata: {get_param: metadata}
      networks:
        - port: { get_resource: port }

  port:
    type: OS::Neutron::Port
    properties:
      network: {get_param: network}
      security_groups:
        - default

  floating_ip:
    type: OS::Neutron::FloatingIP
    properties:
      floating_network: {get_param: external_network}

  floating_ip_assoc:
    type: OS::Neutron::FloatingIPAssociation
    properties:
      floatingip_id: { get_resource: floating_ip }
      port_id: { get_resource: port }

  volume:
    type: OS::Cinder::Volume
    properties:
      image: {get_param: image}
      size: 1
```

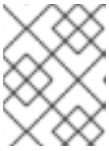
2. **在 ~/stacks/example1/environment.yaml 中注册编排资源：**

```
resource_registry:

  "OS::Nova::Server::Cirros": ~/stacks/example1/cirros.yaml
```

-

3. 创建堆栈模板，描述要监视的 CPU 阈值以及要添加的实例数量。也会创建一个实例组，定义参与此模板的最小和最大实例数量。



注意

granularity 参数需要根据 gnocchi cpu_util 指标粒度来设置。如需更多信息，请参阅 此解决方案文章。

将以下值保存到 ~/stacks/example1/template.yaml 中：

```
heat_template_version: 2016-10-14
description: Example auto scale group, policy and alarm
resources:
  scaleup_group:
    type: OS::Heat::AutoScalingGroup
    properties:
      cooldown: 300
      desired_capacity: 1
      max_size: 3
      min_size: 1
      resource:
        type: OS::Nova::Server::Cirros
        properties:
          metadata: {"metering.server_group": {get_param: "OS::stack_id"}}

  scaleup_policy:
    type: OS::Heat::ScalingPolicy
    properties:
      adjustment_type: change_in_capacity
      auto_scaling_group_id: { get_resource: scaleup_group }
      cooldown: 300
      scaling_adjustment: 1

  scaledown_policy:
    type: OS::Heat::ScalingPolicy
    properties:
      adjustment_type: change_in_capacity
      auto_scaling_group_id: { get_resource: scaleup_group }
      cooldown: 300
      scaling_adjustment: -1

  cpu_alarm_high:
    type: OS::Aodh::GnocchiAggregationByResourcesAlarm
    properties:
      description: Scale up if CPU > 80%
      metric: cpu_util
      aggregation_method: mean
      granularity: 300
      evaluation_periods: 1
      threshold: 80
      resource_type: instance
      comparison_operator: gt
      alarm_actions:
        - str_replace:
```

```
          template: trust+url
          params:
            url: {get_attr: [scaleup_policy, signal_url]}
      query:
        str_replace:
          template: '{"=": {"server_group": "stack_id"}}'
          params:
            stack_id: {get_param: "OS::stack_id"}

  cpu_alarm_low:
    type: OS::Aodh::GnocchiAggregationByResourcesAlarm
    properties:
      metric: cpu_util
      aggregation_method: mean
      granularity: 300
      evaluation_periods: 1
      threshold: 5
      resource_type: instance
      comparison_operator: lt
      alarm_actions:
        - str_replace:
            template: trust+url
            params:
              url: {get_attr: [scaledown_policy, signal_url]}
      query:
        str_replace:
          template: '{"=": {"server_group": "stack_id"}}'
          params:
            stack_id: {get_param: "OS::stack_id"}

outputs:
  scaleup_policy_signal_url:
    value: {get_attr: [scaleup_policy, signal_url]}

  scaledown_policy_signal_url:
    value: {get_attr: [scaledown_policy, signal_url]}
```

4. 运行以下 OpenStack 命令，以构建环境并部署实例：

```
$ openstack stack create  -t template.yaml -e environment.yaml example
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| id                  | 248a98bb-f56e-4934-a281-fffde62d78d8 |
| stack_name          | example                              |
| description         | Example auto scale group, policy and alarm |
| creation_time       | 2017-03-06T15:00:29Z                 |
| updated_time        | None                                 |
| stack_status        | CREATE_IN_PROGRESS                   |
| stack_status_reason | Stack CREATE started                 |
+---------------------+--------------------------------------+
```

5. 编排创建堆栈并启动定义的最少 cirros 实例数量，如 scaleup_group 定义中的 min_size 参数中所述。验证实例是否已成功创建：

```
$ openstack server list
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+----------------------------------+
| ID                                   | Name                                                  | Status | Task State | Power
State | Networks                         |
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+----------------------------------+
| e1524f65-5be6-49e4-8501-e5e5d812c612 | ex-3gax-5f3a4og5cwn2-png47w3u2vjd-server-
vaajhuv4mj3j | ACTIVE | -          | Running     | internal1=10.10.10.9, 192.168.122.8 |
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+----------------------------------+
```

6. **编排也创建两个 cpu 警报，它们用于触发纵向扩展或缩减事件，如 cpu_alarm_high 和 cpu_alarm_low 中定义的。验证触发器是否存在：**

```
$ openstack alarm list
+--------------------------------------+----------------------------------------+-------------------------
----------+------------------+----------+---------+
| alarm_id                             | type                                   | name                          | state
| severity | enabled |
+--------------------------------------+----------------------------------------+-------------------------
----------+------------------+----------+---------+
| 022f707d-46cc-4d39-a0b2-afd2fc7ab86a | gnocchi_aggregation_by_resources_threshold |
example-cpu_alarm_high-odj77qpbld7j | insufficient data | low      | True    |
| 46ed2c50-e05a-44d8-b6f6-f1ebd83af913 | gnocchi_aggregation_by_resources_threshold |
example-cpu_alarm_low-m37jvnm56x2t  | insufficient data | low      | True    |
+--------------------------------------+----------------------------------------+-------------------------
----------+------------------+----------+---------+
```

1.2.1. 测试自动扩展实例

编排可以根据 cpu_alarm_high 阈值定义自动扩展实例。当 CPU 使用率达到阈值 参数中定义的值后，将启动另一个实例来均衡负载。以上 template.yaml 文件中的 阈值 被设置为 80%。

1. **登录到实例并运行多个 dd 命令来生成负载：**

```
$ ssh -i ~/mykey.pem cirros@192.168.122.8
$ sudo dd if=/dev/zero of=/dev/null &
$ sudo dd if=/dev/zero of=/dev/null &
$ sudo dd if=/dev/zero of=/dev/null &
```

2. **运行 dd 命令，预期在 cirros 实例中有 100% CPU 使用率。验证警报是否已触发：**

```
$ openstack alarm list
+--------------------------------------+----------------------------------------+-------------------------
----------+-------+----------+---------+
| alarm_id                             | type                                   | name                          | state |
severity | enabled |
+--------------------------------------+----------------------------------------+-------------------------
----------+-------+----------+---------+
| 022f707d-46cc-4d39-a0b2-afd2fc7ab86a | gnocchi_aggregation_by_resources_threshold |
example-cpu_alarm_high-odj77qpbld7j | alarm | low      | True    |
| 46ed2c50-e05a-44d8-b6f6-f1ebd83af913 | gnocchi_aggregation_by_resources_threshold |
```

```
example-cpu_alarm_low-m37jvnm56x2t  | ok    | low      | True    |
+-------------------------------------+---------------------------------------+-----------------------
-----------+-------+----------+---------+
```

3. 一段时间后（大约 60 秒），编排将启动另一个实例并将它添加到组中。您可以使用 **nova list** 命令验证这一点：

```
$ openstack server list
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+--------------------------------------+
| ID                                   | Name                                                  | Status | Task State | Power
State | Networks                             |
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+--------------------------------------+
| 477ee1af-096c-477c-9a3f-b95b0e2d4ab5 | ex-3gax-4urpikl5koff-yrxk3zxzfmpf-server-
2hde4tp4trnk | ACTIVE | -          | Running     | internal1=10.10.10.13, 192.168.122.17 |
| e1524f65-5be6-49e4-8501-e5e5d812c612 | ex-3gax-5f3a4og5cwn2-png47w3u2vjd-server-
vaajhuv4mj3j | ACTIVE | -          | Running     | internal1=10.10.10.9, 192.168.122.8   |
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+--------------------------------------+
```

4. 在另一个短时间内后，您将观察编排已再次自动缩放到三个实例。将配置设置为最多三个实例，因此它将不会扩展任何更高（**scaleup_group** 定义：**max_size**）。同样，您可以使用上述命令验证：

```
$ openstack server list
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+--------------------------------------+
| ID                                   | Name                                                  | Status | Task State | Power
State | Networks                             |
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+--------------------------------------+
| 477ee1af-096c-477c-9a3f-b95b0e2d4ab5 | ex-3gax-4urpikl5koff-yrxk3zxzfmpf-server-
2hde4tp4trnk | ACTIVE | -          | Running     | internal1=10.10.10.13, 192.168.122.17 |
| e1524f65-5be6-49e4-8501-e5e5d812c612 | ex-3gax-5f3a4og5cwn2-png47w3u2vjd-server-
vaajhuv4mj3j | ACTIVE | -          | Running     | internal1=10.10.10.9, 192.168.122.8   |
| 6c88179e-c368-453d-a01a-555eae8cd77a | ex-3gax-fvxz3tr63j4o-36fhftuja3bw-server-
rhl4sqkjuy5p | ACTIVE | -          | Running     | internal1=10.10.10.5, 192.168.122.5   |
+--------------------------------------+-------------------------------------------------------+--------+------
------+------------+--------------------------------------+
```

1.2.2. 自动缩放实例

编排也可以根据 **cpu_alarm_low** 阈值来自动缩减实例。在本例中，实例在 CPU 使用率低于 5% 后缩减。

1. 终止正在运行的 **dd** 进程，并观察编配开始缩减实例：

```
$ killall dd
```

2. 停止 **dd** 进程会导致 **cpu_alarm_low** 事件触发。因此，编配开始自动缩减和移除实例。验证已触发对应的警报：

```
$ openstack alarm list
+--------------------------------------+---------------------------------------+-----------------------
```

```
----------+-------+----------+---------+
| alarm_id                             | type                                      | name                          | state |
severity | enabled |
+--------------------------------------+-------------------------------------------+-------------------------
----------+-------+----------+---------+
| 022f707d-46cc-4d39-a0b2-afd2fc7ab86a | gnocchi_aggregation_by_resources_threshold |
example-cpu_alarm_high-odj77qpbld7j | ok    | low      | True    |
| 46ed2c50-e05a-44d8-b6f6-f1ebd83af913 | gnocchi_aggregation_by_resources_threshold |
example-cpu_alarm_low-m37jvnm56x2t  | alarm | low      | True    |
+--------------------------------------+-------------------------------------------+-------------------------
----------+-------+----------+---------+
```

五分钟后，编排会持续将实例数量减少到 scaleup_group 定义中 min_size 参数中定义的最小值。在这种情况下，min_size 参数设为 1。

1.2.3. 对设置进行故障排除

如果您的环境无法正常工作，您可以在日志文件和历史记录记录中查找错误。

1. **要获取状态转换的信息，您可以列出堆栈事件记录：**

```
$ openstack stack event list example

2017-03-06 11:12:43Z [example]: CREATE_IN_PROGRESS  Stack CREATE started
2017-03-06 11:12:43Z [example.scaleup_group]: CREATE_IN_PROGRESS  state changed
2017-03-06 11:13:04Z [example.scaleup_group]: CREATE_COMPLETE  state changed
2017-03-06 11:13:04Z [example.scaledown_policy]: CREATE_IN_PROGRESS  state
changed
2017-03-06 11:13:05Z [example.scaleup_policy]: CREATE_IN_PROGRESS  state changed
2017-03-06 11:13:05Z [example.scaledown_policy]: CREATE_COMPLETE  state changed
2017-03-06 11:13:05Z [example.scaleup_policy]: CREATE_COMPLETE  state changed
2017-03-06 11:13:05Z [example.cpu_alarm_low]: CREATE_IN_PROGRESS  state changed
2017-03-06 11:13:05Z [example.cpu_alarm_high]: CREATE_IN_PROGRESS  state changed
2017-03-06 11:13:06Z [example.cpu_alarm_low]: CREATE_COMPLETE  state changed
2017-03-06 11:13:07Z [example.cpu_alarm_high]: CREATE_COMPLETE  state changed
2017-03-06 11:13:07Z [example]: CREATE_COMPLETE  Stack CREATE completed
successfully
2017-03-06 11:19:34Z [example.scaleup_policy]: SIGNAL_COMPLETE  alarm state
changed from alarm to alarm (Remaining as alarm due to 1 samples outside threshold, most
recent: 95.4080102993)
2017-03-06 11:25:43Z [example.scaleup_policy]: SIGNAL_COMPLETE  alarm state
changed from alarm to alarm (Remaining as alarm due to 1 samples outside threshold, most
recent: 95.8869217299)
2017-03-06 11:33:25Z [example.scaledown_policy]: SIGNAL_COMPLETE  alarm state
changed from ok to alarm (Transition to alarm due to 1 samples outside threshold, most
recent: 2.73931707966)
2017-03-06 11:39:15Z [example.scaledown_policy]: SIGNAL_COMPLETE  alarm state
changed from alarm to alarm (Remaining as alarm due to 1 samples outside threshold, most
recent: 2.78110858552)
```

2. **读取警报历史记录日志：**

```
$ openstack alarm-history show 022f707d-46cc-4d39-a0b2-afd2fc7ab86a
+--------------------------+-----------------+-----------------------------------------------------------------
-----------------------------------+----------------------------------+
```

```
| timestamp                  | type            | detail
| event_id                             |
+----------------------------+-----------------+-------------------------------------------------------------
-------------------------------------+----------------------------------+
| 2017-03-06T11:32:35.510000 | state transition | {"transition_reason": "Transition to ok due
to 1 samples inside threshold, most recent:              | 25e0e70b-3eda-466e-abac-
42d9cf67e704 |
|                            |                  | 2.73931707966", "state": "ok"}
|                                  |
| 2017-03-06T11:17:35.403000 | state transition | {"transition_reason": "Transition to alarm
due to 1 samples outside threshold, most recent:          | 8322f62c-0d0a-4dc0-9279-
435510f81039 |
|                            |                  | 95.0964497325", "state": "alarm"}
|                                  |
| 2017-03-06T11:15:35.723000 | state transition | {"transition_reason": "Transition to ok due
to 1 samples inside threshold, most recent:              | 1503bd81-7eba-474e-b74e-
ded8a7b630a1 |
|                            |                  | 3.59330523447", "state": "ok"}
|                                  |
| 2017-03-06T11:13:06.413000 | creation         | {"alarm_actions":
["trust+http://fca6e27e3d524ed68abdc0fd576aa848:delete@192.168.122.126:8004/v1/fd |
224f15c0-b6f1-4690-9a22-0c1d236e65f6 |
|                            |                  |
1c345135be4ee587fef424c241719d/stacks/example/d9ef59ed-b8f8-4e90-bd9b-
|                                  |
|                            |                  | ae87e73ef6e2/resources/scaleup_policy/signal"], "user_id":
"a85f83b7f7784025b6acdc06ef0a8fd8",      |                                      |
|                            |                  | "name": "example-cpu_alarm_high-odj77qpbld7j", "state":
"insufficient data", "timestamp":           |                                      |
|                            |                  | "2017-03-06T11:13:06.413455", "description": "Scale up if
CPU > 80%", "enabled": true,               |                                      |
|                            |                  | "state_timestamp": "2017-03-06T11:13:06.413455", "rule":
{"evaluation_periods": 1, "metric":        |                                      |
|                            |                  | "cpu_util", "aggregation_method": "mean", "granularity": 300,
"threshold": 80.0, "query": "{\"=\":   |                                      |
|                            |                  | {\"server_group\": \"d9ef59ed-b8f8-4e90-bd9b-
ae87e73ef6e2\"}}", "comparison_operator": "gt",        |                                      |
|                            |                  | "resource_type": "instance"}, "alarm_id": "022f707d-46cc-
4d39-a0b2-afd2fc7ab86a",                   |                                      |
|                            |                  | "time_constraints": [], "insufficient_data_actions": null,
"repeat_actions": true, "ok_actions":    |                                      |
|                            |                  | null, "project_id": "fd1c345135be4ee587fef424c241719d",
"type":                                   |                                      |
|                            |                  | "gnocchi_aggregation_by_resources_threshold", "severity":
"low"}                                  |                                      |
+----------------------------+-----------------+-------------------------------------------------------------
-------------------------------------+----------------------------------
```

3. **要查看 heat 为现有堆栈收集的 scale-out 或 scale-down 操作的记录，您可以使用 awk 解析 heat-engine.log ：**

```
$ awk '/Stack UPDATE started/,/Stack CREATE completed successfully/ {print $0}'
/var/log/containers/heat/heat-engine.log
```

4. **要查看与 aodh 相关的信息，请检查 evaluator.log ：**

```
$ grep -i alarm /var/log/containers/aodh/evaluator.log | grep -i transition
```