GPU设备透传 使用教程

产品版本：ZStack 3.10.0

文档版本：V3.10.0

ZStack

版权声明

商标说明

ZStack商标和其他云轴科技商标均为上海云轴信息科技有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受云轴科技公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，云轴科技公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目录

1 概述

ZStack支持物理GPU透传功能，物理GPU可携带其上全部外设（包括：GPU显卡、GPU声卡、以及其它GPU上的小设备）以**组**为单位整体透传给云主机使用，让云主机享有物理机强劲的GPU并行计算能力。该功能适用于3D渲染、高清转解码、以及具备高密集运算特点的高性能计算（HPC）场景。

ZStack支持以下型号的物理GPU透传：

NVIDIA	AMD
• Tesla T4 • Tesla M10/M60 • Tesla P100/P40/P6/P4 • Tesla V100 • RTX 6000/8000 • 更多请参考*NVIDIA*官方文档	• FirePro S7150 • FirePro S7150X2

2 注意事项

使用GPU设备透传功能，需要注意以下情况：

- 一台云主机支持同时加载多个物理GPU设备，但不支持同时加载物理GPU和vGPU设备。
- GPU透传给云主机后，迁移、存储迁移、高可用功能可能无法正常工作。
- 建议停止云主机再执行卸载GPU操作，否则可能导致蓝屏以及暂停。
- 针对Windows云主机透传GPU设备场景，需要通过UEFI方式为云主机安装操作系统。
- 全局设置**PCI设备热插拔开关**默认为true，若热插拔时出现硬件兼容性错误，或不支持该硬件设备时，建议关闭此功能（设置为false）。
- 指定GPU规格方式支持批量创建云主机，但指定GPU设备方式仅支持单个创建云主机。

3 准备工作

ZStack云平台使用GPU设备透传功能，需要准备以下工作：

- 确保物理机BIOS中开启Intel VT-d / AMD IOMMU功能，且物理机内核已开启IOMMU支持。
- 确保已获取到GPU设备对应的驱动。

  注: 相关驱动以及安装方法请联系GPU设备提供厂商获取帮助。

- 按需设置全局设置：进入**设置** > **全局设置** > **高级设置**界面，按需调整全局设置，以下几个全局设置与物理GPU透传相关：
  - PCI设备热插拔开关：用于设置是否允许云主机热插拔GPU设备，默认为true。若热插拔时出现硬件兼容性错误，或不支持该硬件设备时，建议关闭此功能（设置为false）。
  - GPU设备配额：用于设置账户/项目使用GPU设备（包括：物理GPU和vGPU）数量配额，默认为20。

4 典型使用流程

背景信息

GPU设备透传的典型使用流程如下：

1. 物理机启用IOMMU设置；
2. 设置ROM（可选）；
3. 云主机加载物理GPU；
4. 云主机安装GPU设备驱动。

使用GPU设备透传功能前，请务必确保所有准备工作已完成且准确无误。以下详细介绍GPU设备透传功能的操作步骤：

操作步骤

1. 物理机启用IOMMU设置

   确保物理机BIOS已开启Intel VT-d / AMD IOMMU配置的前提下，在ZStack云平台开启物理机IOMMU设置。

   - 新添加物理机：在**硬件设施** > **物理机**界面添加物理机过程，勾选**扫描物理机IOMMU设置**配置，添加物理机的同时开启IOMMU设置。如图 *1:* 新添加物理机并启用*IOMMU*设置所示：

     图 1: 新添加物理机并启用IOMMU设置

- 已添加物理机：在物理机详情页，启用**IOMMU启用状态**配置，针对已添加物理机开启IOMMU设置，重启物理机生效。如图 *2: 已添加物理机启用IOMMU设置*所示：

图 2: 已添加物理机启用IOMMU设置

注: 物理机开启IOMMU设置后，还需在物理机详情页确保**IOMMU就绪状态**为可用，否则也无法正常使用SR-IOV功能。若IOMMU启用状态为启用，但IOMMU就绪状态不可用，可能以下原因：

- 开启IOMMU设置但未重启物理机，手动重启物理机即可。
- 物理机配置错误，请进入物理机BIOS并开启Intel VT-d / AMD IOMMU配置。

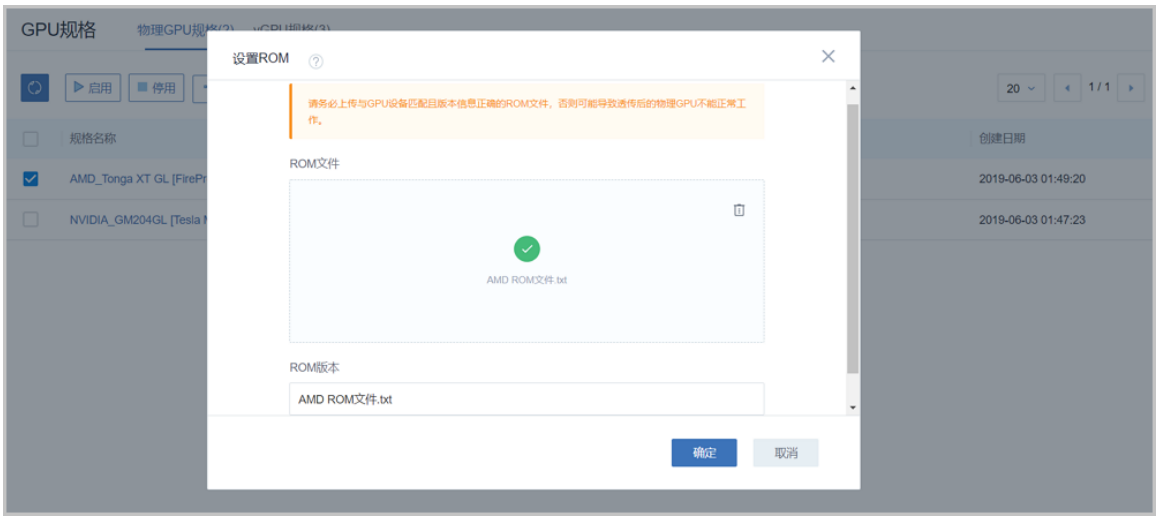
2. 设置ROM（可选）

ROM文件是用于物理GPU透传的配置文件。ROM文件上传后，将直接更新到已添加的规格对应的物理GPU中。

ZStack云平台已内置基础ROM文件，满足绝大部分物理GPU透传。若用户需要使用其他ROM文件，请自行在显卡供应商官网获取所需的ROM文件并上传。

在ZStack私有云主菜单，点击**云资源池** > **GPU规格**，进入**GPU规格**界面，选中需要设置ROM的物理GPU规格并点击**更多操作** > **设置ROM**按钮，在弹出的**设置ROM**页面上传ROM文件，如图 *3: 设置ROM*所示：

图 3: 设置ROM



注: 上传ROM文件需要注意以下情况：

- 请务必上传与物理GPU匹配且版本信息正确的ROM文件，否则可能导致透传后的物理GPU不能正常工作。
- 最新上传的ROM文件会覆盖之前上传的ROM文件。

3. 云主机加载物理GPU

云主机加载物理GPU即可将物理GPU直接透传给云主机使用，ZStack云平台支持以下几种方式为云主机加载物理GPU：

- 方式一：创建云主机并加载物理GPU

  在**云资源池** > **云主机**界面创建云主机过程，基础参数配置完成后在**高级**选项中加载物理GPU，支持指定规格和指定设备两种加载方式。可参考以下示例输入相应内容：

  - 指定规格：创建云主机时指定物理GPU规格，通过规格自动分配GPU设备。支持关机自动卸载设备功能（默认不勾选），若勾选表示云主机关机后自动卸载GPU设备，下次重启后根据GPU规格重新分配新的GPU设备；若不勾选表示云主机关机后保留已加载的GPU设备，下次重启后继续使用原来的GPU设备。如图 *4:* 指定规格所示：

    图 4: 指定规格

- 指定设备：创建云主机时指定固定的物理GPU设备，为云主机加载所选GPU设备，如图 *5:* 指定设备所示：

图 5: 指定设备

配置完成后，点击**确定**按钮，即可创建一台加载物理GPU的云主机。

- 方式二：单个已有云主机加载物理GPU

在**云主机**管理界面点击云主机名称，进入云主机界面的**配置信息**子页面的GPU设备栏，点击**操作** > **加载**按钮手动加载物理GPU，如图 *6:* 加载物理*GPU*所示：

图 6: 加载物理GPU

- 一台云主机支持同时加载多个物理GPU，暂不支持将物理GPU和vGPU同时加载到同一台云主机使用。

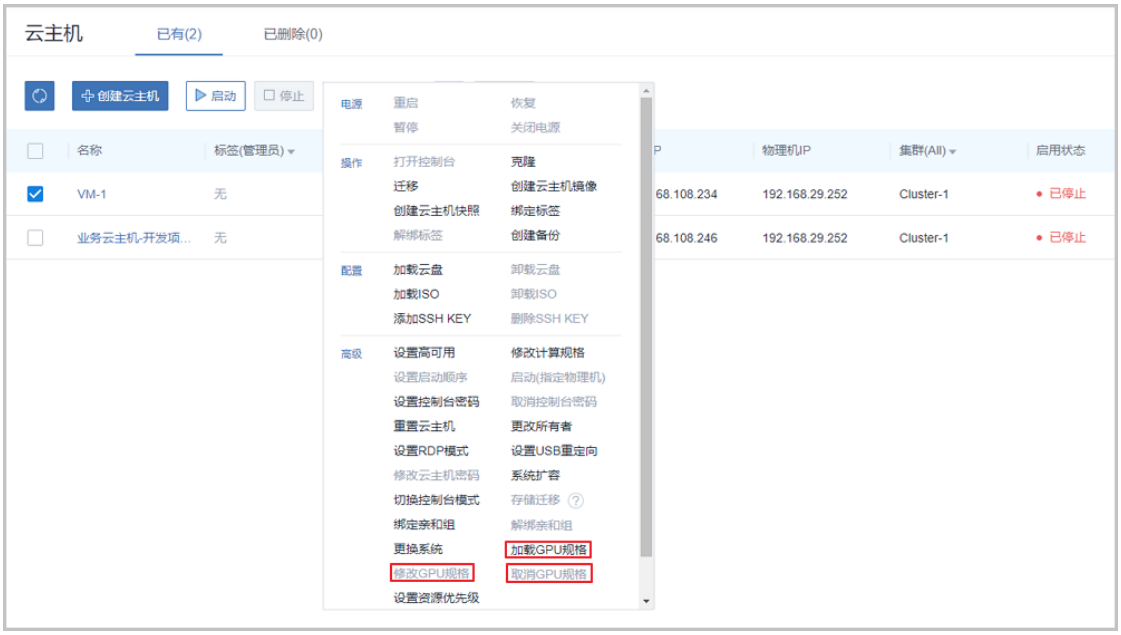
- 若需要释放GPU设备，选中GPU设备，点击**操作** > **卸载**按钮，释放GPU设备。

  注: 对运行中的云主机卸载物理GPU，可能导致蓝屏以及暂停，建议停止云主机再执行卸载操作。

- 方式三：批量为已有云主机加载物理GPU

  在**云主机**管理界面选择一台或多台云主机，点击**更多操作** > **加载GPU规格**按钮，批量为云主机加载GPU规格。如图 7: 批量加载*GPU*规格所示：

  图 7: 批量加载GPU规格



  注:

  - 已加载GPU规格、已加载GPU设备或运行中的云主机无法加载GPU规格。
  - 单台或多台云主机同时支持修改GPU规格和取消GPU规格操作，在云主机操作中点击修改GPU规格\取消GPU规格按钮即可。
    - 未加载GPU规格或运行中的云主机无法修改GPU规格或取消GPU规格。
    - 修改GPU规格后，下次启动云主机将使用最新GPU规格重新加载GPU设备，并卸载原GPU规格相关的GPU设备。

4. 云主机安装GPU设备驱动

   云主机加载GPU设备后，需要安装对应的驱动程序才能正常使用。AMD或NVIDIA驱动程序参考下载路径如下：

- Linux支持AMD的计算卡、游戏卡、专业卡。Linux自带社区驱动，如需支持计算加速和显示加速功能。请点击这里安装官方驱动。
- Linux支持NVIDIA的计算卡、游戏卡、专业卡。Linux自带社区驱动，如需支持计算加速和显示加速功能。请点击这里安装官方驱动。
- Windows支持AMD的计算卡、游戏卡、专业卡和NVIDIA的计算卡。请点击这里根据显卡类型和Windows操作系统版本下载合适的显卡驱动。
- Windows仅支持NVIDIA的计算卡。请点击这里根据显卡类型和Windows操作系统版本下载合适的显卡驱动。

不同GPU设备的驱动以及安装方法可能不同，详情请联系GPU设备提供厂商获取帮助。本章节以Linux云主机安装NVIDIA GPU驱动为例介绍参考操作流程：

1. 获取驱动安装相关文件：

   获取GPU设备匹配的显卡驱动和CUDA toolkit文件。

2. 禁用nouveau驱动：

   NVIDIA显卡的官方驱动和系统自带的nouveau驱动存在冲突。执行**lsmod | grep nouveau**命令，若有输出内容表示存在nouveau驱动，可参考以下方法禁用nouveau驱动；若不存在nouveau驱动，跳过此步骤即可。

   ```
   # touch  /etc/modprobe.d/nvidia-installer-disable-nouveau.conf  #创建文件，将以下两行内容保存至文件中

   blacklist nouveau
   options nouveau modeset=0
   ```

3. 安装gcc、kernel-devel、kernel-headers：

   依次执行以下命令，安装gcc、与**内核版本一致**的kernel-devel和kernel-headers。建议使用相同版本的ISO配置本地源安装。

   ```
   # yum install gcc kernel-devel-$(uname -r)  kernel-headers-$(uname -r)     #重构 initramfs 镜像
   # cp /boot/initramfs-$(uname -r).img /boot/initramfs-$(uname -r).img.bak
   # dracut /boot/initramfs-$(uname -r).img $(uname -r) --force       #只使用文本模式重启云主机
   # systemctl set-default multi-user.target
   # init 3
   # reboot
   # lsmod | grep nouveau    #云主机重新启动后，检查nouveau驱动应该没有被使用
   ```

4. 安装NVIDIA 驱动：

将下载的驱动包拷贝至云主机系统内，依次执行以下命令运行驱动文件：

```
# chmod +x NVIDIA-Linux-x86_64-346.47.run    #配置可执行权限
# ./NVIDIA-Linux-x86_64-346.47.run      #运行驱动文件
```

命令执行后将开始解压驱动包并进入安装步骤，安装过程可能出现一些警告，依次确认即可，不影响驱动安装。若出现error报错，请参考表 *1:* 报错处理检查环境。

表 1: 报错处理

报错	解决方案
ERROR: Unable to find the kernel source tree for the currently running kernel. Please make sure you have installed the kernel source files for your kernel and that they are properly configured; on Red Hat Linux systems, for example, be sure you have the 'kernel-source' or 'kernel-devel' RPM installed. If you know the correct kernel source files are installed, you may specify the kernel source path with the '--kernel-source-path' command line option.	需要确保kernel、kernel-headers、kernel-devel是否均已安装，并且版本号完全一致
ERROR: The Nouveau kernel driver is currently in use by your system. This driver is incompatible with the NVIDIA driver, and must be disabled before proceeding. Please consult the ow to correctly disable the Nouveau kernel driver.	需要禁用nouveau驱动
ERROR: Failed to find dkms on the system! ERROR: Failed to install the kernel module through DKMS. No kernel module was installed; please try installing again without DKMS, or check the DKMS logs for more information.	需要安装DKMS，它可以帮我们维护内核外的驱动程序，在内核版本变动之后可以自动重新生成新的模块
ERROR: Unable to load the kernel module 'nvidia.ko'. This happens most frequently when this kernel module was built against the wrong or improperly configured kernel sources, with a version of gcc that differs from the one used to build the target kernel, or if a driver such as rivafb, nvidiafb, or nouveau is present and prevents the NVIDIA kernel module from obtaining ownership of the NVIDIA graphics device(s), or no NVIDIA GPU installed	执行命令./NVIDIA-Linux-x86_64-384.98.run --kernel-source-path=/usr/src/kernels/3.10.0-XXX.x86_64/ -k $(uname -r)即可

报错	解决方案
in this system is supported by this NVIDIA Linux graphics driver release.	

5. 检查驱动安装情况：

   分别执行以下两条命令，检查驱动安装情况。若返回结果能够显示显卡的型号信息，说明驱动已经安装成功。

```
# lspci |grep NVIDIA
# nvidia-smi
```

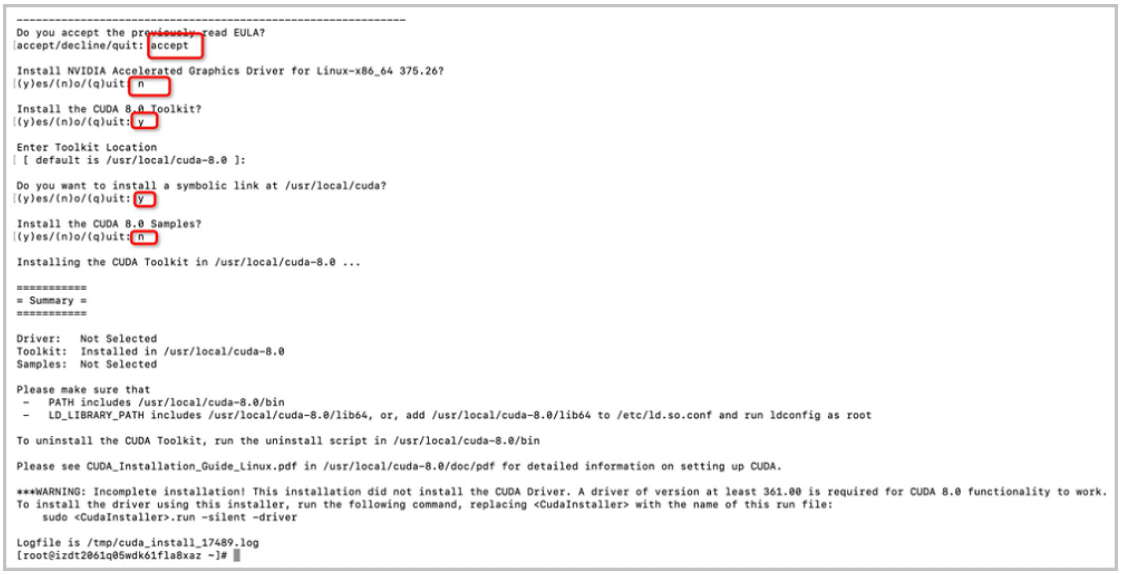
6. 安装CUDA toolkit：

   将下载的驱动包拷贝至云主机系统内，依次执行以下命令执行驱动文件：

```
# chmod +x cuda_8.0.61_375.26_linux.run      #配置可执行权限
# ./cuda_8.0.61_375.26_linux.run     #运行驱动文件
```

   安装过程需要配置一些参数，请参考下图进行配置，如图 *8: 安装CUDA toolkit*所示：

图 8: 安装CUDA toolkit



7. 配置环境变量：

   执行vim /root/.bashrc命令，将以下内容保存至此文件，完成环境变量配置：

```
#gpu driver
export CUDA_HOME=/usr/local/cuda-8.0
export PATH=/usr/local/cuda-8.0/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-8.0/lib64:$LD_LIBRARY_PATH
```

```
export LD_LIBRARY_PATH="/usr/local/cuda-8.0/lib:${LD_LIBRARY_PATH}"
```

环境变量添加完成后立即生效，可执行以下命令进行验证测试：

```
# source ~/.bashrc
# cd /usr/local/cuda-8.0/samples/1_Utilities/deviceQuery
# make
# ./deviceQuery
```

5 典型应用场景

GPU透传功能通过云主机透传物理机强劲的GPU计算能力，可适用于3D渲染、人工智能、云游戏、VDI等典型应用场景。

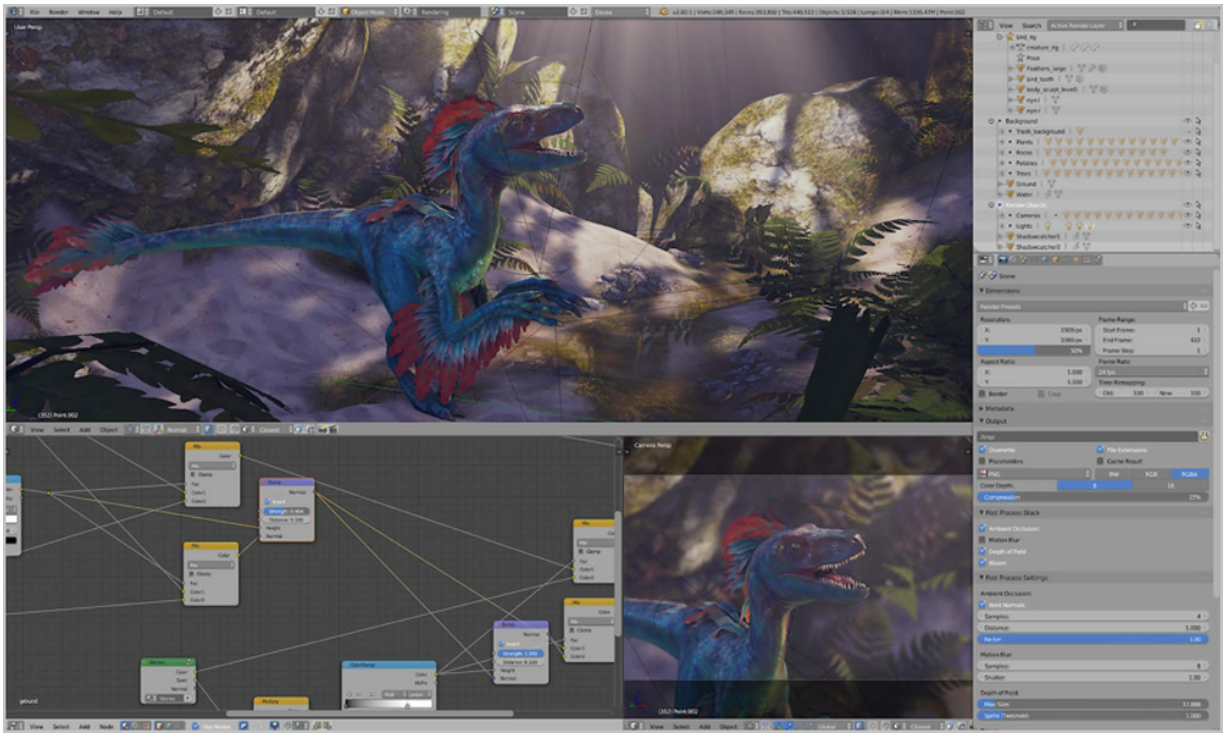
3D渲染

三维计算机图形的预渲染（Pre-rendering、Offline rendering）或实时渲染（Real-time rendering、Online rendering）速度都很缓慢。预渲染常用于电影制作，要求很高的计算强度，需要大量的服务器提供运算能力；实时渲染常用于三维视频游戏，通常依靠图形处理器（GPU）完成这个过程。

现在由于GPU的高速发展，已经有相当多的3D渲染是在GPU服务器集群中完成。结合ZStack的GPU透传功能，在性能损失极低的情况下（5%以内）同时可获得集中高效的集群管理功能，再配合智能监控软件以及ZStack自带的计费功能，可以形成一整套更便捷高效的渲染农场方案。

如图 *9: 3D渲染*所示：

图 9: 3D渲染



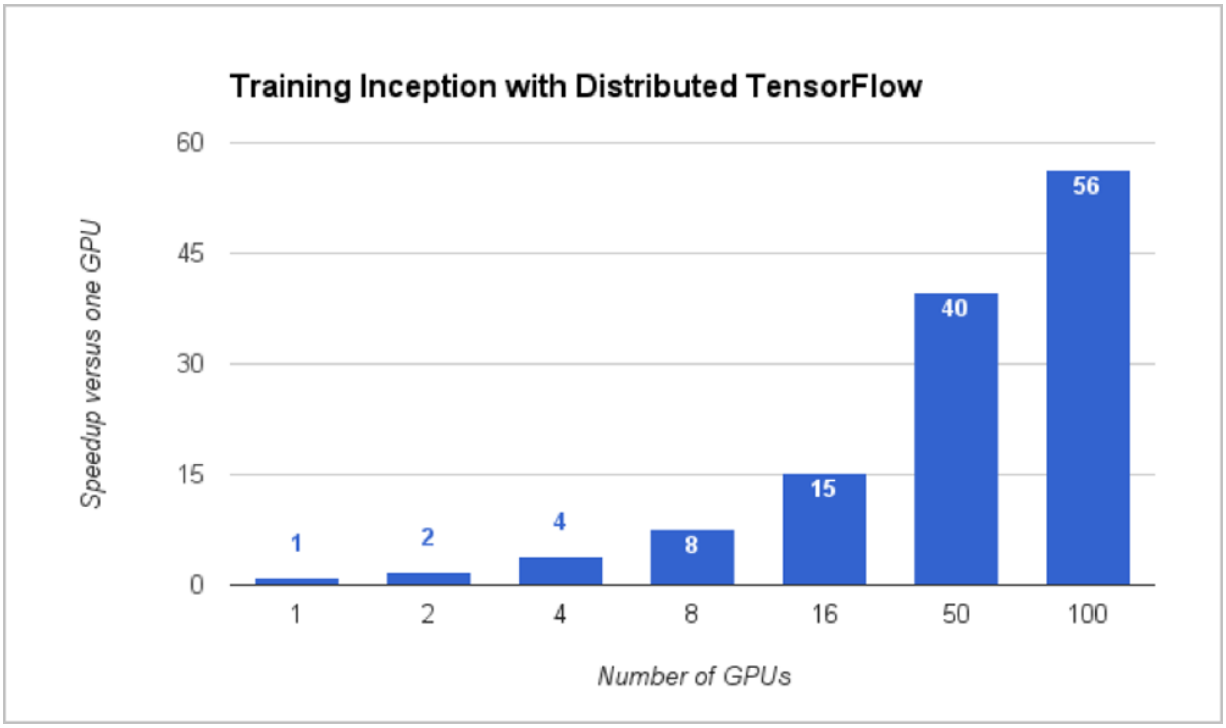
人工智能

GPU的计算能力可以应用于深度学习。自Google推出神经网络工具TensorFlow后，许多科研机构以及企业应用都日渐明显偏向使用GPU作为基础设施。

以规格较高的NVIDIA P100显卡为例，通过ZStack的GPU透传功能，将其透传至云主机后，性能测试结果显示，几乎与标称完全一致，能够充分满足大规模模型训练对基础设施的要求。

如图 *10:* 人工智能所示：

图 10: 人工智能



云游戏

随着宽带网络的发展，以及移动终端设备的普及，将游戏计算至于云端，客户端仅仅负责显示与控制的游戏模式也悄然开始流行。云端服务器上渲染3D游戏，即时为每一帧进行编码，将结果以流的形式传输至任何接驳有线或无线网络的设备。

这种云游戏模式，可以借助GPU以及服务器CPU能力，通过ZStack的GPU透传功能，为游戏创造隔离性更佳的虚拟环境，从而保证计算与渲染的流畅度，为用户提供更好的游戏体验。

如图 *11:* 云游戏所示：

图 11: 云游戏

VDI（**桌面云**）

GPU一直是VDI（桌面云）中非常重要的设备，它不仅能够改善桌面视觉体验，同时在特殊的应用程序中承担主力计算角色，从而完全代替传统PC图站，让用户在更为安全的环境中进行3D设计。

通过ZStack的GPU透传功能，以及配合RDP、PCoIP等协议，可充分利用显卡能力，比如3D设计、游戏等流畅运行，提供更逼近本地物理机的用户体验。

如图 *12: VDI#桌面云#*所示：

图 12: VDI（**桌面云**）

术语表

区域（Zone）

ZStack中最大的一个资源定义，包括集群、二层网络、主存储等资源。

集群（Cluster）

一个集群是类似物理主机（Host）组成的逻辑组。在同一个集群中的物理主机必须安装相同的操作系统（虚拟机管理程序，Hypervisor），拥有相同的二层网络连接，可以访问相同的主存储。在实际的数据中心，一个集群通常对应一个机架（Rack）。

管理节点（Management Node）

安装系统的物理主机，提供UI管理、云平台部署功能。

计算节点（Compute Node）

也称之为物理主机（或物理机），为云主机实例提供计算、网络、存储等资源的物理主机。

主存储（Primary Storage）

用于存储云主机磁盘文件的存储服务器。支持本地存储、NFS、 Ceph、Shared Mount Point、Shared Block类型。

镜像服务器（Backup Storage）

也称之为备份存储服务器，主要用于保存镜像模板文件。建议单独部署镜像服务器。支持ImageStore、Sftp（社区版）、Ceph类型。

镜像仓库（Image Store）

镜像服务器的一种类型，可以为正在运行的云主机快速创建镜像，高效管理云主机镜像的版本变迁以及发布，实现快速上传、下载镜像，镜像快照，以及导出镜像的操作。

云主机（VM Instance）

运行在物理机上的虚拟机实例，具有独立的IP地址，可以访问公共网络，运行应用服务。

镜像（Image）

云主机或云盘使用的镜像模板文件，镜像模板包括系统云盘镜像和数据云盘镜像。

云盘（Volume）

云主机的数据盘，给云主机提供额外的存储空间，共享云盘可挂载到一个或多个云主机共同使用。

计算规格（Instance Offering）

启动云主机涉及到的CPU数量、内存、网络设置等规格定义。

云盘规格（Disk Offering）

创建云盘容量大小的规格定义。

二层网络（L2 Network）

二层网络对应于一个二层广播域，进行二层相关的隔离。一般用物理网络的设备名称标识。

三层网络（L3 Network）

云主机使用的网络配置，包括IP地址范围、网关、DNS等。

公有网络（Public Network）

由因特网信息中心分配的公有IP地址或者可以连接到外部互联网的IP地址。

私有网络（Private Network）

云主机连接和使用的内部网络。

L2NoVlanNetwork

物理主机的网络连接不采用Vlan设置。

L2VlanNetwork

物理主机节点的网络连接采用Vlan设置，Vlan需要在交换机端提前进行设置。

VXLAN网络池（VXLAN Network Pool）

VXLAN网络中的 Underlay 网络，一个 VXLAN 网络池可以创建多个 VXLAN Overlay 网络（即 VXLAN 网络），这些 Overlay 网络运行在同一组 Underlay 网络设施上。

VXLAN网络（VXLAN）

使用 VXLAN 协议封装的二层网络，单个 VXLAN 网络需从属于一个大的 VXLAN 网络池，不同 VXLAN 网络间相互二层隔离。

云路由（vRouter）

云路由通过定制的Linux云主机来实现的多种网络服务。

安全组（Security Group）

针对云主机进行第三层网络的防火墙控制，对IP地址、网络包类型或网络包流向等可以设置不同的安全规则。

弹性IP（EIP）

公有网络接入到私有网络的IP地址。

快照（Snapshot）

某一时间点某一磁盘的数据状态文件。包括手动快照和自动快照两种类型。