Red Hat OpenStack Platform 16.2

Overcloud 的外部负载均衡

配置 Red Hat OpenStack Platform 环境以使用外部负载均衡器

Last Updated: 2024-02-07

Red Hat OpenStack Platform 16.2 Overcloud 的外部负载均衡

配置 Red Hat OpenStack Platform 环境以使用外部负载均衡器

OpenStack Team
rhos-docs@redhat.com

法律通告

摘要

配置 Red Hat OpenStack Platform (RHOSP)环境，为 overcloud 使用外部负载均衡器。这包括使用 Red Hat OpenStack Platform director 的负载均衡器和配置 overcloud 的配置指南。

目录

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 的信息。

对红帽文档提供反馈

我们感谢您对文档提供反馈信息。与我们分享您的成功秘诀。

在 JIRA 中提供文档反馈

使用 Create Issue 表单对文档提供反馈。JIRA 问题将在 Red Hat OpenStack Platform Jira 项目中创建，您可以在其中跟踪您的反馈进度。

1. 确保您已登录到 JIRA。如果您没有 JIRA 帐户，请创建一个帐户来提交反馈。
2. 点击以下链接打开 **Create Issue** 页面： Create Issue
3. 完成 **Summary** 和 **Description** 字段。在 **Description** 字段中，包含文档 URL、章节或章节号以及问题的详细描述。不要修改表单中的任何其他字段。
4. 点 **Create**。

第 1 章 将 OVERCLOUD 配置为使用外部负载均衡器

在 Red Hat OpenStack Platform (RHOSP)中，overcloud 将多个 Controller 节点用作高可用性集群，以确保 OpenStack 服务的最大操作性能。集群还为 OpenStack 服务提供负载均衡，该服务均匀地将流量分发到 Controller 节点，并减少每个节点的服务器过载。

默认情况下，overcloud 使用名为 HAProxy 的开源工具来管理负载平衡。HAProxy 负载均衡到运行 OpenStack 服务的 Controller 节点的流量。**haproxy** 软件包包含侦听传入流量的 **haproxy** 守护进程，包括日志记录功能和示例配置。

overcloud 还使用高可用性资源管理器 Pacemaker 来控制 HAProxy 作为高可用性服务。这意味着 HAProxy 在每个 Controller 节点上运行，并根据您在每个配置中定义的一组规则分发流量。

您还可以使用外部负载均衡器来执行此分发。例如，您的机构可能使用基于硬件的专用负载均衡器来处理到 Controller 节点的流量分布。要定义外部负载均衡器和 overcloud 创建的配置，您可以执行以下过程：

1. 安装和配置外部负载均衡器。
2. 使用 heat 模板参数配置和部署 overcloud，将 overcloud 与外部负载均衡器集成。这需要负载均衡器和潜在节点的 IP 地址。

在将 overcloud 配置为使用外部负载均衡器之前，请确保在 overcloud 上部署并运行高可用性。

1.1. 为外部负载均衡器准备您的环境

要为外部负载均衡器准备您的环境，首先请创建一个节点定义模板，并使用 director 注册空白节点。然后，检查所有节点的硬件，并将节点标记为配置集。

使用以下工作流准备您的环境：

- 创建节点定义模板，并在 Red Hat OpenStack Platform director 中注册空白节点。stackenv **.json 中的** 节点定义模板是一个 JSON 格式文件，其中包含用于注册节点的硬件和电源管理详情。
- 检查所有节点的硬件。这样可确保所有节点都处于 manageable 状态。
- 手动将节点标记为配置集。这些配置集标签将节点与类别匹配。然后，类别会被分配到部署角色。

流程

1. 以 **stack** 用户身份登录 director 主机，并提供 director 凭证：

```
$ source ~/stackrc
```

2. 在stackenv **.json** 中创建节点定义模板，并根据您的环境复制并编辑以下示例：

```
{
   "nodes":[
      {
         "mac":[
            "bb:bb:bb:bb:bb:bb"
         ],
         "cpu":"4",
         "memory":"6144",
```

```
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.205"
        },
        {
            "mac":[
                "cc:cc:cc:cc:cc:cc"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.206"
        },
        {
            "mac":[
                "dd:dd:dd:dd:dd:dd"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.207"
        },
        {
            "mac":[
                "ee:ee:ee:ee:ee:ee"
            ],
            "cpu":"4",
            "memory":"6144",
            "disk":"40",
            "arch":"x86_64",
            "pm_type":"ipmi",
            "pm_user":"admin",
            "pm_password":"p@55w0rd!",
            "pm_addr":"192.0.2.208"
        }
    ]
}
```

3. 将文件保存到 **stack** 用户的主目录 **/home/stack/instackenv.json** 中，然后将其导入到 director，然后将节点注册到 director：

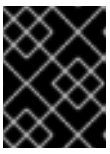
```
$ openstack overcloud node import ~/instackenv.json
```

4. 将内核和 ramdisk 镜像分配给所有节点：

```
$ openstack overcloud node configure
```

5. 检查每个节点的属性：

```
$ openstack overcloud node introspect --all-manageable
```



重要

节点必须处于可管理的状态。确保此进程完成运行。它可能需要 15 分钟来检查这些裸机节点。

6. 获取节点列表以识别它们的 UUID：

```
$ openstack baremetal node list
```

7. 通过在每个节点的 **properties/capabilities** 参数中添加 profile 选项来手动将每个节点标记到特定的配置集。例如，要将三个节点标记为使用 Controller 配置集和一个节点以使用 Compute 配置集，请使用以下命令：

```
$ openstack baremetal node set 1a4e30da-b6dc-499d-ba87-0bd8a3819bc0 --property
capabilities='profile:control,boot_option:local'
$ openstack baremetal node set 6faba1a9-e2d8-4b7c-95a2-c7fbdc12129a --property
capabilities='profile:control,boot_option:local'
$ openstack baremetal node set 5e3b2f50-fcd9-4404-b0a2-59d79924b38e --property
capabilities='profile:control,boot_option:local'
$ openstack baremetal node set 58c3d07e-24f2-48a7-bbb6-6843f0e8ee13 --property
capabilities='profile:compute,boot_option:local'
```

profile:compute 和 **profile:control** 选项将节点标记为对应的配置集。

其他资源

- 规划您的 overcloud

1.2. 为外部负载均衡器配置 OVERCLOUD 网络

要为 overcloud 配置网络，隔离您的服务以使用特定的网络流量，然后为本地环境配置网络环境文件。此文件是一个 heat 环境文件，用于描述 overcloud 网络环境，指向网络接口配置模板，并为您的网络和 IP 地址范围定义子网和 VLAN。

流程

1. 要为各个角色配置节点接口，请自定义以下网络接口模板：
   - 要使用每个角色的 VLAN 配置单个 NIC，请使用以下目录中的示例模板：

     ```
     /usr/share/openstack-tripleo-heat-templates/network/config/single-nic-vlans
     ```

   - 要为每个角色配置绑定 NIC，请使用以下目录中的示例模板：

     ```
     /usr/share/openstack-tripleo-heat-templates/network/config/bond-with-vlans
     ```

2. 通过从 **/home/stack/network-environment.yaml** 中复制文件，并根据您的环境编辑内容来创建网络环境文件。

其他资源

- 基本网络隔离
- 自定义可组合网络
- 自定义网络接口模板
- Overcloud 网络

1.3. 创建外部负载均衡器环境文件

要使用外部负载均衡器部署 overcloud，请使用所需配置创建新环境文件。在本例中，在外部负载均衡器上配置了几个虚拟 IP，每个隔离网络上的一个虚拟 IP，另一个用于 Redis 服务，再启动 overcloud 部署。如果 overcloud 节点 NIC 配置支持配置，则某些虚拟 IP 可能会相同。

流程

- 使用以下示例环境文件 **external-lb.yaml** 创建环境文件，并根据您的环境编辑内容。

```
parameter_defaults:
  ControlFixedIPs: [{'ip_address':'192.0.2.250'}]
  PublicVirtualFixedIPs: [{'ip_address':'172.16.23.250'}]
  InternalApiVirtualFixedIPs: [{'ip_address':'172.16.20.250'}]
  StorageVirtualFixedIPs: [{'ip_address':'172.16.21.250'}]
  StorageMgmtVirtualFixedIPs: [{'ip_address':'172.16.19.250'}]
  RedisVirtualFixedIPs: [{'ip_address':'172.16.20.249'}]
  # IPs assignments for the Overcloud Controller nodes. Ensure these IPs are from each
respective allocation pools defined in the network environment file.
  ControllerIPs:
    external:
    - 172.16.23.150
    - 172.16.23.151
    - 172.16.23.152
    internal_api:
    - 172.16.20.150
    - 172.16.20.151
    - 172.16.20.152
    storage:
    - 172.16.21.150
    - 172.16.21.151
    - 172.16.21.152
    storage_mgmt:
    - 172.16.19.150
    - 172.16.19.151
    - 172.16.19.152
    tenant:
    - 172.16.22.150
    - 172.16.22.151
    - 172.16.22.152
    # CIDRs
    external_cidr: "24"
```

```
  internal_api_cidr: "24"
  storage_cidr: "24"
  storage_mgmt_cidr: "24"
  tenant_cidr: "24"
RedisPassword: p@55w0rd!
ServiceNetMap:
  NeutronTenantNetwork: tenant
  CeilometerApiNetwork: internal_api
  AodhApiNetwork: internal_api
  GnocchiApiNetwork: internal_api
  MongoDbNetwork: internal_api
  CinderApiNetwork: internal_api
  CinderIscsiNetwork: storage
  GlanceApiNetwork: storage
  GlanceRegistryNetwork: internal_api
  KeystoneAdminApiNetwork: internal_api
  KeystonePublicApiNetwork: internal_api
  NeutronApiNetwork: internal_api
  HeatApiNetwork: internal_api
  NovaApiNetwork: internal_api
  NovaMetadataNetwork: internal_api
  NovaVncProxyNetwork: internal_api
  SwiftMgmtNetwork: storage_mgmt
  SwiftProxyNetwork: storage
  HorizonNetwork: internal_api
  MemcachedNetwork: internal_api
  RabbitMqNetwork: internal_api
  RedisNetwork: internal_api
  MysqlNetwork: internal_api
  CephClusterNetwork: storage_mgmt
  CephPublicNetwork: storage
  ControllerHostnameResolveNetwork: internal_api
  ComputeHostnameResolveNetwork: internal_api
  BlockStorageHostnameResolveNetwork: internal_api
  ObjectStorageHostnameResolveNetwork: internal_api
  CephStorageHostnameResolveNetwork: storage
```

注意

- **parameter_defaults** 部分包含每个网络的 VIP 和 IP 分配。这些设置必须与负载均衡器上各个服务相同的 IP 配置匹配。
- **parameter_defaults** 部分为 Redis 服务(RedisPassword)定义管理密码，其中包含 **ServiceNetMap** 参数，该参数将每个 OpenStack 服务映射到特定的网络。负载平衡配置需要此服务重新映射。

1.4. 为外部负载平衡配置 SSL

要为外部负载均衡器配置加密端点，请创建额外的环境文件来启用 SSL 访问端点，然后在外部负载平衡服务器上安装 SSL 证书和密钥副本。默认情况下，overcloud 使用未加密的端点服务。

先决条件

- 如果您使用 IP 地址或域名访问公共端点，请选择以下环境文件之一，以便在 overcloud 部署中包括：

- 要使用域名服务(DNS)访问公共端点，请使用文件 **/usr/share/openstack-tripleo-heat-templates/environments/tls-endpoints-public-dns.yaml**。
- 要使用 IP 地址访问公共端点，请使用文件 **/usr/share/openstack-tripleo-heat-templates/environments/tls-endpoints-public-ip.yaml**。

流程

1. 如果您使用自签名证书，或者证书签名者不在 overcloud 镜像的默认信任存储中，请通过从 heat 模板集合中复制 **inject-trust-anchor.yaml** 环境文件将证书注入 overcloud 镜像：

```
$ cp -r /usr/share/openstack-tripleo-heat-templates/environments/inject-trust-anchor.yaml
~/templates/
```

2. 在文本编辑器中打开文件，并将 root 证书颁发机构文件的内容复制到 **SSLRootCertificate** 参数中：

```
parameter_defaults:
  SSLRootCertificate: |
  -----BEGIN CERTIFICATE-----
  MIIDgzCCAmugAwIBAgIJAKk46qw6ncJaMA0GCSqGSIb3DQEBCwUAMFgxCzAJBgNV
  ...
  sFW3S2roS4X0Af/kSSD8mlBBTFTCMBAj6rtLBKLaQbIxEpIzrgvp
  -----END CERTIFICATE-----
```

重要

证书颁发机构内容为所有新行需要相同的缩进级别。

3. 将 **OS::TripleO::NodeTLSCAData:** 参数的资源 URL 更改为绝对 URL：

```
resource_registry:
  OS::TripleO::NodeTLSCAData: /usr/share/openstack-tripleo-heat-
templates/puppet/extraconfig/tls/ca-inject.yaml
```

4. 可选：如果您使用 DNS 主机名通过 SSL/TLS 访问 overcloud，请创建一个新的环境文件 **~/templates/cloudname.yaml**，并在以下参数中定义 overcloud 端点的主机名：

```
parameter_defaults:
  CloudName: overcloud.example.com
  DnsServers: 10.0.0.1
```

将以下值替换为您环境中的实际值：

- **cloud** Name ：将 **overcloud.example.com** 替换为 overcloud 端点的 DNS 主机名。
- **DnsServers** ：您要使用的 DNS 服务器列表。配置的 DNS 服务器必须包含与公共 API IP 匹配的 **CloudName** 的条目。

1.5. 使用外部负载均衡器部署 OVERCLOUD

若要部署使用外部负载均衡器的 overcloud，请运行 **openstack overcloud deploy**，并为外部负载均衡器包含额外的环境文件和配置文件。

先决条件

- 为外部负载均衡器准备环境。有关如何准备您的环境的更多信息，请参阅 第 1.1 节 “为外部负载均衡器准备您的环境”
- Overcloud 网络是为外部负载均衡器配置的。有关如何配置网络的详情，请参考 第 1.2 节 “为外部负载均衡器配置 overcloud 网络”
- 准备好外部负载均衡器环境文件。有关如何创建环境文件的详情，请参考 第 1.3 节 “创建外部负载均衡器环境文件”
- 为外部负载平衡配置 SSL。有关如何为外部负载平衡配置 SSL 的详情，请参考 第 1.4 节 “为外部负载平衡配置 SSL”

流程

1. 使用外部负载均衡器的所有环境和配置文件部署 overcloud：

```
$ openstack overcloud deploy --templates /
  -e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml /
  -e ~/network-environment.yaml /
  -e /usr/share/openstack-tripleo-heat-templates/environments/external-loadbalancer-vip.yaml
/
  -e ~/external-lb.yaml --control-scale 3 --compute-scale 1 --control-flavor control --compute-
flavor compute /
  -e <SSL/TLS endpoint environment file> /
  -e <DNS hostname environment file> /
  -e <root certificate injection environment file> /
  -e <additional_options_if_needed>
```

将括号 <& **gt**; 中的值替换为您为环境定义的文件路径。

重要

您必须按照本例中列出的顺序将网络环境文件添加到命令中。

这个命令包括以下环境文件：

- **network-isolation.yaml**: 网络隔离配置文件。
- **network-environment.yaml**: 网络配置文件。
- **external-loadbalancer-vip.yaml**: 外部负载均衡虚拟 IP 地址配置文件。
- **external-lb.yaml** ：外部负载均衡器配置文件。您还可以为此文件设置以下选项并调整环境的值：
  - **--control-scale 3**: 将 Controller 节点扩展为 3。
  - **--compute-scale 3**: 将 Compute 节点扩展为 3。
  - **--control-flavor control** ：将特定类别用于 Controller 节点。
  - **--compute-flavor compute** ：将特定类别用于 Compute 节点。
- SSL/TLS 环境文件：

- **SSL/TLS 端点环境文件** ：定义如何连接到公共端点inst 的环境文件。Use **tls-endpoints-public-dns.yaml** or **tls-endpoints-public-ip.yaml**.
- （可选） **DNS 主机名环境文件** ：用于设置 DNS 主机名的环境文件。
- **root 证书注入环境文件** ：注入 root 证书颁发机构的环境文件。

在 overcloud 部署过程中，Red Hat OpenStack Platform director 会置备您的节点。完成此过程需要一些时间。

2. 要查看 overcloud 部署的状态，请输入以下命令：

```
$ source ~/stackrc
$ openstack stack list --nested
```

1.6. 其他资源

- 使用 HAProxy 负载均衡流量。

第 2 章 示例配置：带有外部 HAPROXY 负载均衡器的 OVERCLOUD

本例配置演示了使用联邦 HAProxy 服务器来提供外部负载均衡器的 overcloud。您可以根据环境要求选择不同的外部负载均衡器。

示例配置包括以下元素：

- 运行 HAProxy 的外部负载均衡服务器。
- 一个 Red Hat OpenStack Platform (RHOSP) director 节点。
- 由高可用性集群和 1 个 Compute 节点中的一个由 3 个 Controller 节点组成的 overcloud。
- 使用 VLAN 进行网络隔离。

这个示例为每个网络使用以下 IP 地址分配：

- 内部 API: **172.16.20.0/24**
- 租户： **172.16.22.0/24**
- Storage: **172.16.21.0/24**
- 存储管理： **172.16.19.0/24**
- 外部： **172.16.23.0/24**

这些 IP 范围包括负载均衡器绑定到 OpenStack 服务的 Controller 节点和虚拟 IP 的 IP 分配。

2.1. HAPROXY 配置文件示例

示例文件显示了内部 HAProxy 配置参数。您可以使用示例配置参数作为配置外部负载均衡器的基础。

HAProxy 配置文件包含以下部分：

- 全局配置
- 默认配置
- 服务配置

director 在每个 Controller 节点上的 **/etc/haproxy/haproxy.conf** 文件中为非容器化环境以及容器化环境的 **/var/lib/config-data/puppet-generated/haproxy/etc/haproxy/haproxy.cfg** 文件中提供此配置。

注意

除了 global、default 和 services 参数外，还必须配置其他 HAProxy 参数。有关 HAProxy 参数的更多信息，请参阅位于 Controller 节点上的 **/usr/share/doc/haproxy114/configuration.txt** 的 *HAProxy 配置手册* 或安装 **haproxy** 软件包的任何系统上。

HAProxy 配置文件示例

```
global
```

```
  daemon
  group  haproxy
  log  /dev/log local0
  maxconn  10000
  pidfile  /var/run/haproxy.pid
  user  haproxy

defaults
  log  global
  mode  tcp
  retries  3
  timeout  http-request 10s
  timeout  queue 1m
  timeout  connect 10s
  timeout  client 1m
  timeout  server 1m
  timeout  check 10s

listen aodh
  bind 172.16.20.250:8042
  bind 172.16.20.250:8042
  mode http
  server overcloud-controller-0 172.16.20.150:8042 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8042 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.252:8042 check fall 5 inter 2000 rise 2

listen ceilometer
  bind 172.16.20.250:8777
  bind 172.16.23.250:8777
  server overcloud-controller-0 172.16.20.150:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8777 check fall 5 inter 2000 rise 2

listen cinder
  bind 172.16.20.250:8776
  bind 172.16.23.250:8776
  server overcloud-controller-0 172.16.20.150:8776 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8776 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8776 check fall 5 inter 2000 rise 2

listen glance_api
  bind 172.16.23.250:9292
  bind 172.16.21.250:9292
  server overcloud-controller-0 172.16.21.150:9292 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.21.151:9292 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.21.152:9292 check fall 5 inter 2000 rise 2

listen glance_registry
  bind 172.16.20.250:9191
  server overcloud-controller-0 172.16.20.150:9191 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:9191 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:9191 check fall 5 inter 2000 rise 2

listen gnocchi
  bind 172.16.23.250:8041
  bind 172.16.21.250:8041
```

```
  mode http
  server overcloud-controller-0 172.16.20.150:8041 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8041 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8041 check fall 5 inter 2000 rise 2

listen heat_api
  bind 172.16.20.250:8004
  bind 172.16.23.250:8004
  mode http
  server overcloud-controller-0 172.16.20.150:8004 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8004 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8004 check fall 5 inter 2000 rise 2

listen heat_cfn
  bind 172.16.20.250:8000
  bind 172.16.23.250:8000
  server overcloud-controller-0 172.16.20.150:8000 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.152:8000 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.151:8000 check fall 5 inter 2000 rise 2

listen heat_cloudwatch
  bind 172.16.20.250:8003
  bind 172.16.23.250:8003
  server overcloud-controller-0 172.16.20.150:8003 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8003 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8003 check fall 5 inter 2000 rise 2

listen horizon
  bind 172.16.20.250:80
  bind 172.16.23.250:80
  mode http
  cookie SERVERID insert indirect nocache
  server overcloud-controller-0 172.16.20.150:80 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:80 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:80 check fall 5 inter 2000 rise 2

listen keystone_admin
  bind 172.16.23.250:35357
  bind 172.16.20.250:35357
  server overcloud-controller-0 172.16.20.150:35357 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:35357 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:35357 check fall 5 inter 2000 rise 2

listen keystone_admin_ssh
  bind 172.16.20.250:22
  server overcloud-controller-0 172.16.20.150:22 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:22 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:22 check fall 5 inter 2000 rise 2

listen keystone_public
  bind 172.16.20.250:5000
  bind 172.16.23.250:5000
  server overcloud-controller-0 172.16.20.150:5000 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:5000 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:5000 check fall 5 inter 2000 rise 2
```

```
listen mysql
  bind 172.16.20.250:3306
  option tcpka
  option httpchk
  stick on dst
  stick-table type ip size 1000
  timeout client 0
  timeout server 0
  server overcloud-controller-0 172.16.20.150:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
  server overcloud-controller-1 172.16.20.151:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
  server overcloud-controller-2 172.16.20.152:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2

listen neutron
  bind 172.16.20.250:9696
  bind 172.16.23.250:9696
  server overcloud-controller-0 172.16.20.150:9696 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:9696 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:9696 check fall 5 inter 2000 rise 2

listen nova_ec2
  bind 172.16.20.250:8773
  bind 172.16.23.250:8773
  server overcloud-controller-0 172.16.20.150:8773 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8773 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8773 check fall 5 inter 2000 rise 2

listen nova_metadata
  bind 172.16.20.250:8775
  server overcloud-controller-0 172.16.20.150:8775 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8775 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8775 check fall 5 inter 2000 rise 2

listen nova_novncproxy
  bind 172.16.20.250:6080
  bind 172.16.23.250:6080
  balance source
  server overcloud-controller-0 172.16.20.150:6080 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:6080 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:6080 check fall 5 inter 2000 rise 2

listen nova_osapi
  bind 172.16.20.250:8774
  bind 172.16.23.250:8774
  server overcloud-controller-0 172.16.20.150:8774 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8774 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8774 check fall 5 inter 2000 rise 2

listen nova_placement
  bind 172.16.20.250:8778
  bind 172.16.23.250:8778
  mode http
  server overcloud-controller-0 172.16.20.150:8778 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8778 check fall 5 inter 2000 rise 2
```

```
  server overcloud-controller-2 172.16.20.152:8778 check fall 5 inter 2000 rise 2

listen panko
  bind 172.16.20.250:8779 transparent
  bind 172.16.23.250:8779 transparent
  server overcloud-controller-0 172.16.20.150:8779 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8779 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8779 check fall 5 inter 2000 rise 2

listen redis
  bind 172.16.20.249:6379
  balance first
  option tcp-check
  tcp-check send AUTH\ p@55w0rd!\r\n
  tcp-check send PING\r\n
  tcp-check expect string +PONG
  tcp-check send info\ replication\r\n
  tcp-check expect string role:master
  tcp-check send QUIT\r\n
  tcp-check expect string +OK
  server overcloud-controller-0 172.16.20.150:6379 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:6379 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:6379 check fall 5 inter 2000 rise 2

listen swift_proxy_server
  bind 172.16.23.250:8080
  bind 172.16.21.250:8080
  server overcloud-controller-0 172.16.21.150:8080 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.21.151:8080 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.21.152:8080 check fall 5 inter 2000 rise 2
```

2.1.1. 全局配置参数： HAProxy 配置文件示例

全局配置参数部分为负载均衡器定义一组进程范围参数。您可以使用配置文件中的示例参数来配置外部负载均衡器。根据您的环境调整参数值。

全局配置参数

```
global
  daemon
  user  haproxy
  group  haproxy
  log  /dev/log local0
  maxconn  10000
  pidfile  /var/run/haproxy.pid
```

示例显示以下参数：

- **守护进程** ：作为后台进程运行。
- **用户 haproxy 和组 haproxy** ：定义拥有进程的 Linux 用户和组。
- **log**: 定义要使用的 syslog 服务器。
- **maxconn** ：设置到进程的最大并发连接数。

- **pidfile** ：设置用于进程 ID 的文件。

2.1.2. 默认值配置参数： HAProxy 配置文件示例

默认值配置参数部分定义了在运行外部负载均衡器服务时要使用的一组默认值。您可以使用配置文件中的示例参数来配置外部负载均衡器。根据您的环境调整参数值。

默认值配置参数

```
defaults
  log  global
  mode  tcp
  retries  3
  timeout  http-request 10s
  timeout  queue 1m
  timeout  connect 10s
  timeout  client 1m
  timeout  server 1m
  timeout  check 10s
```

示例显示以下参数：

- **log** ：为该服务启用日志记录。**global** 值表示日志记录功能使用 **global** 部分中的 **log** 参数。
- **模式** ：定义要使用的协议。在这种情况下，默认值为 TCP。
- **retries**：在报告连接失败前设置服务器执行的重试次数。
- **Timeout** ：设置等待特定功能的最长时间。例如，**超时 http-request** 设置等待完整 HTTP 请求的最长时间。

2.1.3. 服务级别配置参数： HAProxy 配置文件示例

服务级别配置参数部分定义了在发送到特定 Red Hat OpenStack Platform (RHOSP)服务时使用的一组参数。您可以使用配置文件中的示例参数来配置外部负载均衡器。根据您的环境调整参数值，并为您要负载均衡的每个服务复制 部分。

服务级别配置参数

本例显示 **ceilometer** 服务的配置参数。

```
listen ceilometer
  bind 172.16.20.250:8777
  bind 172.16.23.250:8777
  server overcloud-controller-0 172.16.20.150:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8777 check fall 5 inter 2000 rise 2
```

要负载均衡的每个服务都必须与配置文件中的某个部分对应。每个服务配置包括以下参数：

- **listen** ：侦听请求的服务名称。
- **bind** ：服务侦听的 IP 地址和 TCP 端口号。每个服务绑定一个不同的地址，它代表不同的网络流量类型。

- **服务器** ：提供服务的每个服务器的名称、服务器 IP 地址和侦听端口以及连接参数：
- **检查** ：（可选）启用健康检查。
- **回退 5** ：（可选）在五个失败的健康检查失败后，该服务被视为离线。
- **2000** 之间：（可选）连续健康检查设置为 2000 毫秒或 2 秒之间的间隔。
- **最后 2：**（可选）在两个成功健康检查后，服务被视为可操作。

在 **ceilometer** 示例中，服务标识 ceilometer 服务提供的 IP 地址和端口为 **172.16.20.2500:8777** 和 **172.16.23.250:8777**。HAProxy 将这些地址的请求定向到 **overcloud-controller-0** (172.16.20.150:8777), **overcloud-controller-1** (172.16.20.151:8777), 或 **overcloud-controller-2** (172.16.0.152:8777)。

其他资源

- 第 2.2 节 “使用负载均衡的服务的配置参数”

2.2. 使用负载均衡的服务的配置参数

对于使用负载均衡的 overcloud 中的每个服务，请使用以下示例作为配置外部负载均衡器的指南。根据您的环境调整参数值，并为您要负载均衡的每个服务复制 部分。

注意

大多数服务都会使用默认的健康检查配置：

- 两个连续健康检查之间的间隔设置为 2000 毫秒，或 2 秒。
- 两个成功健康检查后，服务器被视为可操作。
- 在五个失败的健康检查后，该服务被视为离线。

每个服务都指示该服务 **的其他信息** 部分中的默认健康检查或附加选项。

aodh

端口号： 8042

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen aodh
  bind 172.16.20.250:8042
  bind 172.16.23.250:8042
  server overcloud-controller-0 172.16.20.150:8042 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8042 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8042 check fall 5 inter 2000 rise 2
```

-

opendoi

端口号： 8777

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen ceilometer
  bind 172.16.20.250:8777
  bind 172.16.23.250:8777
  server overcloud-controller-0 172.16.20.150:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8777 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8777 check fall 5 inter 2000 rise 2
```

cinder

端口号： 8776

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen cinder
  bind 172.16.20.250:8776
  bind 172.16.23.250:8776
  server overcloud-controller-0 172.16.20.150:8776 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8776 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8776 check fall 5 inter 2000 rise 2
```

glance_api

端口号： 9292

绑定到： 存储，外部

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的存储

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen glance_api
  bind 172.16.23.250:9292
  bind 172.16.21.250:9292
  server overcloud-controller-0 172.16.21.150:9292 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.21.151:9292 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.21.152:9292 check fall 5 inter 2000 rise 2
```

glance_registry

端口号： 9191

绑定至： internal_api

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen glance_registry
  bind 172.16.20.250:9191
  server overcloud-controller-0 172.16.20.150:9191 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:9191 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:9191 check fall 5 inter 2000 rise 2
```

gnocchi

端口号： 8041

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen gnocchi
  bind 172.16.20.250:8041
  bind 172.16.23.250:8041
  server overcloud-controller-0 172.16.20.150:8041 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8041 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8041 check fall 5 inter 2000 rise 2
```

heat_api

端口号： 8004

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。
- 此服务使用 HTTP 模式而不是默认的 TCP 模式。

HAProxy 示例：

```
listen heat_api
  bind 172.16.20.250:8004
  bind 172.16.23.250:8004
  mode http
  server overcloud-controller-0 172.16.20.150:8004 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8004 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8004 check fall 5 inter 2000 rise 2
```

heat_cfn

端口号： 8000

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen heat_cfn
  bind 172.16.20.250:8000
  bind 172.16.23.250:8000
  server overcloud-controller-0 172.16.20.150:8000 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.152:8000 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.151:8000 check fall 5 inter 2000 rise 2
```

heat_cloudwatch

端口号： 8003

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen heat_cloudwatch
  bind 172.16.20.250:8003
  bind 172.16.23.250:8003
  server overcloud-controller-0 172.16.20.150:8003 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8003 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8003 check fall 5 inter 2000 rise 2
```

Horizon

端口号： 80

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。
- 此服务使用 HTTP 模式而不是默认的 TCP 模式。
- 此服务使用基于 Cookie 的持久性来与 UI 交互。

HAProxy 示例：

```
listen horizon
  bind 172.16.20.250:80
  bind 172.16.23.250:80
  mode http
  cookie SERVERID insert indirect nocache
  server overcloud-controller-0 172.16.20.150:80 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:80 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:80 check fall 5 inter 2000 rise 2
```

keystone_admin

端口号： 35357

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen keystone_admin
  bind 172.16.23.250:35357
  bind 172.16.20.250:35357
  server overcloud-controller-0 172.16.20.150:35357 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:35357 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:35357 check fall 5 inter 2000 rise 2
```

keystone_admin_ssh

端口号： 22

绑定至： internal_api

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen keystone_admin_ssh
  bind 172.16.20.250:22
  server overcloud-controller-0 172.16.20.150:22 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:22 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:22 check fall 5 inter 2000 rise 2
```

keystone_public

端口号： 5000

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen keystone_public
  bind 172.16.20.250:5000
  bind 172.16.23.250:5000
  server overcloud-controller-0 172.16.20.150:5000 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:5000 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:5000 check fall 5 inter 2000 rise 2
```

mysql

端口号： 3306

绑定至： internal_api

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。但是，健康检查使用端口 9200。
- 此服务正一个时间点上只会负载均衡一台服务器。

- 只有在所有其他非备份服务器都不可用时，每个服务器才会用于负载平衡。
- 如果服务器离线，则所有连接都会立即终止。
- 您必须在两端启用 TCP keepalive 数据包发送。
- 您必须启用 HTTP 协议以检查服务器健康状况。
- 您可以配置粘性表来存储 IP 地址，以帮助维护持久性。

重要

mysql 服务使用 Galera 提供高度可用的数据库集群。Galera 支持主动-主动配置，但为了避免锁定争用，您必须使用由负载均衡器强制执行的主动 - 被动配置。

HAProxy 示例：

```
listen mysql
  bind 172.16.20.250:3306
  option tcpka
  option httpchk
  stick on dst
  stick-table type ip size 1000
  timeout client 0
  timeout server 0
  server overcloud-controller-0 172.16.20.150:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
  server overcloud-controller-1 172.16.20.151:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
  server overcloud-controller-2 172.16.20.152:3306 backup check fall 5 inter 2000 on-marked-down
shutdown-sessions port 9200 rise 2
```

neutron

端口号： 9696

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen neutron
  bind 172.16.20.250:9696
  bind 172.16.23.250:9696
  server overcloud-controller-0 172.16.20.150:9696 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:9696 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:9696 check fall 5 inter 2000 rise 2
```

nova_ec2

端口号： 8773

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen nova_ec2
  bind 172.16.20.250:8773
  bind 172.16.23.250:8773
  server overcloud-controller-0 172.16.20.150:8773 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8773 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8773 check fall 5 inter 2000 rise 2
```

nova_metadata

端口号： 8775

绑定至： internal_api

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen nova_metadata
  bind 172.16.20.250:8775
  server overcloud-controller-0 172.16.20.150:8775 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8775 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8775 check fall 5 inter 2000 rise 2
```

nova_novncproxy

端口号： 6080

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。
- 默认负载平衡方法是 round-robin。但是，对于此服务，请使用 **源** 方法。这个方法哈希源 IP 地址，并根据正在运行的服务器的总权重进行划分。此方法还指定接收请求的服务器，并确保同一客户端 IP 地址始终到达同一服务器，除非服务器停机或启动。如果因为正在运行的服务器数量的

更改而更改哈希结果，负载均衡器会将客户端重定向到不同的服务器。

HAProxy 示例：

```
listen nova_novncproxy
  bind 172.16.20.250:6080
  bind 172.16.23.250:6080
  balance source
  server overcloud-controller-0 172.16.20.150:6080 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:6080 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:6080 check fall 5 inter 2000 rise 2
```

nova_osapi

端口号： 8774

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen nova_osapi
  bind 172.16.20.250:8774
  bind 172.16.23.250:8774
  server overcloud-controller-0 172.16.20.150:8774 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8774 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8774 check fall 5 inter 2000 rise 2
```

nova_placement

端口号： 8778

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen nova_placement
  bind 172.16.20.250:8778
  bind 172.16.23.250:8778
  server overcloud-controller-0 172.16.20.150:8778 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8778 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8778 check fall 5 inter 2000 rise 2
```

panko

端口号： 8779

绑定至： internal_api, external

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen panko
  bind 172.16.20.250:8779
  bind 172.16.23.250:8779
  server overcloud-controller-0 172.16.20.150:8779 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.20.151:8779 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.20.152:8779 check fall 5 inter 2000 rise 2
```

redis

端口号： 6379

绑定到： internal_api (redis 服务 IP)

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的 internal_api

其它信息：

- 每个目标服务器都使用默认的健康检查。
- 使用 **tcp-check** send/expect 序列执行健康检查。要发送的字符串是 **info\ replication\r\n**，响应是 **role:master**。
- Redis 服务使用密码进行身份验证。例如，HAProxy 配置使用带有 AUTH 方法和 Redis 管理密码的 **tcp-check**。director 通常会生成随机密码，但您可以定义自定义 Redis 密码。
- 默认平衡方法是 **round-robin**。但是，对于此服务，请使用 **第一种方法**。这样可确保有可用连接插槽的第一个服务器接收连接。

HAProxy 示例：

```
listen redis
  bind 172.16.20.249:6379 transparent
  balance first
  option tcp-check
  tcp-check send AUTH\ p@55w0rd!\r\n
  tcp-check send PING\r\n
  tcp-check expect string +PONG
  tcp-check send info\ replication\r\n
  tcp-check expect string role:master
  tcp-check send QUIT\r\n
  tcp-check expect string +OK
```

```
server overcloud-controller-0 172.16.20.150:6379 check fall 5 inter 2000 rise 2
server overcloud-controller-1 172.16.20.151:6379 check fall 5 inter 2000 rise 2
server overcloud-controller-2 172.16.20.152:6379 check fall 5 inter 2000 rise 2
```

swift_proxy_server

端口号： 8080

绑定到： 存储，外部

目标网络或服务器： overcloud-controller-0、overcloud-controller-1 和 overcloud-controller-2 上的存储

其它信息：

- 每个目标服务器都使用默认的健康检查。

HAProxy 示例：

```
listen swift_proxy_server
  bind 172.16.23.250:8080
  bind 172.16.21.250:8080
  server overcloud-controller-0 172.16.21.150:8080 check fall 5 inter 2000 rise 2
  server overcloud-controller-1 172.16.21.151:8080 check fall 5 inter 2000 rise 2
  server overcloud-controller-2 172.16.21.152:8080 check fall 5 inter 2000 rise 2
```