背景

1. cinder从镜像创建卷时，即使镜像为Raw/Bare格式，也会偶现cinder下载镜像到本地，而不是在后端直接拷贝！
2. Cinder后端配置了两个Volume

从镜像创建卷流程分析：

实际镜像创建流程在CreateVolumeFromSpecTask任务中，开始创建前会输创建日志：

```
LOG.info(_LI("Volume %(volume_id)s: being created as %(create_type)s "
             "with specification: %(volume_spec)s"),
         {'volume_spec': volume_spec, 'volume_id': volume_id,
          'create_type': create_type})
```

该条日志的输出信息对定位问题有很大帮助：

volume_spec　　：包含将要拷贝的源信息、目标信息

volume_id　　：目标卷ID

create_type　　：创建方式

典型的日志输出如下：

```
Volume24404271-dfe0-40d7-8154-f21efba4e733: being created as image
with specification:
{
  'status': u'creating',
  'image_location': (None,[{u'url': u'cinder://0df99357-9e1f-4462-bf30-aabfd3f01f40',u'metadata': {}}]),
  'volume_size': 38,
  'volume_name': 'volume-24404271-dfe0-40d7-8154-f21efba4e733',
  'image_id': 'aeb33544-cdc8-42c8-9102-5023745e6a23',
  'image_service': <cinder.image.glance.GlanceImageServiceobjectat0x6137c10>,
  'image_meta': {
    u'status': u'active',
    u'file': u'/v2/images/aeb33544-cdc8-42c8-9102-5023745e6a23/file',
    u'name': u'release-ubu14_tw_from_vhdx_raw',
    u'tags': [],
    u'container_format': u'bare',
    u'created_at': datetime.datetime(2017,5,11,3,12,55,tzinfo=<iso8601.Utc>),
    u'disk_format': u'raw',
    u'updated_at': datetime.datetime(2017,5,12,3,41,45,tzinfo=<iso8601.Utc>),
    u'visibility': u'public',
    u'locations': [{ u'url': u'cinder://0df99357-9e1f-4462-bf30-aabfd3f01f40',u'metadata': {}}],
    u'owner': u'406cd353135e44f0ade98f53d92d5d8b',
    u'protected': False,
    u'id': u'aeb33544-cdc8-42c8-9102-5023745e6a23',
    u'min_ram': 0,
    u'checksum': u'de6479f0dd69e4de3a2a4b1477a0fb42',
    u'min_disk': 35,
    u'virtual_size': None,
    'properties': { },
    u'size': 37579915264
  }
}
```

CreateVolumeFromSpecTask会根据不同的create_type类型，选择创建时的分支方法：

　　_create_raw_volume

_create_from_snapshot

_create_from_source_volume

_create_from_source_replica

_create_from_image

关注从镜像创建卷时的分支方法：cinder.volume.flows.manager.CreateVolumeFromSpecTask._create_from_image

方法的流程：

1.打印拷贝时的源和目的信息：

```
LOG.debug("Cloning %(volume_id)s from image %(image_id)s "
          " at location %(image_location)s.",
          {'volume_id': volume.id,
           'image_location': image_location, 'image_id': image_id})
```

2.检查CONF.image_conversion_dir的剩余空间是否足够，包括image_meta['size']和image_meta['virtual_size']

3.判断目标卷是否加密（volume的encryption_key_id字段），如果是执行self.driver.clone_image！说明：加密卷的复制需要后端来处理！！

4.如果3无法Clone，判断CONF.allowed_direct_url_schemes是否包含cinder，执行self._clone_image_volume

5.如果4无法成功，执行self._create_from_image_cache_or_download从缓存clone，或者通过glance下载镜像

6.执行self._handle_bootable_volume_glance_meta拷贝一些启动信息，和meta信息到目标卷

分析上述流程中的4，该步骤直接在后端进行LUN的拷贝，镜像数据不过volume节点

关注从镜像创建卷时的分支方法：cinder.volume.flows.manager.CreateVolumeFromSpecTask._clone_image_volume

1.image_location是否为空

2.container_format=bare 以及 disk_format=raw

3.在目标volume的HOST下查询image_volume_ids，如果没有查到，直接返回！进行对卷的所有者进行检查

4.调用self.driver.create_cloned_volume进行拷贝

问题根因分析：

当前问题的根本原因是：从raw格式镜像创建卷时走下载流程，因此下载超大镜像时可能造成Nova端判断块设备映射超时（超时时间是180s，在nova侧通

问题的根本原因是：_clone_image_volume方法中，下面这条语句没有返回值。

这里volume_get_all_by_host方法的入参是volume['host']，格式为back_end_host@back_end_name#volume_path，方法要求volume的host字段等于输入

显然当输入的image_volume_ids的location不在指定的volume['host']时方法没有返回！

```
image_volumes = self.db.volume_get_all_by_host(
    context, volume['host'], filters={'id': image_volume_ids})
```

解决方案：

利用glance的multiple location特性！

注意到上的image_volume_ids中不止可以指定一个id，也就是说image的**locations**属性中应该可以指定多个地址，对**cinder**来说就是多个**LUN**，如果对一个

那么问题应该可以解决

```
direct_url, locations = image_location
urls = set([direct_url] + [loc.get('url') for loc in locations or []])
image_volume_ids = [url[9:] for url in urls
                    if url and url.startswith('cinder://')]
```

实施方案：

glance_api配置show_multiple_locations

cinder拷贝对应卷

将拷贝加入image的location：使用glance location-add

手工操作步骤：

#注意创建时添加的metadata需要参考原镜像卷，下面列出的都是必须项

cinder create --name <target-volid> --source-volid <source-id> --metadata image_owner=<image_owner> readonly=True image_size=<image-s

cinder migrate <target-volid> <target-pool>

glance location-add <image-id> --url cinder://<target-volid>

代码实现：

方案一：在glance侧实现

glance上传镜像时，调用cinder get-pools在所有pool创建镜像卷，之后添加到glance的location

方案二：cinder侧上传Image实现

cinder判断这是glance创建镜像的请求，在所有pool创建镜像卷，之后调用glance location-add

方案三：cinder创建卷时实现

1.同一个镜像只在一个pool克隆卷；

check_max_pool_luns_threshold=True

考虑：卷是动态变化的，动态变化时需要复制镜像卷！

```
[root@node1 ~]# glance help location-add
usage: glance location-add --url <URL> [--metadata <STRING>] <IMAGE_ID>

Add a location (and related metadata) to an image.

Positional arguments:
  <IMAGE_ID>           ID of image to which the location is to be added.

Optional arguments:
  --url <URL>          URL of location to add.
  --metadata <STRING>  Metadata associated with the location. Must be a valid
                       JSON object (default: {})
```