



Red Hat OpenStack Platform 8

コンピュートインスタンスの高可用性

コンピュートインスタンスの高可用性設定

Red Hat OpenStack Platform 8 コンピュートインスタンスの高可用性

コンピュートインスタンスの高可用性設定

OpenStack Team
rhos-docs@redhat.com

法律上の通知

Copyright © 2018 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, 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 Software Collections 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

本ガイドでは、Red Hat OpenStack Platform 8 で、コンピュートインスタンスに高可用性を設定する方法について説明します。

目次

第1章 概要 3

第2章 環境の要件および前提条件 4

 2.1. 共有ストレージの例外 4

第3章 インストール 5

第4章 高可用性のテスト 10

第5章 参考資料 11

第1章 概要

本ガイドでは、高可用性 (HA) を使用してインスタンスを保護する方法について説明します。HA のバックエンドは **Pacemaker Cluster Manager** です。Pacemaker により、コンピュータノードおよびアプリケーションの障害を検知し、障害発生を予測して対応する機能が追加されます。



注記

本ガイドは、**director** を使用してデプロイ済みの既存の Red Hat OpenStack Platform 8 環境が完全に HA の状態に設定されていることを前提とします。



警告

インスタンスの高可用性を実装している間は、アップグレードやスケールアップの操作はできません。アップグレードやスケールアップを試行しても失敗します。

また、インスタンスの高可用性を有効にすると、将来、director を使用したアップグレードはできなくなります。これは、メジャーアップグレードとマイナーアップグレードの両方に該当します。詳しくは、<https://access.redhat.com/solutions/2661641> を参照してください。

第2章 環境の要件および前提条件

本ガイドで使用する環境の要件と前提条件は以下のとおりです。

- Red Hat OpenStack Platform director を使用して環境がデプロイされていること。
- コントロールプレーンでフェンシングが手動で有効化されていること。
- インスタンスを HA に設定した後に **overcloud** stack の更新が実行されないこと。
- 以下のパッケージが全ノードにインストールされていること。
 - **fence-agents-4.0.11-13.el7_1.1** (以降)
 - **pacemaker-1.1.12-22.el7_1.4.x86_64** (以降)
 - **resource-agents-3.9.5-54.el7_2.18** (以降)
- コンピュートおよびコントロールプレーンの両方を完全に停止する必要がある。
- 一時ストレージおよびブロックストレージの共有ストレージが環境内で有効になっていること。
- メッセージブローカー (AMQP) が各コンピュータノードのホスト名を有効と認識していること。

コンピュータノードのホスト名を確認するには、以下のコマンドを実行します。

```
heat-admin@compute-n # sudo openstack-config --get /etc/nova/nova.conf  
DEFAULT host
```

2.1. 共有ストレージの例外

通常、インスタンスに高可用性を設定するには、共有ストレージが必要です。**no-shared-storage** のオプションの使用を試みると、退避中に **InvalidSharedStorage** エラーが表示されて、別のノードではインスタンスの電源がオンにならない可能性が高くなります。ただし、全インスタンスが Block Storage (**cinder**) ボリュームから起動するように設定されている場合には、インスタンスのディスクイメージを保管するための共有ストレージは必要ないため、**no-shared-storage** オプションを使用して全インスタンスを退避することができます。

退避中に、インスタンスが Block Storage ボリュームから起動するように設定されている場合には、退避されるインスタンスはいずれも、同じボリュームから起動することが予想されますが、別のコンピュータノード上で起動することになります。そのため、OS イメージとアプリケーションデータはその Block Storage ボリューム上に保持されているので、退避されたインスタンスは、ジョブを即時に再起動することができます。

第3章 インストール

1. コンピュートノード上の **libvirt** および全 OpenStack サービスを無効にすることから開始します。

```
heat-admin@compute-n # sudo openstack-service stop
heat-admin@compute-n # sudo openstack-service disable
heat-admin@compute-n # sudo systemctl stop libvirt
heat-admin@compute-n # sudo systemctl disable libvirt
```

2. **pacemaker-remote** に使用するための認証キーを作成します。
コンピュートノードで以下のステップを実行します。

```
heat-admin@compute-1 # sudo mkdir -p /etc/pacemaker/
heat-admin@compute-1 # sudo dd if=/dev/urandom of=/etc/pacemaker/authkey
bs=4096 count=1
heat-admin@compute-1 # sudo cp /etc/pacemaker/authkey ./
heat-admin@compute-1 # sudo chown heat-admin:heat-admin authkey
```

3. このキーを director ノードにコピーしてから、残りのコンピュートノードおよびコントローラーノードにコピーします。

```
stack@director # scp authkey heat-admin@node-n:~/
heat-admin@node-n # sudo mkdir -p --mode=0750 /etc/pacemaker
heat-admin@node-n # sudo chgrp haclient /etc/pacemaker
heat-admin@node-n # sudo chown root:haclient /etc/pacemaker/authkey
```

4. 全コンピュートノードで **pacemaker-remote** を有効化します。

```
heat-admin@compute-n # sudo systemctl enable pacemaker_remote
heat-admin@compute-n # sudo systemctl start pacemaker_remote
```

5. 必要とされるバージョンの **pacemaker (1.1.12-22.el7_1.4.x86_64)** および **resource-agents (3.9.5-40.el7_1.5.x86_64)** パッケージがコントローラーノードとコンピュートノードにインストールされていることを確認します。

```
heat-admin@controller-n # sudo rpm -qa | egrep '(pacemaker|resource-agents)'
```

6. [BZ#1257414](#) に必要とされている、制約の回避策を適用します。



注記

この問題は、[RHSA-2015:1862](#) で対処されており、お使いの環境では必要ない可能性があります。

```
heat-admin@controller-1 # sudo pcs constraint order start openstack-nova-novncproxy-clone then openstack-nova-api-clone
heat-admin@controller-1 # sudo pcs constraint order start rabbitmq-clone then openstack-keystone-clone
heat-admin@controller-1 # sudo pcs constraint order promote galera-master then openstack-keystone-clone
heat-admin@controller-1 # sudo pcs constraint order start haproxy-clone then openstack-keystone-clone
```

```
heat-admin@controller-1 # sudo pcs constraint order start memcached-clone
then openstack-keystone-clone
heat-admin@controller-1 # sudo pcs constraint order promote redis-master
then start openstack-ceilometer-central-clone require-all=false
heat-admin@controller-1 # sudo pcs resource defaults resource-
stickiness=INFINITY
```

7. **overcloudrc** ファイルを使用して **NovaEvacuate** Active/Passive リソースを作成し、**auth_url**、**username**、**tenant**、**password** の値を指定します。

```
stack@director # scp overcloudrc heat-admin@controller-1:~/
heat-admin@controller-1 # . ~/overcloudrc
heat-admin@controller-1 # sudo pcs resource create nova-evacuate
ocf:openstack:NovaEvacuate auth_url=$OS_AUTH_URL username=$OS_USERNAME
password=$OS_PASSWORD tenant_name=$OS_TENANT_NAME \ ❶
```

❶ 共有ストレージを使用していない場合には、**no_shared_storage=1** オプションを追加してください。詳しい説明は、「[共有ストレージの例外](#)」を参照してください。

8. **nova-evacuate** が Floating IP リソース、Image サービス (glance)、OpenStack Networking サービス (neutron)、Compute サービス (nova) の後に起動されることを確認します。

```
heat-admin@controller-1 # for i in $(sudo pcs status | grep IP | awk '{
print $1 }\''); do sudo pcs constraint order start $i then nova-evacuate ;
done
heat-admin@controller-1 # for i in openstack-glance-api-clone neutron-
metadata-agent-clone openstack-nova-conductor-clone; do sudo pcs
constraint order start $i then nova-evacuate require-all=false ; done
```

9. コントロールプレーン全体ですべての OpenStack リソースを無効にします。

```
heat-admin@controller-1 # sudo pcs resource disable openstack-keystone --
wait=540
```

上記のコマンドのタイムアウト値 (**--wait=540**) は、例として使用しているだけです。Identity サービスを停止するのに必要な時間 (およびハードウェアの処理能力) に応じて、タイムアウト時間を増やすことを検討してください。

10. **cibadmin** データを使用して現在のコントローラー一覧を作成します。

```
heat-admin@controller-1 # controllers=$(sudo cibadmin -Q -o nodes | grep
uname | sed s/.*uname.// | awk -F\" '{print $1}')
heat-admin@controller-1 # echo $controllers
```

11. **osprole=controller** プロパティーでこれらのノードをコントローラーとしてタグ付けします。

```
heat-admin@controller-1 # for controller in ${controllers}; do sudo pcs
property set --node ${controller} osprole=controller ; done
```

12. 環境内にすでに存在する **stonith** デバイスの一覧を作成します。

```
heat-admin@controller-1 # stonithdevs=$(sudo pcs stonith | awk '{print
$1}')
```

```
heat-admin@controller-1 # echo $stonithdevs
```

13. コントロールプレーンサービスをタグ付けし、一覧内の stonith デバイスをスキップして、上記で特定したコントローラーのみで実行されるようにします。

```
heat-admin@controller-1 # for i in $(sudo cibadmin -Q --xpath //primitive
--node-path | tr ' ' '\n' | awk -F "id='" '{print $2}' | awk -F '"'
'{print $1}' | uniq); do
    found=0
    if [ -n "$stonithdevs" ]; then
        for x in $stonithdevs; do
            if [ $x = $i ]; then
                found=1
            fi
        done
    fi
    if [ $found = 0 ]; then
        sudo pcs constraint location $i rule resource-discovery=exclusive
        score=0 osprole eq controller
    fi
done
```

14. **pacemaker** 内にコンピュータノードのリソースの追加を開始します。最初に **neutron-openvswitch-agent** を追加します。

```
heat-admin@controller-1 # sudo pcs resource create neutron-openvswitch-
agent-compute systemd:neutron-openvswitch-agent op start timeout 200s stop
timeout 200s --clone interleave=true --disabled --force
heat-admin@controller-1 # sudo pcs constraint location neutron-
openvswitch-agent-compute-clone rule resource-discovery=exclusive score=0
osprole eq compute
heat-admin@controller-1 # sudo pcs constraint order start neutron-server-
clone then neutron-openvswitch-agent-compute-clone require-all=false
```

次に Compute の **libvirtd** リソースを追加します。

```
heat-admin@controller-1 # sudo pcs resource create libvirtd-compute
systemd:libvirtd op start timeout 200s stop timeout 200s --clone
interleave=true --disabled --force
heat-admin@controller-1 # sudo pcs constraint location libvirtd-compute-
clone rule resource-discovery=exclusive score=0 osprole eq compute
heat-admin@controller-1 # sudo pcs constraint order start neutron-
openvswitch-agent-compute-clone then libvirtd-compute-clone
heat-admin@controller-1 # sudo pcs constraint colocation add libvirtd-
compute-clone with neutron-openvswitch-agent-compute-clone
```

次に **openstack-ceilometer-compute** リソースを追加します。

```
heat-admin@controller-1 # sudo pcs resource create ceilometer-compute
systemd:openstack-ceilometer-compute op start timeout 200s stop timeout
200s --clone interleave=true --disabled --force
heat-admin@controller-1 # sudo pcs constraint location ceilometer-compute-
clone rule resource-discovery=exclusive score=0 osprole eq compute
heat-admin@controller-1 # sudo pcs constraint order start openstack-
ceilometer-notification-clone then ceilometer-compute-clone require-
```

```
all=false
heat-admin@controller-1 # sudo pcs constraint order start libvirtd-
compute-clone then ceilometer-compute-clone
heat-admin@controller-1 # sudo pcs constraint colocation add ceilometer-
compute-clone with libvirtd-compute-clone
```

次に **nova-compute** リソースを追加します。

```
heat-admin@controller-1 # . /home/heat-admin/overcloudrc
heat-admin@controller-1 # sudo pcs resource create nova-compute-
checkevacuate ocf:openstack:nova-compute-wait auth_url=$OS_AUTH_URL
username=$OS_USERNAME password=$OS_PASSWORD tenant_name=$OS_TENANT_NAME
domain=localdomain op start timeout=300 --clone interleave=true --disabled
--force
heat-admin@controller-1 # sudo pcs constraint location nova-compute-
checkevacuate-clone rule resource-discovery=exclusive score=0 osprole eq
compute
heat-admin@controller-1 # sudo pcs constraint order start openstack-nova-
conductor-clone then nova-compute-checkevacuate-clone require-all=false
heat-admin@controller-1 # sudo pcs resource create nova-compute
systemd:openstack-nova-compute --clone interleave=true --disabled --force
heat-admin@controller-1 # sudo pcs constraint location nova-compute-clone
rule resource-discovery=exclusive score=0 osprole eq compute
heat-admin@controller-1 # sudo pcs constraint order start nova-compute-
checkevacuate-clone then nova-compute-clone require-all=true
heat-admin@controller-1 # sudo pcs constraint order start nova-compute-
clone then nova-evacuate require-all=false
heat-admin@controller-1 # sudo pcs constraint order start libvirtd-
compute-clone then nova-compute-clone
heat-admin@controller-1 # sudo pcs constraint colocation add nova-compute-
clone with libvirtd-compute-clone
```

15. コンピュートノード用に stonith デバイスを追加します。各コンピュータノードで以下のコマンドを実行します。

```
heat-admin@controller-1 # sudo pcs stonith create ipmilan-overcloud-
compute-N fence_ipmilan pcmk_host_list=overcloud-compute-0
ipaddr=10.35.160.78 login=IPMILANUSER passwd=IPMILANPW lanplus=1 cipher=1
op monitor interval=60s;
```

ここで、

- **N** は各コンピュータノードを識別するための番号に置き換えます (例: **ipmilan-overcloud-compute-1**、**ipmilan-overcloud-compute-2** など)。
- **IPMILANUSER** および **IPMILANPW** は、IPMI デバイスのユーザー名とパスワードに置き換えます。

16. 別の **fence-nova** stonith デバイスを作成します。

```
heat-admin@controller-1 # . overcloudrc
heat-admin@controller-1 # sudo pcs stonith create fence-nova fence_compute
\
                                auth-url=$OS_AUTH_URL \
                                login=$OS_USERNAME \
```

```
passwd=$OS_PASSWORD \
tenant-name=$OS_TENANT_NAME \
record-only=1 action=off --force
```

17. コンピュートノードがフェンシング後に回復できるようにします。

```
heat-admin@controller-1 # sudo pcs property set cluster-recheck-
interval=1min
```

18. コンピュートノードのリソースを作成して、stonith **level 1** にノードの物理フェンスデバイスと **fence-nova** の両方が含まれるように設定します。各コンピュートノードに対して以下のコマンドを実行します。

```
heat-admin@controller-1 # sudo pcs resource create overcloud-compute-N
ocf:pacemaker:remote reconnect_interval=60 op monitor interval=20
heat-admin@controller-1 # sudo pcs property set --node overcloud-compute-N
osprole=compute
heat-admin@controller-1 # sudo pcs stonith level add 1 overcloud-compute-N
ipmilan-overcloud-compute-N,fence-nova
heat-admin@controller-1 # sudo pcs stonith
```

N は、各コンピュートノードを識別するための番号に置き換えます (例: **overcloud-compute-1**、**overcloud-compute-2** など)。これらの識別番号を使用して、前のステップで作成した stonith デバイスに各コンピュートノードを照合します (例: **overcloud-compute-1** および **ipmilan-overcloud-compute-1**)。

19. コントロールおよびコンピュートプレーンサービスを有効にします。

```
heat-admin@controller-1 # sudo pcs resource enable openstack-keystone
heat-admin@controller-1 # sudo pcs resource enable neutron-openvswitch-
agent-compute
heat-admin@controller-1 # sudo pcs resource enable libvirtd-compute
heat-admin@controller-1 # sudo pcs resource enable ceilometer-compute
heat-admin@controller-1 # sudo pcs resource enable nova-compute-
checkevacuate
heat-admin@controller-1 # sudo pcs resource enable nova-compute
```

20. 環境が安定するまでしばらく待ってから、失敗したリソースをクリーンアップします。

```
heat-admin@controller-1 # sleep 60
heat-admin@controller-1 # sudo pcs resource cleanup
heat-admin@controller-1 # sudo pcs status
heat-admin@controller-1 # sudo pcs property set stonith-enabled=true
```

第4章 高可用性のテスト



注記

以下の手順を実行すると、コンピュートノードは警告なしで意図的に再起動されます。

1. オーバークラウド上でインスタンスを1つまたは複数起動し、それらのインスタンスをホストしているコンピュートノードをクラッシュさせます。

```
stack@director # . overcloudrc
stack@director # nova boot --image cirros --flavor 2 test-failover
stack@director # nova list --fields name,status,host
stack@director # . stackrc
stack@director # ssh -lheat-admin compute-n
heat-admin@compute-n # echo c > /proc/sysrq-trigger
```

2. しばらくすると、それらのインスタンスはオンラインのコンピュートノードで再起動されるはずです。

```
stack@director # nova list --fields name,status,host
stack@director # nova service-list
```

第5章 参考資料

- <http://blog.clusterlabs.org/blog/2015/openstack-ha-compute/>
- <https://github.com/beekhof/osp-ha-deploy/blob/master/pcmk/compute-managed.scenario>
- <https://github.com/beekhof/osp-ha-deploy/blob/master/pcmk/controller-managed.scenario>