# 目 录

## 1 分析 resize 流程前的必要知识

### 1.1 nova 中的 RPC 机制

### 1.2 重要的数据类型

#### 1.2.1 req

#### 1.2.2 context

```
# 根据req创建环境上下文context
# context是nova/context.py中的RequestContext类
context = req.environ["nova.context"]
```

#### 1.2.3 instance

```
# 根据req和instance_id创建instance
# instance是nova/context/instance.py中的Instance类
instance = self._get_server(context, req, instance_id)


instance_type
flavor_id
deltas
quotas
vm_state
```

## 2 nova-api 阶段

入口函数为:

```
# 这个函数在nova/api/openstack/compute/servers.py
def _resize(self, req, instance_id, flavor_id, **kwargs):
    ...
    try:
        # compute_api是nova/compute/api.py中的API类
        self.compute_api.resize(context, instance, flavor_id, **kwargs)
    ...
```

进一步看 API.resize() 函数:

```
# nova/compute/api.py API.resize()
def resize(self, context, instance, flavor_id=None,
           **extra_instance_updates):
```

```
    ...

    # filter_properties与选择本地扩容或选择异地扩容有关
    filter_properties = {'ignore_hosts': []}

    if not CONF.allow_resize_to_same_host:
        filter_properties['ignore_hosts'].append(instance['host'])

    if (not flavor_id and not CONF.allow_migrate_to_same_host):
        filter_properties['ignore_hosts'].append(instance['host'])

    ...

    # scheduler_hint挺重要的，是nova-scheduler的参数
    scheduler_hint = {'filter_properties': filter_properties}
    self.compute_task_api.resize_instance(context, instance,
            extra_instance_updates, scheduler_hint=scheduler_hint,
            flavor=new_instance_type,
            reservations=quotas.reservations or [])
```

```
# nova/conductor/api.py ComputeTaskAPI.resize_instance()
def resize_instance(self, context, instance, extra_instance_updates,
                    scheduler_hint, flavor, reservations):
    self.conductor_compute_rpcapi.migrate_server(
        context, instance, scheduler_hint, False, False, flavor,
        None, None, reservations)
```

```
# nova/conductor/rpcapi.py ComputeTaskAPI.migrate_server()
def migrate_server(self, context, instance, scheduler_hint, live, rebuild,
                   flavor, block_migration, disk_over_commit,
                   reservations=None):
    ...
    cctxt = self.client.prepare(version=version)
    return cctxt.call(context, 'migrate_server',
                      instance=instance, scheduler_hint=scheduler_hint,
                      live=live, rebuild=rebuild, flavor=flavor_p,
                      block_migration=block_migration,
                      disk_over_commit=disk_over_commit,
                      reservations=reservations)
```

## 3 nova-conductor 部分

```
# nova/conductor/manager.py ComputeTaskManager.migrate_server()
def migrate_server(self, context, instance, scheduler_hint, live, rebuild,
        flavor, block_migration, disk_over_commit, reservations=None):
    ...
    if live and not rebuild and not flavor:
        self._live_migrate(context, instance, scheduler_hint,
                           block_migration, disk_over_commit)
    elif not live and not rebuild and flavor:
        ...
        with compute_utils.EventReporter(context, 'cold_migrate',
                                         instance_uuid):
            self._cold_migrate(context, instance, flavor,
                               scheduler_hint['filter_properties'],
                               reservations)
    ...
```

## 4 冷迁移

### 4.1 冷迁移中的 nova-conductor 部分

```
    # nova/conductor/manager.py ComputeTaskManager._cold_migrate()
    def _cold_migrate(self, context, instance, flavor, filter_properties,
                      reservations):
        ...
        try:
            ...
            # 选择目的主机
            hosts = self.scheduler_client.select_destinations(
                    context, request_spec, filter_properties)
            host_state = hosts[0]
        ...

        try:
            ...
            (host, node) = (host_state['host'], host_state['nodename'])
            self.compute_rpcapi.prep_resize(
                context, image, instance,
                flavor, host,
                reservations, request_spec=request_spec,
                filter_properties=filter_properties, node=node)
        ...
```

```
    # nova/compute/rpcapi.py ComputeAPI.prep_resize()
    def prep_resize(self, ctxt, image, instance, instance_type, host,
                    reservations=None, request_spec=None,
                    filter_properties=None, node=None):
        ...
        cctxt = self.client.prepare(server=host, version=version)
        cctxt.cast(ctxt, 'prep_resize',
                   instance=instance,
                   instance_type=instance_type_p,
                   image=image_p, reservations=reservations,
                   request_spec=request_spec,
                   filter_properties=filter_properties,
                   node=node)
```

## 4.2 冷迁移中的 nova-compute 部分

### 4.2.1 目的主机上的操作：prep_resize

```
    # nova/compute/manager.py ComputeManager.prep_resize()
    def prep_resize(self, context, image, instance, instance_type,
                    reservations, request_spec, filter_properties, node):
        ...
        with self._error_out_instance_on_exception(context, instance,
                                                   quotas=quotas):
            ...
            try:
                self._prep_resize(context, image, instance,
                                  instance_type, quotas,
                                  request_spec, filter_properties,
                                  node)
            ...
```

```
    # nova/compute/manager.py ComputeManager._prep_resize()
    def _prep_resize(self, context, image, instance, instance_type,
            quotas, request_spec, filter_properties, node):

        ...
        with rt.resize_claim(context, instance, instance_type,
                             image_meta=image, limits=limits) as claim:
            ...
            self.compute_rpcapi.resize_instance(
                    context, instance, claim.migration, image,
                    instance_type, quotas.reservations)
```

```
    # nova/compute/rpcapi.py ComputeAPI.resize_instance()
    def resize_instance(self, ctxt, instance, migration, image, instance_type,
                        reservations=None):
        ...
        cctxt = self.client.prepare(server=_compute_host(None, instance),
                version=version)
        cctxt.cast(ctxt, 'resize_instance',
                   instance=instance, migration=migration,
                   image=image, reservations=reservations,
                   instance_type=instance_type_p)
```

### 4.2.2 源主机的操作：resize_instance

```
    # nova/compute/manager.py ComputeManager.resize_instance()
    def resize_instance(self, context, instance, image,
                        reservations, migration, instance_type,
                        clean_shutdown=True):
        ...
        with self._error_out_instance_on_exception(context, instance,
                                                   quotas=quotas):
```

```
            ...
            # 获得虚拟机块设备的信息
            block_device_info = self._get_instance_block_device_info(
                                context, instance, bdms=bdms)

            # 关闭虚拟机并迁移虚拟机的增量文件
            disk_info = self.driver.migrate_disk_and_power_off(
                    context, instance, migration.dest_host,
                    instance_type, network_info,
                    block_device_info,
                    timeout, retry_interval)
            ...
            self.compute_rpcapi.finish_resize(context, instance,
                    migration, image, disk_info,
                    migration.dest_compute, reservations=quotas.reservations)
            ...
```

migrate_disk_and_power_off() 是源主机上将虚拟机迁移给目的主机的实现函数，主要利用了 libvirt API。这个函数的分析在《nova 调用 libvirt》中的“nova 扩容时对 libvirt 的调用”一节。

```
# nova/compute/rpcapi.py(690) ComputeAPI.finish_resize()
def finish_resize(self, ctxt, instance, migration, image, disk_info,
        host, reservations=None):
    ...
    cctxt = self.client.prepare(server=host, version=version)
    cctxt.cast(ctxt, 'finish_resize',
               instance=instance, migration=migration,
               image=image, disk_info=disk_info, reservations=reservations)
```

### 4.2.3 目的主机上的操作：finish_resize

```
# nova/compute/manager.py ComputeManager.finish_resize()
def finish_resize(self, context, disk_info, image, instance,
                  reservations, migration):
    quotas = quotas_obj.Quotas.from_reservations(context,
                                                 reservations,
                                                 instance=instance)
    try:
        self._finish_resize(context, instance, migration,
                            disk_info, image)
    ...
```

```
# nova/compute/manager.py ComputeManager._finish_resize()
def _finish_resize(self, context, instance, migration, disk_info,
                   image):
    ...
    try:
        self.driver.finish_migration(context, migration, instance,
                                     disk_info,
                                     network_info,
                                     image ,  resize_instance ,
                                     block_device_info ,  power_on)
    ...
```