nova-compute 虚机创建的入口函数 ：*nova.compute.manager.ComputeManager* 的 *_build_and_run_instance* 方法

该方法会执行虚机创建时的块设备映射，虚机孵化等诸多操作，并且方法会catch各种异常。_build_and_run_instance会对异常进行简单处理，最后向外抛出BuildAbortException或者RescheduledException，前者直接宣告创建虚机失败，后者会重新进行调度。

与块设备相关的信息封装在block_device_mapping变量中，_builde_and_run_instance会对block_device_mapping进行必要的类型检查，之后调用_build_resources执行块设备映射。

_build_resources先向neutron请求网络，之后向执行块设备映射，并打印如下日志：

```
LOG.debug('Start building block device mappings for instance.',instance=instance)
```

所以显然，对于块设备映射失败的虚机，定位问题时可以搜索下面的关键字，快速定位到块设备映射的开始位置：

[instance: <vm-id>] Start building block device mappings for instance.

_build_resources会调用_prep_block_device执行块设备映射。

```
def _prep_block_device(self, context, instance, bdms,
                       do_check_attach=True):
    """Set up the block device for an instance with error logging."""
    try:
        self._add_missing_dev_names(bdms, instance)
        block_device_info = driver.get_block_device_info(instance, bdms)

        mapping = driver.block_device_info_get_mapping(block_device_info)

        driver_block_device.attach_block_devices(
            mapping, context, instance, self.volume_api, self.driver,
            do_check_attach=do_check_attach,
            wait_func=self._await_block_device_map_created)

        self._block_device_info_to_legacy(block_device_info)
        return block_device_info

    except exception.OverQuota:
        msg = _LW('Failed to create block device for instance due to '
                  'being over volume resource quota')
        LOG.warning(msg, instance=instance)
        raise exception.VolumeLimitExceeded()

    except Exception:
        LOG.exception(_LE('Instance failed block device setup'),
                      instance=instance)
        raise exception.InvalidBDM()
```

*_prep_block_device*中 *driver_block_device.attach_block_devices*（*nova/nova/virt/block_device.py*）执行块设备映射。

在_log_and_attach方法中会打印卷的创建源，不同创建源对应着不同的bdm类型：

bdm的类型	对应卷类型	
DriverSnapshotBlockDevice	从快照启动卷（包括卷快照和磁盘快照）	
DriverImageBlockDevice	从镜像启动卷	
DriverBlankBlockDevice	创建卷	作为VM的附加数据卷

```
def attach_block_devices(block_device_mapping, *attach_args, **attach_kwargs):
    def _log_and_attach(bdm):
        instance = attach_args[1]
        if bdm.get('volume_id'):
            LOG.info(_LI('Booting with volume %(volume_id)s at '
                         '%(mountpoint)s'),
                     {'volume_id': bdm.volume_id,
                      'mountpoint': bdm['mount_device']},
                     instance=instance)
        elif bdm.get('snapshot_id'):
            LOG.info(_LI('Booting with volume snapshot %(snapshot_id)s at '
                         '%(mountpoint)s'),
                     {'snapshot_id': bdm.snapshot_id,
                      'mountpoint': bdm['mount_device']},
                     instance=instance)
        elif bdm.get('image_id'):
            LOG.info(_LI('Booting with volume-backed-image %(image_id)s at '
                         '%(mountpoint)s'),
                     {'image_id': bdm.image_id,
                      'mountpoint': bdm['mount_device']},
                     instance=instance)
        else:
            LOG.info(_LI('Booting with blank volume at %(mountpoint)s'),
                     {'mountpoint': bdm['mount_device']},
                     instance=instance)
        bdm.attach(*attach_args, **attach_kwargs)

    for device in block_device_mapping:
        _log_and_attach(device)
    return block_device_mapping
```

bdm的attach方法会调用cinder客户端发送创建卷的请求，并调用_await_block_device_map_created等待创建完成，当知晓卷创建完成后执行attach

_await_block_device_map_created等待卷创建完成

_await_block_device_map_created会对creating，或者downloading状态的卷，进行等待直到其为available，而发现卷为其他状态时，会直接抛出异常

_await_block_device_map_created等待的时间和参数CONF.block_device_allocate_retries（60）以及CONF.block_device_allocate_retries_interva

显然，发现块设备映射失败请第一时间搜索该方法打印的日志：Volume id: %(vol_id)s finished being created but its status is %(vol_status)s

```
def _await_block_device_map_created(self, context, vol_id):
    # TODO(yamahata): creating volume simultaneously
    #                 reduces creation time?
    # TODO(yamahata): eliminate dumb polling
    start = time.time()
    retries = CONF.block_device_allocate_retries
    if retries < 0:
        LOG.warning(_LW("Treating negative config value (%(retries)s) for "
                        "'block_device_retries' as 0."),
                    {'retries': retries})
    # (1) treat  negative config value as 0
    # (2) the configured value is 0, one attempt should be made
    # (3) the configured value is > 0, then the total number attempts
    #      is (retries + 1)
    attempts = 1
```

```
    if retries >= 1:
        attempts = retries + 1
    for attempt in range(1, attempts + 1):
        volume = self.volume_api.get(context, vol_id)
        volume_status = volume['status']
        if volume_status not in ['creating', 'downloading']:
            if volume_status == 'available':
                return attempt
            LOG.warning(_LW("Volume id: %(vol_id)s finished being "
                            "created but its status is %(vol_status)s."),
                        {'vol_id': vol_id,
                         'vol_status': volume_status})
            break
        greenthread.sleep(CONF.block_device_allocate_retries_interval)
    raise exception.VolumeNotCreated(volume_id=vol_id,
                                     seconds=int(time.time() - start),
                                     attempts=attempt,
                                     volume_status=volume_status)
```

_build_and_run_instance 方法捕获的异常列表：

异常类型	原因	其他
InstanceNotFound		
UnexpectedDeletingTaskStateError		
ComputeResourcesUnavailable		
BuildAbortException		
FixedIpLimitExceeded		
NoMoreNetworks		
NoMoreFixedIps		
VirtualInterfaceCreateException		
VirtualInterfaceMacAddressException		
FixedIpInvalidOnHost		
UnableToAutoAllocateNetwork		
FlavorDiskTooSmall		
FlavorMemoryTooSmall		
ImageNotActive		
ImageUnacceptable		
InvalidDiskInfo		
InvalidDiskFormat		
SignatureVerificationError		

VolumeEncryptionNotSupported		
InvalidInput		
Exception		