目录

实例创建

基础配置1

在选择配置过程中，用户可根据业务需求，选择使用3种套餐中的1种，详细配置可以参考价格介绍。

1. 用户登录云服务器控制台，
2. 点击**新建实例**,系列选择GPU，按业务需要选择GPU实例套餐。完成后单击**下一步：网络配置**。 在配置过程中，需要关注以下参数：
    - 计费方式： 根据需求选择**包年包月**、**按量付费（按日月结）**、或**按量付费**。
    - 数据中心： 只支持北京、上海、广州的VPC数据中心。
    - 镜像类型： 支持标准镜像、自定义镜像、共享镜像和镜像市场的镜像。
    - 系统盘： 选择北京，上海区GPU云服务器时，使用SSD云硬盘，并可调整系统盘大小。Windows镜像类型的系统盘可调整范围为50GB-500GB，Linux镜像类型的系统盘可调整范围为20GB~500GB（广州VPC数据中心仅支持本地SSD）。有关SSD云硬盘的收费标准，请参见云硬盘价格。
    - 数据盘： 如果需要增加云硬盘类型的数据盘，单击**添加数据盘**，选择云硬盘类型，并配置硬盘大小。每个云服务器实例最多可以配置8块云硬盘，每块云硬盘带有**随实例删除**选项，可根据实际需求选择是否随实例删除该云硬盘。

网络配置

1. 在**网络配置**页面，配置弹性IP，用户按业务需求选择**购买新的弹性IP**或**稍后购买**。 注意：弹性IP的购买数量必须与云服务器的数量保持一致。
2. 设置主网卡VPC、关联子网、IP地址、安全组等信息，并可根据需要添加辅助网卡。
3. 单击**下一步：系统配置** 。

系统配置

在**系统配置**页面，设置**服务器名称**和**登录方式**等参数。

完成购买

设置完基本信息后，点击**购买**，您就完成了GPU云服务器实例的创建。

注：您还需要完成订单确认和支付，最终才能获得GPU云服务器的管理和使用权限。

查看实例

操作步骤

1. 登录云服务器控制台。
2. 在**实例**页面， 可以对实例进行如下操作：
    - 查看实例详情：单击实例名称，进入“实例详情”页面，可查看实例的详细信息并对实例进行管理操作。
    - 过滤实例列表：单击列表右上方的“过滤列”图标，选择想要展示的表格列，包括实例属性项和标签项。
    - 导出实例：单击列表右上方的“导出”图标，将当前实例列表导出为 .csv 文件，便于在本地查看和统计。

修改实例名称

用户可根据实际需求更改实例名称。

操作步骤

1. 登录云服务器控制台。
2. 在实例列表中，鼠标指向想要修改的实例名称。实例名称旁边出现**编辑**图标。
3. 单击**编辑**图标，并输入新的实例名称。

开启实例

概述

GPU云服务器实例一经创建，将自动开启并进入“运行中”状态。在控制台，用户可以像操作本地服务器一样启动实例。

注意：只能对“已关闭”状态的实例进行启动操作。

操作步骤

1. 登录云服务器控制台。
2. 在实例列表中，选择一个或多个需要启动的实例。单击实例列表左上角"开启"按钮。
3. 确认信息并单击**确定**。

关闭实例

概述

用户如果想停止服务或完成某些需要关机的操作（如修改密码），可从控制台关闭实例。 控制台提供“关闭”和“强制关闭”功能。

- “关闭”功能效果相当于正常关闭本地计算机。
- “强制关闭”功能效果相当于关闭本地计算机的电源，仅在GPU云服务器无法正常关机时使用。

注意事项

- 只有处于“运行中”的实例才支持关闭操作。
- 强制关闭可能会导致GPU云服务器数据丢失或者文件系统损坏，需谨慎使用。
- 如果强制关闭失败，请联系客服人员。
- GPU云服务器所有套餐暂不支持“关机不收费”模式。

操作步骤

1. 登录云服务器控制台。
2. 在实例列表中，选择一个或多个需要关闭的实例，并单击**关闭**。
3. 在新窗口中，选择关闭方式：**关闭**或**强制关闭**。

4. 单击“确定”。

操作过程中需要注意如下情况：如果选择**强制关闭**，需要进一步勾选**确认要强制关闭**。

重启实例

概述

用户可以在控制台重启实例，效果相当于重启本地计算机的操作系统。如果GPU云服务器无法正常重启，可使用**强制重启**功能。

注意事项

- 只有处于“运行中”的实例才支持重启操作。如果实例处于“已关闭”状态，必须先开启实例才能重启。
- 重启操作会造成实例停止工作从而中断业务，需谨慎执行。

操作步骤

1. 登录云服务器控制台。
2. 在实例列表中，选择一个或多个需要重启的实例，并单击**重启**。
3. 在弹出窗口中，选择重启方式：**重启**或**强制重启**。 如果选择**强制重启**，需要勾选**确认要强制重启**。

4. 单击**确定**。

重装系统

概述

GPU云服务器实例支持重装系统，可用于故障恢复或者更换操作系统类型和版本。

注意：仅支持已关闭状态的GPU云服务器重装系统。

重装操作系统类型

所有地域和可用区的GPU云服务器实例均支持Windows、Linux操作系统的任意版本在控制台的重装。即重装系统不限制系统类型和版本。 Linux操作系统，Windows操作系统均免费赠送容量50G。

计费说明

如果不调整系统盘大小，重装系统不额外收取费用。

对于在重装时调整系统盘大小的服务器，在重装系统的同时对系统盘进行了扩容，将产生磁盘容量费用差额。关于此项的收费标准，请参考云硬盘价格。

操作步骤

1. 登录云服务器控制台。
2. 在实例列表中，选择一台需要重装系统的实例，并选择**更多操作** > **重装系统**。
3. 在**重装系统**页面，完成参数配置。 需要注意如下参数的配置：
   - **镜像类型**： 可选的镜像类型包括标准镜像、自定义镜像、共享镜像和镜像市场。其中，镜像市场提供的镜像需用户购买。如果对购买的镜像有任何疑问，需联系镜像提供商。
   - **系统盘**： 支持调整系统盘大小，新的系统盘容量不得低于原有系统盘容量。
   - **登录方式**： 可选的登录方式包括密码、密钥和保留镜像设置。其中，“标准镜像”类型不支持“保留镜像设置”登录方式。
   - **管理员密码/确认密码**： 如果选择“密码”登录方式，需要设置系统登录密码。具体格式要求为8-32个字符，必须包含大小写字母和数字，支持英文特殊字符!”$%()*+-/;:<=>?@[]^_`{|}~。
   - **SSH密钥**： 如果选择“密钥”登录方式，需要在SSH密钥列表中选择登录密钥，或单击“创建SSH密钥”创建新的密钥对。有关SSH密钥的详细步骤，请参考管理密钥对。
4. 单击**确定**。

重置密码

概述

用户可以根据需求重置实例密码。

注意：重置密码需在GPU云服务器关闭状态下执行。

操作步骤

1. 在实例列表中选择要重置密码的实例。
2. 在**操作**列选择 **更多** > **密钥密码** > **重置密码**。
3. 在弹出框中确认输入两次新密码。
4. 点击**确定**，重置密码生效。

设置弹性IP

概述

本篇介绍如何绑定和解绑弹性IP。

操作步骤

1. 登录云服务器控制台，默认进入**云服务器** > **实例**页面。
2. 点击实例名称/ID，选择**更多** >**网络/安全组** >**设置弹性IP**。
3. 在弹性IP列表中，选择想要绑定的弹性IP。如果想解绑弹性IP，单击已绑定的弹性IP旁边的删除图标“ X ”。

4. 单击**确定**。

设置网络配置

概述

用户可以为主网卡和辅网卡修改对应的内网IP地址。云服务器如果已经设置了弹性IP，必须先解绑弹性IP，否则无法修改内网IP地址。

操作步骤

1. 登录云服务器控制台，默认进入**云服务器** > **实例**页面。
2. 在目标实例对应的操作列中，选择**更多** > **网络/安全组** >**更改网络配置**。 配置如下参数：
   - 选择网卡：选择新的内网IP地址所对应的的网卡，选项包括主网卡和辅网卡。主网卡和辅网卡使用的详细信息可在弹性网卡概述中查找。
   - 关联VPC和关联子网：分别选择新的内网IP地址所属的VPC和子网
   - 内网IP：输入或者选择新的内网IP地址。
   - DNS1和DNS2：配置DNS地址。
3. 单击**下一步**。
4. 设置安全组，并单击**确定**。

删除实例

概述

用户可以根据需求随时删除不再使用的实例。

注意：计费方式为“包年包月”的实例不支持本节描述的操作。

GPU云服务器实例在被彻底删除时，相关的数据处理方式如下：

- 系统盘和本地数据盘上的全部数据同时被删除，并无法恢复。
- 挂载的云数据盘如果选择了**随实例删除**，则该云硬盘上的数据同时被删除。
- 挂载的云数据盘如果未选择**随实例删除**，则该云硬盘与云服务器解绑，数据无影响。

立即删除

计费方式为“按量付费（按日月结）”实例和“按量付费”的实例支持“立即删除”功能。

操作步骤

1. 登录云服务器控制台。
2. 在实例列表中，根据实际需求选择如下操作：
   - 删除单个实例：在列表中找到需要删除的实例，并选择对应操作列的**更多** > **实例状态** > **立即删除**。
   - 批量删除实例：在列表中选择多个需要删除的实例，并选择**更多操作** >**立即删除**。
3. 选择是否保留弹性IP，并单击**删除**。

注： 删除GPU云服务器现暂时不支持进入回收站。

定时删除

用户可以提前设定实例的删除时间，到达设定时间后，实例将自动删除。

需要注意的是：

- 只有“按量付费”与”按量付费（按日月结）“实例支持开启“定时删除”功能。
- 如果云服务器实例绑定了弹性IP，随着实例的自动删除，弹性IP将自动解绑并保留在弹性IP列表中，保留的弹性IP将继续产生费用，如后续无需使用，请及时删除。
- 在设置的删除时间未到达之前，用户可以随时调整定时删除时间或关闭“定时删除”功能。

为新建实例开启定时删除

1. 在云服务器控制台单击**新建实例**。
2. 完成**基础设置**、**网络设置**等相关参数的配置，下一步设置**系统配置**时，开启**定时删除**，并设置**删除时间**。

   设置的删除时间将至少晚于当前时间60分钟。

3. 设置定时删除时，可选择是否随主机定时删除所有绑定的弹性IP。

   选中后，当主机到达定时删除时间时，绑定的弹性IP也将随主机一起删除（计费类型为包年包月的弹性IP只解绑，不删除）

关于新建实例的详细步骤，请参考实例创建。

为已有实例开启定时删除

1. 登录云服务器控制台。
2. 在实例列表中，根据实际需求选择如下操作：
   - 为单个实例开启定时删除：在列表中找到目标实例，并选择对应操作列的**更多 > 实例状态 > 立即删除**。
   - 批量开启实例定时删除：在列表中选择多个目标实例，并选择**更多操作 > 定时删除**。
3. 在弹出页面中开启“定时删除”，设置删除时间，并单击**确定**。

   设置的删除时间要至少晚于当前时间30分钟。
4. 设置定时删除时，可选择是否随主机定时删除所有绑定的弹性IP。

   选中后，当主机到达定时删除时间时，绑定的弹性IP也将随主机一起删除（计费类型为包年包月的弹性IP只解绑，不删除）

实例启动模板

概述

实例启动模板可以存储除密码以外的任意配置信息，可以帮助您快速创建有相同配置的GPU云服务器实例。

使用限制

- 同一账号可以为每个地域最多创建50个实例启动模板。
- 实例启动模板中的弹性IP仅支持自动分配。

操作步骤

用户使用实例启动模板可参考云服务器（KEC）实例启动模板 您可按照以下步骤使用实例模板：

1. 创建实例启动模板
   - 您可以提前创建实例启动模板。
   - 您可以与实例同时创建实例启动模板。
2. 修改实例启动模板。
3. 删除实例启动模板。

实例分配至项目

概述

用户可以通过建立项目制，实现项目维度的批量实例管理。项目制详细信息参考项目管理。

相关操作

实例与项目分配相关操作可参考云服务器（KEC）实例分配至项目 相关操作如下：

1. 新建实例加入项目。
2. 按项目筛选实例列表。
3. 更改实例所属项目。

调整配置

概述

创建GPU云服务器后，用户可以根据业务需求调整云服务器配置，可调整配置有CPU，内存，系统盘，数据盘的大小。

配置变更可能需要进行较长时间，请根据业务需求自行选择合适的配置类型进行调配，详见下表。

配置类型	GPU云服务器类型	GPU云服务器状态	调整对象	业务影响	生效条件
升级配置	GN6I P3 P3I P3IN P4V	“已关闭”	CPU 内存 系统盘 数据盘	中断业务	调整配置成功后开启云服务器

升级配置

1. 登录云服务器KEC控制台，默认进入**云服务器** > **实例**页面，选择并关闭需要升级配置的GPU云服务器。

2. 在目标云服务器对应的**操作**列中，选择**更多** > **资源调整** > **调整配置**。

3. 选择配置信息后，单击**确定**。

4. 确认变更后的配置，单击**确定**，提交订单并支付。 5. 待调整配置成功后，根据提示重启云服务器使配置生效。

快照

概述

GPU云服务器支持快照功能，允许用户随时备份数据，可实现重要操作前的数据备份和操作后的数据恢复。

注意：仅支持为“运行中”或“已关闭”状态的云服务器创建快照。

使用步骤

1. 创建快照。
2. 用户可以创建自动快照策略来定期备份硬盘数据，有需求可以修改自动快照策略。
3. 不需要备份时，用户可以选择删除快照或者删除自动快照策略。

应用方向

- 使硬盘的数据恢复到创建快照时的状态可以参考回滚快照。
- 可以为本地硬盘设置自动快照策略，详细参考为硬盘设置自动快照策略。

镜像

概述

控制台镜像相关操作都是围绕自定义镜像进行的，用户可使用自定义镜像快速部署业务。

使用步骤

您可以通过以下方式获取镜像开机

- 通过实例制作镜像。
- 导入私有镜像。
- 跨地域复制自定义镜像。
- 可以将自定义镜像共享给同一地域内的其他用户，详细请参考共享镜像。

硬盘

概述

GPU云服务器可以选择挂载云硬盘，或GPU云服务器创建之后挂载本地盘扩充实例存储量。

使用步骤

1. 新创建云服务器之后，可以登录到服务器的操作系统挂载本地数据盘。详情见挂载本地数据盘。
2. 所有GPU云服务器套餐支持挂载云硬盘SSD3.0。详情见挂载云硬盘。
3. GPU云服务器支持在线扩容云盘系统盘，支持离线扩容云盘系统盘或本地系统盘。当现有系统盘的容量无法满足需求，您可以通过实例>更多>资源调整>调整配置扩容系统盘，或者通过云硬盘界面扩容云盘系统盘。详细步骤请参见扩容云硬盘。

4. 如现有数据盘的容量无法满足需求，您可以扩容数据盘。详细步骤请参见扩容本地数据盘，及扩容云硬盘。
5. 用户可以对回收站内的云硬盘进行恢复操作，详细见云硬盘回收站。

安全组

概述

安全组是金山云提供的重要的网络安全隔离手段，是一种有状态的服务器虚拟防火墙，用于设置云服务器KEC、裸金属服务器、GPU产品、以及云数据库产品的网络访问控制。

安全组作为逻辑上的分组，支持用户将同一地域内具有相同网络安全隔离需求的实例加到同一个安全组内，并可以配合安全组策略对实例的出入流量进行安全过滤。详情见安全组概述

使用步骤

您可对安全组有如下操作，详情见设置安全组。

1. 创建安全组。
2. 编辑安全组入站规则/出站规则。
3. 复制安全组，用户可以将已有安全组快速地复制到其它地域或VPC。
4. 删除安全组，用户可以删除不再使用的安全组，但是VPC中的默认安全组不能删除。
5. 更改安全组。
6. 管理GPU云服务器成员。

弹性网卡

概述

弹性网卡是可以附加到实例上的虚拟网卡，通过弹性网卡，可以实现低成本的故障转移和精细化的网络管理。每个弹性网卡之间是独立的，可以在多个云服务器之间自由迁移，实现业务的灵活扩展和迁移。GPU云服务器（GEC）支持弹性网卡功能。

GEC弹性网卡的相关属性和功能和云服务器（KEC）相同。

相关属性

- **网卡类型**：

  主网卡：创建实例时随实例一起创建的弹性网卡，创建完成后无法与实例解绑

  辅网卡：用户手动创建的弹性网卡，支持绑定实例或从实例上解绑

- **所属VPC**：单个实例不同网卡可以处于不同VPC下也可以处于同一VPC下
- **子网**：单个实例的不同网卡可处于相同子网或者不同子网下
- **可用区**：弹性网卡所属子网与绑定的实例必须属于同一可用区
- **安全组**：加入到一个安全组中，由安全组控制进出弹性网卡的流量
- **子网IPv4地址**：支持自动分配和手动分配IP地址
- **MAC地址**：弹性网卡有全局唯一的MAC地址

功能特点

- **多网卡**：除随实例一起创建的主网卡外，一台实例还支持绑定多个辅网卡。不同弹性网卡可属于相同或不同子网，每个网卡支持配置独立的安全组。
- **灵活迁移**：弹性网卡可以自由地在相同VPC、可用区下的云服务器间自由迁移，弹性网卡与云服务器解绑时，保留已绑定内网 IP、弹性公网 IP 和安全组策略，迁移后无需重新配置关联关系。
- **热插拔**：弹性网卡支持热插拔，切换弹性网卡绑定的实例时无需重启实例，不影响实例上运行的业务。

应用场景

- **搭建高可用集群** ：满足单实例多网卡的需求从而搭建高可用集群。
- **低成本故障迁移**：通过将弹性网卡从实例解绑后再绑定到另外一台实例，将故障实例上的业务流量快速转移到备用实例，实现服务快速恢复。
- **精细化网络管理** ：可以为实例配置多个弹性网卡，例如，用于内部管理的弹性网卡及用于公网业务访问的弹性网卡等，完成管理数据和业务数据间的隔离。也可以根据源IP地址、应用层协议、端口等对每张弹性网卡配置精准的安全组规则，从而

对每张弹性网卡的流量进行安全访问控制。

使用步骤

您可以通过以下方式使用弹性网卡：

- 创建弹性网卡
- 绑定弹性网卡
- 配置弹性网卡
- 更改网络配置
- 卸载弹性网卡
- 删除弹性网卡

密钥对登录

概述

SSH密钥对是一种安全便捷的登录认证方式，由公钥和私钥组成。Linux实例绑定创建的密钥对后，才可以使用私钥通过SSH命令连接实例。

相较于用户名和密码认证方式，SSH密钥对认证更为安全可靠，可以杜绝暴力破解威胁，便于远程登录大量Linux实例。

使用步骤

您可以按如下操作使用密钥对，详情见创建与删除密钥对。

1. 创建新密钥对。
2. 使用已有公钥创建密钥对。
3. 删除密钥对。

一对SSH密钥可支持多台Linux云服务器的登录使用。详情见实例绑定与解绑密钥对。

安装CUDA驱动指南

CUDA (Compute Unified Device Architecture) 是显卡厂商 NVIDIA 推出的运算平台。 CUDA™ 是一种由 NVIDIA 推出的通用并行计算架构，该架构使 GPU 能够解决复杂的计算问题。 它包含了 CUDA 指令集架构（ISA）以及 GPU 内部的并行计算引擎。 开发人员现在可以使用 C 语言, C++ , FORTRAN 来为 CUDA™ 架构编写程序，所编写出的程序可以在支持 CUDA™ 的处理器上以超高性能运行。GPU 云服务器采用 NVIDIA 显卡，需要安装 CUDA 开发运行环境。

您可选择下载金山云提供的CUDA安装脚本进行安装。也可直接到Nvidia官网进行下载安装。

使用脚本安装驱动

背景信息

配备NVIDIA GPU的实例仅涉及GPU驱动，且仅非vGPU的GPU实例支持安装GPU驱动。脚本可支持基础标准镜像安装Nvidia最新的GPU驱动，支持镜像如下：

- CentOS-7.* 64位（所有Centos 7 版本的镜像均支持）
- Ubuntu18.04 64位镜像

安装步骤

以Ubuntu 18.0.4为例，可参照以下步骤进行安装

1. 下载CUDA安装脚本至本地环境。
2. 登录 GPU 实例，打开管理员权限：

   ```
   sudo -i
   ```

3. 上传CUDA安装脚本至GPU云服务器实例。

   ```
   rz
   ```

4. 在弹出的上传窗口中选择目标脚本，点击确定上传。此时脚本已上传到当前目录。
5. 上传完毕后点击关闭。

6. 运行脚本

   ```
   sh auto_install_kingsoft.sh
   ```

7. 检查是否安装成功 命令行输入nvcc --version，若输出CUDA版本号，则CUDA安装成功。

登录Nvidia官网下载驱动

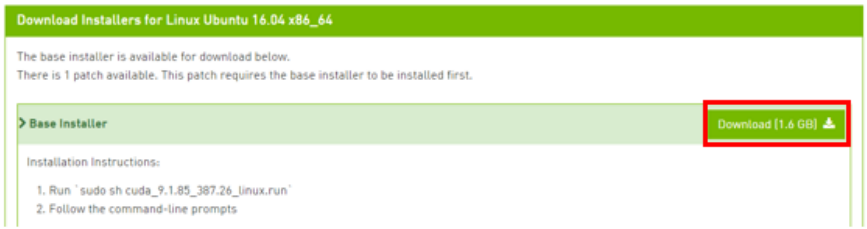
以Ubuntu 18.0.4为例，可参照以下步骤进行安装

1. 登录GPU实例，进行CUDA驱动下载或复制链接 https://developer.nvidia.com/cuda-downloads
2. 选择与自己的操作系统相匹配的安装包。以Ubuntu 18.0.4 64 位为例，可按如下方式进行选择：

   注意：

   - Installer Type 这里推荐选择 runfile（local）。
   - network：网络安装包，安装包较小，需要在主机内联网下载实际的安装包。
   - local：本地安装包。安装包较大，包含每一个下载安装组件的安装包。

3. 点击【Download】，选择下载存放地址：



4. 切换到CUDA安装包所在的目录，执行以下命令：

   ```
   sudo sh cuda_9.1.85_387.26_linux.run
   ```

   根据提示选择accept -yes -enter。

   注意：

   - 若执行后出现如下结果:

     ```
     Driver: Installed require reboot
     Toolkit: install skip
     Samples: install skip
     ```

     说明这个CUDA安装包包含了Driver，Toolkit和Samples三部分，但此次安装时只把驱动装上了。

     此时需重新安装，再次执行以下命令：

     ```
     sudo sh cuda_9.1.85_387.26_linux.run
     ```

   - CUDA安装成功结果如下：

     ```
     Driver: Installed
     Toolkit: Installed in /usr/local/cuda
     Samples: Installed in /home/XX
     ```

5. 在 /usr/local/cuda/samples/1_Utilities/deviceQuery 目录下，执行 make 命令，可以编译出 deviceQuery 程序。执行 deviceQuery 正常显示如下设备信息，此刻认为 CUDA 安装正确。

```
./deviceQuery Starting...

 CUDA Device Query (Runtime API) version (CUDART static linking)

Detected 1 CUDA Capable device(s)

Device 0: "Tesla M40 24GB"
  CUDA Driver Version / Runtime Version          8.0 / 7.5
  CUDA Capability Major/Minor version number:    5.2
  Total amount of global memory:                 24505 MBytes (25695092736 bytes)
  (24) Multiprocessors, (128) CUDA Cores/MP:     3072 CUDA Cores
  GPU Max Clock rate:                            1112 MHz (1.11 GHz)
  Memory Clock rate:                             3004 Mhz
  Memory Bus Width:                              384-bit
  L2 Cache Size:                                 3145728 bytes
  Maximum Texture Dimension Size (x,y,z)         1D=(65536), 2D=(65536, 65536), 3D=(4096, 4096, 4096)
  Maximum Layered 1D Texture Size, (num) layers  1D=(16384), 2048 layers
  Maximum Layered 2D Texture Size, (num) layers  2D=(16384, 16384), 2048 layers
  Total amount of constant memory:               65536 bytes
  Total amount of shared memory per block:       49152 bytes
  Total number of registers available per block: 65536
  Warp size:                                     32
  Maximum number of threads per multiprocessor:  2048
  Maximum number of threads per block:           1024
  Max dimension size of a thread block (x,y,z): (1024, 1024, 64)
  Max dimension size of a grid size    (x,y,z): (2147483647, 65535, 65535)
  Maximum memory pitch:                          2147483647 bytes
  Texture alignment:                             512 bytes
  Concurrent copy and kernel execution:          Yes with 2 copy engine(s)
  Run time limit on kernels:                     No
  Integrated GPU sharing Host Memory:            No
  Support host page-locked memory mapping:       Yes
  Alignment requirement for Surfaces:            Yes
  Device has ECC support:                        Disabled
  Device supports Unified Addressing (UVA):      Yes
  Device PCI Domain ID / Bus ID / location ID:   0 / 0 / 7
  Compute Mode:
     < Default (multiple host threads can use ::cudaSetDevice() with device simultaneously) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 8.0, CUDA Runtime Version = 7.5, NumDevs = 1, Device0 = Tesla M40
Result = PASS
```

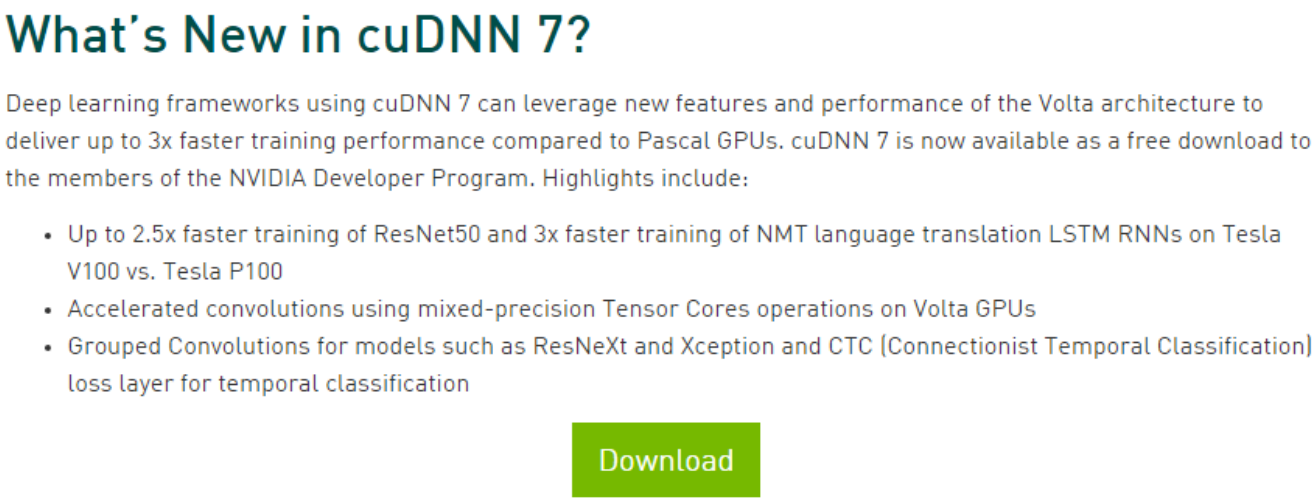
安装cuDNN和NCCL指南

安装cuDNN

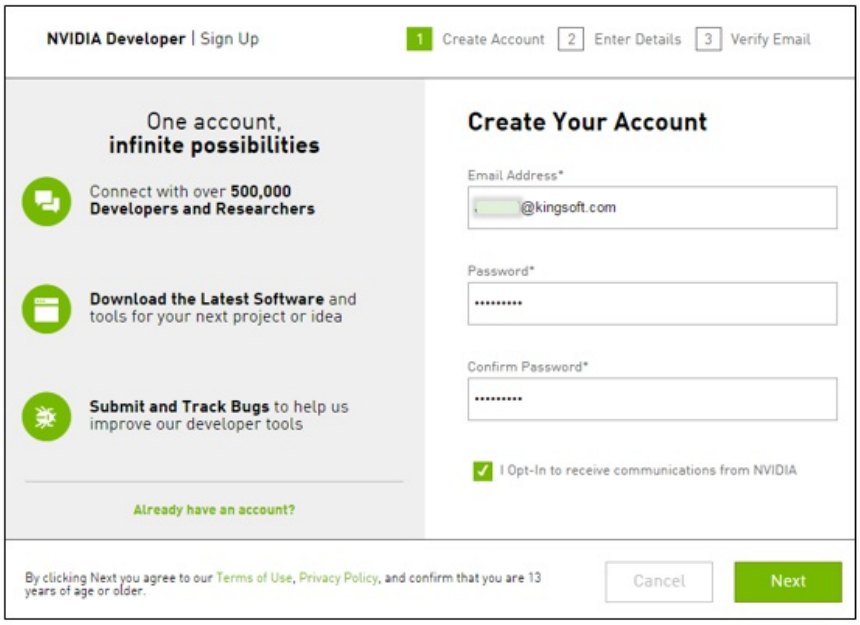
cuDNN的全称为NVIDIA CUDA® Deep Neural Network library，是NVIDIA专门针对深度神经网络（Deep Neural Networks）中的基础操作而设计基于GPU的加速库。cuDNN为深度神经网络中的标准流程提供了高度优化的实现方式，例如convolution、pooling、normalization以及activation layers的前向以及后向过程。

下面以Ubuntu 16.0.4为例说明如何配置cuDNN进行神经网络的加速。

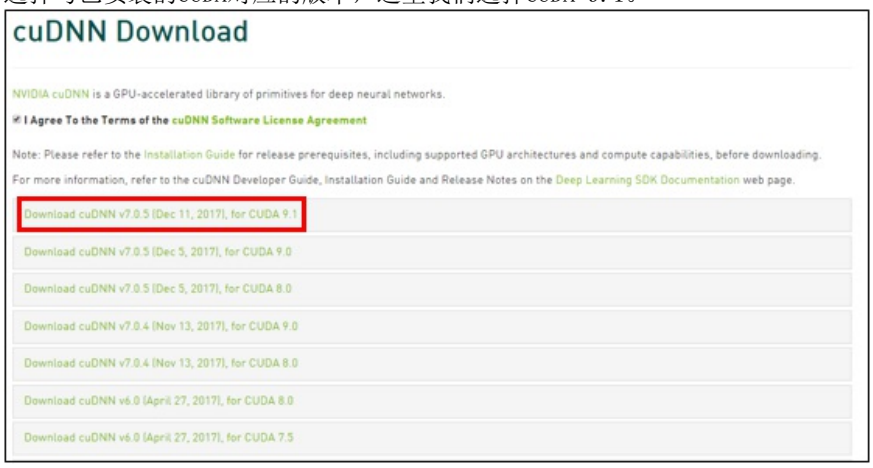
1. 登录GPU实例，进行cuDNN下载或复制下载地址 https://developer.nvidia.com/cudnn ，点击【Download】进行下载。



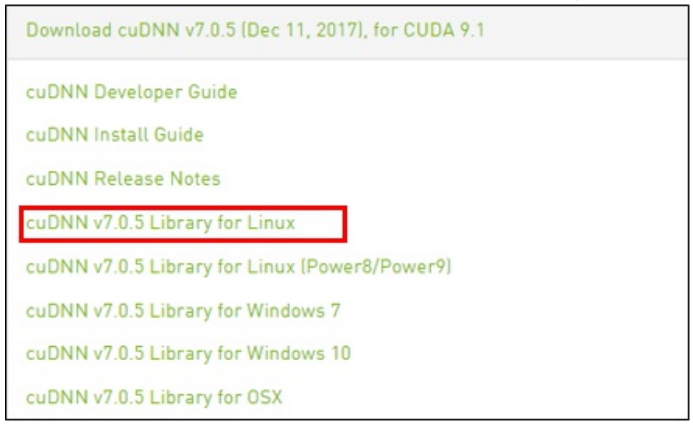
2. 需要登录/注册，请按步骤提示，完成输入邮箱和密码，用户等信息，确认邮件等步骤。



3. 选择与已安装的CUDA对应的版本，这里我们选择CUDA 9.1。



4. 选择与操作系统对应的版本，这里我们选择【cuDNN v7.0.5 Library for Linux】，下载得到压缩文件cudnn-9.1-linux-x64-v7.tgz。



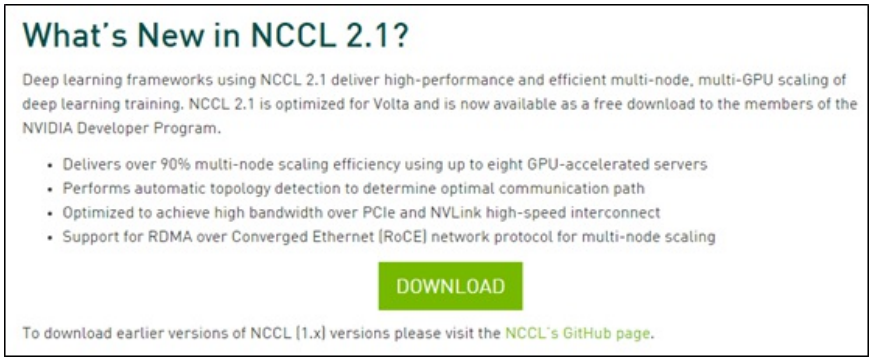
5. 切换到cuDNN压缩包所在目录，在命令行输入以下指令进行解压。

```
sudo tar -xvf cudnn-9.1-linux-x64-v7.tgz -C /usr/local          ---- 完成安装
```

安装NCCL

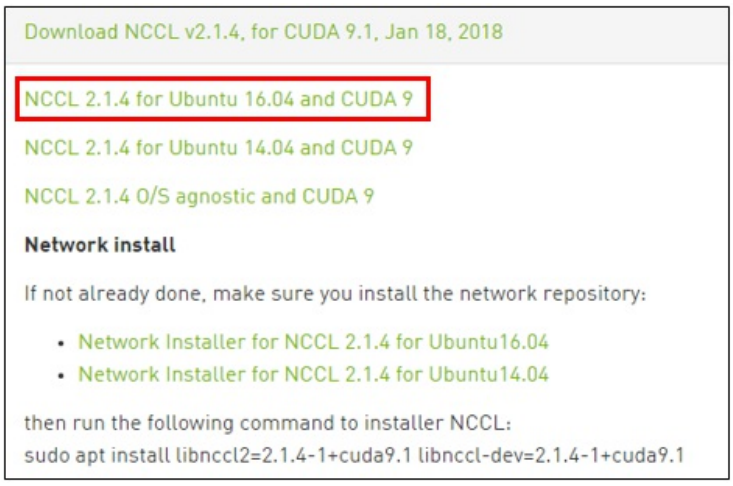
NCCL是Nvidia Collective multi-GPU Communication Library的简称，它是一个实现多GPU的collective communication通信库，Nvidia做了很多优化，以在PCIe、Nvlink、InfiniBand上实现较高的通信速度。

1. 登录GPU实例，进行NCCL下载或复制下载地址 https://developer.nvidia.com/nccl ，点击【Download】进行下载。



2. 需要登录/注册，请按步骤提示耐心操作。登录后，选择对应版本的 NCCL，这里我们选择【Download NCCL v2.1.4, for CUDA 9.1, Jan 18, 2018】。

3. 选择与操作系统对应的版本，以Ubuntu 16.0.4为例，这里我们选择【NCCL 2.1.4 for Ubuntu 16.04 and CUDA 9】，下载得到nccl-repo-ubuntu1604-2.1.4-ga-cuda9.1_1-1_amd64.deb。



4. 切换到NCCL文件所在目录，运行以下命令：

```
sudo dpkg -i nccl-repo-ubuntu1604-2.1.4-ga-cuda9.1_1-1_amd64.deb
```

完成解压安装，将NCCL的 include 和 lib 文件夹下文件放到对应 /usr/local/include/usr/local/lib 目录下。

安装CAFFE指南

前提条件

1. 已安装成功CUDA，若未安装请查看教程 安装CUDA驱动指南。
2. 已安装cuDNN和NCCL优化库，若未安装请查看教程 安装cuDNN和NCCL指南。

3. 安装必要的依赖库，输入以下指令：

```
sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libhdf5-serial-dev protobuf-compiler libgflags-dev libgoogle-glog-dev liblmdb-dev libatlas-base-dev git  -y
sudo apt-get install --no-install-recommends libboost-all-dev  -y
```

安装CAFFE

1. 从 Github下载CAFFE的对应发布版本，下载地址如下： https://github.com/NVIDIA/caffe/releases，选择【Source code (tar.gz)】。 或者复制链接地址，利用指令进行下载：

   ```
   wget https://github.com/NVIDIA/caffe/archive/v0.16.6.tar.gz
   ```

2. 进行Openblas下载与安装，输入如下指令：

   ```
   wget  https://github.com/xianyi/OpenBLAS/archive/v0.2.20.tar.gz
   tar -xvf v0.2.20.tar.gz
   cd OpenBLAS-0.2.20
   make PRFIX=/opt/OpenBLAS
   make install
   ```

3. 配置lib路径，使lib库生效，输入以下指令：

   ```
   echo “/opt/OpenBLAS/lib/
         /usr/local/cuda/lib64“ >> /etc/ld.so.conf.d/cuda.conf
   ldconfig
   ldconfig -v
   ```

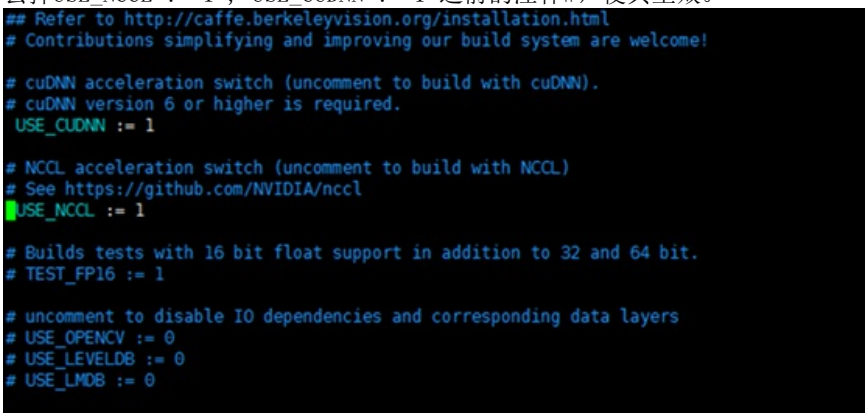
4. 解压CAFFE压缩包，请把文件放到/home 目录下，输入以下指令：

   ```
   tar -xvf v0.16.6.tar.gz
   cd caffe-0.16.6
   cp Makefile.config.example Makefile.config
   ```

5. 对Makefile.config进行编辑，输入如下指令：

   ```
   vi Makefile.config
   ```

   去掉USE_NCCL := 1 ， USE_CUDNN := 1 之前的注释#，使其生效。

   注：

   - 取消对行 USE_CUDNN := 1 的注释可以启用 cuDNN 加速。
   - 取消对行 USE_NCCL := 1 的注释可以启用在多个 GPU 上运行 Caffe 所需的 NCCL。

6. 保存并关闭文件，对CAFFE进行编译，输入指令：

   ```
   make -all -j4
   ```

等待编译打开编辑器，添加bin 路径，输入指令：

```
vi /etc/profile
export  PATH=$PATH:/home/ caffe-0.16.6/build/tools
```

保存并退出，输入指令，使文件生效。

```
source /etc/profile
```

配置pycaffe

1. 根据caffe根目录python文件夹下的requirements.txt的清单文件，安装上面列出的需要的依赖库。输入如下指令查看requirements.txt：

   ```
   cd caffe-0.16.6/python
   cat requirements.txt
   ```

   显示如下信息：

   ```
   cython>=0.19.2
   numpy>=1.7.1,<=1.11.0
   scipy>=0.13.2
   scikit-image>=0.9.3
   matplotlib>=1.3.1
   ipython>=3.0.0,<=5.4.1
   h5py>=2.2.0
   leveldb>=0.191
   networkx>=1.8.1
   nose>=1.3.0
   pandas>=0.12.0
   protobuf>=2.5.0
   pydot2
   python-dateutil>=1.4,<2
   python-gflags>=2.0
   pyyaml>=3.10
   Pillow>=2.3.0
   six>=1.1.0
   ```

2. 执行安装代码：

   ```
   for seq in $(cat requirements.txt  )
   do
   pip install $seq
   done
   ```

3. 安装完成以后，再次回到caffe根目录，可以执行如下指令：

   ```
   cd ..  ----退出软件python 目录，回到caffe根目录
   sudo pip install -r python/requirements.txt
   ```

   安装成功的，都会显示Requirement already satisfied； 没有安装成功的，会继续安装。

4. 编译python接口，输入如下指令：

   ```
   make pycaffe
   ```

   配置路径，添加到 /etc/profile，输入以下指令：

   ```
   vi /etc/profile
   export PYTHONPATH=/home/ caffe-0.16.6/python:$PYTHONPATH
   ```

   保存并退出，输入指令，使文件生效：

   ```
   source /etc/profile
   ```

5. 输入python，开启pyhton测试pycaffe，若运行如下三行不出错 说明配置完成。

   ```
   import caffe
   from caffe import layers as L
   from caffe import params as P
   ```

安装TensorFlow指南

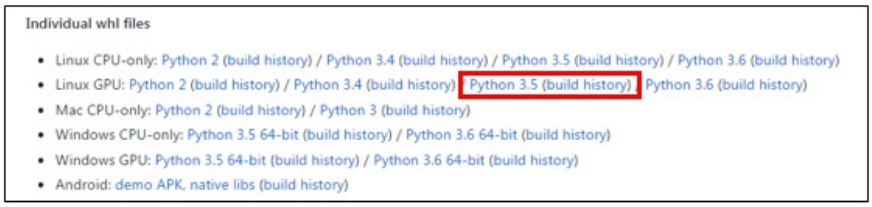
网络安装

网络良好时，可直接指令安装：

```
pip3 install --upgrade tensorflow-gpu
```

源码安装

1. 进行tensorflow 1.5.0下载或复制下载地址 https://github.com/tensorflow/tensorflow
2. 下载对应 GPU版本，这里我们选择python 3.5 版本，得到文件【tf_nightly_gpu-1.head-cp35-cp35m-linux_x86_64.whl】。



3. 输入以下指令，进行安装。

```
pip3 install tf_nightly_gpu-1.head-cp35-cp35m-linux_x86_64.whl
```

验证环境

TensorFlow安装完成后，执行python脚本，根据输出结果，确认是否有异常。Python代码如下：

```
import tensorflow as tf
import numpy as np
x_data = np.float32(np.random.rand(2, 100))
y_data = np.dot([0.100, 0.200], x_data) + 0.300
b = tf.Variable(tf.zeros([1]))
W = tf.Variable(tf.random_uniform([1, 2], -1.0, 1.0))
y = tf.matmul(W, x_data) + b
loss = tf.reduce_mean(tf.square(y - y_data))
optimizer = tf.train.GradientDescentOptimizer(0.5)
train = optimizer.minimize(loss)
init = tf.initialize_all_variables()
sess = tf.Session()
sess.run(init)
for step in range(0, 201):
    sess.run(train)
    if step % 20 == 0:
        print (step, sess.run(W), sess.run(b))*
```

使用以上代码写入python 文件，Python 指向文件运行即可，如果运行成功则结果如下：

更加详细的安装流程，可以参考tensorflow社区文档。

vGPU简介

vGPU基于NVIDIA推出的GPU虚拟化技术提供了不同于传统云服务器的GPU搭载模式。vGPU真正实现了GPU资源的虚拟化，提供在云服务器上搭载虚拟GPU的能力。

相较于传统的GPU Passthrough（直通模式）搭载，GPU虚拟化能够提供更加适宜的能力，具体包括：

- 对GPU资源实现更加细粒度的切割，提供1/2、1/4等更加细粒度的GPU资源。
- vGPU通过配置不同的License，可适用于各种细分应用场景。

  例如，

  - vPC License：适用于图形/图像处理场景，如视频编辑处理、简单的渲染场景等。
  - vCS License：适用于深度学习处理场景，如小规模灵活的训练、推理、AI教学场景等。
  - vDWS License：适用于专业级图形/图像处理场景、适用于深度学习场景。

vGPU用户指南概述

vGPU提供在云服务器上搭载虚拟GPU的能力，vGPU实例需要配合如下产品使用：

- GRID驱动：安装在vGPU实例上，是实现vGPU功能的必备需求程序。
- License Server：为vGPU实例授权License，统一管理License。
- License：用于激活vGPU实例的GRID驱动。

不同类别的License可实现不同的功能，具体包括：

- vDWS License：同时支持CUDA计算、专业级图形图像处理工作站，且除了支持vGPU外，vDWS也可用于GPU计算型实例对OpenGL的支持。
- vPC License：最大支持4K图形图像处理，可用于视频编辑、渲染等。
- vCS License：支持深度学习场景，通过对GPU资源的切割，提升GPU利用率。

各产品的关系如图所示。

要正确运行和使用vGPU功能，需要完成如下任务：

1. 购买vGPU实例。

   购买vGPU类型的实例时，可以选择标准镜像，或联系金山云客服获取已经预装GRID驱动的镜像。

   有关购买的详细步骤，请参考GPU云服务器实例创建。

2. 获取License文件。可以直接联系NVIDIA购买，或咨询金山云客服。

3. 搭建License Server并导入License文件。

   有关操作的详细步骤，请参考搭建License Server。

4. 为vGPU实例安装并激活GRID驱动。

   根据实例的操作系统，选择对应的安装和激活方式：

   - 如果使用Windows，请参考安装/激活GRID驱动（Windows）。
   - 如果使用Linux，请参考安装/激活GRID驱动（Linux）。

5. （可选）如果想在其他GPU实例上实现vGPU的图形/图像计算能力，可为其安装GRID驱动。

   有关操作的详细介绍，请参考GPU计算型实例安装GRID驱动。

搭建License Server

License Server用于实现License的统一管理和授权，vGPU实例需要与License Server通信以激活GRID驱动。

本节以Linux服务器为例，介绍如何搭建License Server并管理License。

注意事项

为保证License的安全，建议将License Server和vGPU实例创建在同一个VPC内，并使用内网IP地址访问License Server。

如果确实需要跨地域部署vGPU实例，建议为License Server绑定公网IP地址后，通过安全组严格控制访问的源IP地址。

License Server的搭建步骤

License Server的推荐配置如下
CPU：4核
内存：8GB
数据盘：100GB，用于存储日志，可根据需求调整容量。

1. 您需要按照NVIDIA官方文档，自己搭建License Server。

2. 向License Server中导入License。

   a. 登录云服务器。

   b. 检查FNE服务是否为“running”状态。

   ```
   systemctl status flexnetls-nvidia.service
   ```

   如果服务状态错误，则需要重新启动服务并再次检查服务状态。

   ```
   systemctl stop flexnetls-nvidia.service
   rm -rf  /var/opt/flexnetls/nvidia
   systemctl start flexnetls-nvidia.service
   systemctl status flexnetls-nvidia.service
   ```

   c. 向NVIDIA购买或者申请试用获取License，并从“NVIDIA SOFTWARE LICENSING CENTER”页面获取License文件。 申请NVIDIA试用版License：NVIDIA 90-day vGPU trial

注意： 下载的License文件要在24小时内尽快使用；否则文件失效，需要重新下载。

d. 打开licserver管理界面，地址为http://localhost:8080/licserver。

e. 在licserver管理界面中，选择“License Server > License Management”，上传License文件。

文件上传成功后，License Server可以开始为vGPU实例授权License。

3. 利用licserver管理界面查看License的使用情况。

选择“License Server > Licensed Clients”，查看已经申请到License的设备，如下图所示。

安装/激活GRID驱动（Windows）

本节介绍在使用Windows标准镜像的GPU云服务器上，如何手动安装并激活GRID驱动。

如果vGPU实例使用的是预装GRID驱动的操作系统镜像，则无需手动安装GRID驱动，只需要激活GRID驱动。

安装GRID驱动

1. 根据操作系统版本，下载对应的GRID驱动安装包：
   - 适用于Windows 10、Windows Server 2016
   - 适用于Windows 8、Windows 7、Windows Server 2012R2、Windows Server 2008R2
2. 双击文件，按照向导完成GRID驱动的安装。

注意： GRID驱动安装生效后，Windows的远程链接协议（RDP）可能出现不支持DirectX、OpenGL等应用、控制台VNC失效的情况。要解决这些问题，可安装VNC服务（如tightVNC等软件），再连接到vGPU渲染输出的高清屏幕。

激活GRID驱动

在开始本节操作之前，确保已完成License Server的搭建和License的配置。有关操作的详细描述，请参考搭建License Server。

激活GRID驱动的操作步骤为：

1. 右击桌面打开NVIDIA控制面板。

2. 选择“许可 > 管理许可证”，输入License Server的地址和端口号。

3. 待GRID激活后，查看授权情况。

安装/激活GRID驱动（Linux）

本节介绍在使用Linux标准镜像的云服务器上，如何手动安装并激活GRID驱动。

如果vGPU实例使用的是预装GRID驱动的操作系统镜像，则无需手动安装GRID驱动，只需要激活GRID驱动。

安装GRID驱动

1. 下载GRID驱动安装包。
2. 赋予安装包文件可执行权限。

```
chmod +x NVIDIA-Linux-x86_64-430.30-grid.run
```

3. 执行安装包文件。

```
./NVIDIA-Linux-x86_64-430.30-grid.run
```

4. 重启GPU云服务器。

   ```
   reboot
   ```

激活GRID驱动

在开始本节操作之前，确保已完成License Server的搭建和License的配置。有关操作的详细描述，请参考搭建License Server。

激活GRID驱动的操作步骤为：

1. 进入目录/etc/nvidia。

   ```
   cd /etc/nvidia
   ```

2. 创建或打开该目录下的gridd.conf文件。

   - 如果文件不存在，创建该文件。

     ```
     cp gridd.conf.template gridd.conf
     ```

   - 如果文件已存在，打开该文件。

     ```
     vim gridd.conf
     ```

3. 在文件中添加License Server配置信息。

   ```
   ServerAddress=      /* License Server的IP地址*/
   ServerPort=7070     /* License Server的端口号，默认为7070*/
   FeatureType=        /* 如果是用于CUDA计算，填“4”即可*/
   ```

4. 重启。

   ```
   systemctl restart nvidia-gridd
   ```

5. 验证License的激活状态。

   ```
   systemctl status nvidia-gridd
   ```

如果GRID驱动已经成功激活，可以看到“License acquired successfully”字样。

GPU计算型实例安装GRID驱动

GPU加速计算型实例，如P3、P3I、P3IN、和P4V、GN6I实例，如果需要OpenGL图形支持，必须在实例中安装GRID驱动。

注意： 直通模式的GPU云主机，其GRID驱动只支持通过vDWS License激活。

关于详细的操作步骤，请参考安装/激活GRID驱动 (Windows)。

关于直通模式的GPU实例类型的介绍，请参考GPU产品类型。