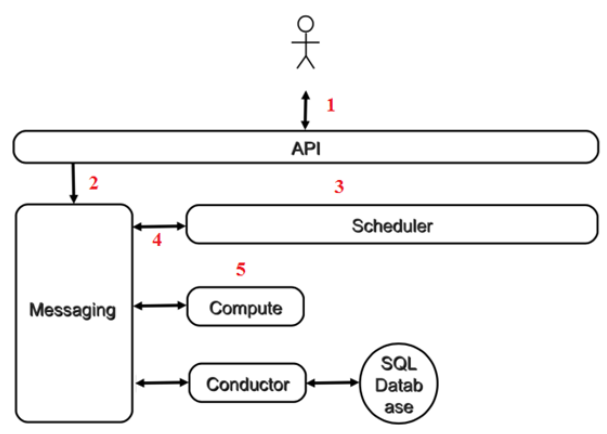


Migrate 操作的作用是将 instance 从当前的计算节点迁移到其他节点上。

Migrate 不要求源和目标节点必须共享存储，当然共享存储也是可以的。 Migrate 前必须满足一个条件：计算节点间需要配置 nova 用户无密码访问。

下面是 Migrate instance 的流程图

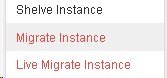


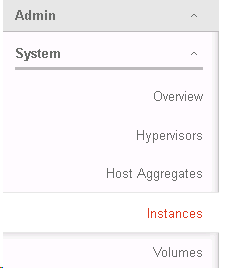
1. 向 nova-api 发送请求
2. nova-api 发送消息
3. nova-scheduler 执行调度
4. nova-scheduler 发送消息
5. nova-compute 执行操作

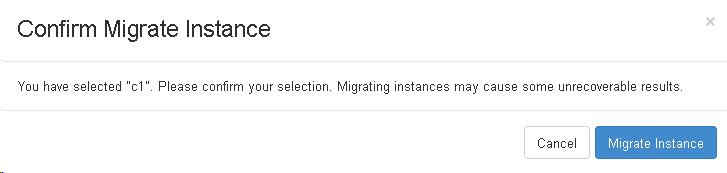
下面我们详细讨论每一个步骤。

**向 nova-api 发送请求**

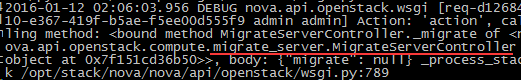
客户（可以是 OpenStack 最终用户，也可以是其他程序）向 API（nova-api）发送请求：“帮我迁移这个 Instance” Migrate 操作是特权操作，只能在 Admin 的 instance 菜单中执行







查看日志 /opt/stack/logs/n-api.log

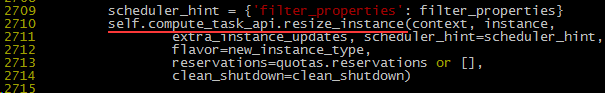


**nova-api 发送消息**

nova-api 向 Messaging（RabbitMQ）发送了一条消息：“迁移这个 Instance”

查看源代码 /opt/stack/nova/nova/compute/api.py，方法是 resize。

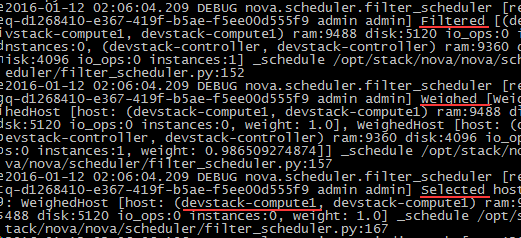
**没错，是 resize 而非 migrate。**这是由于 migrate 实际上是通过 resize 操作实现的，至于为什么要这样设计，我们会在下一节 resize 中详细分析。



**nova-scheduler 执行调度**

nova-scheduler 收到消息后，会为 instance 选择合适的目标计算节点。

查看日志 /opt/stack/logs/n-sch.log



可以看到，因为 devstack-compute1 的权值比 devstack-controller 大，最终选择 devstack-compute1 作为目标节点。

看到上面的日志，大家发现什么问题没有？

在分析这段日志的时候，我发现 scheduler 选出来的计算节点有可能是当前节点源节点！

因为 scheduler 并没在初始的时候将源节点剔除掉，而是与其他节点放在一起做 filter，按照这个逻辑，只要源节点的权值足够大，是有可能成为目标节点的。

那紧接着的问题是：如果源节点和目标节点是同一个，migrate 操作会怎样进行呢？

实验得知，nova-compute 在做 migrate 的时候会检查目标节点，如果发现目标节点与源节点相同，会抛出 UnableToMigrateToSelf 异常。Nova-compute 失败之后，scheduler 会重新调度，由于有 RetryFilter，会将之前选择的源节点过滤掉，这样就能选到不同的计算节点了。

关于 RetryFilter，大家还有印象吗？如果生疏了可以看前面章节。

好，言归正传。在上面的操作中 sheduler 选择的目标节点是 devstack-compute1，意味着 instance 将从 devstack-controller 迁移到 devstack-compute1。

**nova-scheduler 发送消息**

nova-scheduler 发送消息，通知计算节点可以迁移 instance 了。

源代码在 /opt/stack/nova/nova/scheduler/filter\_scheduler.py 第 95 行，方法为 select\_destinations

image254.png

**nova-compute 执行操作**

nova-compute 会在源计算节点和目标计算节点上分别执行操作。

**源计算节点 devstack-controller**

迁移操作在源节点上首先会关闭 instance，然后将 instance 的镜像文件传到目标节点上。

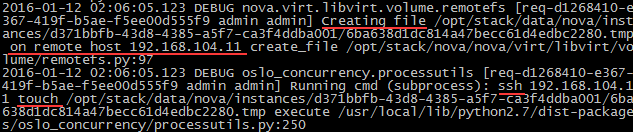
日志在 /opt/stack/logs/n-cpu.log，具体步骤如下：

**开始 migrate**

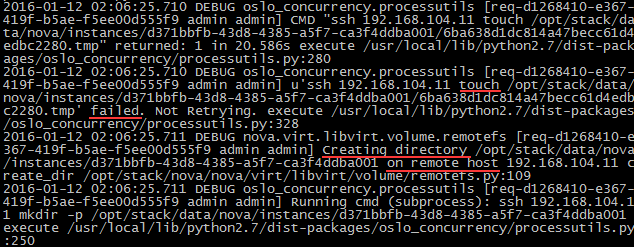
http://7xo6kd.com1.z0.glb.clouddn.com/upload-ueditor-image-20160528-1464447097395055198.jpg

**在目标节点上创建 instance 的目录**

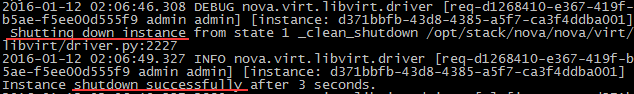
nova-compute 首先会尝试通过 ssh 在目标节点上的 instance 目录里 touch 一个临时文件，日志如下



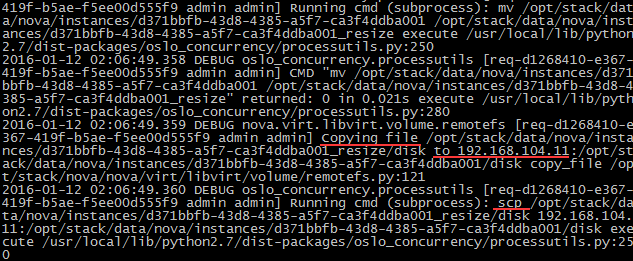
如果 touch 失败，说明目标节点上还没有该 instance 的目录，也就是说，源节点和目标节点没有共享存储。那么接下来就要在目标节点上创建 instance 的目录，日志如下



**关闭 instance**



**将 instance 的镜像文件通过 scp 传到目标节点上**



**目标计算节点 devstack-compute1**

在目标节点上启动 instance，过程与 launch instance 非常类似。

会经过如下几个步骤：

1. 为 instance 准备 CPU、内存和磁盘资源

2. 创建 instance 镜像文件

3. 创建 instance 的 XML 定义文件

4. 创建虚拟网络并启动 instance

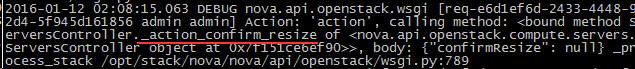
日志记录在 /opt/stack/logs/n-cpu.log，分析留给大家练习。

**Confirm**

这时，instance 会处于 “Confirm or Revert Resize/Migrate”状态，需要用户确认或者回退当前的迁移操作，实际上给了用户一个反悔的机会。

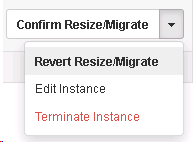
http://7xo6kd.com1.z0.glb.clouddn.com/upload-ueditor-image-20160528-1464447098328016910.jpg

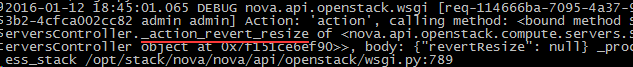
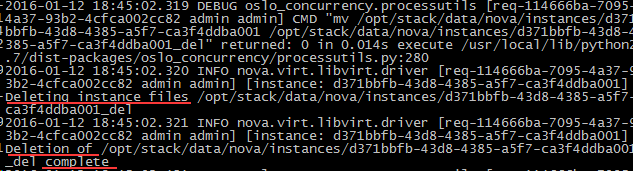
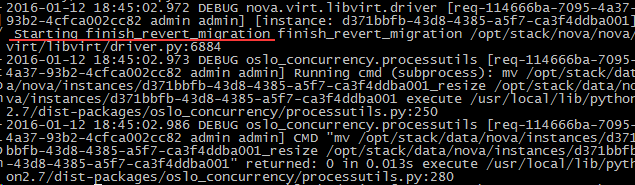
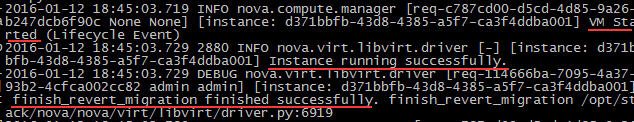
当我们按下 Confirm 按钮后，会发生如下事情：

1. nova-api 接收到 confirm 的消息  
   
2. 源计算节点删除 instance 的目录，并在 Hypervisor 上删除 instance。  
   http://7xo6kd.com1.z0.glb.clouddn.com/upload-ueditor-image-20160528-1464447098678033862.jpg  
     
   http://7xo6kd.com1.z0.glb.clouddn.com/upload-ueditor-image-20160528-1464447098788085342.jpg
3. 目标计算节点不需要做任何事情

**Revert**

如果执行的是 Revert 操作会发生什么事情呢？



1. nova-api 接收到 revert 的消息  
   
2. 在目标计算节点上关闭 instance，删除 instance 的目录，并在 Hypervisor 上删除 instance。  
   http://7xo6kd.com1.z0.glb.clouddn.com/upload-ueditor-image-20160528-1464447099273097687.jpg
3. 源计算节点上启动 instance
4. 因为之前迁移的时候只是在源节点上关闭了该 instance，revert 操作只需重新启动 instance。  
     
     
   

以上是 Migrate 操作的完整流程，这里有一点需要特别注意：

迁移过程中源和目标节点之前需要使用 ssh 和 scp，为了使操作顺利进行，必须要保证 nova-compute 进程的启动用户（通常是 nova，也可能是 root，可以通过 ps 命令确认）能够在计算节点之间无密码访问。否则 nova-compute 会等待密码输入，但后台服务是无法输入密码的，迁移操作会一直卡在那里。  
  
以上是 Migrate 操作的详细分析，下一节我们讨论 Resize。