

弹性云主机 ECS

目录

最佳实践

云主机基础操作实践.....	2
Windows基础最佳实践.....	26
Linux基础最佳实践.....	48
搭建网站/应用最佳实践.....	71
搭建环境相关最佳实践.....	160
网络相关最佳实践.....	187
数据库相关最佳实践.....	197
云主机迁移相关实践.....	282
OpenClaw 部署最佳实践.....	287
部署Palworld幻兽帕鲁服务器最佳实践.....	328
大模型学习机最佳实践.....	353
AIGC实践.....	419

云主机基础操作实践

麒麟系统云主机配置图形化界面

操作场景

为了提供纯净的弹性云云主机系统给客户，麒麟系统云主机默认没有安装图形化界面，如果您需要图形化界面，请参见本节内容进行安装。

操作步骤

1. 配置网络，确保yum源可以正常使用。银河麒麟云主机操作系统在有外网的环境下可以直接使用yum源，而在无外网的环境下需要挂载镜像作为本地源。
2. 在字符界面执行以下命令，列出可安装的图形化包组。

```
#yum group list
```

```
[root@ec2-8411 ~]# yum group list
Kylin Linux Advanced Server 10 - Os                98 kB/s | 10 MB    01:46
Kylin Linux Advanced Server 10 - Updates          135 kB/s | 12 MB    01:30
Last metadata expiration check: 0:00:02 ago on Fri 25 Aug 2023 11:35:21 AM CST.
Available Environment Groups:
  Server
  File and Print Server
  Basic Web Server
  Virtualization Host
  Server with UKUI GUI
Installed Environment Groups:
  Minimal Install
Available Groups:
  Container Management
  Development Tools
  Headless Management
  Legacy UNIX Compatibility
  Network Servers
  Scientific Support
  Security Tools
  System Tools
  Smart Card Support
```

3. 在字符界面执行以下命令，安装图形化桌面相关包。

```
#yum groupinstall -y "Server with UKUI GUI"
```

或者

```
#yum groupinstall -y "带GUI的云主机"
```

最佳实践

```
dnsnsg-help-2.82-10.ky10.x86_64
emacs-1:26.1-13.p02.ky10.x86_64
expat-devel-2.2.9-8.ky10.x86_64
fcitx-gtk2-4.2.9.1-2.p05.ky10.x86_64
fcitx-gtk3-4.2.9.1-2.p05.ky10.x86_64
fcitx-libs-4.2.9.1-2.p05.ky10.x86_64
freetype-devel-2.10.2-4.ky10.x86_64
ghostscript-9.52-6.p01.ky10.x86_64
git-2.27.0-14.ky10.x86_64
gufs-1.40.2-6.p03.ky10.x86_64
java-1.8.0-openjdk-headless-1:1.8.0.342.b07-0.p02.ky10.x86_64
kernel-core-4.19.90-23.37.u2101.ky10.x86_64
kernel-modules-extra-4.19.90-23.37.u2101.ky10.x86_64
kernel-modules-internal-4.19.90-23.37.u2101.ky10.x86_64
kfs-kwindowsystem-5.55.0-1.ky10.01.ky10.x86_64
libitm-7.3.0-20190804.h30.p03.ky10.x86_64
libtntative-1-0-1.2.23-2.ky10.x86_64
libuio-1:1.42.0-1.ky10.x86_64
marco-1.16.1-4.p01.ky10.x86_64
nariadb-common-3:10.3.35-1.p01.ky10.x86_64
nminzip-1.2.11-20.ky10.x86_64
openldap-servers-2.4.50-7.ky10.x86_64
perl-DBI-1.643-2.ky10.x86_64
poppler-0.67.0-8.ky10.x86_64
poppler-glib-0.67.0-8.ky10.x86_64
python3-rtplib-2.1.70-4.ky10.noarch
qt5-qtsvg-5.11.1-6.ky10.x86_64
sqlite-devel-3.32.3-6.ky10.x86_64
target-restore-2.1.70-4.ky10.noarch
tzdata-java-2020a-1.p02.ky10.noarch
uorbis-tools-1:1.4.0-31.ky10.x86_64
dracut-squash-050-3.p04.se.07.ky10.x86_64
emacs-common-1:26.1-13.p02.ky10.x86_64
fcitx-data-4.2.9.1-2.p05.ky10.noarch
fcitx-gtk3-4.2.9.1-2.p05.ky10.x86_64
freetds-1.00.38-8.ky10.x86_64
freibidi-devel-1.0.10-2.ky10.x86_64
giflib-5.1.4-6.p01.ky10.x86_64
gststreamer1-plugins-bad-free-1.16.2-2.ky10.x86_64
gufs-client-1.40.2-6.p03.ky10.x86_64
java-11-openjdk-headless-1:11.0.17.8-1.ky10.x86_64
kernel-modules-4.19.90-23.37.u2101.ky10.x86_64
kernel-modules-internal-4.19.90-23.37.u2101.ky10.x86_64
libexif-0.6.21-24.ky10.x86_64
libpsl-devel-0.21.1-1.ky10.x86_64
libadisks2-2.8.1-4.1.p01.ky10.x86_64
libmnc-2.31.0-1.p01.ky10.x86_64
mariadb-3:10.3.35-1.p01.ky10.x86_64
mariadb-errmsg-3:10.3.35-1.p01.ky10.x86_64
openjpeg2-2.3.1-6.ky10.x86_64
paho-c-1.3.2-1.p02.ky10.x86_64
perl-Git-2.27.0-14.ky10.noarch
poppler-cpp-0.67.0-8.ky10.x86_64
postgresql-libs-10.5-23.p02.ky10.x86_64
qt-1:4.8.7-47.p04.ky10.x86_64
speex-1.2.0-5.ky10.x86_64
system-config-printer-libs-1.5.11-16.01.p01.ky10.noarch
targetcli-help-2.1.54-1.ky10.noarch
udisks2-2.8.1-4.1.p01.ky10.x86_64
xz-devel-5.2.5-2.ky10.x86_64
Skipped:
ntp-4.2.8p14-5.p01.ky10.x86_64
Complete!
```

4. 在字符界面执行以下命令，设置系统默认的启动方式为graphical.target，即图形化启动。

```
#systemctl set-default graphical.target
```

```
[root@ec2-8411 ~]# systemctl set-default graphical.target
Removed /etc/systemd/system/default.target.
Created symlink /etc/systemd/system/default.target → /usr/lib/systemd/system/graphical.target.
[root@ec2-8411 ~]#
```

5. 在字符界面执行以下命令，重启系统使其生效。

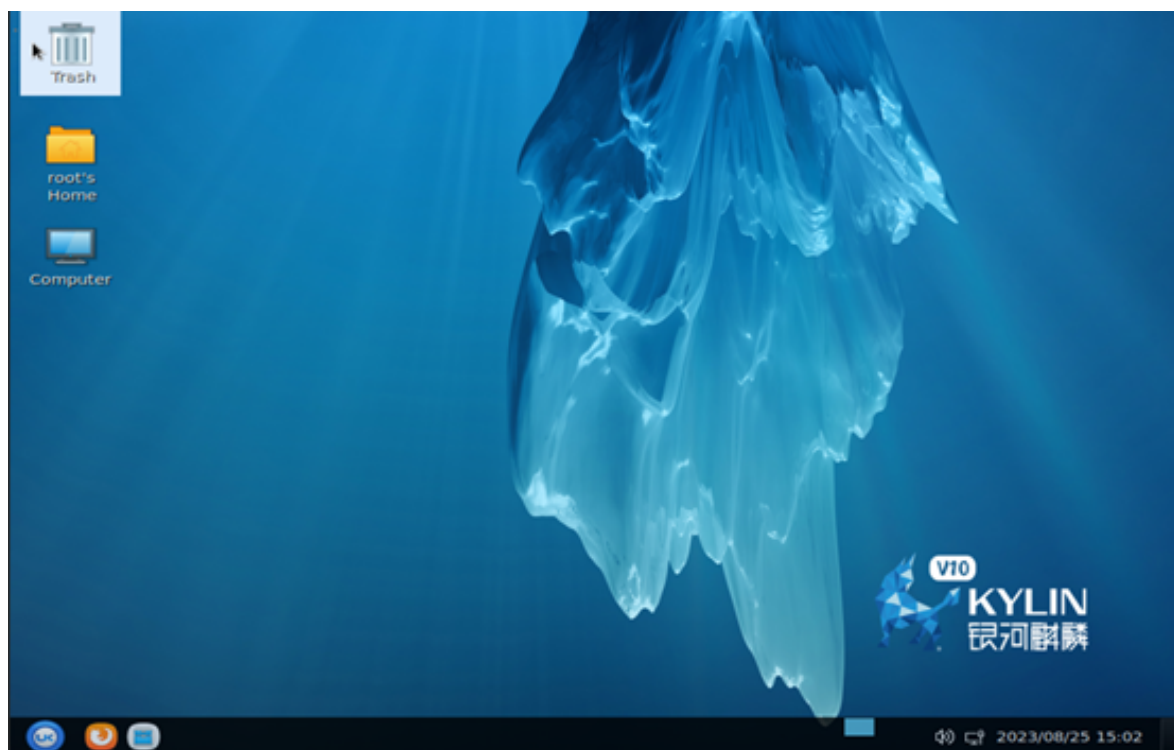
```
#reboot
```

最佳实践

6. 重启系统后即可进入图形化界面。



7. 输入用户名与密码，进入系统。



创建云主机最佳实践

操作场景一

通过控制台开通单台云主机。

操作步骤

本文以南昌5资源池为例，创建单台Linux/Windows云主机。

计算控制台点击“弹性云主机”，点击右上角“创建云主机”。

“基础配置”页

1. 计费模式选择按量付费；
2. 地域选择江西-南昌5；
3. 可用区选择可用区1；
4. 企业项目选择default；
5. 虚拟私有云（VPC）选择默认default-vpc；
6. 实例名称和主机名称按照要求填写即可；
7. 选择所需开通规格，本次以s7.xlarge.2为例；
8. 选择所需镜像类型。

“网络配置”页

1. 网卡会关联上述选择的虚拟私有云（VPC）下的子网，选择相应的子网即可；
2. 安全组可以选择default安全组；
3. 弹性IP选择“不使用”。

“高级配置”页

1. 登录方式默认为密码，创建密码为稍后创建；
2. 用户数据选择“暂不配置”；
3. 左下角购买量选择1台。

“确认配置”页

确认上述选择和填写的配置项符合预期，勾选“已阅读并同意相关协议”，点击右下角的“立即购买”完成订购。

对于单台Linux/Windows云主机，通常情况下可在一分钟内完成开通。本次单台Linux云主机开通耗时17秒，单台Windows云主机开通耗时16秒。

订单号: 20240826170006268242 订单类型: 订购 创建时间: 2024-08-26 17:00:13 更新时间: 2024-08-26 17:00:30

已完成
订单开通构成:

已下单 已交付 开通中 已完成

产品	配置	订购数量	所属资源池	规格	金额 (元)
弹性云主机(云主机(OVMFS))	CPU: 4核 内存: 8G 系统盘: 40GB 操作系统: Linux	1	南昌5	按需计费	0.00元
云硬盘(块存储)	磁盘类型: 普通IO 容量: 40GB 磁盘: 云硬盘				0.00元

最佳实践



操作场景二

通过控制台开通多台云主机。

操作步骤

本文以南昌5资源池为例，批量创建20台Linux云主机。

控制台点击“弹性云主机”，点击右上角“创建云主机”。

“基础配置”页

1. 计费模式选择按量付费；
2. 地域选择江西-南昌5；
3. 可用区选择可用区1；
4. 企业项目选择default；
5. 虚拟私有云（VPC）选择默认自己创建的VPC；
6. 实例名称和主机名称按照要求填写即可；
7. 选择所需规格，本次以s7.xlarge.2为例；
8. 选择所需镜像类型。

“网络配置”页

1. 网卡会关联上述选择的虚拟私有云（VPC）下的子网，选择相应的子网即可；
2. 安全组可以选择default安全组；
3. 弹性IP选择“不使用”。

“高级配置”页

1. 登陆方式默认为密码，创建密码为稍后创建；
2. 用户数据选择“暂不配置”；
3. 左下角购买量选择20台。

“确认配置”页

确认上述选择和填写的配置项符合预期，勾选“已阅读并同意相关协议”，点击右下角的“立即购买”完成订购。

对于多台Linux云主机，通常情况下可实现分钟级开通。本次开通20台Linux云主机耗时22秒。

最佳实践

订单号: 20240825171702004508 订单类型: 订购 创建时间: 2024-08-26 17:17:07 更新时间: 2024-08-26 17:17:29

产品	配置	订购数量	所属资源池	周期	金额 (元)
云硬盘(非IOPS优化)	磁盘类型: 普通IO 容量: 40GB 磁盘: 云硬盘	20	高5		¥0.02
弹性云服务器(ECS)	CPU: 4核 内存: 8G 系统盘: 40GB 操作系统: Linux			按量计费	¥0.02

操作场景三

通过OpenAPI接口并发开通1000台云主机。

接口约束

1. 目前不支持预付费账户创建按需付费类型云主机。
2. 计费模式选择包年包月计费方式时，需要填写订购周期类型与订购时长。
3. 自动分配弹性IP (extIP="1") 时，需要填写弹性IP版本 (ipVersion) 与带宽大小 (bandwidth)；使用已有的弹性IP (extIP="2") 时，需要填写弹性IP的版本 (ipVersion)，和对应弹性IP的ID (eipID或ipv6AddressID)。
4. 挂载网卡时，子网与虚拟私有云存在对应关系，确保子网属于当前虚拟私有云。
5. 云主机绑定多个标签时，标签键 (参数labelKey) 不可重复，单台云主机最多可绑定10个标签。

操作步骤

本文以南昌5资源池为例，创建1000台按需计费的云主机，其中每批创建50台，并发创建20批次。

1. 计算控制台点击“镜像服务”，在公共镜像Tab选择CentOS Linux 7.6 64位镜像，获取镜像ID。
2. 选择需要开通的云主机规格，本次使用s7.large.2，根据OpenAPI的[查询云主机规格资源接口](#)获取该规格对应的flavorID。
3. 网络控制台点击“虚拟私有云”，创建虚拟私有云 (VPC)，同时在该虚拟私有云 (VPC) 下创建subnet，保证该subnet可使用的IP数量超过1000个。
4. 在官网帮助中心查找OpenAPI[批量创建云主机接口](#)。
5. 批量开通函数封装。

```
@atomic.action_timer("openapi.batch_create_instance")
def _batch_create_instance(self, region, az, flavor, image, vpc, subnet,
    ext_ip, disk_type, disk_size,ondemand, order_count, **kwargs):
    """Returns user servers list."""
    body = self.client.servers.batch_create(region, az, flavor, image, vpc, subnet,
        ext_ip, disk_type, disk_size,ondemand, order_count, **kwargs)
    order = body["returnObj"]["masterOrderID"]
    res = self._query_uuid(order, interval=2, timeout=600)
    return res
```

```
def batch_create(self, region, az, flavor, image, vpc, subnet,
    ext_ip, disk_type, disk_size, ondemand, order_count,
```

```
    **kwargs):
name = "s-rally-" + self._random_digits(5)
params = {
    "clientToken": str(uuid.uuid4()),
    "regionID": region,
    "azName": az,
    "instanceName": name,
    "displayName": name,
    "flavorID": flavor,
    "imageType": 1,
    "imageID": image,
    "bootDiskType": disk_type,
    "bootDiskSize": disk_size,
    "onDemand": ondemand,
    "orderCount": order_count,
    "vpcID": vpc,
    "extIP": ext_ip,
    "networkCardList": [
        {
            "subnetID": subnet,
            "isMaster": True
        }
    ],
}
params.update(kwargs)
return self._post("/v4/ecs/batch-create-instances", params=params)
```

6. 编写yaml执行脚本。

- scenario选择上述编写的batch_create_instance方法；
- region、az对应南昌5资源池ID和可用区；
- flavor、image为上述选择的规格和镜像；
- vpc、subnet是上述创建的虚拟私有云和子网；
- ext_ip默认为0，不创建弹性IP；
- disk_type选择SATA普通IO，disk_size选择40；
- ondemand选择true，选择创建按需云主机；
- projectID为相关的企业项目；
- order_count选择50，表示单批次批量创建50台云主机；

最佳实践

- runner中times设置20，代表共批量创建20批次；concurrency设置20，代表并发20。
yaml可执行脚本如图所示。

```
{% set flavor_name = flavor_name or "m1.tiny" %}
{% set image_name = image_name or "CentOS7.5-test" %}
{% set size = size or 20 %}
{% set volume_type = volume_type or "default_volume_type" %}
{% set external_network = external_network or "ext-net" %}
{% set to_image_name = to_image_name or "CentOS7.5" %}
{% set to_flavor_name = to_flavor_name or "m1.small" %}
{% set volume_size = volume_size or 20 %}
{% set enable_instance_backup = enable_instance_backup or false %}
{% set regionID = regionID or "200000002527" %}

---
version: 2
title: Task for SDK test
description: >
  Test suit for SDK test.
subtasks:
  -
    title: SDK ComputeFlavors.create_flavor
    scenario:
      #OpenAPI.list_servers_eop:
      OpenAPI.batch_create_instance:
        region: "200000002527"
        az: "cn-jx-nc5-jxnc1A-public-ctcloud"
        flavor: "bf5e0057-6f08-c886-82f6-dd944e5fd531"
        image: "34c74243-dd70-46ed-901d-093965a8b0ec"
        vpc: "vpc-dksqlk6aml"
        subnet: "subnet-vfaps9aszl"
        ext_ip: "0"
        disk_type: "SATA"
        disk_size: 40
        ondemand: true
        projectID: "6d94f8f6396a48858d92e91ac53d4b0d"
        order_count: 50
    runner:
      constant:
        times: 20
        concurrency: 20
```

- 部署相关脚本到对应的运行机器，执行yaml脚本，开始创建云主机。

```
rally task start ***.yaml
```

最佳实践

8. 当20个订单创建完成后，可在控制台查看“运行中”的云主机数量。对于千台云主机，通常情况下可在十分分钟内完成开通。本次批量开通千台云主机耗时138秒，开通成功率100%。



弹性云主机配置

背景介绍

为了帮助用户最大程度地安全、可靠地使用弹性云主机，通常可以从以下几个方面进行弹性云主机配置的最佳实践：

- 考虑业务负载需求
- 网络与安全
- 数据备份与恢复策略
- 系统监控与告警

考虑业务负载需求

在选择弹性云主机时，需要根据应用程序的需求选择适当的实例类型和规格，考虑性能需求、应用需求以及成本效益等因素。

性能需求

性能需求是选择弹性云主机配置的首要考虑因素。性能需求包括计算性能、存储性能和网络性能。

- 计算性能越强，服务器能够同时处理的请求就越多。如果应用需要大量计算资源，比如进行大规模数据分析、科学计算等，就需要选择配置较高的CPU和内存。
- 存储性能是指服务器对数据的读写速度。对于需要频繁读写大量数据的应用，如数据库、视频处理等，需要选择配置较高的存储设备。
- 网络性能是指服务器的网络带宽和延迟。对于需要大量数据传输的应用，如视频直播、游戏等，需要选择具有较高网络带宽的配置。此外，延迟也是一个重要指标，特别是对于对实时性要求较高的应用，如在线交易、远程会议等。

应用需求

不同的应用有不同的特点和需求，因此需要选择适合应用的配置。

- 操作系统：弹性云主机支持多种操作系统，如Windows、Linux等。选择操作系统需要根据应用的实际需求来决定，比如是否支持特定软件、是否需要特定的开发环境等。
- 数据库：如果应用需要使用数据库，需要考虑数据库的类型和规模。对于大型数据库，需要选择配置较高的服务器，以保证数据库的性能和稳定性。

- 应用软件：如果应用需要运行特定的软件，比如Web服务器、应用服务器、大数据平台等，需要选择配置适合运行这些软件的服务器。

成本效益

在云计算领域，用户只需按需购买和使用资源，不需要投资大额资金购买硬件设备，因此成本效益非常重要。

- 价格：弹性云主机的价格与配置相关，一般来说，配置越高，价格越高。在选择配置时，需要根据应用的需求和预算来综合考虑。
- 弹性伸缩：弹性云主机搭配弹性伸缩使用，可以根据应用的需求进行资源灵活调整。在选择配置时，可以考虑未来应用的发展和变化。可以搭配负载均衡实现业务负载的均衡分布。

选择弹性云主机配置需要综合考虑性能需求、应用需求和成本效益等因素，选择最适合业务负载需求的云主机。

网络与安全

弹性云主机通常需要与网络结合使用，因此需要重点考虑到网络设计安全性，同时需要配置多项安全措施保障云主机的安全。

网络逻辑隔离

虚拟私有云提供了一个隔离的虚拟网络环境，可以在云平台上创建自己的私有网络。将弹性云主机部署在虚拟私有云中，可以更好地控制网络访问和保护数据的安全性。主要通过下列方法进行网络安全设置：

- 使用安全组：安全组是一种虚拟防火墙，用于控制弹性云主机的网络流向。配置适当的安全组规则，限制入站和出站流量，只允许必要的端口和协议访问。配置详细步骤参见[弹性云主机-安全组配置示例](#)。
- 使用网络ACL：网络ACL是一种在子网级别控制流量的安全策略。通过配置网络ACL规则，实现对子网内所有ECS实例的访问控制，防止未经授权的访问。配置详细步骤参见[虚拟私有云-创建ACL](#)。

安全登录

为了确保弹性云主机创建成功且状态正常，建议您第一次通过VNC方式远程登录弹性云主机，若登录成功则表明弹性云主机资源状态正常。

尽量选用密钥方式登录Linux弹性云主机。同时可以采取相应措施对Linux服务器SSH登录进行安全加固，详细操作，请参见[Linux服务器SSH登录的安全加固](#)。

使用密码登录的实例需要定期修改密码。

详细操作，请参见：

[弹性云主机-登录Linux弹性云主机](#)

[弹性云主机-登录Windows弹性云主机](#)

安全产品配置

安全防护：为了实现弹性云主机的高阶安全防护，建议您根据防护需求开通服务器安全卫士服务，服务器安全卫士通过对主机信息和行为进行持续监控和分析，快速精准地发现安全威胁和入侵事件，保证云主机安全。具体操作参见[服务器安全卫士-快速入门](#)。

网络防护：为了防止弹性云主机绑定的弹性公网IP被攻击，建议您配合使用Anti-DDoS服务，具体操作参见[Anti-DDoS流量清洗-开启Anti-DDoS防护](#)。

数据备份与恢复策略

数据备份与恢复

为了在发生病毒入侵、人为误删除、软硬件故障等事件时，能够快速恢复数据，建议您定期进行数据备份，可以使用天翼云云备份服务，云备份提供简单易用的备份能力，当发生异常时，可将数据恢复到任意备份点，减轻经济损失。具体操作参见[云备份-快速入门](#)。

高可用容灾业务部署

为了实现较高的业务容灾能力，建议您将云主机资源部署在同一区域的不同可用区内。具体详情参见[地域和可用区](#)。

系统监控与告警

监控管理

为了使用户能够及时获取弹性云主机的各项性能指标及运行状态，建议您使用云监控服务，对实例进行实时监控，实例异常时进行告警和通知。详细内容参见[云监控服务](#)。

日志管理

为了提高弹性云主机和应用程序的可用性和性能，建议您通过日志审计服务收集日志数据，通过对海量日志数据的分析和处理，实现实时决策分析、设备运维管理、用户业务趋势分析等能力。

创建双栈云主机

1. 目前双栈云主机未面向线上用户开放，线下客户请向云公司受理岗申请双栈公测权限，注明账号名称，账号归属（哪个分公司、哪个客户），公测的资源池。

最佳实践

2. 双栈是针对子网开启的，需在VPC内创建一个新的双栈子网，或对原有子网开启IPv6。申请完双栈公测权限后，创建子网时可看到选项。

创建子网 ✕

⚠️ 标准裸金属子网 仅限于绑定标准裸金属，不可绑定云主机与弹性裸金属；标准裸金属子网下虚拟IP仅可绑定标准裸金属。

* VPC:
当前VPC还可以创建 5 个子网。

网段 192.168.0.0/16(主)

可用区 全部
子网在全部可用区下部署，子网在全部可用区下流量互通。

* 子网类型 普通子网 标准裸金属子网

* 子网名称 ?

描述

* 子网网段 / ?

* 网关

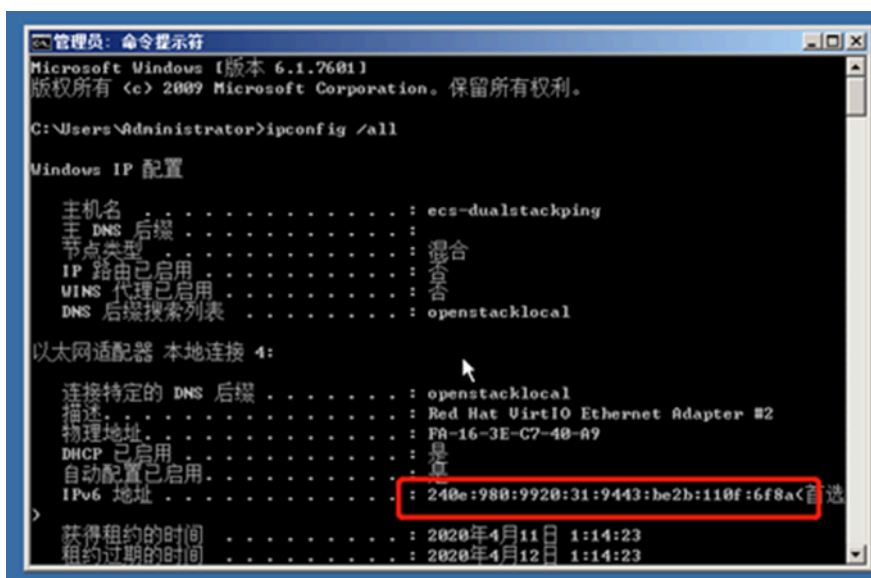
子网IPv6网段 开启IPv6 ?

* DNS服务器地址: [重置](#) ?

3. 云主机必须创建在已开启IPv6的子网中，云主机目前只有在新创建的时刻支持开启双栈，存量主机暂无法增加IPv6地址。
4. 新建一台云主机，选择支持双栈的规格，并非所有主机规格都支持双栈，具体见后续列表，但也存在双栈POD中主机资源售罄而无法开出，如果1-4步都正常走完，选择支持的主机类型还是看不到IPv6选项，就是双栈资源售罄了。

最佳实践

- 弹性IP仅用于访问IPv4公网，如果不访问IPv4公网，无需使用。
- 牢记：弹性IP与双栈功能没有关系，不要把弹性IP的NAT64转换功能与双栈混为一谈。双栈云主机是通过云主机上绑定的双栈网卡获得内网IPv4地址、内外合一的IPv6地址。如果访问IPv4公网，就申请弹性IP；如果访问IPv6公网，就把双栈网卡的IPv6地址加入共享带宽。
- 申请双栈主机成功后，主机将获得IPv6地址，此地址是全球唯一的，内网公网都相同，但此时只支持内网访问。
- 以上都操作完成后，可登录主机尝试使用 `ipconfig /all` 或 `ifconfig` 来查看主机是否获取了该IPv6地址。如果没有，需要手动开启IPv6协议。



```
管理员: 命令提示符
Microsoft Windows [版本 6.1.7601]
版权所有 (c) 2009 Microsoft Corporation。保留所有权利。

C:\Users\Administrator>ipconfig /all

Windows IP 配置

主机名 . . . . . : ecs-dualstackping
主 DNS 后缀 . . . . . :
节点类型 . . . . . : 混合
IP 路由已启用 . . . . . : 否
WINS 代理已启用 . . . . . : 否
DNS 后缀搜索列表 . . . . . : openstacklocal

以太网适配器 本地连接 4:

   连接特定的 DNS 后缀 . . . . . : openstacklocal
   描述 . . . . . : Red Hat VirtIO Ethernet Adapter #2
   物理地址 . . . . . : FA-16-3E-C7-48-A9
   DHCP 已启用 . . . . . : 是
   自动配置已启用 . . . . . : 是
   IPv6 地址 . . . . . : 240e:980:9920:31:9443:be2b:110f:6f8a<首选>

   获得租约的时间 . . . . . : 2020年4月11日 1:14:23
   租约过期的时间 . . . . . : 2020年4月12日 1:14:23
```

最佳实践

- 如果要访问IPv6公网，将IPv6地址加入共享带宽，并查看网络ACL和安全组的出入方向是否已放通所需IPv6协议。::/0 表示全部IPv6地址，0.0.0.0/0 表示全部IPv4地址。

添加弹性IP

i 弹性IP和IPv6双栈网卡添加到共享带宽后，原本的带宽峰值无效，并和共享带宽相同。
原本的计费方式无效，不额外计流量和宽带费用。
包年/包月弹性IP暂时不支持添加到共享带宽。
因到期导致共享带宽销毁时，弹性IP会从共享带宽移出并重新计费；到期欠费后请及时续订续费。

共享带宽： bandwidth-bf1c

当前共享带宽还可以添加弹性IP数:20

单个共享带宽最多可以添加弹性IP的个数：20

弹性IP：

弹性IP

IPv6双栈网卡

所属VPC：

vpc-csmab

子网：

jumpserver-subnet-152f

请输入网卡的IP地址

<input type="checkbox"/>	IPv6地址	所属VPC	子网	所属实例	所绑定带宽名称
--------------------------	--------	-------	----	------	---------

网络控制台>访问控制>网络ACL，点击“添加规则”。

最佳实践

添加规则



* IP版本

IPv4 IPv6

* 方向

入方向 出方向

* 授权策略:

允许 拒绝

* 优先级:

- 1 + ?

* 协议

TCP

TCP、UDP报文分片后，分片不带有端口信息，需要将端口范围指定为1-65535，不进行端口过滤。

* 端口范围

例如20或20-33

* 源地址:

IP 安全组 前缀列表

::/0

描述

确定

取消

10. 尝试访问网站，正常显示网页则成功。如果失败，可将IPv6 DNS设置为 240c::6666 再尝试。




修改云主机默认远程端口

弹性云主机实例的默认远程端口是22。这个端口是SSH（Secure Shell）协议的默认端口，用于远程登录和管理云主机实例。通过SSH协议，您可以通过终端或SSH客户端与云主机进行安全的远程连接，并执行各种管理任务和操作。

修改Windows系统实例默认远程端口

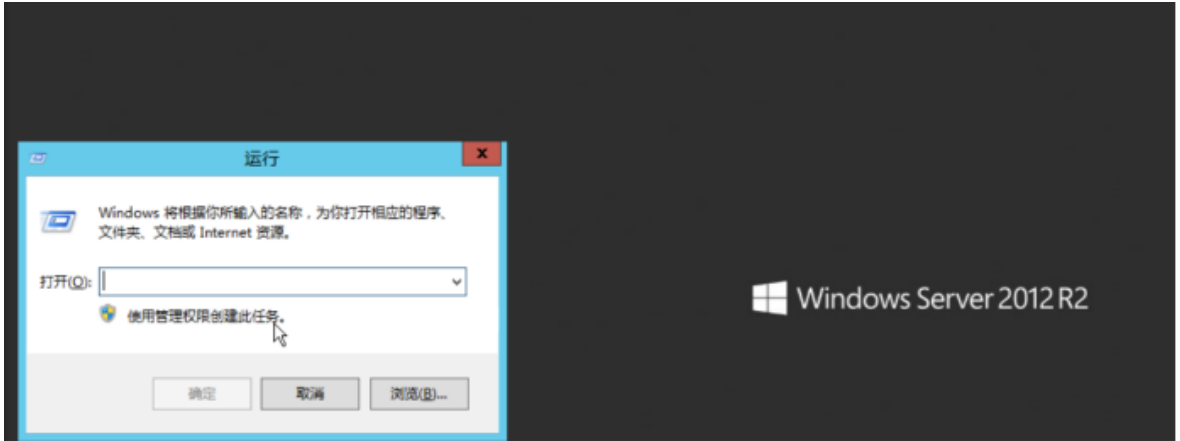
本章以Windows Server 2012 R2为例介绍如何修改Windows系统实例默认远程端口。

1. 登录控制中心。
2. 选择区域  华东-华东1。
3. 单击“计算>弹性云主机”，进入弹性云主机页面。
4. 参考[登录Windows弹性云主机](#)，远程登录待修改端口的Windows弹性云主机实例。

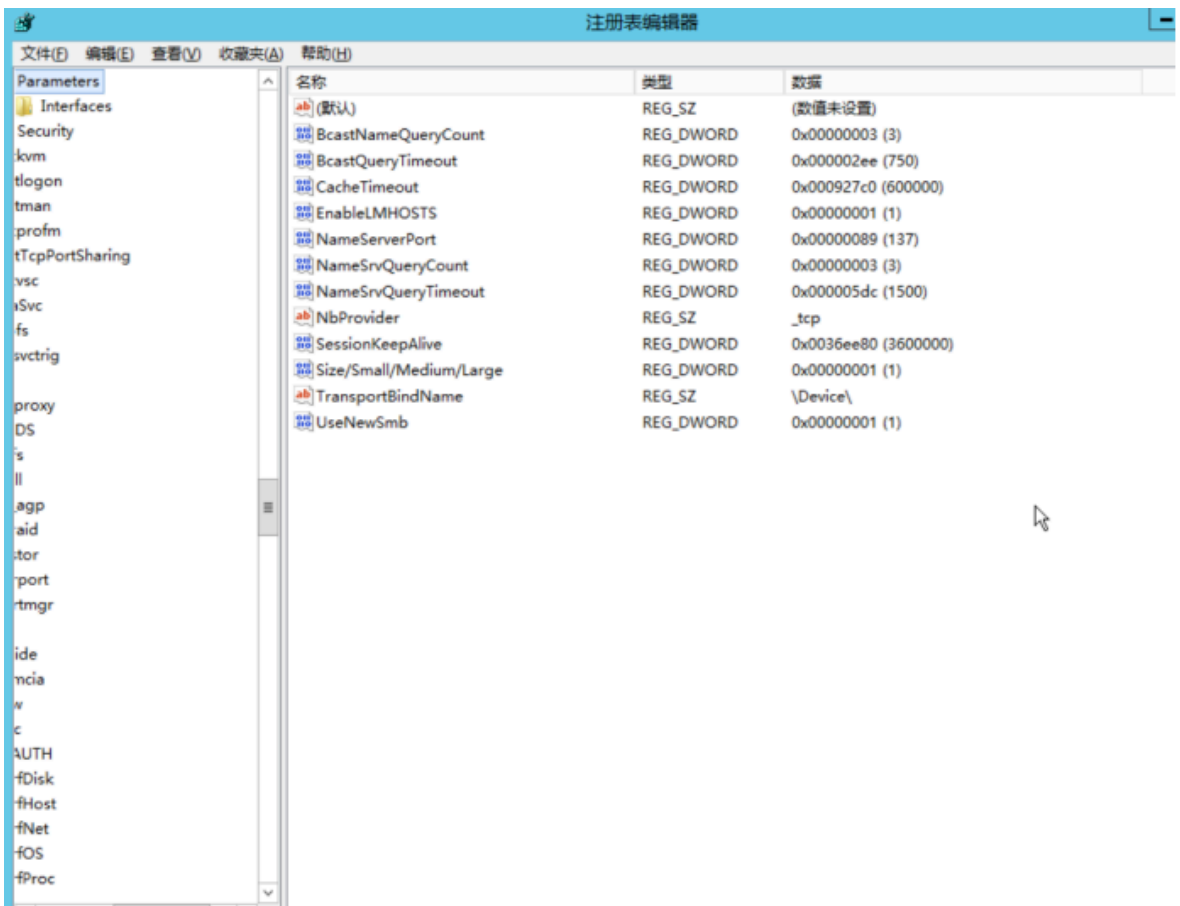
最佳实践

5. 修改注册表子项PortNumber的值。

- a. 右键单击Windows 徽标键，选择“运行”，启动运行窗口。

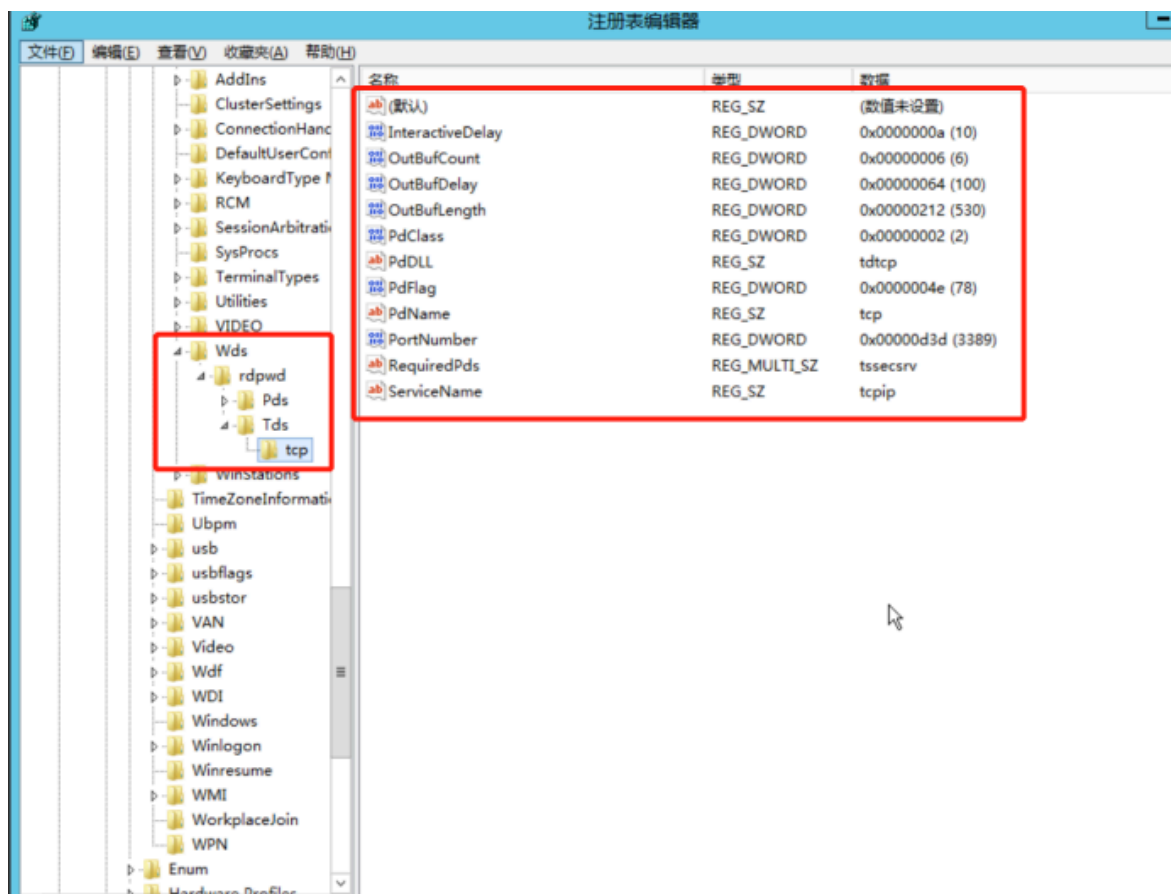


- b. 在运行窗口的文本框内输入regedit.exe后按回车键，打开“注册表编辑器”。

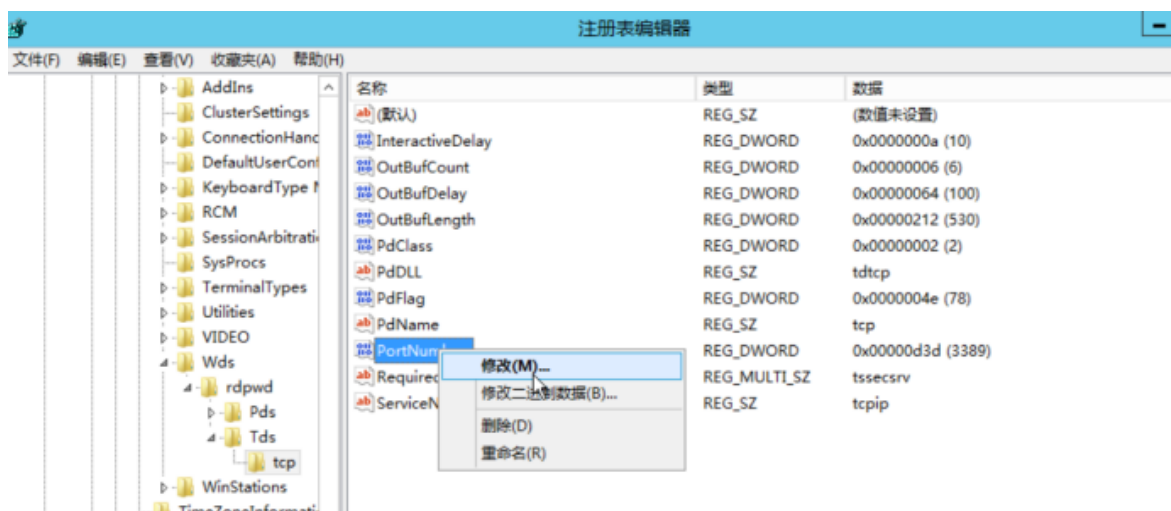


- c. 在左侧导航树中逐层选择HKEY_LOCAL_MACHINE > System > CurrentControlSet > Control > Terminal Server > Wds > rdpwd > Tds > tcp，如下图所示：

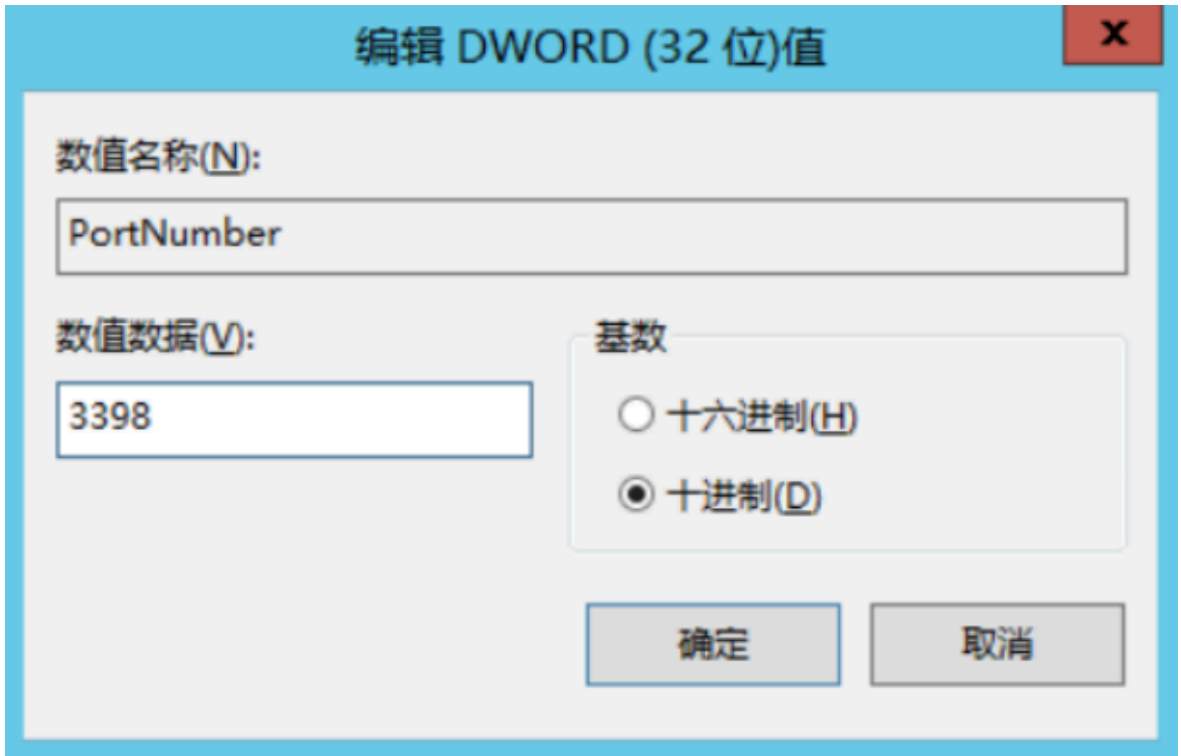
最佳实践



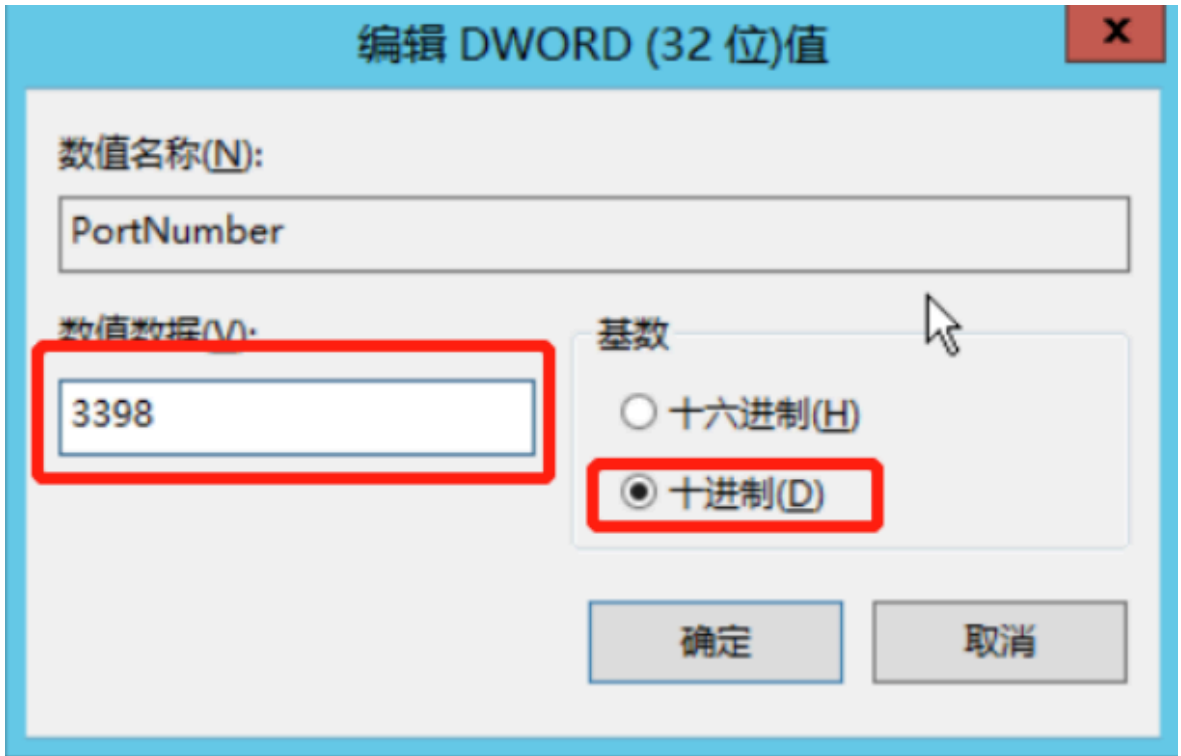
d. 在列表中找到注册表子项PortNumber并右键单击，选择“修改”，进入修改窗口。



e. 在弹出的修改窗口中，在数值数据的文本框中输入新的远程端口号，以3398为例。在基数框中勾选十进制，然后单击“确定”。



- f. 在左侧导航树中逐层选择HKEY_LOCAL_MACHINE > System > CurrentControlSet > Control > Terminal Server > WinStations > RDP-Tcp。
- g. 在右侧列表中找到注册表子项PortNumber并右键单击，选择“修改”（可以使用键盘方向键向下寻找）。
- h. 在弹出的对话框中，在数值数据的文本框中输入新的远程端口号，在本示例中即3398。在基数框中勾选十进制，然后单击“确定”。



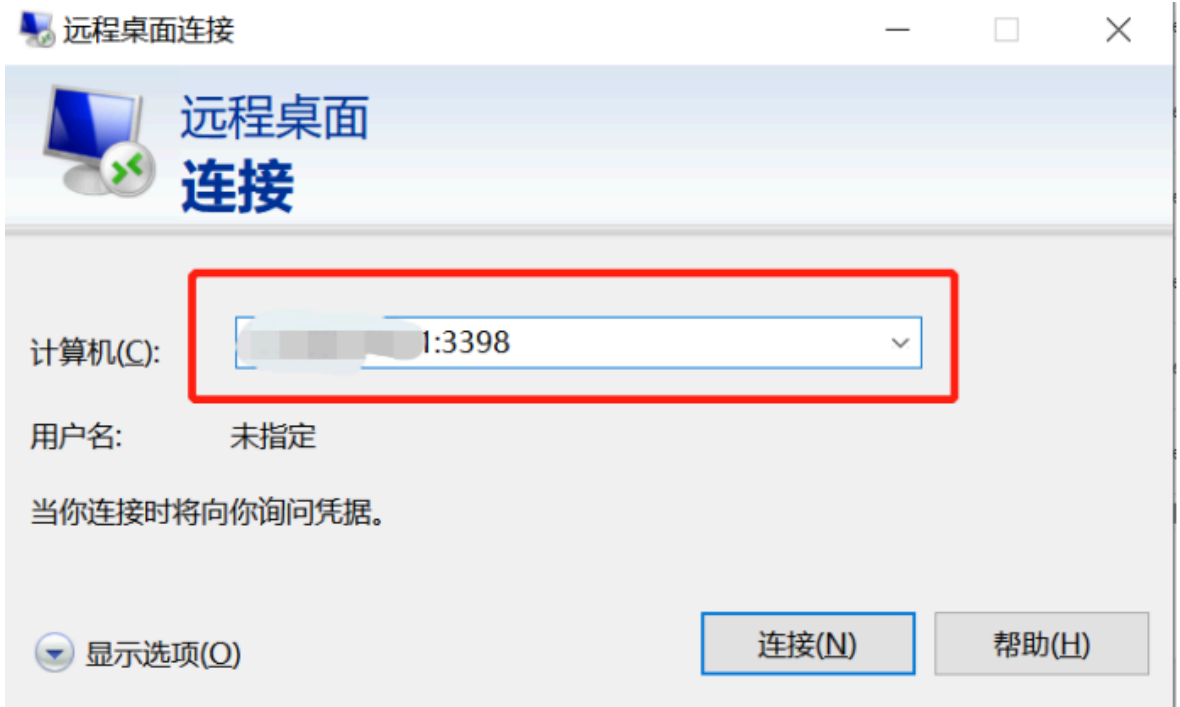
6. 在弹性云主机管理控制台中将此台云主机进行重启，具体如下图：



7. 为该实例添加安全组规则，允许新配置的3398远程端口进行连接。
具体操作，请参见[配置安全组规则](#)。


最佳实践

8. 最后使用远程桌面连接功能远程访问实例，在远程地址后面添加新端口号3398即可连接实例。



修改Linux系统实例默认远程端口

本章以CentOS 8.0 64位为例介绍如何修改Linux系统实例默认远程端口。

1. 登录控制中心。
2. 选择区域  华东-华东1。
3. 单击“计算>弹性云主机”，进入弹性云主机页面。
4. 单击待修改的弹性云主机行的“操作>远程登录”按钮，远程连接弹性云主机实例。
5. 输入用户名root，密码为购买弹性云主机时用户自定义的密码，登录成功之后如图：



```
CentOS Linux 8 (Core)
Kernel 4.18.0-88.el8.x86_64 on an x86_64

ecm-ctrlx login: root
Password:
Last login: Thu Jun 29 14:55:51 on tty1
[root@ecm-ctrlx ~]# _
```

最佳实践

6. 因sshd_config是Linux中重要的配置文件，为了避免误操作所带来的故障，在运行前首先对sshd_config进行备份，请运行以下命令：

```
cp /etc/ssh/sshd_config /etc/ssh/sshd_config_bak
```

进入到/etc/ssh路径下输入ll命令查看，具体信息如下图：

```
[root@ecm-ctrlx ~]# cp /etc/ssh/sshd_config /etc/ssh/sshd_config_bak
[root@ecm-ctrlx ~]# cd /etc/ssh
[root@ecm-ctrlx ssh]# ll
total 684
-rw-r--r--. 1 root root 563386 May 11 2019 moduli
-rw-r--r--. 1 root root 1727 May 11 2019 ssh_config
drwxr-xr-x. 2 root root 28 Mar 8 2021 ssh_config.d
-rw----- 1 root root 4443 Mar 10 2022 sshd_config
-rw----- 1 root root 4443 Jun 29 16:25 sshd_config_bak
-rw----- 1 root root 1381 Sep 1 2022 ssh_host_dsa_key
-rw-r--r-- 1 root root 601 Sep 1 2022 ssh_host_dsa_key.pub
-rw----- 1 root root 505 Sep 1 2022 ssh_host_ecdsa_key
-rw-r--r-- 1 root root 173 Sep 1 2022 ssh_host_ecdsa_key.pub
-rw----- 1 root root 399 Sep 1 2022 ssh_host_ed25519_key
-rw-r--r-- 1 root root 93 Sep 1 2022 ssh_host_ed25519_key.pub
-rw----- 1 root root 1811 Sep 1 2022 ssh_host_rsa_key
-rw-r--r-- 1 root root 393 Sep 1 2022 ssh_host_rsa_key.pub
[root@ecm-ctrlx ssh]#
```

7. 修改sshd服务的端口号，因为已经在/etc/ssh路径下，因此直接运行vim sshd_config编辑sshd_config配置文件，在键盘上按i键，进入编辑状态，添加新的远程服务端口，本节以2222端口为例。在Port 22下输入Port 2222。

```
# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

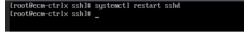
```
# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
#Port 22
port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

8. 在键盘上按Esc键，输入:wq后保存并退出编辑模式。

最佳实践

9. 运行以下命令重启sshd服务。

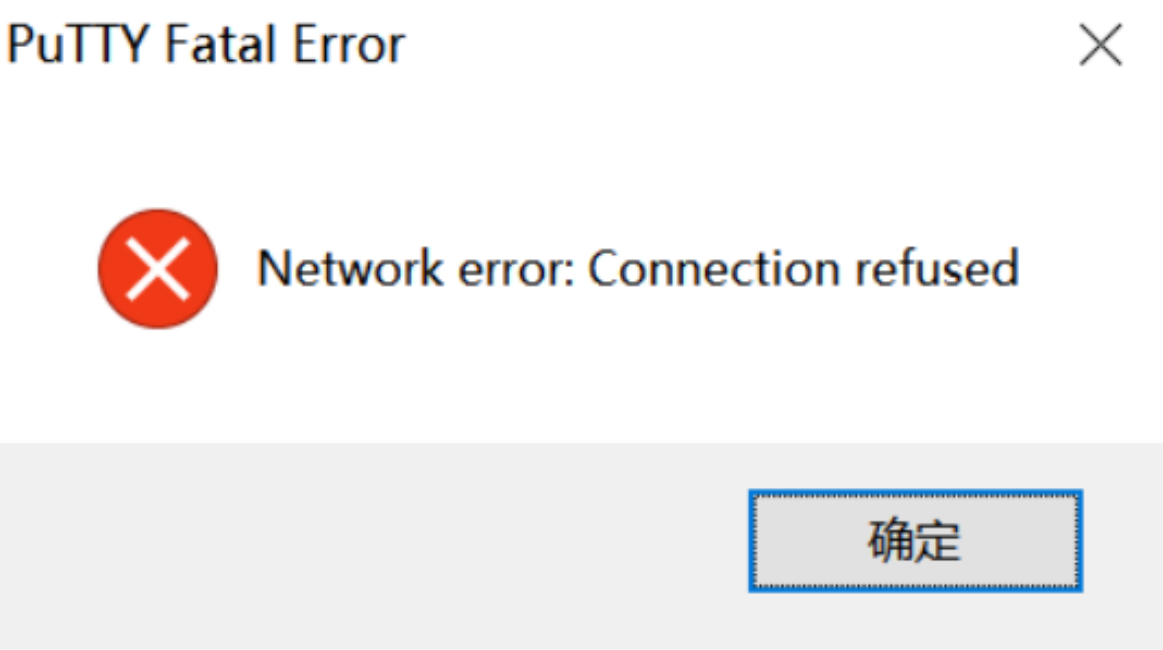
```
systemctl restart sshd
```



```
[root@ec2-ctr ~]# systemctl restart sshd
[root@ec2-ctr ~]#
```

至此，此弹性云主机的远程登录默认端口已经从22改为了2222。

功能验证

1. 打开远程连接工具putty，输入此弹性云主机实例的弹性公网IP，先默认使用22端口登录，可以看到连接被拒绝，网络已中断，说明目前22端口已经无法进行远程登录。

PutTY Fatal Error



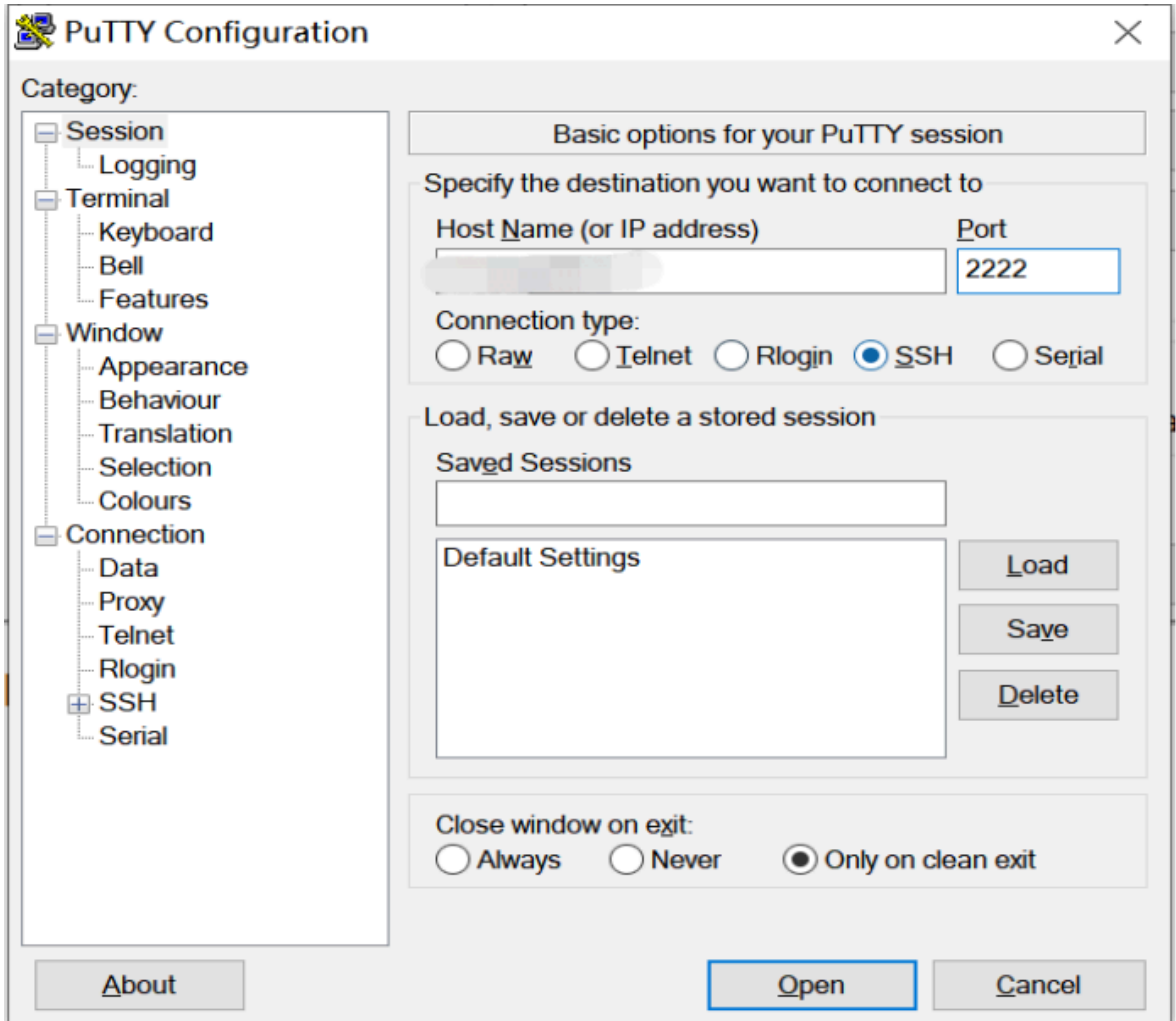
Network error: Connection refused

确定

2. 配置实例的安全组，放行TCP协议2222端口，具体操作，请参见[配置安全组规则](#)

最佳实践

3. 配置完成之后，使用SSH工具连接新端口2222，在port下输2222，可看到能够成功连接。





Windows基础最佳实践

镜像部署Windows环境

简介

本节介绍如何使用天翼云镜像，通过重装系统部署Windows环境。如果您已经购买了弹性云主机，想切换到Windows系统，或者想使用镜像重新部署已经预装了其它软件的环境，可以参考本文档的介绍和操作指导。

重装系统的约束与限制可以参见[重装操作系统](#)。

操作步骤

1. 登录天翼云控制台。
2. 选择“计算>弹性云主机”。
3. 选中指定的云主机，将云主机进行关机操作。

如果云主机已经处于关机状态，则可以直接执行重装系统操作。重装系统后，系统盘数据将会丢失，如果想要保留某些数据，可以先进行备份。



最佳实践

4. 选择云主机列表操作栏中的“更多”选项，选择“一键重装”。



5. 如果需要批量云主机重装系统，勾选多个云主机实例，选择云主机列表上方的“更多”选项，选择“一键重装”。



6. 仔细阅读一键重装页面的提示信息，选择需要的Windows镜像。镜像可以选择公共镜像、私有镜像、共享镜像、安全产品镜像。

7. 设置密码和用户数据。密码为必填项，用户数据可以暂不配置。点击“确定”按钮，即可进行重装系统的操作。
8. 可以在云主机列表看到重装云主机的状态。

9. 重装完成后，云主机的状态会恢复成“关机”，此时云主机可以正常使用。

为Windows Server添加AD、DHCP、DNS、IIS服务

操作场景

为Windows Server添加AD、DHCP、DNS、IIS服务，具体操作步骤如下。

最佳实践

操作步骤

1. 打开“服务器管理器”，点击“本地服务器”，选择对应的“关闭”选项，之后点击“确定”完成配置。

服务器管理器 · 本地服务器

仪表板
本地服务器
所有服务器
传真服务器
打印服务
文件和存储服务

计算机名	00gjjic-P3YRn0	上次安装的更新	状态未知
工作组	WORKGROUP	Windows 更新	状态未知
		上次检查更新的时间	状态未知
Windows 防火墙	公用: 启用	Windows Defender	已禁用
远程管理	已启用	反馈和诊断	设置
远程桌面	已禁用	IE 增强的安全配置	关闭
NIC 组合	已禁用	时区	(UTC+08:00) 北京, 重庆, 香港特别行政区, 乌鲁木齐
以太网 2	由 DHCP 分配	产品 ID	00377-60000-00000-AA934(已激活)
以太网 3	未连接		
以太网 5	由 DHCP 分配的 IPv4 地址, IPv6 已启用		
以太网 7	2.0.0.1, IPv6 已启用		
操作系统版本	Microsoft Windows Server 2016 Standard	处理器	Intel Xeon Processor (Skylake, IBRS), Intel Xeon Processor
硬件信息	RDO OpenStack Compute	安装的内存(RAM)	16 GB
		总磁盘空间	479.51 GB

事件
所有事件 | 共 113 个

筛选器

服务器名称	ID	严重性	源	日志	日期和时间
00GJJIC-P3YRNO	5973	错误	Microsoft-Windows-Apps	应用程序	2023/8/30 9:38:59
00GJJIC-P3YRNO	2005	错误	Microsoft-Windows-PerfNet	应用程序	2023/8/30 9:38:32
00GJJIC-P3YRNO	5973	错误	Microsoft-Windows-Apps	应用程序	2023/8/30 9:36:17
00GJJIC-P3YRNO	2005	错误	Microsoft-Windows-PerfNet	应用程序	2023/8/30 9:22:29
00GJJIC-P3YRNO	10016	错误	Microsoft-Windows-DistributedCOM	系统	2023/8/30 9:20:32
00GJJIC-P3YRNO	10016	错误	Microsoft-Windows-DistributedCOM	系统	2023/8/30 9:09:07
00GJJIC-P3YRNO	2005	错误	Microsoft-Windows-PerfNet	应用程序	2023/8/30 9:06:26

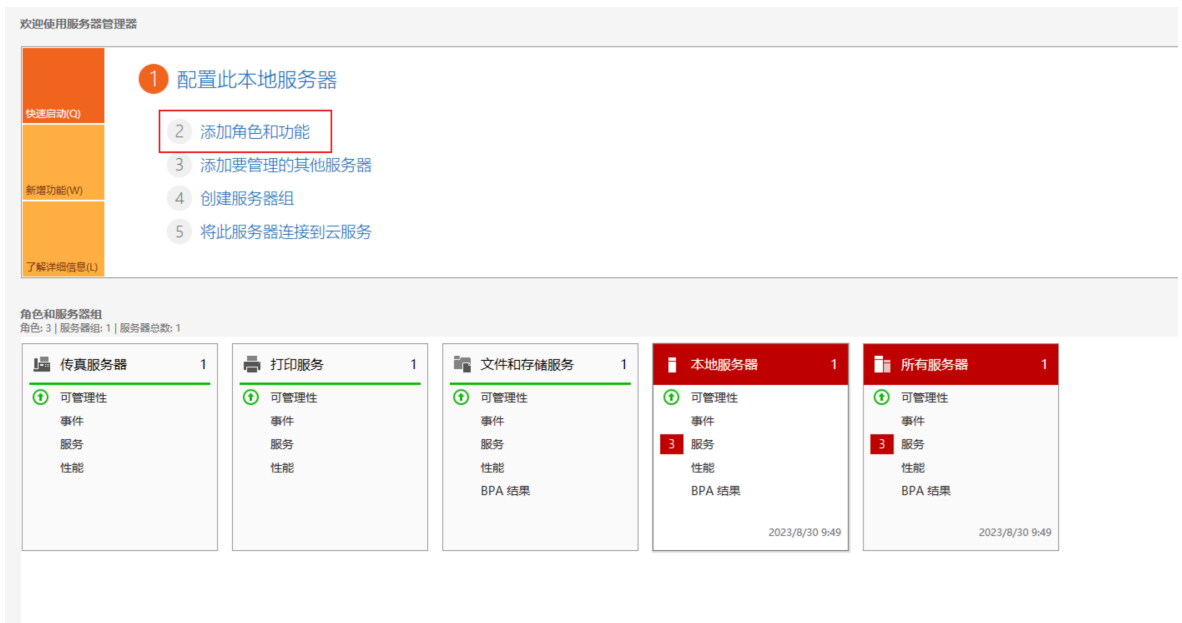


最佳实践

2. 进入“服务器管理器”的页面点击“仪表板”。

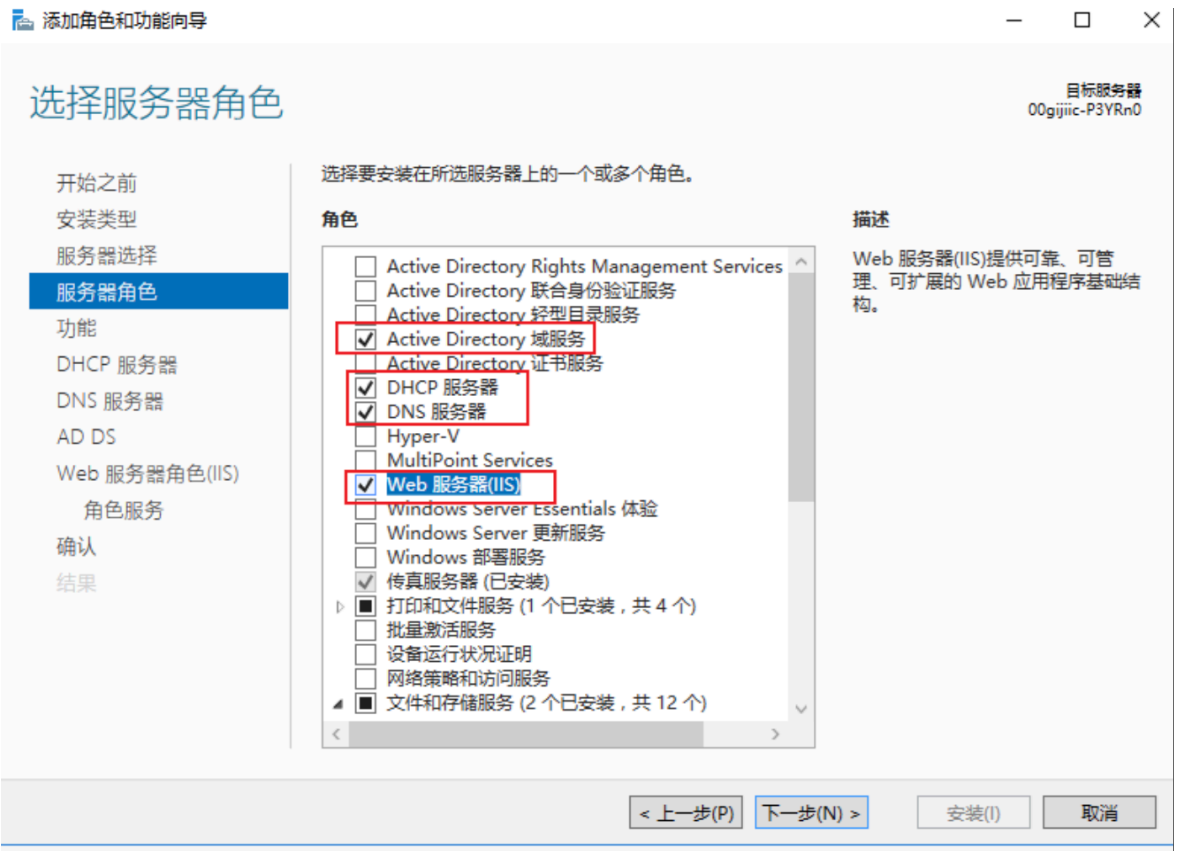


3. 单击“添加角色和功能”，为服务器添加新的角色和功能，包括DNS、DHCP、IIS、NET Framework 3.5。



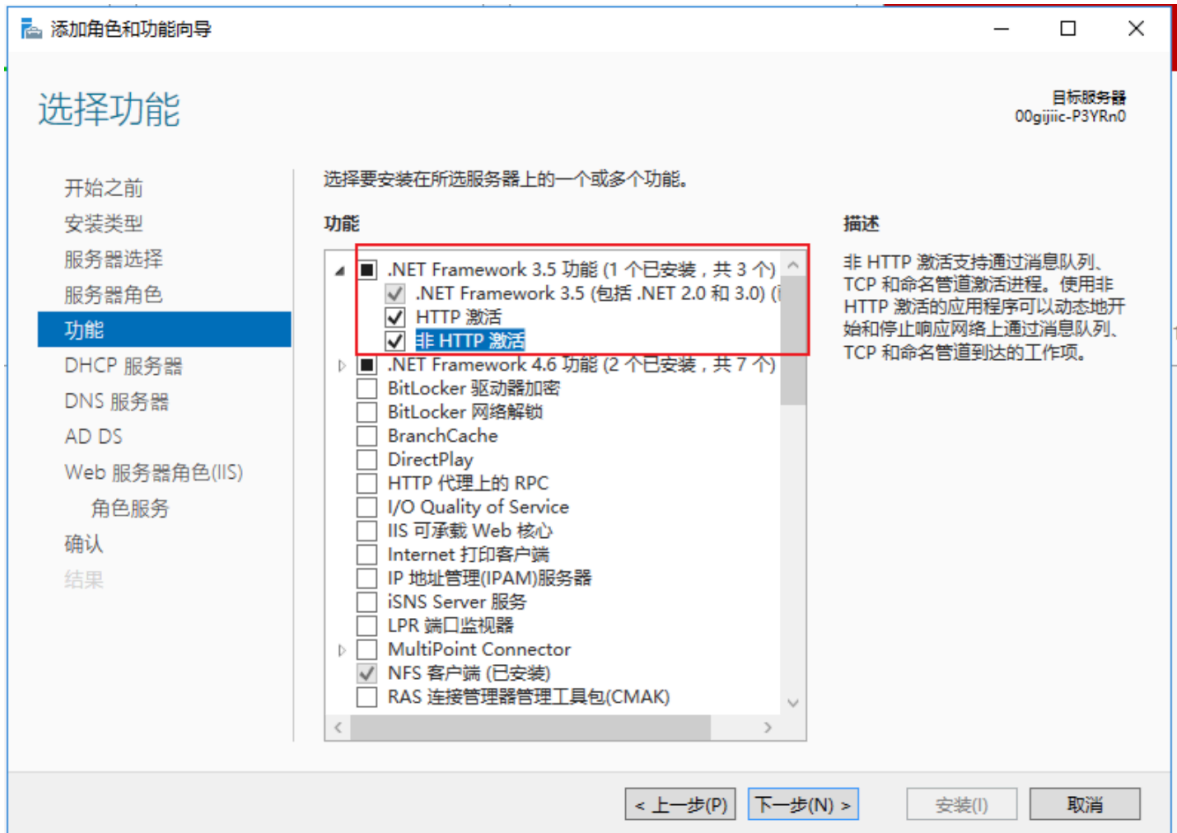
最佳实践

- 在“服务器角色”导航栏中，勾选“Active Directory 域服务”、“DHCP 服务器”、“DNS 服务器”和“Web 服务器(IIS)”。



- 单击“下一步”。

6. 在“功能”导航栏中，选中“.NET Framework 3.5 功能”。



7. 单击“下一步”。

最佳实践

8. 在“确认”导航栏中，勾选“如果需要，自动重新启动目标服务器”。



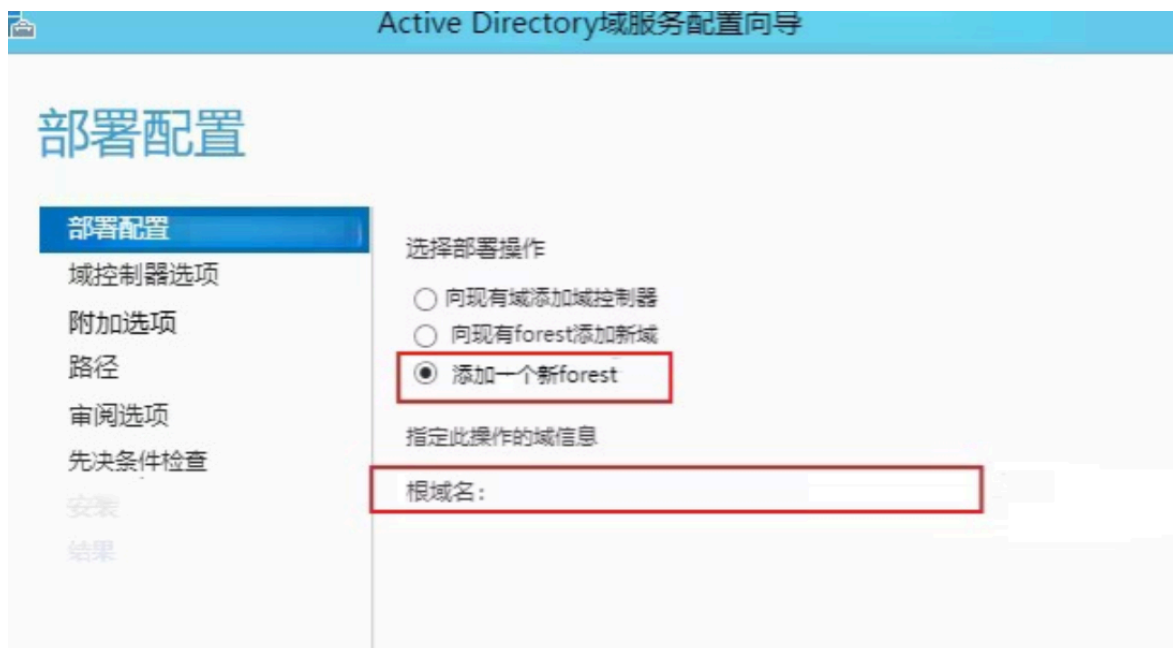
9. 单击“安装”，开始安装。

最佳实践

10. 安装完成后，单击“将此服务器升级为域控制器”，开始配置AD服务。

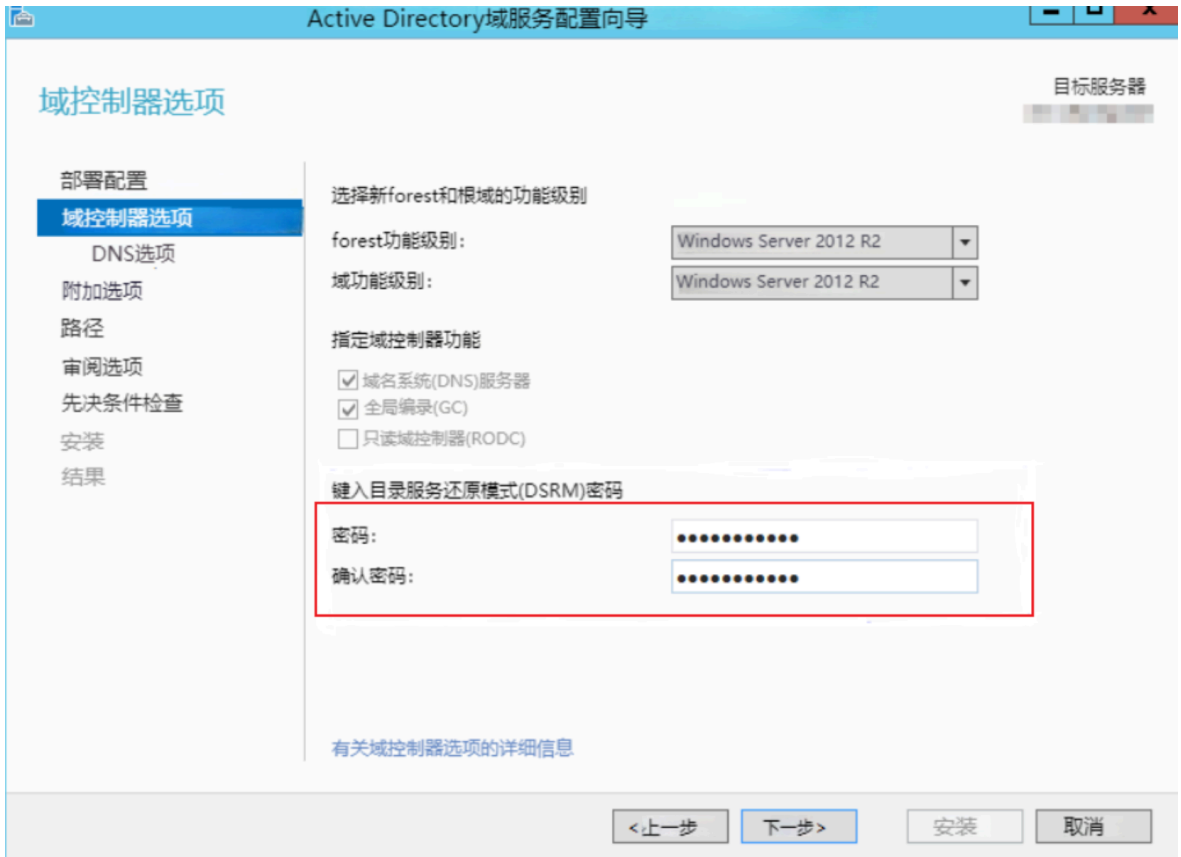


11. 因为没有已经存在的域环境，选择“添加一个新forest”，创建一个新的forest并设置根域名。如“根域名”可以设置为“sp160.com.cn”。



最佳实践

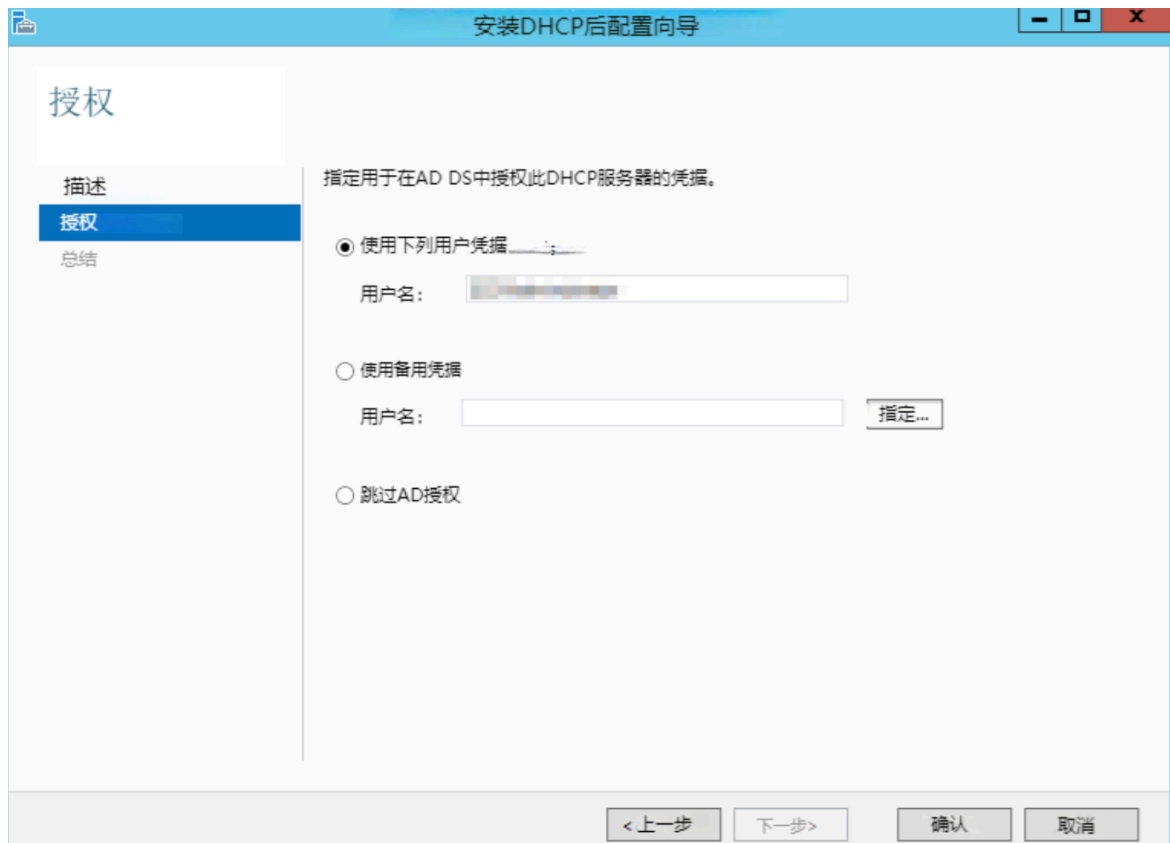
- 单击“下一步”。
- 设置密码，该密码用于备份还原域控。



- 单击“下一步”，直至安装完成。
- 单击“实现 DHCP 配置”，开始配置DHCP功能。



16. 单击“下一步”，选择默认配置。



17. 单击“确认”。

18. 配置完成，单击“关闭”。

设置Windows操作系统首选语言


前提条件

- 已经注册并登录天翼云账号，未完成的可参见[注册天翼云账号](#)。
- 已经在天翼云上购买了弹性云主机，且购买过程中镜像选择为Windows 2016 数据中心版操作系统。

背景信息

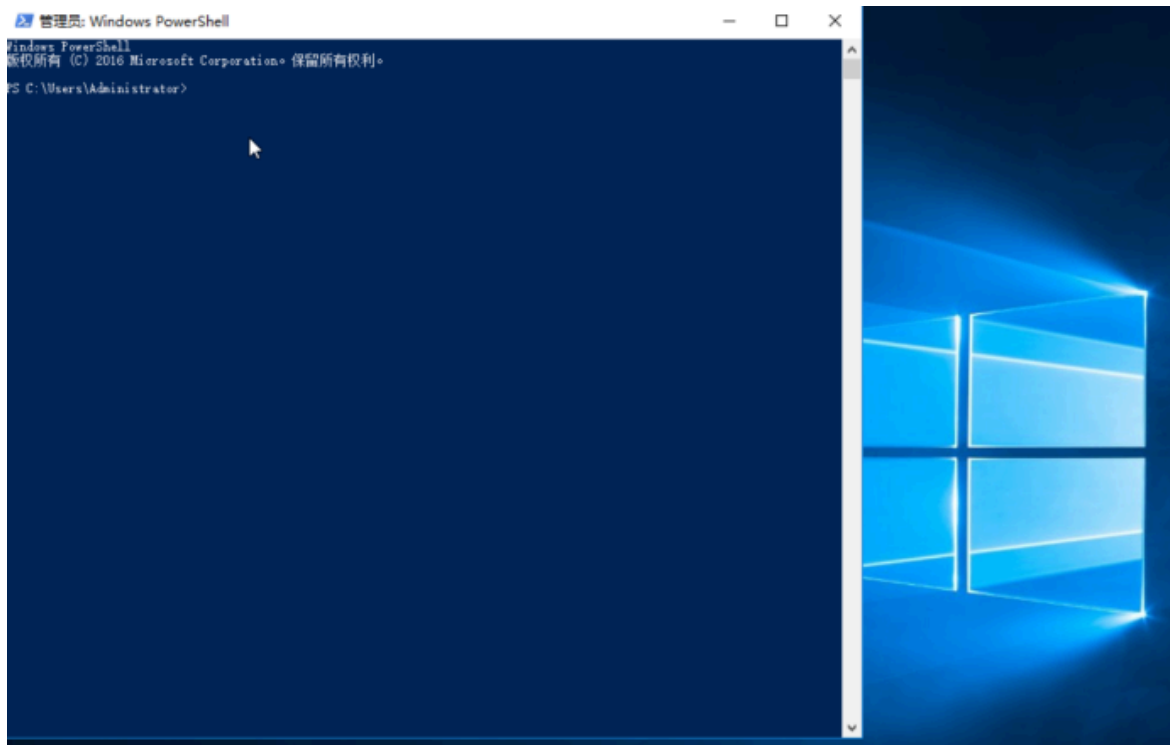
天翼云服务器ECS目前仅提供中文版的Windows Server公共镜像。如果您因工作需求而需要使用其他的语言版本，例如英文，那么您可以参考本章节为弹性云主机实例设置新的首选语言。

操作步骤

1. 登录控制中心。
2. 选择区域  华东-华东1。
3. 单击“计算>弹性云主机”，进入弹性云主机页面。
4. 参考[登录Windows弹性云主机](#)，远程登录待设置操作系统首选语言的Windows弹性云主机实例。

最佳实践

5. 打开PowerShell模块。



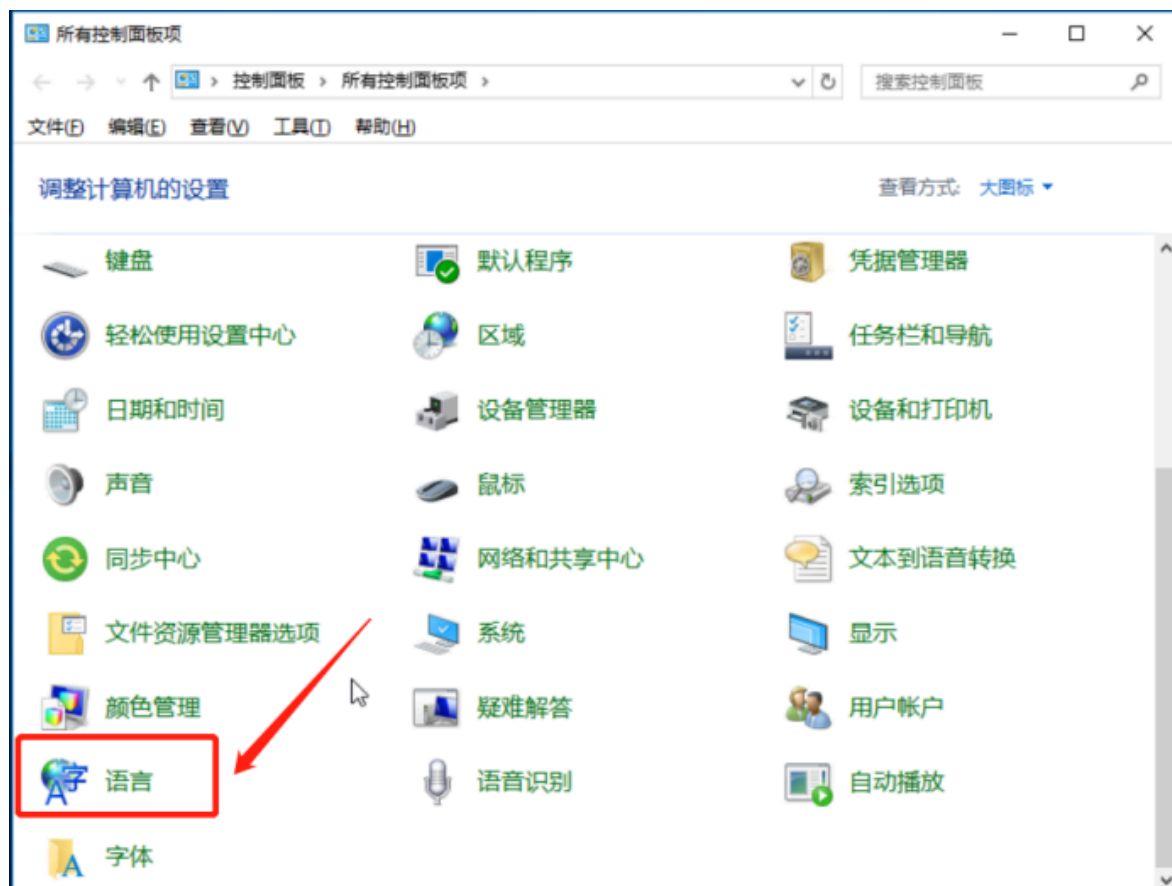
6. 在操作之前首先需要临时禁用掉WSUS（Windows Server Update Services）更新源，请您运行以下命令：

```
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU' -  
Name UseWUServer -Value 0  
Restart-Service -Name wuauclt
```



最佳实践

7. 找到控制面板，单击“语言”>“添加语言”。



最佳实践



语言

控制面板 > 所有控制面板项 > 语言

控制面板主页

高级设置

更改日期、时间或数字格式

更改语言首选项

可以使用已添加到列表中的任何语言键入。Windows、应用和网站将以列表中受支持的第一种语言进行显示。

添加语言 删除 上移 下移

中文(中华人民共和国)	Windows 显示语言: 已启用 输入法: 微软拼音 日期、时间和数字格式	选项
-------------	--	----

另请参阅

字体

位置

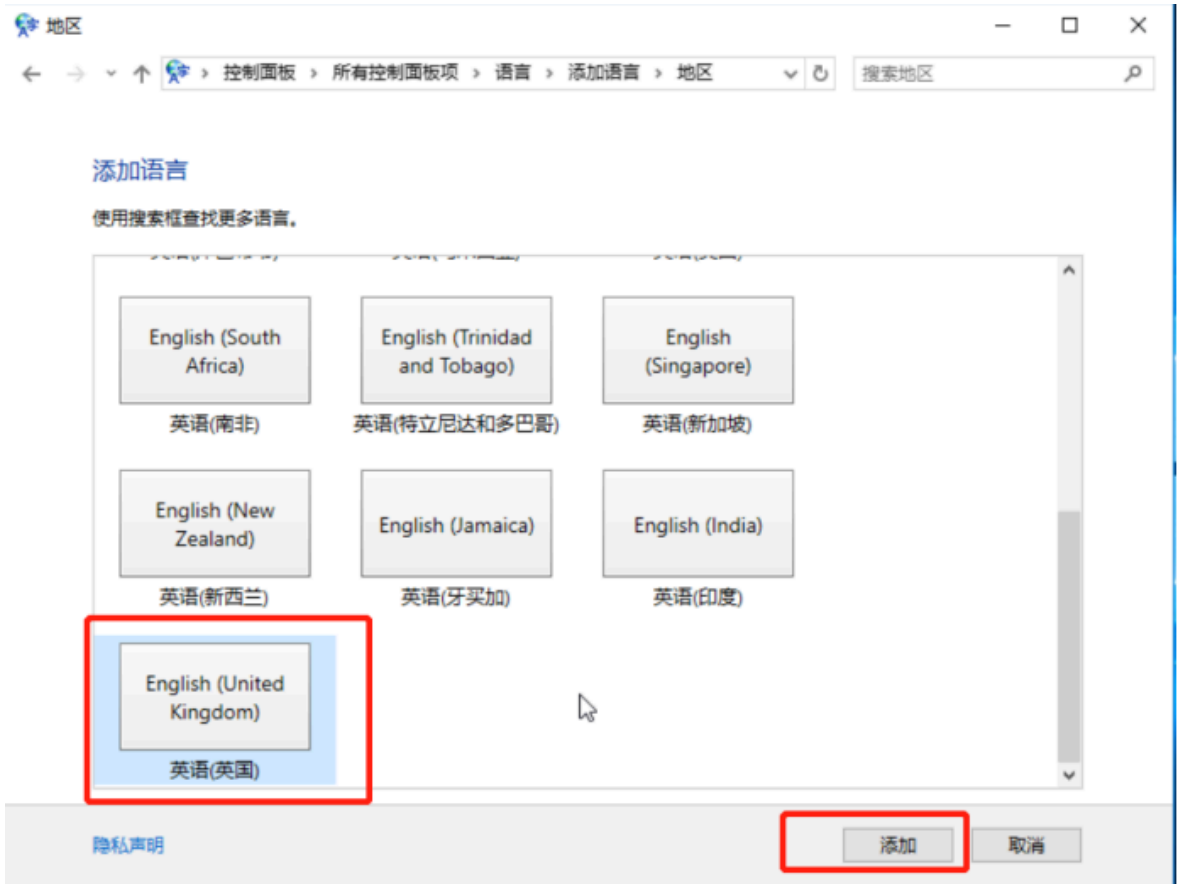
最佳实践

- 在“添加语言”对话框中，选择一种语言，例如English(英语)，单击下方的“打开”。



最佳实践

9. 进入跳转窗口后，选择具体的英语区域，选择English (United kingdom)，点击“添加”。



最佳实践

10. 添加之后，此语言会更新到语言列表。



最佳实践

11. 选中新增的English，单击“上移”更改语言优先级，下图为已经上移之后的状态：

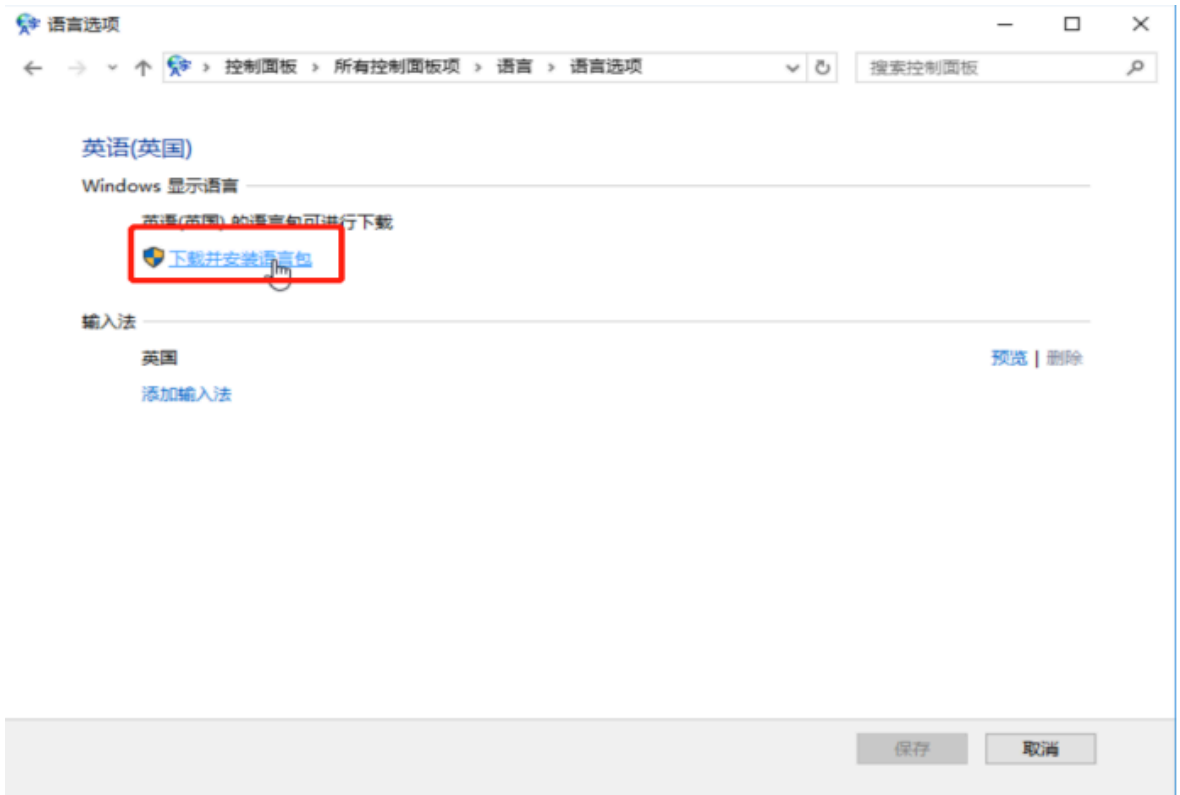


最佳实践

12. 单击所选语言右侧的“选项”按钮，进入语言选项页面，单击“下载并安装语言包”，若没有此链接，请再等待三分钟进行更新。

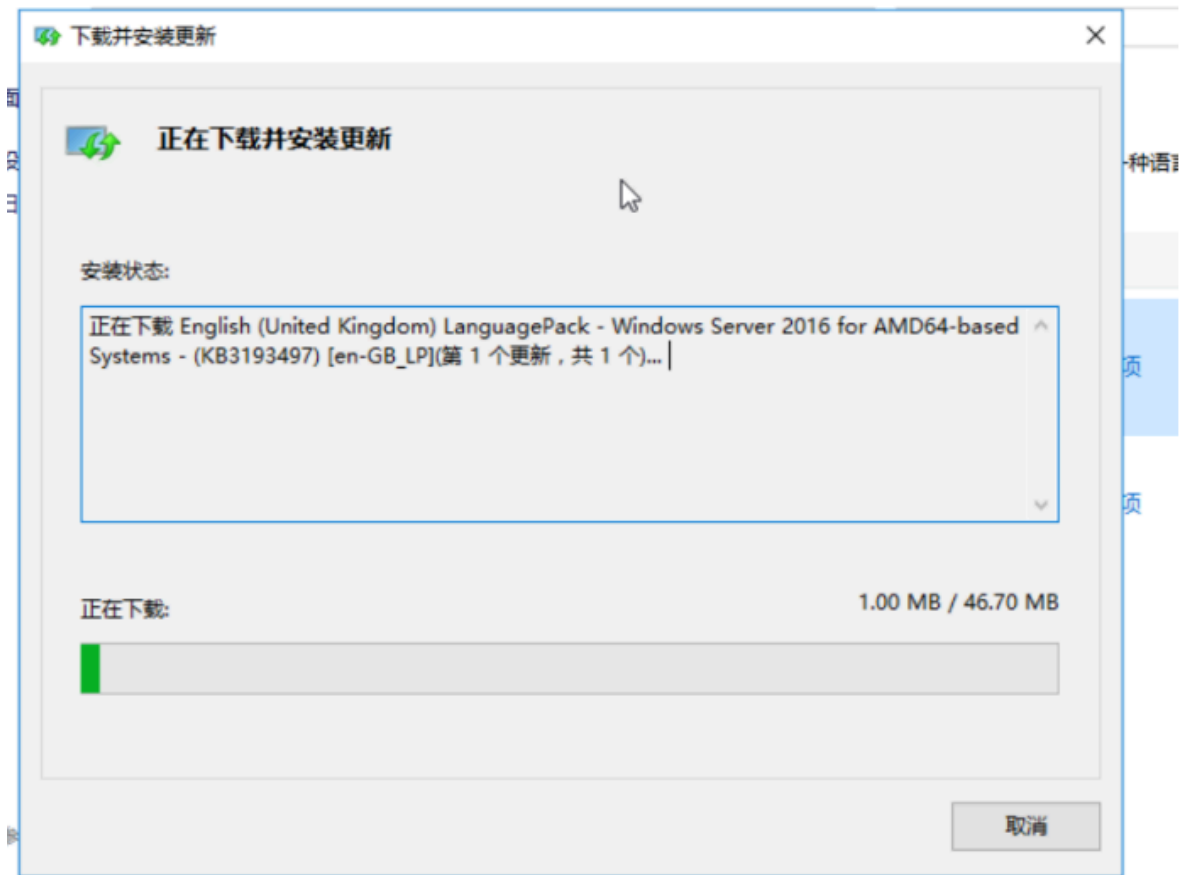


最佳实践

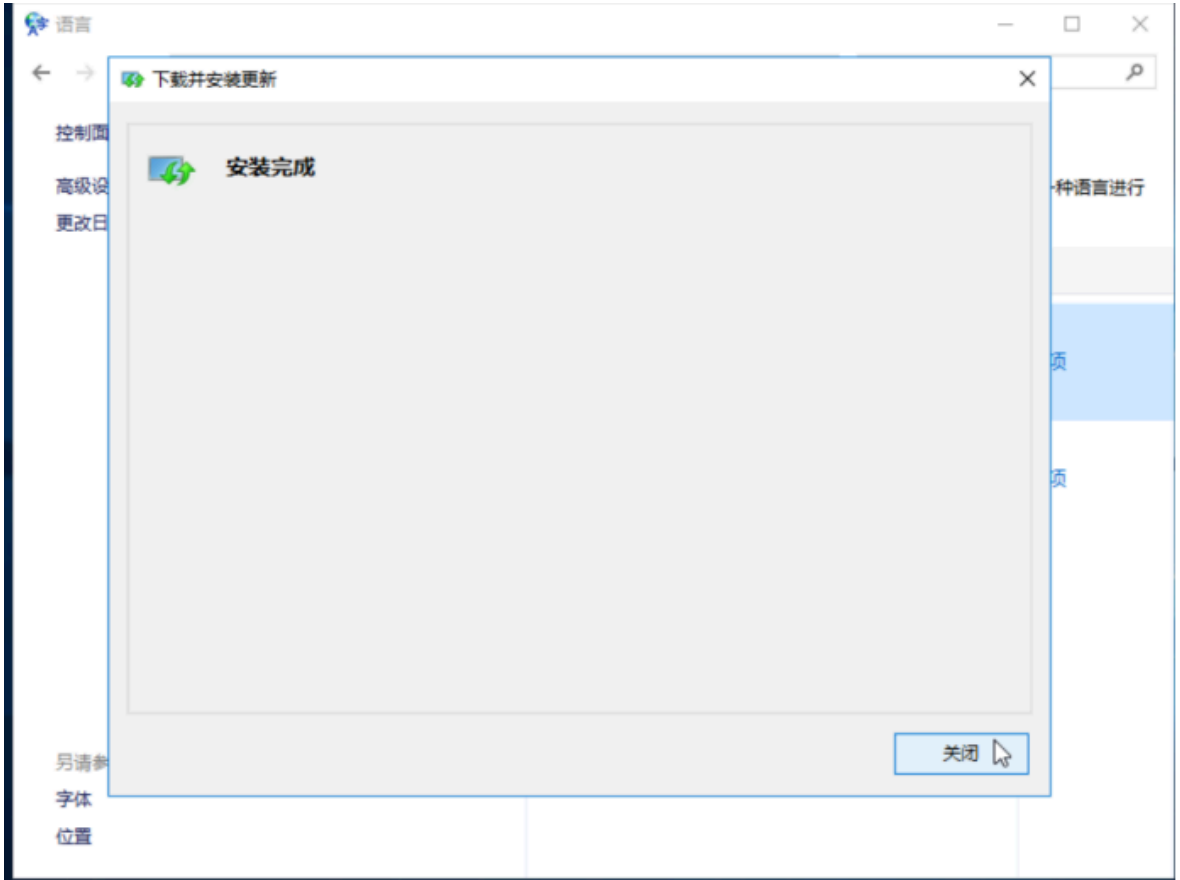


最佳实践

13. 用户此时可以等待下载并安装完成。



最佳实践



14. 在弹性云主机管理控制台中将此台云主机进行重启，具体如下图：



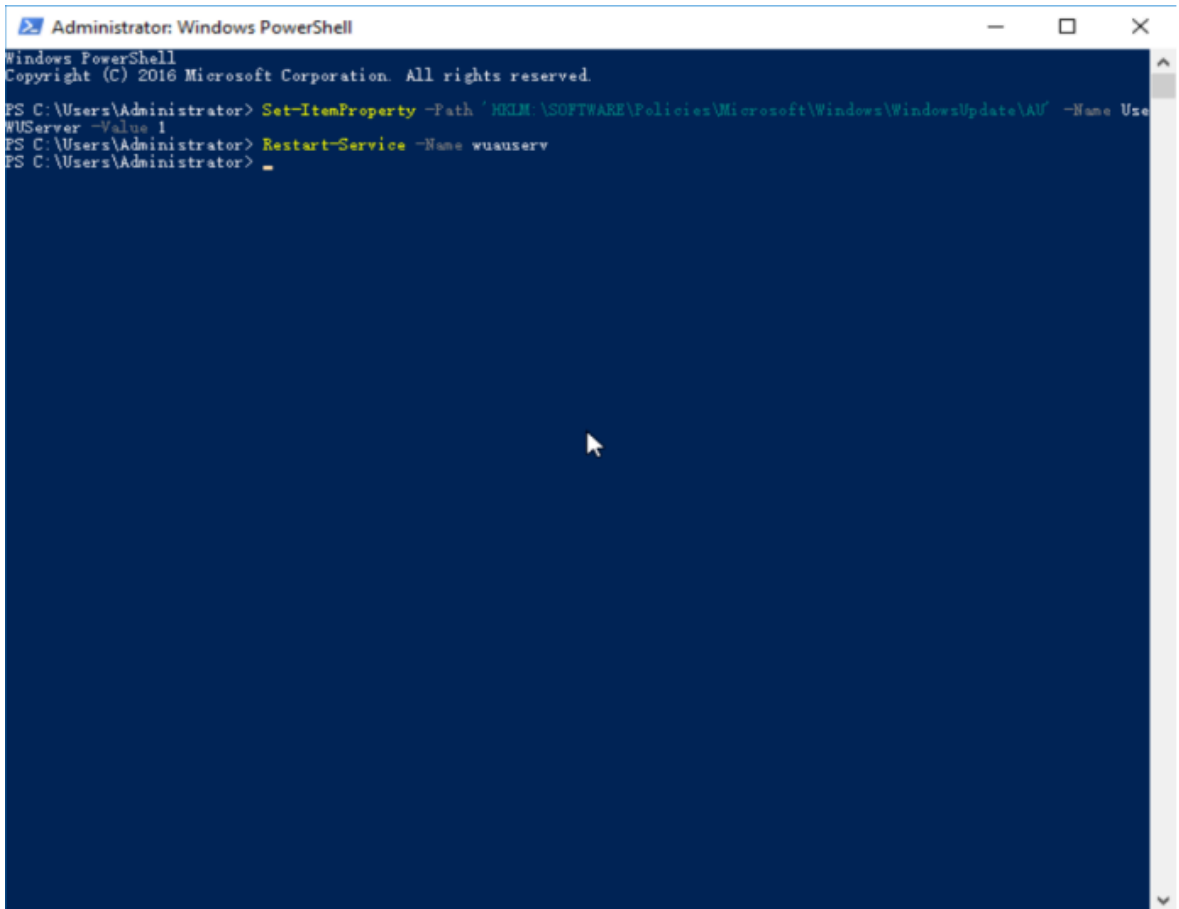
15. 再次连接Windows实例。显示语言更改为英语。



16. 打开Windows PowerShell，运行以下命令重新启用WSUS。

```
Set-ItemProperty -Path 'HKLM:\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU' -  
Name UseWUserver -Value 1
```

Restart-Service -Name wuauclt



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> Set-ItemProperty -Path 'HKLM\SOFTWARE\Policies\Microsoft\Windows\WindowsUpdate\AU' -Name UseWUService -Value 1
PS C:\Users\Administrator> Restart-Service -Name wuauclt
PS C:\Users\Administrator> _
```

Linux基础最佳实践

Linux服务器SSH登录的安全加固

SSH是远程登录Linux服务器的主要方式，但它也是黑客进行暴力破解和恶意攻击的主要入口之一。为了加强Linux云主机的SSH登录安全，您可以采用以下方式进行安全加固：

- 修改默认端口；
- 禁止root用户直接登录；
- 添加安全组规则；
- 采用密钥登录；
- 配置SSH登录限制。

我们以centos为例为大家介绍如何进行安全加固。

修改默认端口

1. 远程登录弹性云主机。

最佳实践

2. 打开 sshd 配置文件 `/etc/ssh/sshd_config`，可以使用编辑器打开该文件，例如使用 `vi` 命令：

```
vi /etc/ssh/sshd_config
```

3. 找到以下代码行：

```
#Port 22
```

将该行代码的注释符号 `#` 去掉，并将 `22` 修改为想要设置的端口号，例如 `2222`：

```
Port 2222
```

4. 按ESC键，输入`:wq`保存修改并退出编辑器。

5. 重启 sshd 服务，使配置生效。可以使用以下命令重启 sshd 服务：

```
systemctl restart sshd
```

6. 确认修改是否生效。可以使用以下命令检查 sshd 服务是否在新端口上监听：

```
netstat -tnl | grep 2222
```

如果输出类似于以下内容，则表示 sshd 服务已经在新端口 2222 上监听：

```
tcp    0    0 0.0.0.0:2222        0.0.0.0:*          LISTEN
```

7. 完成以上步骤后，就成功将 CentOS 系统的 SSH 默认端口号修改为了 2222。注意，在修改 SSH 默认端口号之后，需要使用新的端口号来进行 SSH 连接，例如：

```
ssh username@hostname -p 2222
```

其中，`username` 和 `hostname` 分别是 SSH 登录的用户名和远程主机名或 IP 地址。

禁用root用户直接登录

Linux的默认管理员名即是root，只需要知道root密码即可直接登录SSH。禁止root从SSH直接登录可以提高服务器安全性。经过以下操作后即可实现。

1. 新建用户

```
useradd test
```

2. 配置密码

#使用passwd命令即可给相应帐户设置或修改密码。

```
passwd test
```

#根据图示，设置或修改密码需要填写两次，第二次为效验密码，输入完毕后请回车确认。

```
Changing password for user test.
```

```
New password:
```

```
Retype new password:
```

```
passwd: all authentication tokens updated successfully.
```

3. 配置不允许root用户直接登录，修改相关文件

```
vi /etc/ssh/sshd_config
```

最佳实践

查找 `#PermitRootLogin yes`” 默认为132行
将“#”去掉，末尾“Yes”改为“No”
并:wq保存

4. 重启SSHD服务

```
systemctl restart sshd
```

5. 测试连接，可以看到，直接使用root 连接服务器 ssh会直接拒绝

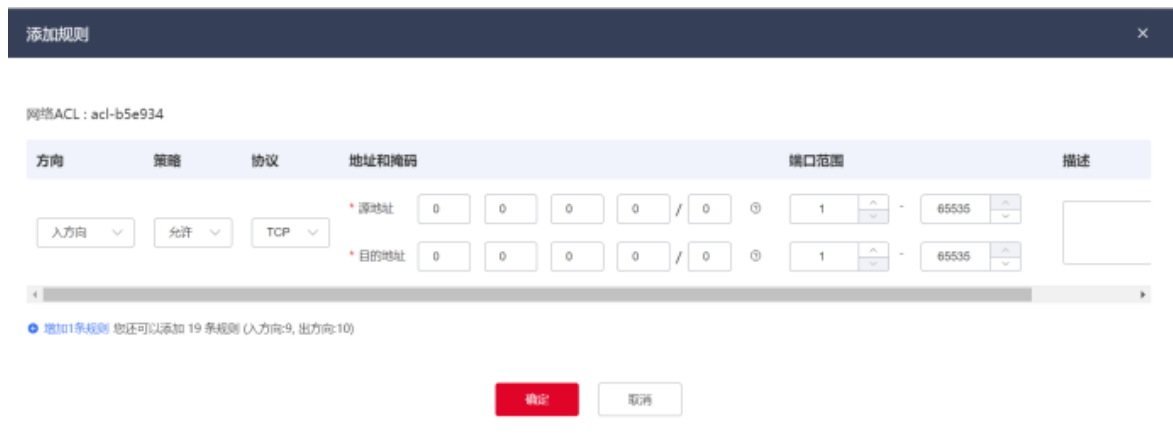


添加安全组规则

安全组中的入方向规则默认开启了22端口，当云服务器的SSH登录端口修改为2222时，需要为安全组新加一条规则。

登录管理控制台。

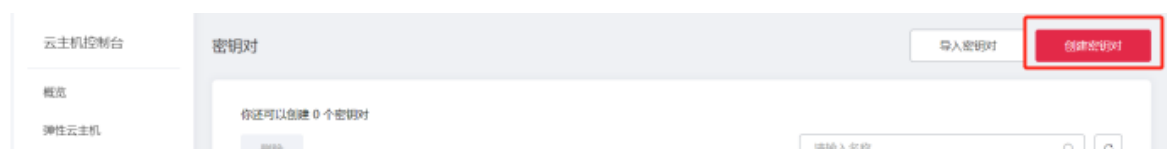
1. 选择“计算 > 弹性云主机”，进入云主机控制台。
2. 单击云服务器名称进入详情页面。
3. 选择“安全组”页签，单击展开安全组规则详情，单击列表右上角的“更改安全组规则”。
4. 添加一条入方向规则。



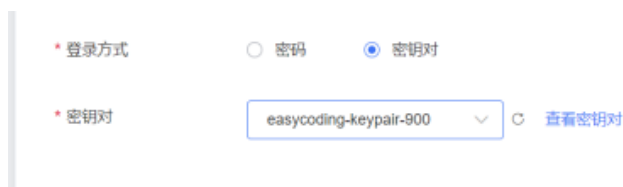
采用密钥登录

密钥对登录比起密码登录更加安全，因此在创建弹性云主机时您可以选择密钥对登录的方式。

1. 新建密钥对，或者选择已有的密钥对，并下载至本地。



2. 创建云主机时，登录方式选择密钥对。



3. 使用xshell登录弹性云主机

通过弹性公网IP，执行以下命令，SSH远程连接弹性云主机。

```
ssh 用户名@弹性公网IP
```

选择“Public Key”，并单击“用户密钥(K)”栏的“浏览”。

在“用户密钥”窗口中，单击“导入”。

选择本地保存的密钥文件，并单击“打开”。

单击“确定”，登录弹性云主机。

配置SSH登录限制

“/etc/hosts.allow”和“/etc/hosts.deny”是控制远程访问的文件，通过配置该文件可以允许或者拒绝某个IP或者IP段的客户访问Linux云服务器的某项服务。

比如SSH服务，通常只对管理员开放，那我们就可以禁用不必要的IP，而只开放管理员可能使用到的IP段。

最佳实践

因为云主机需要在不同地点登录，建议编辑“/etc/hosts.allow”允许所有IP地址登录，这样不会影响正常使用。

```
vim /etc/hosts.allow
```

在最后一行增加“sshd:ALL”。

```
#          either use the tcp_wrappers library or that have been
#          started through a tcp_wrappers-enabled xinetd.
#
#          See 'man 5 hosts_options' and 'man 5 hosts_access'
#          for information on rule syntax.
#          See 'man tcpd' for information on tcp_wrappers
sshd:ALL
```

也可以通过一些方法识别云主机的安全风险，使用tail -f /var/log/secure打开SSH日志，查看疑似恶意登录的IP。

```
[root@ecm-c26e ~]# tail -f /var/log/secure
Sep  8 11:11:51 ecm-c26e sshd[3246]: Received signal 15; terminating.
Sep  8 11:11:51 ecm-c26e sshd[3280]: Server listening on 0.0.0.0 port 22.
Sep  8 11:11:51 ecm-c26e sshd[3280]: Server listening on :: port 22.
Sep  8 11:12:32 ecm-c26e sshd[3280]: Received signal 15; terminating.
Sep  8 11:12:32 ecm-c26e sshd[3442]: Server listening on 0.0.0.0 port 22.
Sep  8 11:12:32 ecm-c26e sshd[3442]: Server listening on :: port 22.
Sep  8 11:12:54 ecm-c26e sshd[3712]: Accepted password for root from 10.0.0.1 port 2140 ssh2
Sep  8 11:12:55 ecm-c26e sshd[3712]: pam_unix(sshd:session): session opened for user root by (uid=0)
Sep  8 11:12:55 ecm-c26e sshd[3714]: Accepted password for root from 10.0.0.1 port 2141 ssh2
Sep  8 11:12:55 ecm-c26e sshd[3714]: pam_unix(sshd:session): session opened for user root by (uid=0)
```

然后在“/etc/hosts.deny”中将这些地址禁止。

```
#
# hosts.deny      This file contains access rules which are used to
#                deny connections to network services that either use
#                the tcp_wrappers library or that have been
#                started through a tcp_wrappers-enabled xinetd.
#
#                The rules in this file can also be set up in
#                /etc/hosts.allow with a 'deny' option instead.
#
#                See 'man 5 hosts_options' and 'man 5 hosts_access'
#                for information on rule syntax.
#                See 'man tcpd' for information on tcp_wrappers
#
sshd:10.0.0.1:deny
~
```

使用VNC Viewer连接Linux云主机

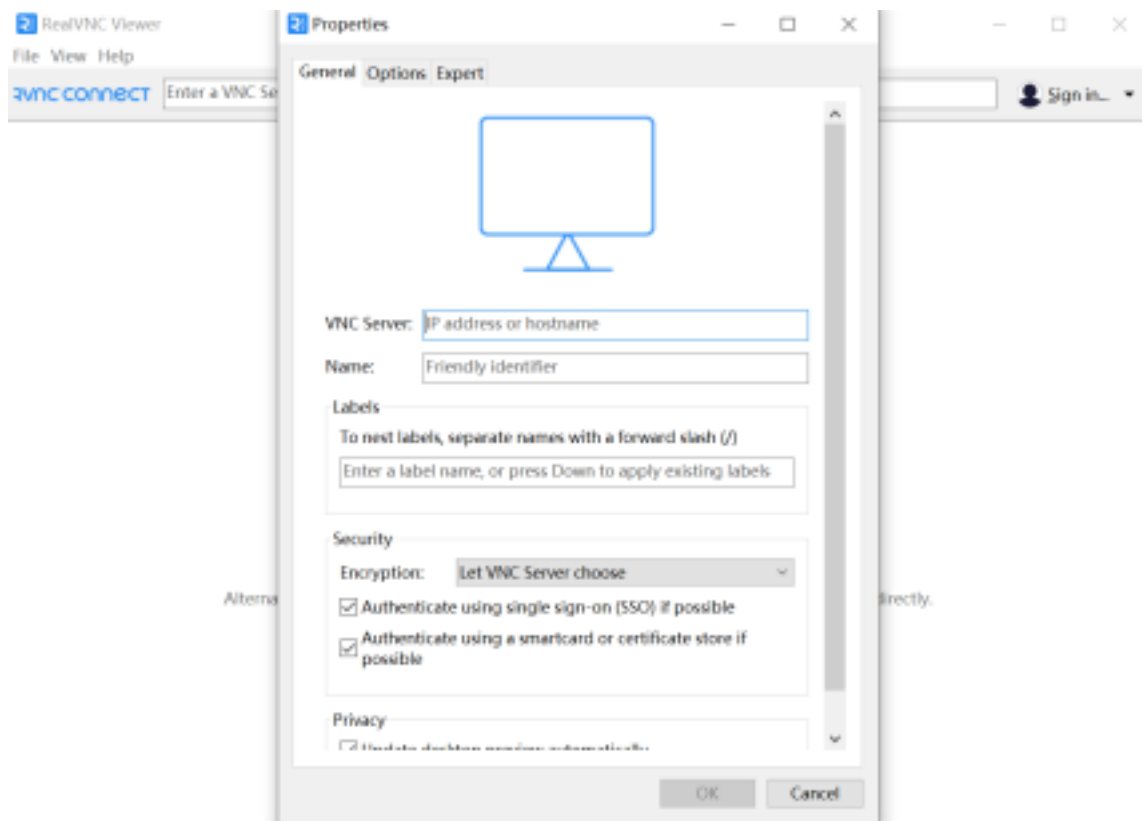
背景介绍

Linux云主机一般采用SSH连接方式，使用密钥对进行安全地无密码访问。但是SSH连接一般都是字符界面，有时我们需要使用图形界面进行一些复杂操作。

VNC Viewer是一款免费，并且开源的远程控制软件，可以实现批量管理，远程监控，批量打开链接，操作一台等于同时操作N台，使用起来十分便捷。

准备工作

1. 已创建一台CentOS7.6操作系统的弹性云主机，并且为云主机绑定弹性公网IP，确保可以连接互联网，具体操作参见[弹性云主机-创建弹性云主机](#)。
2. 已在本地PC安装VNC Viewer客户端。



给CentOS 7.6云主机安装gnome桌面环境

1. 使用SSH工具或控制台远程登录云主机。账号：root；密码：您在创建云主机时设置的密码。
2. 执行如下命令，安装图形化界面。

```
yum groupinstall "GNOME Desktop" "Graphical Administration Tools" -y
```

3. 执行以下命令，设置默认启动图形化界面。

```
ln -sf /lib/systemd/system/runlevel5.target /etc/systemd/system/default.target
```

最佳实践

4. 执行以下命令，重启云主机。

```
reboot
```

5. 通过天翼云控制台远程登录方式登录云主机，详情请参见[登录Linux 弹性云主机](#)。

登录云主机后查看可视化界面即表示搭建成功，根据界面提示进行配置进入桌面后，可按需进行相关操作。如图所示：



安装VNC Server实现本地远程图形化访问

1. 执行如下命令，安装vncserver。

```
yum install -y tigervnc-server
```

2. 配置vncserver执行vncserver命令配置vnc密码（初次启动的时候需要输入两次密码）。

```
vncserver
```

You will require a password to access your desktop.

Password: (输入并记住该密码)

最佳实践

Verify: (重复输入并记住该密码)

```
[root@ ~]# vncserver
You will require a password to access your desktops.
Password:
Verify:
Would you like to enter a view-only password (y/n)? n
A view-only password is not used
New 'instance-ie5uf3jl:1 (root)' desktop is instance-ie5uf3jl:1
Creating default startup script /root/.vnc/xstartup
Creating default config /root/.vnc/config
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/instance-ie5uf3jl:1.log
```

输入并记住该密码

再次重复密码

这句话的意思是是否需要创建一个只能看，不能操作的用户。可根据自身需求选择，我选择的是不需要。

执行以下命令，生成并编辑配置文件。

```
cp -p /lib/systemd/system/vncserver@.service /lib/systemd/system/vncserver@:1.service
```

执行如下命令修改配置：

```
vim /lib/systemd/system/vncserver@:1.service
```

```
[Unit]
Description=Remote desktop service (VNC)
After=syslog.target network.target

[Service]
Type=forking
User=root
# Clean any existing files in /tmp/.X11-unix environment
ExecStartPre=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'
ExecStart=/usr/sbin/runuser -l root -c "/usr/bin/vncserver %i"
PIDFile=/home/root/.vnc/%i.pid
ExecStop=/bin/sh -c '/usr/bin/vncserver -kill %i > /dev/null 2>&1 || :'

[Install]
WantedBy=multi-user.target
```

新增该字符

这两个修改为root

按Esc键退出编辑模式，再输入:wq保存并退出。

3. 运行以下命令重启systemd。

```
systemctl daemon-reload
```

4. 配置为开机自启动。

```
systemctl enable vncserver@:1.service
```

5. 运行以下命令启动VNC Server。

```
systemctl start vncserver@:1.service
```

最佳实践

6. 运行命令 `ps -ef | grep vnc` 确认服务是否已经启动。如果返回以下类似结果，说明服务已经启动。

```
[root@instance-ie5uf3jl ~]# ps -ef | grep vnc
root      5016      1  0 15:00 ?        00:00:00 /usr/bin/Xvnc :1 -auth /root/.Xauthority -desktop instance-ie5uf3jl:1 (root) -fp ca
talogue:/etc/X11/fontpath.d -geometry 1024x768 -pn -rfbauth /root/.vnc/passwd -rfbport 5901 -rfbwait 30000
root      5184      1  0 15:00 ?        00:00:00 /bin/sh /root/.vnc/xstartup
root      8056    7378  0 15:12 pts/0    00:00:00 grep  --color=auto vnc
[root@instance-ie5uf3jl ~]#
```

更改安全组规则

1. 登录弹性云主机管理控制台。
2. 单击弹性云主机名称，进入详情页面。
3. 在“安全组”页签，单击“更改安全组规则”，放行5901端口。

* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略： 允许 拒绝

* 优先级：

* 协议

TCP、UDP报文分片后，分片不带有端口信息，需要将端口范围指定为1-65535，不进行端口过滤。

* 端口范围

* 源地址： IP 安全组

/

描述

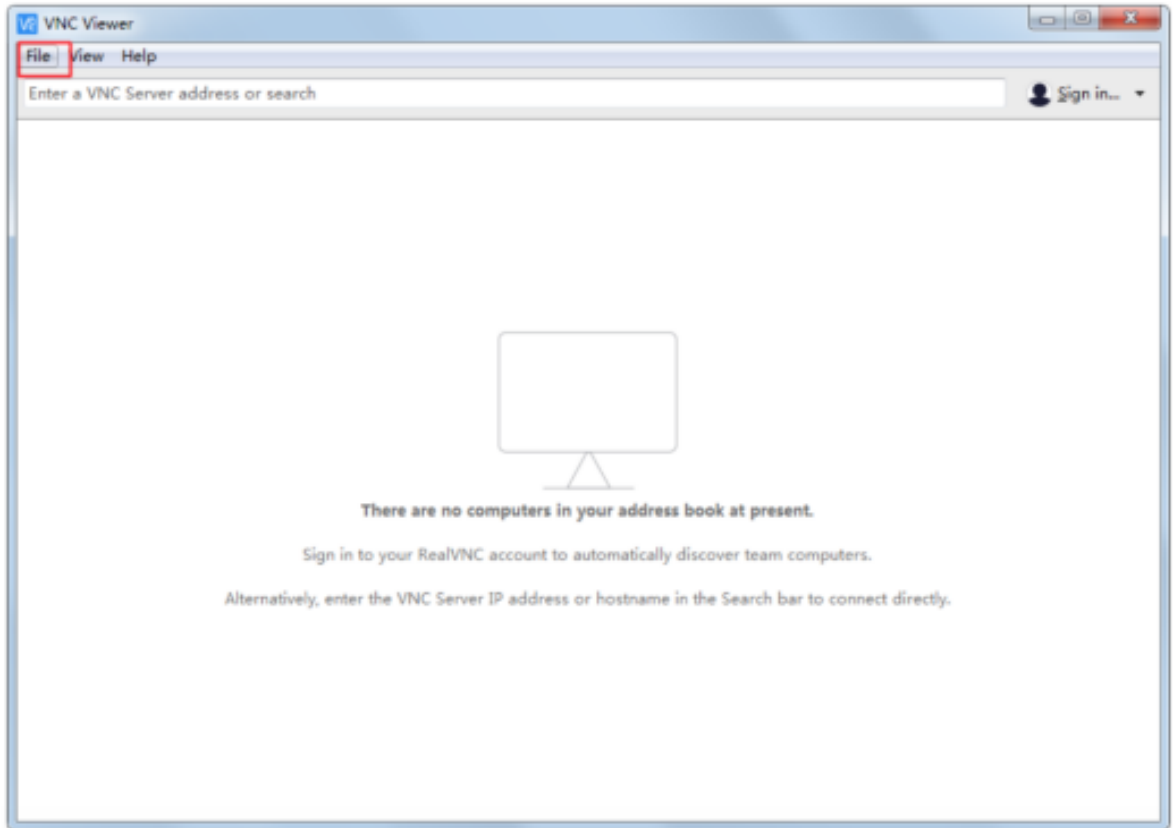
确定

取消

最佳实践

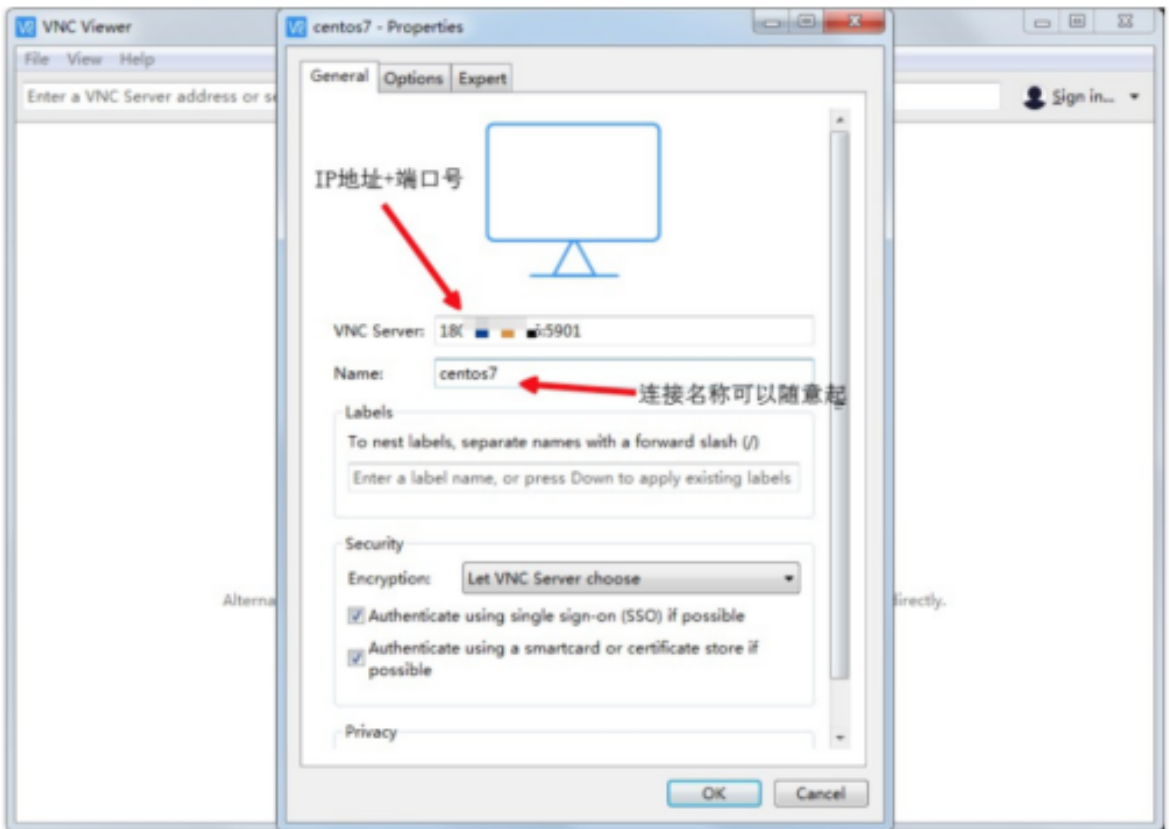
使用viewer客户端连接软件连接该(GentOS7.6)实例

1. 使用准备工作中提供的viewer客户端连接软件连接该实例。点击file——New connection，创建新连接。



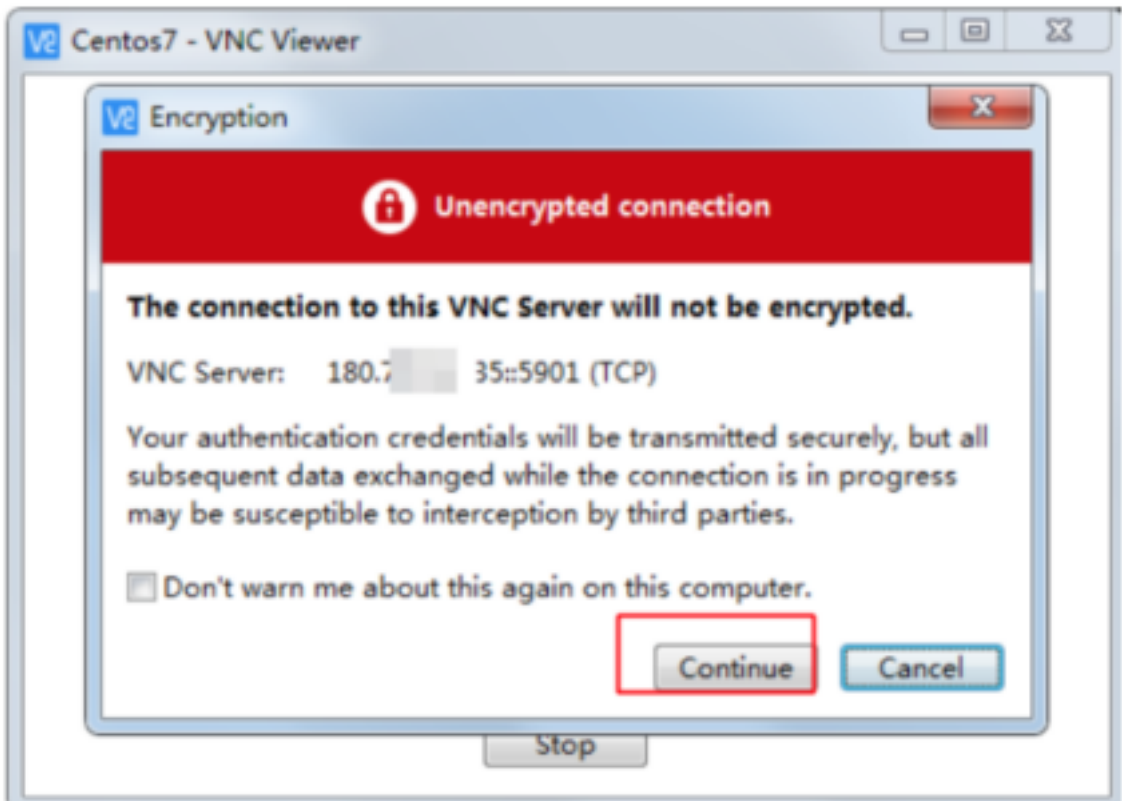
最佳实践

2. 输入对应的信息。



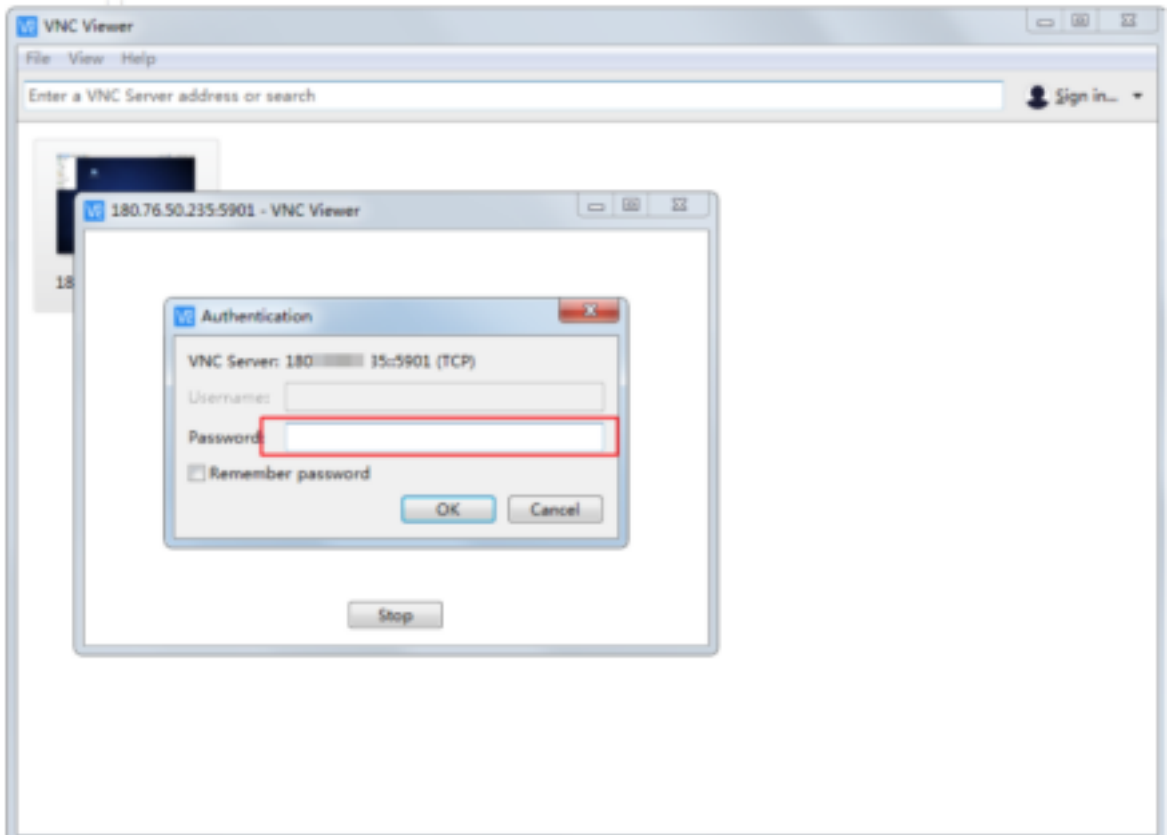
最佳实践

3. 双击之前创建的连接，点击进行连接。

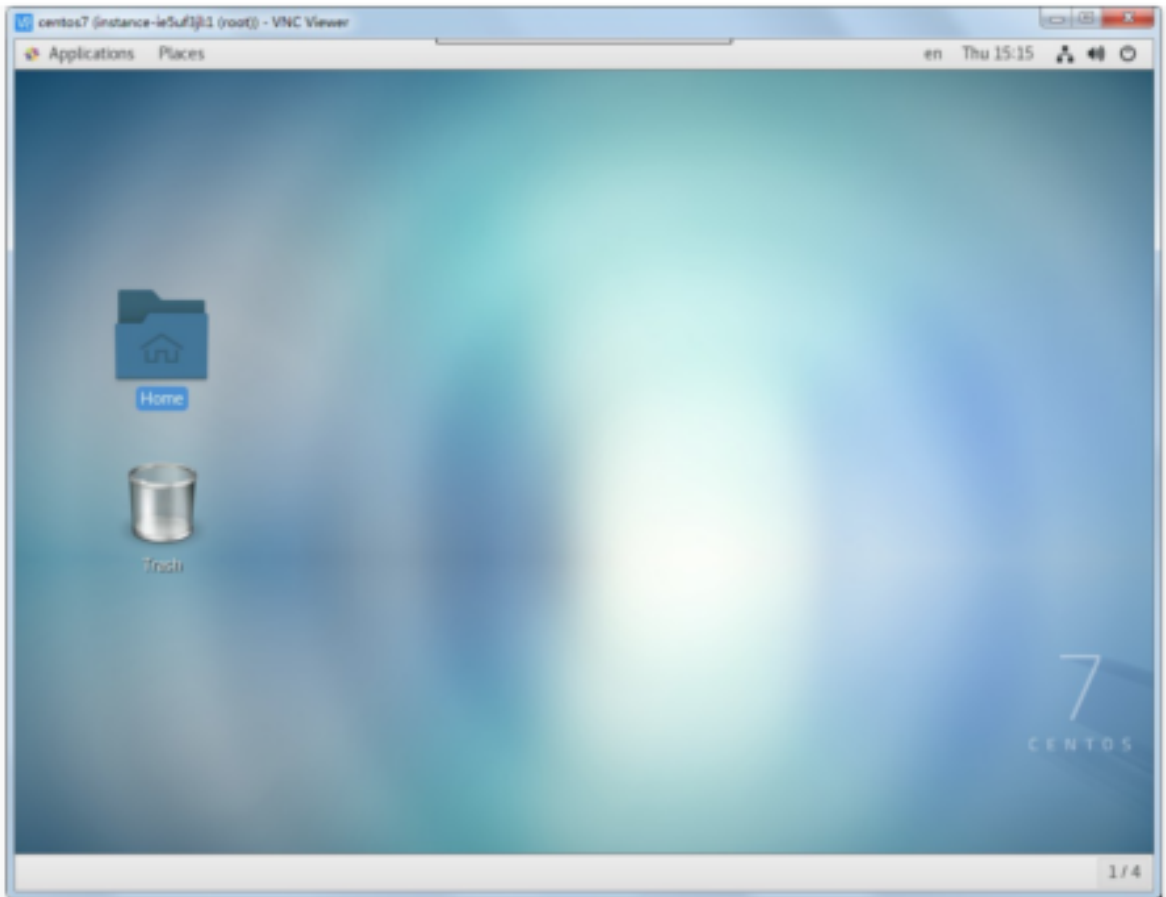


最佳实践

4. 此处输入安装VNC Server时设置的密码。



5. 连接成功。



Linux系统进入单用户模式

背景介绍

用户模式介绍

Linux操作系统有两种重要的用户模式，一种是多用户模式，一种是单用户模式，具体介绍如下：

- 多用户模式（multi-user mode）：多用户模式是Linux系统默认的运行模式。在多用户模式下，系统可以同时支持多个用户登录，并运行各种服务和进程。多用户模式提供了完整的功能和服务，包括网络服务、图形界面等。
- 单用户模式（single-user mode）：单用户模式是一种特殊的启动模式，提供了一个最小化但强大的环境，允许管理员以超级用户权限进行操作。进入单用户模式后，系统只会加载最基本的服务和功能，只有一个命令行界面，不会启动图形界面和网络服务。

单用户模式应用场景

单用户模式通常应用于以下场景：

最佳实践

- **系统故障排除和修复：**当系统遇到启动问题、文件系统错误、网络配置问题或其他故障时。管理员可以以超级用户权限登录，并执行诊断、修复和恢复任务，如修复文件系统、检查硬件、还原配置文件等。
- **系统备份和还原：**单用户模式提供了对系统磁盘的完全访问权限。这使得管理员可以在单用户模式下执行备份和还原操作，包括创建和还原系统快照、复制重要数据、修复损坏的文件系统等。
- **系统配置和维护：**单用户模式提供了一个干净且最小化的系统环境，使管理员能够进行系统配置和维护操作，而不会受到其他服务或用户的干扰。这包括更新软件包、重建启动引导程序、更改网络配置等。
- **安全审计：**单用户模式可以用于进行安全审计。管理员可以检查系统日志、分析安全事件、查找潜在的漏洞，并采取必要的措施来加固系统和提高安全性。

前提条件

要进入Linux系统的单用户模式，通常需要满足以下前提条件：

- 已经注册并登录天翼云账号，未完成的可参见[注册天翼云账号](#)。
- 已经在天翼云上购买了弹性云主机，且购买过程中镜像选择为Linux操作系统，例如CentOS, Ubuntu等。
- 进入单用户模式后，您将作为超级用户（root）登录。因此，您需要知道root用户的密码或者您具有其他获得root权限的方法。

约束与限制


- **当进入Linux操作系统的单用户模式时，存在一些约束与限制，包括：**
 - 提前备份数据：**在进入单用户模式之前，务必提前备份重要数据。单用户模式是一个强大的权限环境，错误的操作可能导致数据丢失或系统不稳定。
 - 只有一个命令行界面：**单用户模式不会加载图形界面，只提供基本的命令行界面。这意味着用户只能使用命令行工具进行操作，无法使用图形化界面。
 - 确保系统文件可读写：**单用户模式需要能够读写系统文件，因此需要确保您的文件系统没有损坏并且没有只读的挂载。
 - 无网络连接：**默认情况下，单用户模式不会启动网络服务，因此无法进行网络连接。这意味着不能访问网络资源，包括互联网、局域网和远程连接。

操作步骤

不同的Linux发行版使用不同的引导加载程序，如GRUB（Grand Unified Bootloader）或LILO（Linux Loader）。本文将通过系统引导器（GRUB）进入单用户模式。

CentOS操作系统进入单用户模式

本示例将会对一台操作系统为CentOS8.0 64位镜像的弹性云主机实例进行单用户模式进入操作。

1. 登录控制中心。
2. 选择区域  华东-华东1。
3. 单击“计算>弹性云主机”，进入弹性云主机页面。
4. 单击待操作的弹性云主机行的“操作>远程登录”按钮，远程连接弹性云主机实例。



最佳实践

5. 输入用户名root，密码为购买弹性云主机时用户自定义的密码，登录成功之后如图：

```
CentOS Linux 8 (Core)
Kernel 4.18.0-80.el8.x86_64 on an x86_64

ecm-ctrlx login: root
Password:
Last login: Thu Jun 29 14:55:51 on tty1
[root@ecm-ctrlx ~]# _
```

6. 输入重启命令reboot，当重启过程中出现让您选择启动系统界面时按下键盘e键（为防止页面自动跳转，需要您及时关注重启页面），跳转至启动项配置界面。下图为启动项配置界面：

```
_load_video
set gfx_payload=keep
insmod gzio
linux ($root)/boot/vmlinuz-4.18.0-80.el8.x86_64 root=UUID=(
ro net.ifnames=0 biosdevname=0 console=tty0 console=ttyS0,115\
200n8 crashkernel=0M-3G:0M,3G-8G:192M,8G-:256M
initrd ($root)/boot/initramfs-4.18.0-80.el8.x86_64.img $tuned_initrd

Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
discard edits and return to the menu. Pressing Tab lists
possible completions.
```

7. 使用键盘的方向键，移动光标到linux开头的那一行命令，将本行中ro至本行末尾的内容删除，并替换为rw
init=/bin/sh crashkernel=auto，具体修改信息可见下图：


8. 截至目前，进入单用户模式的配置已经基本完成，现在需要用户按下键盘的ctrl+x组合键或按F10键，系统会直接进入单用户模式。输入passwd命令重置系统密码，密码输入完成还需要进行密码的二次输入进行确认。示例如图所示。

```
sh-4.4# passwd
Changing password for user root.r
New password: [ 246.150017] psmouseserial:mouseattisa0060/serial/input0klostlsynchronization, throwing 1 bytes away.
Retype new password:
passwd: all authentication tokens updated successfully.
sh-4.4#
```


至此，一台操作系统为CentOS8.0 64位的弹性云主机实例就进入到了单用户模式。

Ubuntu操作系统进入单用户模式

本示例将会对一台操作系统为Ubuntu18.04 64位镜像的弹性云主机实例进行单用户模式进入操作。

1. 登录控制中心。
2. 选择区域  华东-华东1。

最佳实践

3. 单击“计算>弹性云主机”，进入弹性云主机页面。
4. 单击待操作的弹性云主机行的“操作>远程登录”按钮，远程连接弹性云主机实例。
5. 输入用户名root，密码为购买弹性云主机时用户自定义的密码，登录成功之后如图：

```
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-137-generic x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

System information as of Thu Jun 29 15:18:41 CST 2023

System load:  0.23          Processes:    112
Usage of /:   15.2% of 39.12GB  Users logged in:  0
Memory usage: 4%           IP address for eth0: 10.0.0.59
Swap usage:  0%

* Canonical Livepatch is available for installation.
- Reduce system reboots and improve kernel security. Activate at:
  https://ubuntu.com/livepatch

221 updates can be applied immediately.
156 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

root@ecm-ctrlx:~# _
```

最佳实践

6. 输入重启命令reboot，并在重启过程中按shift键，进入GRUB界面，GRUB界面示例如下。

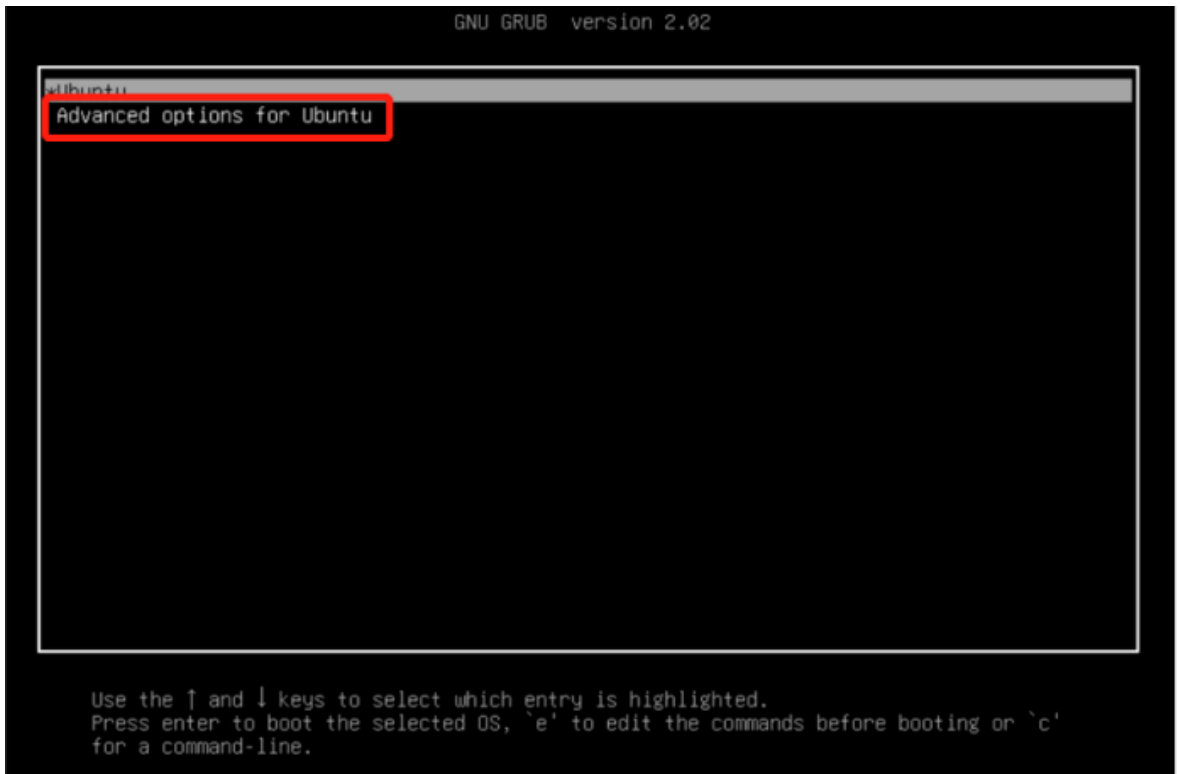
```
GNU GRUB version 2.02

*Ubuntu
Advanced options for Ubuntu

Use the ↑ and ↓ keys to select which entry is highlighted.
Press enter to boot the selected OS, `e' to edit the commands before booting or `c'
for a command-line.
```

最佳实践


7. 使用键盘方向键，向下移动选择GRUB页面第二行的高级选项，并按下键盘的enter键。

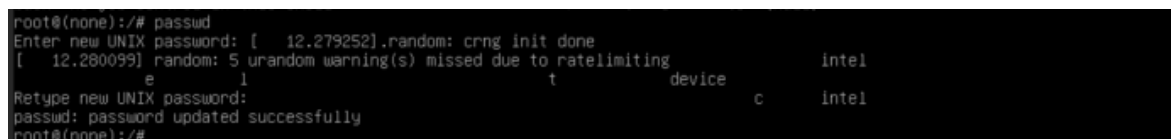


最佳实践

8. 接下来选择新页面中第二行的恢复模式（recovery mode），并按e键编辑启动项。



9. 进入编辑页面，使用键盘的方向键，移动光标向下至linux命令开头的一行，并在本行中将ro至末尾的内容删除并替换为rw single init=/bin/bash。具体的修改信息如下图：
10. 截至目前，进入单用户模式的配置已经基本完成，现在需要用户按下键盘的ctrl+x组合键或按F10键，系统会直接进入单用户模式。输入passwd命令重置系统密码，密码输入完成还需要进行密码的二次输入进行确认。示例如图所示。



至此，一台操作系统为Ubuntu18.04 64位的弹性云主机实例就进入到了单用户模式。

手工安装宝塔面板（CentOS7.2）

应用场景

本文主要介绍如何使用Linux操作系统云主机手工安装宝塔面板。宝塔面板是一款服务器管理软件，支持Windows和Linux系统，可以通过Web端轻松管理服务器，提升运维效率。例如：创建管理网站、FTP、数据库，拥有可视化文件管理器、可视化软件管理器，以及可视化CPU、内存、流量监控图表、计划任务等功能。同时，宝塔面板还提供了简单直观的用户界面，使得初学者也能够轻松地管理自己的服务器。本文以CentOS 7.2 64位操作系统为例，介绍安装宝塔Linux面板过程。

最佳实践

前提条件

云主机资源配置和操作系统要求：

- 云主机规格最低1核1G（软件要求内存最低512MB，推荐768MB以上，纯面板约占系统60MB内存）。
- 云硬盘：任意空间大小（软件要求300M以上可用硬盘空间，纯面板约占20M磁盘空间）。
- 架构：支持x86_64架构（ARM架构不完整兼容，面板环境安装慢，部分软件可能无法安装，请谨慎尝试）。
- 宝塔Linux6.0版本基于Centos7开发，建议使用Centos 7.x系统。
- 云主机全新且环境干净，未安装过Apache、Nginx、php、MySQL。

安装宝塔面板操作流程

Linux实例手工安装宝塔面板的具体操作步骤如下：

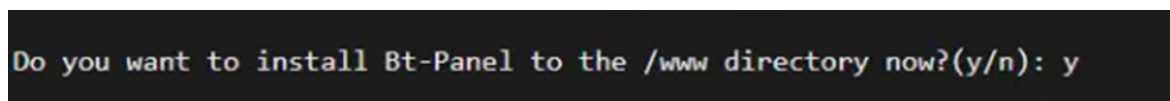
1. 安装宝塔面板。
2. 修改云主机所在安全组规则。
3. 登录宝塔面板。

实施步骤

1. 安装宝塔面板。
 - a. 登录云主机。
 - b. 执行以下命令，下载并安装宝塔面板。

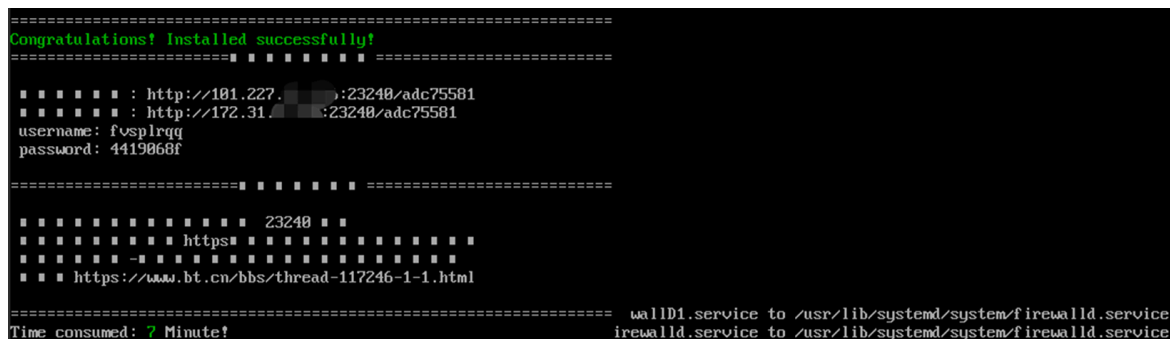
```
yum install -y wget && wget -O install.sh http://download.bt.cn/install/install_6.0.sh && sh install.sh
```

待命令执行完毕。在执行过程中，会提示是否将磁盘挂载到/www，如下图所示：



输入小写y字母确认挂载。

安装完成后，会提示安装成功，并显示面板URL、端口号、“username”、“password”等参数值。



说明

请妥善记录回显信息中URL、端口号、“username”、“password”等参数值，于后续步骤中使用。

最佳实践

2. 修改云主机所在安全组规则。

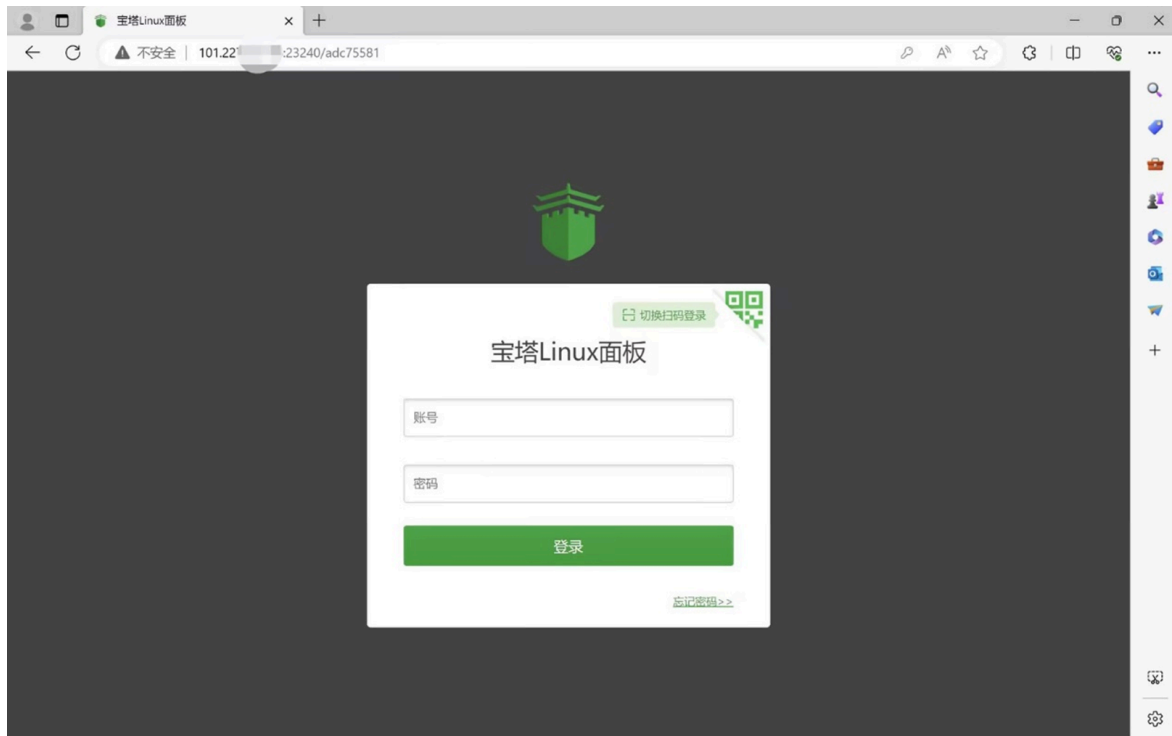
为云主机所在安全组添加安全组规则，规则如下表所示，具体步骤参见[添加安全组规则](#)。

方向	协议	端口/范围	源地址
入方向	TCP	回显信息中的端口号（本示例中为23240）	0.0.0.0/0

最佳实践

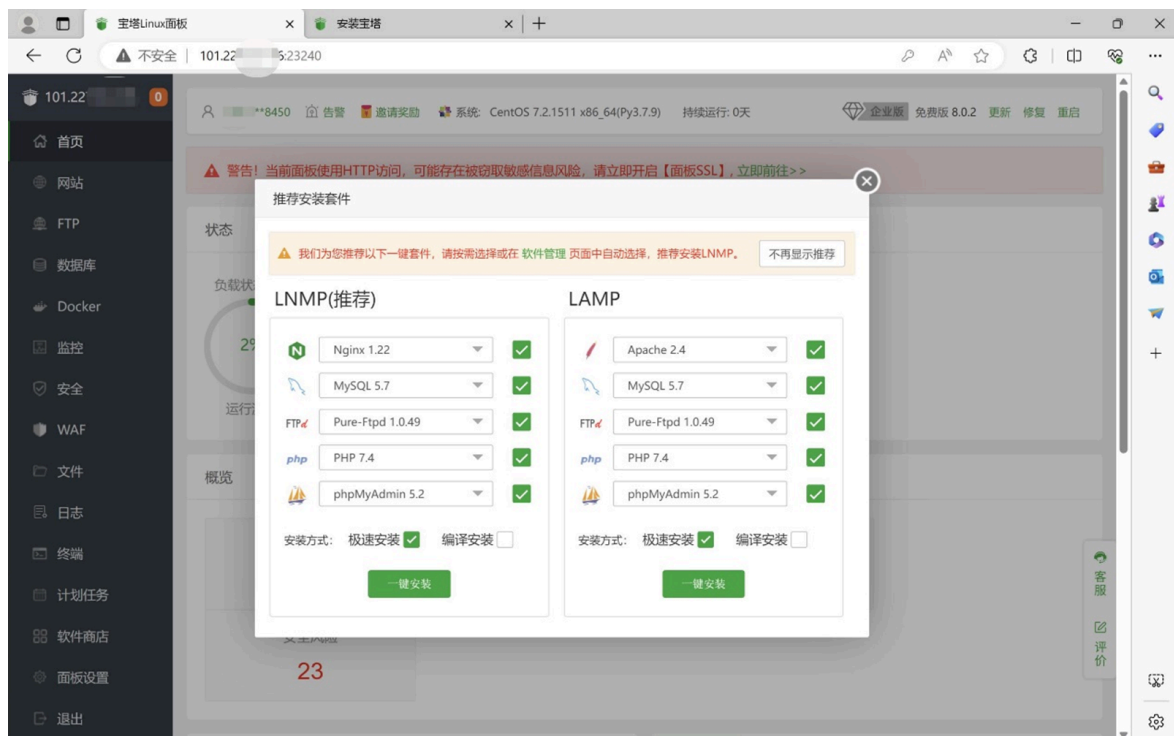
3. 登录宝塔面板。

- a. 在浏览器地址栏中输入回显信息中的URL地址，例如：`http://101.227.xxx.xx:23240/adc75581`。成功访问，显示如下图：



- b. 输入安装步骤中记录的账户和密码，勾选宝塔用户协议，绑定宝塔账号后，进入宝塔管理界面。

最佳实践



c. 根据业务需要可以通过面板安装相关的套件并部署网站。

搭建网站/应用最佳实践

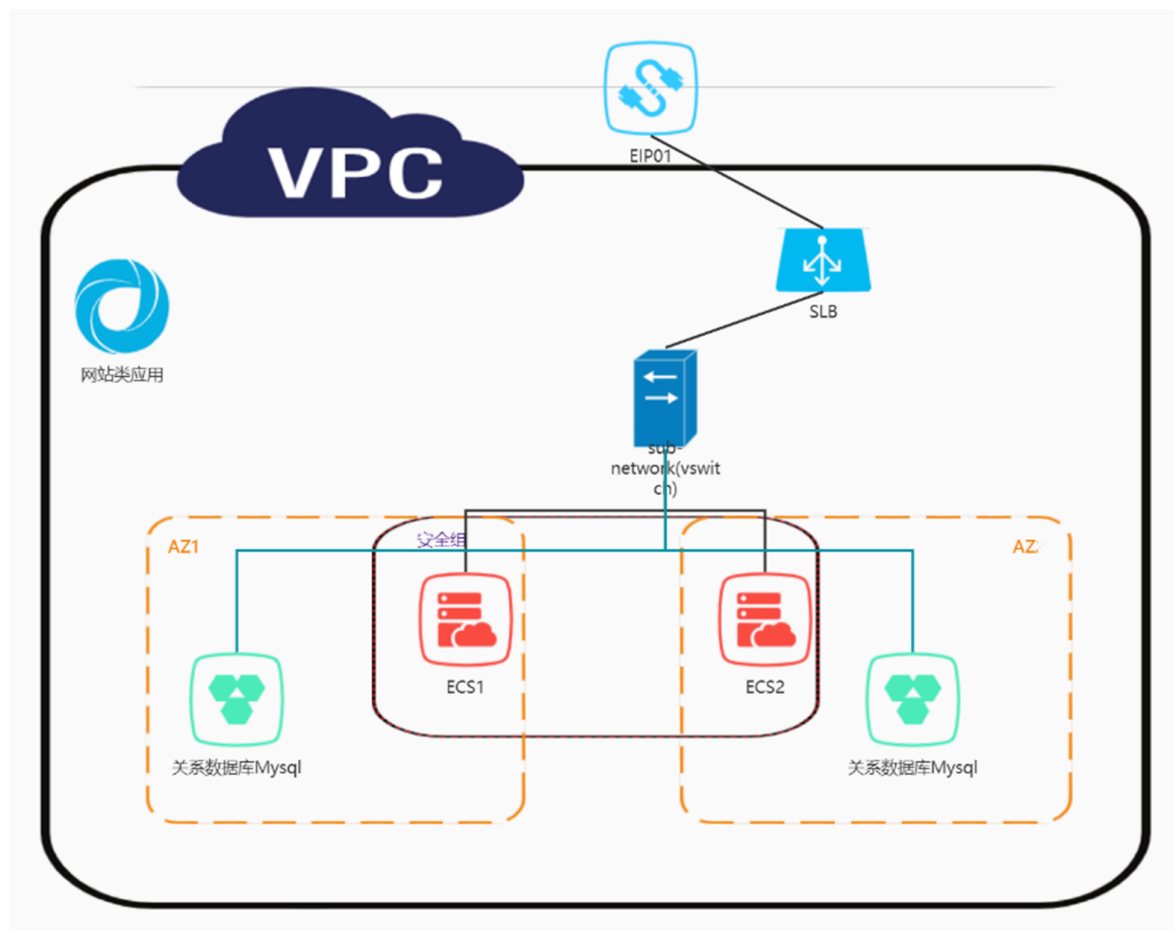
典型网站类场景

场景以WordPress网站高可用场景为例，在不同可用区部署前端云主机，并通过弹性负载均衡实例分摊业务流量，同时后端数据分离到关系数据库。

资源参数

产品	配置	可用区
虚拟私有云	VpcforWebsite	VPC网段192.168.0.0/16，子网192.168.1.0/24
弹性云主机	2*s7.large.4（2核8G）；CentOS 7.6	分别位于可用区1，可用区2
关系型数据库MySQL	主从4核8G；MySQL 5.7；超高IO	-
弹性负载均衡	SLBforWebsite	-
弹性IP	5M带宽	-

网站拓扑



操作步骤

1. 进入控制中心，“网络”>“虚拟私有云”，点击创建虚拟私有云，VPC网段设置为192.168.0.0/16，子网网段设置为192.168.0.0/24。
2. 进入控制中心，“计算”>“弹性云主机”，点击创建云主机ECS1，云主机规格选择s7.large.4，操作系统设置为CentOS 7.6，VPC和子网选择步骤1所创建的VPC和子网。
3. 进入控制中心，“网络”>“弹性IP”，点击创建弹性IP。
4. 在云主机控制台或弹性IP控制台将弹性IP绑定至ECS1。
5. 使用控制台的远程登录功能，登录进入ECS1，部署WordPress。

- 执行如下命令，安装 Apache 和 PHP 组件：

安装 Apache 和 PHP 组件

```
yum install -y httpd php php-mysql php-gd php-imap php-ldap php-odbc php-pear php-xml php-xmlrpc
```

启动 Apache 服务

```
service httpd start
```

设置开机默认启动 Apache

最佳实践

```
sudo chkconfig httpd on
```

- 执行如下命令，创建 info.php 文件，验证php是否安装成功。

```
vim /var/www/html/info.php
```

- 执行如下命令，重启 httpd 服务。

```
service httpd restart
```

- 用可以访问公网的任何节点，访问http://弹性IP/info.php，可见运行成功，（请确保PHP所使用的端口，已在云主机所在安全组中放行）。
- 执行如下命令，卸载当前版本PHP，安装高版本PHP。

```
yum list installed | grep php | awk '{print $1}' | xargs yum remove -y  
rpm -Uvh https://mirror.webtatic.com/yum/el7/epel-release.rpm  
rpm -Uvh https://mirror.webtatic.com/yum/el7/webtatic-release.rpm  
yum install -y php56w.x86_64 php56w-cli.x86_64 php56w-common.x86_64  
php56w-gd.x86_64 php56w-imagick.x86_64 php56w-ldap.x86_64 php56w-  
mysql.x86_64 php56w-pdo.x86_64 php56w-odbc.x86_64 php56w-  
process.x86_64 php56w-xml.x86_64 php56w-xmlrpc.x86_64
```

- 执行如下命令，重启httpd服务。

```
systemctl restart httpd
```

- 执行如下命令，下载并解压WordPress软件包。

```
mkdir -p /opt/WP
```

```
cd /opt/WP
```

```
wget https://cn.wordpress.org/latest-zh_CN.tar.gz
```

```
tar -xzf latest-zh_CN.tar.gz
```

- 拷贝wordpress 目录到/var/www/html/路径下，执行如下命令。

```
cd /var/www/html
```

```
cp -rf /opt/WP/wordpress/* /var/www/html/
```

- 进入控制中心，“数据库” > “关系数据库MySQL版”，创建MySQL数据库实例。
- ECS1中安装mysql客户端，用于创建和访问MySQL数据库实例。

```
yum install mysql
```

- 连接MySQL数据库，创建数据库实例。

```
mysql -h<ip> -P<host> -u<user> -p<password>
```

```
create database <databasename> charset utf8
```

- ECS1中修改wordpress访问数据库配置。

```
cd /var/www/html/
```

```
vim wp-config.php
```

修改配置内容参考下面配置：

```
#####
```

```
// ** MySQL 设置 - 具体信息来自您正在使用的主机 ** //
```

```
/** WordPress 数据库的名称 */
define('DB_NAME', 'db_wordpress');
/** MySQL 数据库用户名 */
define('DB_USER', 'user_wordpress');
/** MySQL 数据库密码 */
define('DB_PASSWORD', 'Hello1234');
/** MySQL 主机 */
define('DB_HOST', '<rds 实例内网地址, 不含端口号="">');
/** 创建数据表时默认的文字编码 */
define('DB_CHARSET', 'utf8');
/** 数据库整理类型。如不确定请勿更改 */
define('DB_COLLATE', '');
#####
```

6. 打开浏览器输入http://弹性IP:端口/wp-admin/install.php，访问WordPress站点。
7. 参考步骤5，申请ECS2并完成相应配置。配置ECS2的过程，可重复利用弹性IP，挂载给ECS2。
8. 进入控制中心，“网络”>“弹性负载均衡”，创建弹性负载均衡，并绑定申请的弹性IP。使用弹性负载均衡后，弹性IP将会从云主机上卸载下并关联到弹性负载均衡上。
9. 执行以下命令，使用ab压测网站，并关闭其中ECS，出现一定概率访问失败的现象。

```
ab. -n 10 -c 10 -t 30 -C 'cookies=xxx' -v 3 http://IP/wp-admin
```

网站搭建指导

网站搭建一般有以下两种方式：

对比项	镜像部署	手动搭建
搭建方式	选择天翼云市场镜像直接安装部署	以手动的方式安装所需软件，可定制化
特点	配套的软件版本相对比较固定	配套版本可以灵活选择
所需时间	比较短，一键部署	比较长，需自行安装相关软件
难易程度	相对比较简单	需要对软件配套版本及安装方法有一定的了解

如果您或您的团队，在建站时，有个性化的部署需求，并且有专业的维护人员，可以采用手动搭建方式。

如果您或您的团队，在建站时，对于操作系统、数据库、中间件等各类软件的选择和配置不是很了解，并且缺少专业维护人员，建议使用镜像部署，后期有镜像服务商的专业售后工程师团队支持。

搭建微信公众号后台

本实践将基于天翼云弹性云主机搭建微信公众号处理后台，使用Python语言编写对应的微信消息处理逻辑代码，接收从微信服务端转发过来的消息，并返回处理结果给最终用户。

准备事项

- 申请微信公众号。

微信公众号申请链接：<https://mp.weixin.qq.com/>

最佳实践

- 购买天翼云弹性云主机服务。

如果没有天翼云账号，需先注册天翼云账号并完成实名认证。

本实践中，使用公共镜像CentOS 7.4。

- 购买弹性IP。

建议同时购买弹性IP，后面需要在微信公众号上配置公网IP的地址。

操作步骤

基础软件安装

本实践中使用Python+Web.py组合完成微信公众号开发，需要安装或升级Python、pip、Web.py框架、WinSCP软件版本。

最佳实践

1. 升级默认Python版本;

CentOS 7.4自带Python版本比较老, 建议升级到Python3。

a. 查看Python版本, 使用如下命令:

```
python - version
```

```
[root@ecs- ~]# python --version
Python 2.7.5
[root@ecs- ~]#
```

b. 下载Python安装包, 这里以Python 3.6.0版本为例, 使用命令:

```
wget https://www.python.org/ftp/python/3.6.0/Python-3.6.0a1.tar.xz
```

```
[root@ecs- ~]# wget https://www.python.org/ftp/python/3.6.0/Python-3.6.0a1.tar.xz
--2020-12-28 09:25:56-- https://www.python.org/ftp/python/3.6.0/Python-3.6.0a1.tar.xz
Resolving www.python.org (www.python.org)... 2a04:4e42:1a::223
Connecting to www.python.org (www.python.org)| :443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 15328032 (15M) [application/octet-stream]
Saving to: 'Python-3.6.0a1.tar.xz'

100%[=====]

2020-12-28 09:25:57 (12.7 MB/s) - 'Python-3.6.0a1.tar.xz' saved [15328032/15328032]

[root@ecs- ~]#
```

c. 解压安装包, 使用如下命令:

```
tar xvf Python-3.6.0a1.tar.xz
```

```
configure: creating ./config.status
config.status: creating Makefile.pre
config.status: creating Modules/Setup.config
config.status: creating Misc/python.pc
config.status: creating Misc/python-config.sh
config.status: creating Modules/ld_so_aix
config.status: creating pyconfig.h
creating Modules/Setup
creating Modules/Setup.local
creating Makefile
```

最佳实践

如果出现“configure: error: no acceptable C compiler found in \$PATH”异常提示，是因为未安装合适的编译器。

解决方法：

执行以下命令，安装/升级gcc及其他依赖的包。

```
sudo yum install gcc-c++
```

并在随后提示安装包是否OK时，输入y并回车。出现如下图提示，说明依赖的包安装成功。

```
Installed:
gcc-c++.x86_64 0:4.8.5-44.el7

Dependency Installed:
libstdc++-devel.x86_64 0:4.8.5-44.el7

Dependency Updated:
cpp.x86_64 0:4.8.5-44.el7          gcc.x86_64 0:4.8.5-44.el7
```

在编译器安装完成后，重新执行 ./configure 命令。

d. 执行命令：

```
make && make install
```

执行成功。但提示pip错误，原因是我的系统中少了openssl-devel包，可以先忽略。

最佳实践

```
rm -f /usr/local/bin/pyvenv
(cd /usr/local/bin; ln -s pyvenv-3.6 pyvenv)
if test "x" != "x" ; then \
    rm -f /usr/local/bin/python3-32; \
    (cd /usr/local/bin; ln -s python3.6-32 python3-32) \
fi
rm -f /usr/local/share/man/man1/python3.1
(cd /usr/local/share/man/man1; ln -s python3.6.1 python3.1)
if test "xupgrade" != "xno" ; then \
    case upgrade in \
        upgrade) ensurepip="--upgrade" ;; \
        install|*) ensurepip="" ;; \
    esac; \
    ./python -E -m ensurepip \
        $ensurepip --root=/ ; \
fi
Ignoring ensurepip failure: pip 8.1.1 requires SSL/TLS
```

e. 查看Python3版本，使用命令：

```
python3 --version
```

```
[root@ecs- Python-3.6.0a1]# python3 --version
Python 3.6.0a1
```

f. 执行命令：

```
python3
```

出现如下提示，则说明Python3安装成功。

```
[root@ecs- Python-3.6.0a1]# python3
Python 3.6.0a1 (default, Dec 18 2020, 15:45:57)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-44)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

最佳实践

2. 升级默认pip版本;

pip是通用的Python包管理工具。提供了对Python包的查找、下载、安装、卸载功能。Python3安装成功后自带pip3，但版本比较老，建议升级到pip最新版本。同时前面安装python3提示“Ignoring ensurepip failure: pip 8.1.1 requires SSL/TLS”错误，导致pip未成功安装，所以需要重新安装pip。

a. 安装openssl-devel包，使用命令：

```
yum install openssl-devel -y
```

```
Installed:
  openssl-devel.x86_64 1:1.0.2k-21.el7_9

Dependency Installed:
  keyutils-libs-devel.x86_64 0:1.5.8-3.el7          krb5-devel.x86_64 0:1.15.1-50.el7
  libselinux-devel.x86_64 0:2.5-15.el7             libsepol-devel.x86_64 0:2.5-10.el7
  zlib-devel.x86_64 0:1.2.7-18.el7

Dependency Updated:
  openssl.x86_64 1:1.0.2k-21.el7_9

Complete!
```

b. 执行命令：

```
make && make install
```

出现如下提示说明pip安装成功。

```
Collecting setuptools
Collecting pip
Installing collected packages: setuptools, pip
Successfully installed pip-8.1.1 setuptools-20.10.1
```

c. 升级pip3，使用命令：

```
pip3 install --upgrade pip
```

出现如下提示说明升级pip到最新版本了。

```
[root@ecs- Python-3.6.0a1]# pip3 install --upgrade pip
Collecting pip
  Downloading https://files.pythonhosted.org/packages/54/eb/4a36
  100% |#####| 1.5MB 32kB/s
Installing collected packages: pip
  Found existing installation: pip 8.1.1
  Uninstalling pip-8.1.1:
    Successfully uninstalled pip-8.1.1
  Successfully installed pip-20.3.3
```

3. 安装Web.py框架;

Web.py官方教程地址: <http://webpy.org/>, 使用如下命令安装web.py:

```
pip3 install web.py==0.40.dev0
```

```
[root@ecs-c438 Python-3.6.0a1]# pip3 install web.py==0.40.dev0
Collecting web.py==0.40.dev0
  Downloading web.py-0.40.dev0.tar.gz (116 kB)
    |████████████████████████████████████████| 116 kB 76 kB/s
Using legacy 'setup.py install' for web.py, since package 'wheel'
Installing collected packages: web.py
  Running setup.py install for web.py ... done
Successfully installed web.py-0.40.dev0
```

4. 安装WinSCP;

通常情况下,我们在本地Windows操作系统上编辑代码,完成后再上传至ECS上(CentOS Linux系统)。WinSCP是一个Windows环境下使用的SSH的开源图形化SFTP客户端,同时支持SCP协议。它的主要功能是在本地与远程计算机间安全地复制文件,并且可以直接编辑文件。

WinSCP安装链接: <https://winscp.net/eng/docs/lang:chs>

上传代码

1. 新建main.py文件,复制如下代码:

```
# -*- coding: utf-8 -*-
# filename: main.py
import web
from handle import Handle
urls = (
    '/wx', 'Handle',
)
if __name__ == '__main__':
    app = web.application(urls, globals())
    app.run()
```

2. 新建handle.py文件,复制如下代码:

```
# -*- coding: utf-8 -*-
# filename: handle.py
import hashlib
import web
import receive
import time
import os
class Handle(object):
    def __init__(self):
        self.app_root = os.path.dirname(__file__)
        self.templates_root = os.path.join(self.app_root, 'templates')
```

```
self.render = web.template.render(self.templates_root)
def GET(self):
    try:
        data = web.input()
        if len(data) == 0:
            return "hello, this is handle view"
        signature = data.signature
        timestamp = data.timestamp
        nonce = data.nonce
        echostr = data.echostr
        token = "此处内容与公众号基本配置里Token字段取值保持一致"
        list = [token, timestamp, nonce]
        list.sort()
        s = list[0] + list[1] + list[2]
        hashcode = hashlib.shal(s.encode('utf-8')).hexdigest()
        print("handle/GET func: hashcode, signature:", hashcode, signature)
        if hashcode == signature:
            return echostr
        else:
            return echostr
    except (Exception) as Argument:
        return Argument
def POST(self):
    try:
        webData = web.data()
        print("Handle Post webdata is:\n", webData)
        #打印消息体日志
        recMsg = receive.parse_xml(webData)
        if isinstance(recMsg, receive.Msg) and recMsg.MsgType == 'text':
            toUser = recMsg.FromUserName
            fromUser = recMsg.ToUserName
            content = "欢迎" + str(recMsg.Content)
            print('Reply message info:\n')
            print('toUser =', toUser)
            print('fromUser =', fromUser)
            print('content =', content)
            return self.render.reply_text(toUser, fromUser, int(time.time()), content)
        else:
            print("不支持的消息类型: ", recMsg.MsgType)
            return "success"
    except (Exception) as Argment:
        return Argment
```

3. 新建receive.py文件，复制如下代码：

```
# -*- coding: utf-8 -*-
# filename: receive.py
```

```
import xml.etree.ElementTree as ET
def parse_xml(web_data):
    if len(web_data) == 0:
        return None
    xmlData = ET.fromstring(web_data)
    msg_type = xmlData.find('MsgType').text
    if msg_type == 'text':
        return TextMsg(xmlData)
    elif msg_type == 'image':
        return ImageMsg(xmlData)
    elif msg_type == 'location':
        return LocationMsg(xmlData)
    elif msg_type == 'event':
        return EventMsg(xmlData)
class Event(object):
    def __init__(self, xmlData):
        self.ToUserName = xmlData.find('ToUserName').text
        self.FromUserName = xmlData.find('FromUserName').text
        self.CreateTime = xmlData.find('CreateTime').text
        self.MsgType = xmlData.find('MsgType').text
        self.Eventkey = xmlData.find('EventKey').text
class Msg(object):
    def __init__(self, xmlData):
        self.ToUserName = xmlData.find('ToUserName').text
        self.FromUserName = xmlData.find('FromUserName').text
        self.CreateTime = xmlData.find('CreateTime').text
        self.MsgType = xmlData.find('MsgType').text
        self.MsgId = xmlData.find('MsgId').text
class TextMsg(Msg):
    def __init__(self, xmlData):
        Msg.__init__(self, xmlData)
        self.Content = xmlData.find('Content').text
class ImageMsg(Msg):
    def __init__(self, xmlData):
        Msg.__init__(self, xmlData)
        self.PicUrl = xmlData.find('PicUrl').text
        self.MediaId = xmlData.find('MediaId').text
class LocationMsg(Msg):
    def __init__(self, xmlData):
        Msg.__init__(self, xmlData)
        self.Location_X = xmlData.find('Location_X').text
        self.Location_Y = xmlData.find('Location_Y').text
class EventMsg(Msg):
    def __init__(self, xmlData):
        Event.__init__(self, xmlData)
        self.Event = xmlData.find('Event').text
```

最佳实践

4. 新建templates文件夹，在文件夹下新建reply_text.xml文件，复制如下代码：

```
$def with (toUser, fromUser, createTime, content)
<xml>
<tousername><!--[CDATA[$toUser]]--></tousername>
<fromusername><!--[CDATA[$fromUser]]--></fromusername>
<createtime>$createTime</createtime>
<msgtype><!--[CDATA[text]]--></msgtype>
<content><!--[CDATA[$content]]--></content>
</xml>
```

5. 最终本地代码文件形成如下：

```
D:\workspace\wx\textProcess
2018/04/21  14:31    <DIR>          .
2018/04/21  14:31    <DIR>          ..
2018/04/20  13:42             2,077 handle.py
2018/04/11  23:13             211 main.py
2018/04/19  23:46             2,008 receive.py
2018/04/20  13:41    <DIR>          templates

D:\workspace\wx\textProcess\templates
2018/04/20  13:41    <DIR>          .
2018/04/20  13:41    <DIR>          ..
2018/04/20  13:14             275 reply_text.xml
```

6. 通过WinSCP工具将上述文件与目录上传至ECS指定目录下：

```
[root@ecs-test1-0001 wx]# ls -lR
.:
total 16
-rw-r--r-- 1 root root 2077 Apr 20 13:42 handle.py
-rw-r--r-- 1 root root 211 Apr 11 23:13 main.py
-rw-r--r-- 1 root root 2008 Apr 19 23:46 receive.py
drwxr-xr-x 2 root root 4096 May 7 22:40 templates

./templates:
total 4
-rw-r--r-- 1 root root 275 Apr 20 13:14 reply_text.xml
```

最佳实践

启动服务

使用如下命令启动服务：

```
python3 main.py 80
```

```
[root@ecs-test1-0001 wx]# python3 main.py 80
http://0.0.0.0:80/
```

启用开发者模式

1. 登录微信公众平台，选择“开发 > 基本配置”，单击“修改配置”。
2. 填写配置信息，单击“提交”。
 - URL：https://ECS的弹性公网IP/wx，无需添加 80 端口。
 - Token：需要与handle.py中对应token取值完全一致。
 - EncodingAESKey：随机生成。
 - 消息加解密方式：此为示例，选择简单的“明文模式”。
3. 验证token成功，单击“启用”。

说明

如果token验证失败，请检查Token配置与handle.py中GET消息处理代码是否一致。

验证

使用微信关注公众号，任意发送一条文本消息，看是否能够收到回复。如能收到回复则表明系统处理正常。

网站性能测试的最佳实践

天翼云通用型S6、计算增强型C6 等新一代云主机上提供超高网络性能，您可通过本实践提供的 netperf 和 DPDK 两种网络性能测试方法，进行云主机高吞吐网络性能测试。

推荐选择 netperf 方法进行测试，netperf 为通常使用的测试方法，可满足大多数测试场景。但当云主机配置较高（pps 超过1000万且带宽大于50Gbps）时，netperf包含云主机内核协议栈的完整处理路径对网络性能损耗较大，而 DPDK 可屏蔽虚拟机内核协议栈的差异，获取虚拟机网卡的网络性能，此时可选择 DPDK 方法进行测试。

netperf测试

工具介绍

- Netperf

HP 开发的网络性能测量工具，主要测试 TCP 及 UDP 吞吐量性能。测试结果主要反应系统向其他系统发送数据的速度，以及其他系统接收数据的速度。

- SAR

用于监控网络流量，运行示例如下：

```
sar -n DEV 1
```

最佳实践

```
02:41:03 PM IFACE rxpck/s txpck/s rxkB/s txkB/s rxcmp/s txcmp/s rxcst/s
02:41:04 PM eth0 1626689.00 8.00 68308.62 1.65 0.00 0.00
0.0002:41:04 PM lo 0.00 0.00 0.00 0.00 0.00 0.00
0.0002:41:04 PM IFACE rxpck/s txpck/s rxkB/s txkB/s rxcmp/s txcmp/s rxcst/s
02:41:05 PM eth0 1599900.00 1.00 67183.30 0.10 0.00 0.00
0.0002:41:05 PM lo 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

sar -n DEV 1: 该命令每秒钟报告一次网络设备的性能统计信息

字段解释

字段	单位	说明
rxpck/s	pps	每秒收包量，即接收 pps
txpck/s	pps	每秒发包量，即发送 pps
rxkB/s	kB/s	接收带宽
txkB/s	kB/s	发送带宽

测试场景

测试场景	客户端运行命令	SAR 监控指标
UDP 64	netperf -t UDP_STREAM -H-1 10000 -- -m 64 -R 1 &	PPS
TCP 1500	netperf -t TCP_STREAM -H-1 10000 -- -m 1500 -R 1 &	带宽
TCP RR	netperf -t TCP_RR -H-1 10000 -- -r 32,128 -R 1 &	PPS

命令说明:

- netperf: 这是 Netperf 工具的命令，用于执行网络性能测试。
- -t UDP_STREAM/TCP_STREAM: 指定测试的类型为 UDP/TCP 流量测试。
- -H: 指定远程主机的地址。在这里，-H 后面应该跟上远程主机的地址或主机名。
- -l: 指定测试的时间限制，后接时间，单位为秒。
- -m: 设置消息大小，单位为字节。
- -R: 指定测试运行的数量。
- &: 将测试命令放入后台运行。

性能指标

指标	说明
64字节 UDP 收发 PPS (包/秒)	表示通过 UDP 进行批量数据传输时的数据传输吞吐量，能反映网络极限转发能力 (可能会存在丢包)。
1500字节 TCP 收发带宽 (Mbits/秒)	表示通过 TCP 进行批量数据传输时的数据传输吞吐量，能反映网络极限带宽能力 (可能会存在丢包)。

最佳实践

指标	说明
TCP-RR (次/秒)	表示在 TCP 长链接中反复进行 Request/Response 操作的交易吞吐量, 能反映 TCP 不丢包网络转发能力。

操作步骤

准备测试环境

1. 准备3台测试机器, 请参见[创建弹性云主机](#)购买测试机器。本文测试机器使用 CentOS 8.2 操作系统。
2. 依次登录测试机器, 并执行以下命令安装 netperf 工具。

```
yum install -y sysstat wget tar automake make gccwget -O netperf-2.7.0.tar.gz -c https://codeload.github.com/HewlettPackard/netperf/tar.gz/netperf-2.7.0tar zxf netperf-2.7.0.tar.gzcd netperf-netperf-2.7.0./autogen.sh && ./configure && make && make install
```

测试发包性能

1. 分别在机器中执行命令 “pkill netserver && pkill netperf”, 停止残余的 netperf 和 netserver 进程。
2. 将其中的机器 a 作为客户端, 机器 b 和机器 c 作为服务端。在服务端中执行命令 “netserver”, 运行 netserver。
 - 若返回结果如下图所示, 则说明仍存在其他 netserver 进程。请执行步骤1中的命令, 停止该进程。

```
[root@VM-2-8-centos ~]# netserver
Unable to start netserver with 'IN(6)ADDR_ANY' port '12865' and family AF_UNSPEC
[root@VM-2-8-centos ~]#
```

- 若返回结果如下图所示, 则说明已成功运行 netserver, 请继续下一步操作。

```
[root@VM-2-8-centos ~]# netserver
Starting netserver with host 'IN(6)ADDR_ANY' port '12865' and family AF_UNSPEC
[root@VM-2-8-centos ~]#
```

3. 在客户端中执行 测试场景 中提供的命令, 不断增减 netperf 进程, 直到客户端发包性能不再增加。
 - 执行以下命令, 启动10个netperf客户端进程

```
bash netperf 10 <serverip1> <serverip2> <serverip3> <serverip4> <serverip5> <serverip6> <serverip7> <serverip8> <serverip9> <serverip10>
```

- 同时向b、c两台或者多台客户端发送数据, 代码如下:

```
#!/bin/bash
count=$1
shift
while [ $# != 0 ];do
for ((i=1;i<=count;i++))
do
server_ip=$1
[ -z $server_ip ] && exit 0
# 下方命令可以替换为测试场景表格中的命令
# -H 后填写服务器 IP 地址;
# -l 后为测试时间, 为了防止 netperf 提前结束, 因此时间设为 10000;
netperf -t UDP_STREAM -H $server_ip -l 10000 -- -m 64 -R 1 &
```

最佳实践

```
echo "$server_ip run $i times"
done
shift
done
```

4. 在客户端执行命令“sar -n DEV 1”，观察客户端发包性能变化，取最大值。

根据所得结果，参考性能指标进行分析，即可测出云主机高吞吐网络性能。

测试收包性能

1. 分别在机器中执行命令“pkill netserver && pkill netperf”，停止残余的 netperf 和 netserver 进程。
2. 将其中的机器 a 作为服务端，机器 b 和机器 c 作为客户端。在服务端中执行命令“netserver”，运行 netserver。

若返回结果如下图所示，则说明仍存在其他 netserver 进程。请执行 步骤1 中的命令，停止该进程。

```
[root@VM-2-8-centos ~]# netserver
Unable to start netserver with 'IN(6)ADDR_ANY' port '12865' and family AF_UNSPEC
[root@VM-2-8-centos ~]#
```

若返回结果如下图所示，则说明已成功运行 netserver，请继续下一步操作。

```
[root@VM-2-8-centos ~]# netserver
Starting netserver with host 'IN(6)ADDR_ANY' port '12865' and family AF_UNSPEC
[root@VM-2-8-centos ~]#
```

3. 在客户端中执行 测试场景 中提供的命令，不断增减 netperf 进程，直到客户端发包性能不再增加。

在服务端执行命令“sar -n DEV 1”，观察服务端收包性能变化，取最大值。

4. 根据所得结果，参考性能指标 进行分析，即可测出云主机高吞吐网络性能。

附录

测试辅助脚本：

执行该脚本，可快速发起多个 netperf 进程。

```
#!/bin/bash
count=$1
for ((i=1;i<=count;i++))
do
echo "Instance:$i-----"
# 下方命令可以替换为测试场景表格中的命令
# -H 后填写服务器 IP 地址;
# -l 后为测试时间，为了防止 netperf 提前结束，因此时间设为 10000;
netperf -t UDP_STREAM -H <server ip=""> -l 10000 -- -m 64 -R 1 &
done
```

DPDK测试

编译安装 DPDK

1. 准备2台测试机器，请参见 自定义配置 Linux 云主机 购买测试机器。本文测试机器使用 CentOS 8.2 操作系统。

最佳实践

2. 依次登录测试机器，并执行以下命令下载 DPDK 工具。

```
yum install -y sysstat wget tar automake make gcc
wget http://git.dpdk.org/dpdk/snapshot/dpdk-17.11.tar.gz
tar -xf dpdk-17.11.tar.gz
mv dpdk-17.11 dpdk
```

3. 修改 txonly 引擎，使每个 DPDK 发包 CPU 上的 UDP 流量的端口变动产生多条流。

- a. 执行以下命令，修改 dpdk/app/test-pmd/txonly.c 文件。

```
vim dpdk/app/test-pmd/txonly.c
```

按 i 进入编辑模式，修改以下内容：

- 找到 `#include "testpmd.h"`，另起一行输入如下内容：

```
RTE_DEFINE_PER_LCORE(struct udp_hdr, lcore_udp_hdr);
```

修改完成后，如下图所示：

```
#include "testpmd.h"
RTE_DEFINE_PER_LCORE(struct udp_hdr, lcore_udp_hdr);

#define UDP_SRC_PORT 1024
#define UDP_DST_PORT 1024
```

- 找到 `ol_flags |= PKT_TX_MACSEC;`，另起一行输入如下内容：

```
/* dummy test udp port */
static uint16_t test_port = 0;
test_port++;
memcpy(&RTE_PER_LCORE(lcore_udp_hdr), &pkt_udp_hdr, sizeof(pkt_udp_hdr));
RTE_PER_LCORE(lcore_udp_hdr).src_port = rte_cpu_to_be_16(rte_lcore_id() * 199 + test_port
% 16);
```

最佳实践

```
RTE_PER_LCORE(lcore_udp_hdr).dst_port = rte_cpu_to_be_16(rte_lcore_id() * 1999 + test_port % 16);
```

修改完成后，如下图所示：

```
if (txp->tx_ol_flags & TESTPMD_TX_OFFLOAD_INSERT_QINQ)
    ol_flags |= PKT_TX_QINQ_PKT;
if (txp->tx_ol_flags & TESTPMD_TX_OFFLOAD_MACSEC)
    ol_flags |= PKT_TX_MACSEC;

/* dummy test udp port */
static uint16_t test_port = 0;
test_port++;
memcpy(&RTE_PER_LCORE(lcore_udp_hdr), &pkt_udp_hdr, sizeof(pkt_udp_hdr));
RTE_PER_LCORE(lcore_udp_hdr).src_port = rte_cpu_to_be_16(rte_lcore_id() * 199 + test_port % 16);
RTE_PER_LCORE(lcore_udp_hdr).dst_port = rte_cpu_to_be_16(rte_lcore_id() * 1999 + test_port % 16);

for (nb_pkt = 0; nb_pkt < nb_pkt_per_burst; nb_pkt++) {
    pkt = rte_mbuf_raw_alloc(mbp);
    if (pkt == NULL) {
        nomore_mbuf:
        if (nb_pkt == 0)
            return;
        break;
    }
}
```

- 找到 `copy_buf_to_pkt(&pkt_udp_hdr, sizeof(pkt_udp_hdr), pkt, ,` 将其替换为如下内容：

```
copy_buf_to_pkt(&RTE_PER_LCORE(lcore_udp_hdr), sizeof(RTE_PER_LCORE(lcore_udp_hdr)), pkt,
```

修改完成后，如下图所示：

```
copy_buf_to_pkt(&eth_hdr, sizeof(eth_hdr), pkt, 0);
copy_buf_to_pkt(&pkt_ip_hdr, sizeof(pkt_ip_hdr), pkt,
    sizeof(struct ether_hdr));
copy_buf_to_pkt(&RTE_PER_LCORE(lcore_udp_hdr), sizeof(RTE_PER_LCORE(lcore_udp_hdr)), pkt,
    sizeof(struct ether_hdr) +
    sizeof(struct ipv4_hdr));
```

按 `Esc` 输入 `:wq` 保存修改并退出。

- 执行以下命令，修改 `dptk/config/common_base` 文件。

```
vim dptk/config/common_base
```

按 `i` 进入编辑模式，找到 `CONFIG_RTE_MAX_MEMSEG=256`，将其修改为`1024`。修改完成后如下图所示：

```
CONFIG_RTE_LIBRTE_EAL=y
CONFIG_RTE_MAX_LCORE=128
CONFIG_RTE_MAX_NUMA_NODES=8
CONFIG_RTE_MAX_MEMSEG=1024
CONFIG_RTE_MAX_MEMZONE=2560
CONFIG_RTE_MAX_TAILQ=32
```

按 `Esc` 输入 `:wq` 保存修改并退出。

4. 执行以下命令，将 dpdk/app/test-pmd/txonly.c 的 IP 地址修改为测试机器所用 IP。

```
vim dpdk/app/test-pmd/txonly.c
```

按 i 进入编辑模式，找到如下内容：

```
#define IP_SRC_ADDR (198U << 24) | (18 << 16) | (0 << 8) | 1;
#define IP_DST_ADDR (198U << 24) | (18 << 16) | (0 << 8) | 2;
```

按数字198、18、0、1替换为机器 IP，SRC_ADDR 为发送端 IP，DST_ADDR 为接收端 IP。

5. 对应机器操作系统，执行以下命令，安装 numa 库。

```
yum install numactl-devel
```

6. 在 dpdk/ 目录下执行以下命令，关闭 KNI。

```
sed -i "s/^(^CONFIG_.KNI.)=y/1=n/g" ./config/*
```

7. 若您的操作系统内核版本较高（例如5.3），则请执行以下命令，屏蔽差异。

```
sed -i "s/^(^WERROR_FLAGS += -Wundef -Wwrite-strings$)/1 -Wno-address-of-packed-member/g" ./mk/toolchain/gcc/rte.vars.mksed -i "s/fall back/falls through -/g" ./lib/librte_eal/linuxapp/igb_uio/igb_uio.c
```

8. 执行以下命令，编译DPDK。

```
make defconfigmake -j
```

配置大页内存

执行以下命令，配置大页内存。

```
let hugepagesall=$(cat /proc/meminfo | grep MemTotal | awk '{print $2}')/1024/2048
#配置时不超过理论可配大页个数
echo $hugepagesall > /sys/kernel/mm/hugepages/hugepages-2048kB/nr_hugepages
```

装载内核模块及绑定接口

1. 切换登录方式为使用VNC。由于将网卡驱动绑定至 igb_uio 用户态驱动后，该网卡将无法通过 ssh 或 IP 访问，仅支持通过 VNC 或 console 方式访问。
2. 依次执行以下命令，装载 UIO 模块及绑定 virtio 接口。

```
ifconfig eth0 0
ifconfig eth0 down
modprobe uio
insmod /root/dpdk/build/kmod/igb_uio.ko
cd /root/dpdk/usertools/
python3 dpdk-devbind.py --bind=igb_uio 00:05.0
```

完成测试后，可通过请执行以下命令，恢复网卡变更。

```
cd /root/dpdk/usertools/
python3 dpdk-devbind.py --bind=virtio-pci 00:05.0
ifconfig eth0 up
```

测试带宽及吞吐量

1. 执行以下命令，发送端采用 TX only 模式启动 testpmd，接收端启用 rxonly 模式。

- 发送端：

```
/root/dpdk/build/app/testpmd -- --txd=128 --rxd=128 --txq=16 --rxq=16 --nb-cores=1 --
forward-mode=txonly --txpkts=1430 --stats-period=1
```

- 接收端：

```
/root/dpdk/build/app/testpmd -- --txd=128 --rxd=128 --txq=48 --rxq=48 --nb-cores=16 --
forward-mode=rxonly --stats-period=1
```

2. 执行以下命令，测试 pps（UDP 64B 小包）。

- 发送端：

```
/root/dpdk/build/app/testpmd -- --txd=128 --rxd=128 --txq=16 --rxq=16 --nb-cores=3 --
forward-mode=txonly --txpkts=64 --stats-period=1
```

- 接收端：

```
/root/dpdk/build/app/testpmd -- --txd=128 --rxd=128 --txq=48 --rxq=48 --nb-cores=16 --
forward-mode=rxonly --stats-period=1
```

得出如下图所示测试结果：

```
Port statistics =====
##### NIC statistics for port 0 #####
RX-packets: 0          RX-missed: 0          RX-bytes: 0
RX-errors: 0
RX-nobuf: 0
TX-packets: 69283890496 TX-errors: 0          TX-bytes: 99075963420720

Throughput (since last show)
Rx-pps: 0
Tx-pps: 31967172
#####

Port statistics =====
##### NIC statistics for port 0 #####
RX-packets: 11855403490 RX-missed: 0          RX-bytes: 16953226942600
RX-errors: 0
RX-nobuf: 0
TX-packets: 0          TX-errors: 0          TX-bytes: 0

Throughput (since last show)
Rx-pps: 4692725
Tx-pps: 0
#####
```

网络带宽计算

可根据接收端 PPS 和测试包长来计算当前网络的接收带宽，公式如下：

$PPS \times \text{packet length} \times 8\text{bit/B} \times 10^{-9} = \text{带宽}$

结合测试得出数据，可得当前带宽为：

$4692725\text{pps} \times 1430\text{B} \times 8\text{bit/B} \times 10^{-9} \approx 53\text{Gbps}$

快速构建FTP站点（Windows）

Windows实例搭建FTP站点具体操作步骤

1. 添加IIS的角色和功能。
2. 创建FTP服务用户名及密码。
3. 设置共享文件权限。
4. 配置FTP站点。
5. 配置FTP防火墙。
6. 配置安全组规则及防火墙策略。

7. 客户端进行连接测试。

示例环境

- 实例类型：s3.large.2 | 2核 | 4G 通用型云主机
- 所在区域：山西
- 系统盘：40GB
- 操作系统：Windows 2012 Standard R2 64位中文版
- 公网弹性IP带宽：1Mbps

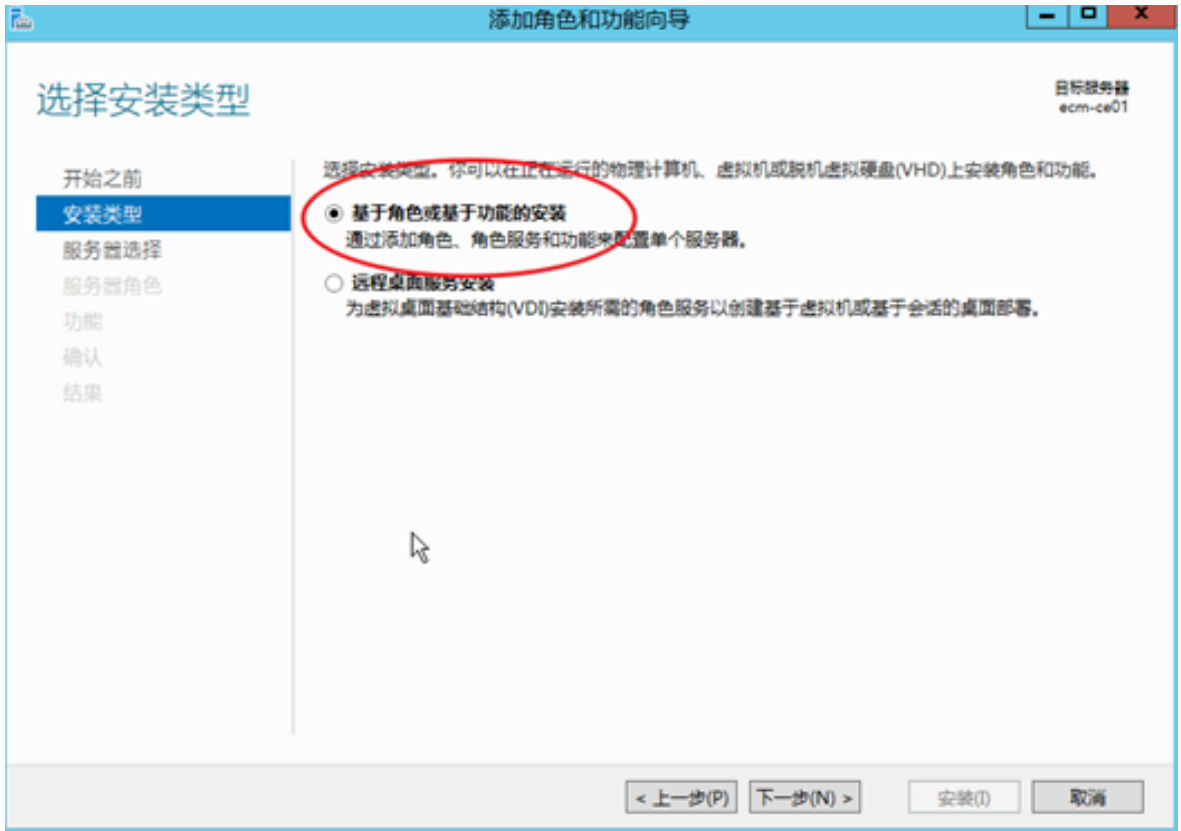
最佳实践

操作步骤

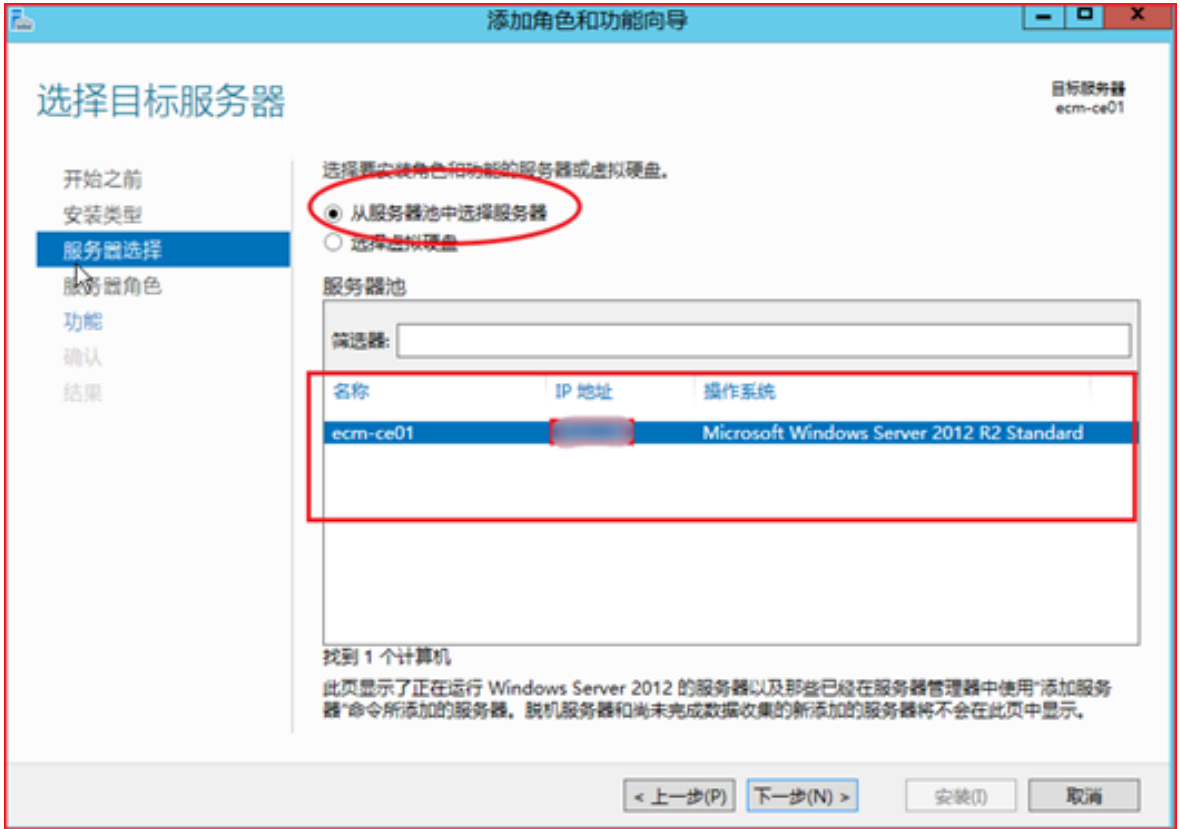
1. 添加IIS的角色和功能。
 - a. 登录弹性云主机。
 - b. 选择“开始 > 服务器管理器”。
 - c. 单击“添加角色和功能”。



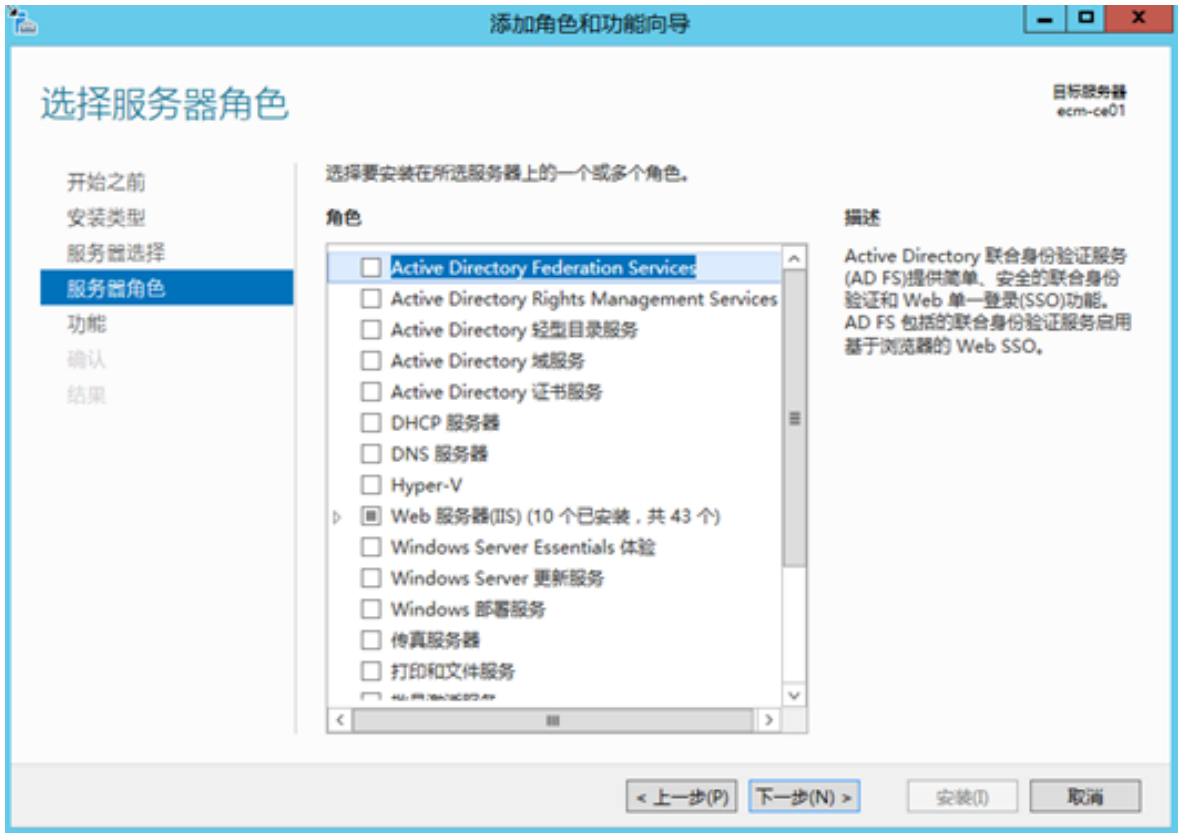
- d. 在弹出的“开始之前”对话框中，单击“下一步”。
 - e. 选择“基于角色或基于功能的安装”，单击“下一步”。



f. 选择需要部署FTP的服务器，单击“下一步”。

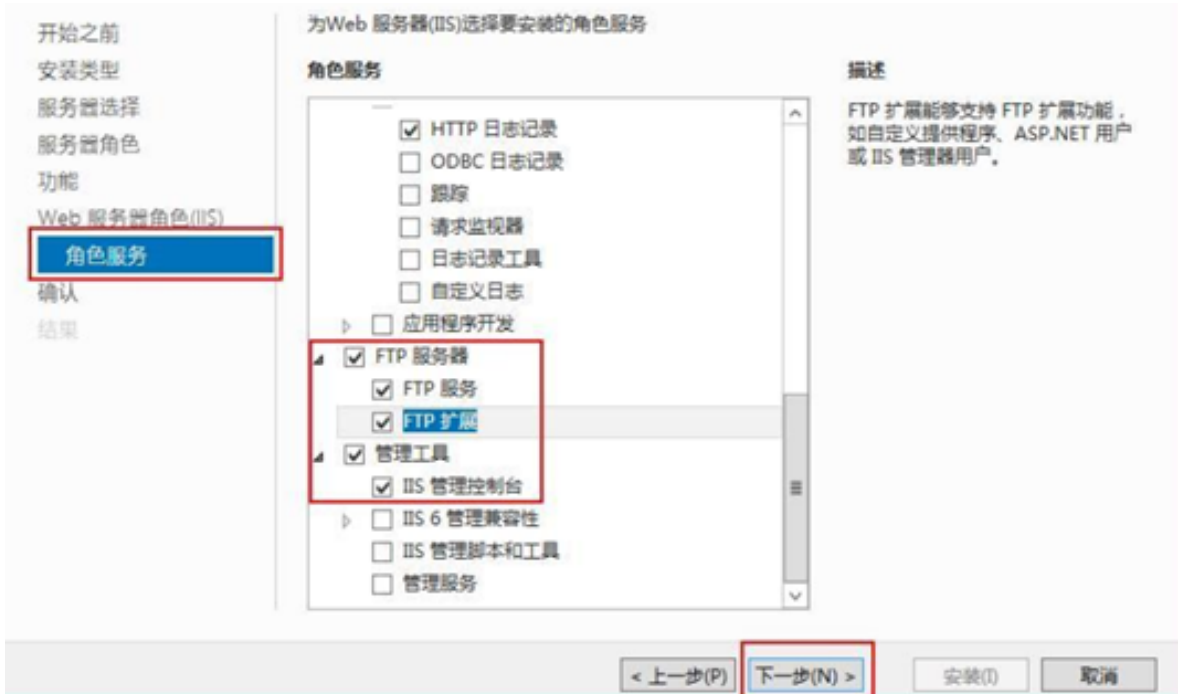


g. 选择“Web服务器（IIS）”，并在弹出的对话框中单击“添加功能”，然后单击“下一步”。



h. 连续单击“下一步”，到“角色服务”页面。

i. 选择“FTP服务器”以及“IIS管理控制台”，单击“下一步”。



- j. 单击“安装”，开始部署服务角色。
- k. 安装完成后，单击“关闭”。
 - 1. 添加IIS的角色和功能
 - 2. 添加IIS的角色和功能
 - 3. 添加IIS的角色和功能
 - 4. 添加IIS的角色和功能

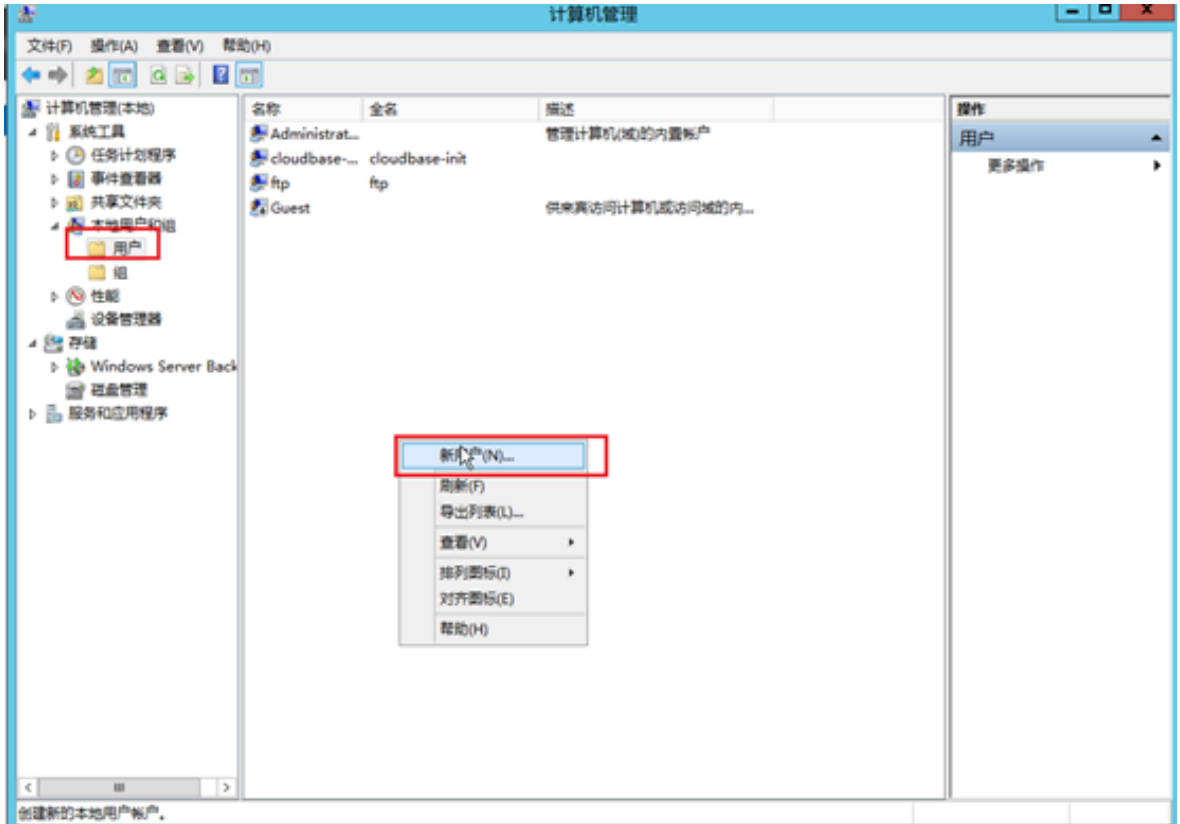
最佳实践

2. 创建FTP服务用户名及密码。

- a. 在“服务器管理器”中，选择“仪表盘 > 工具 > 计算机管理”。

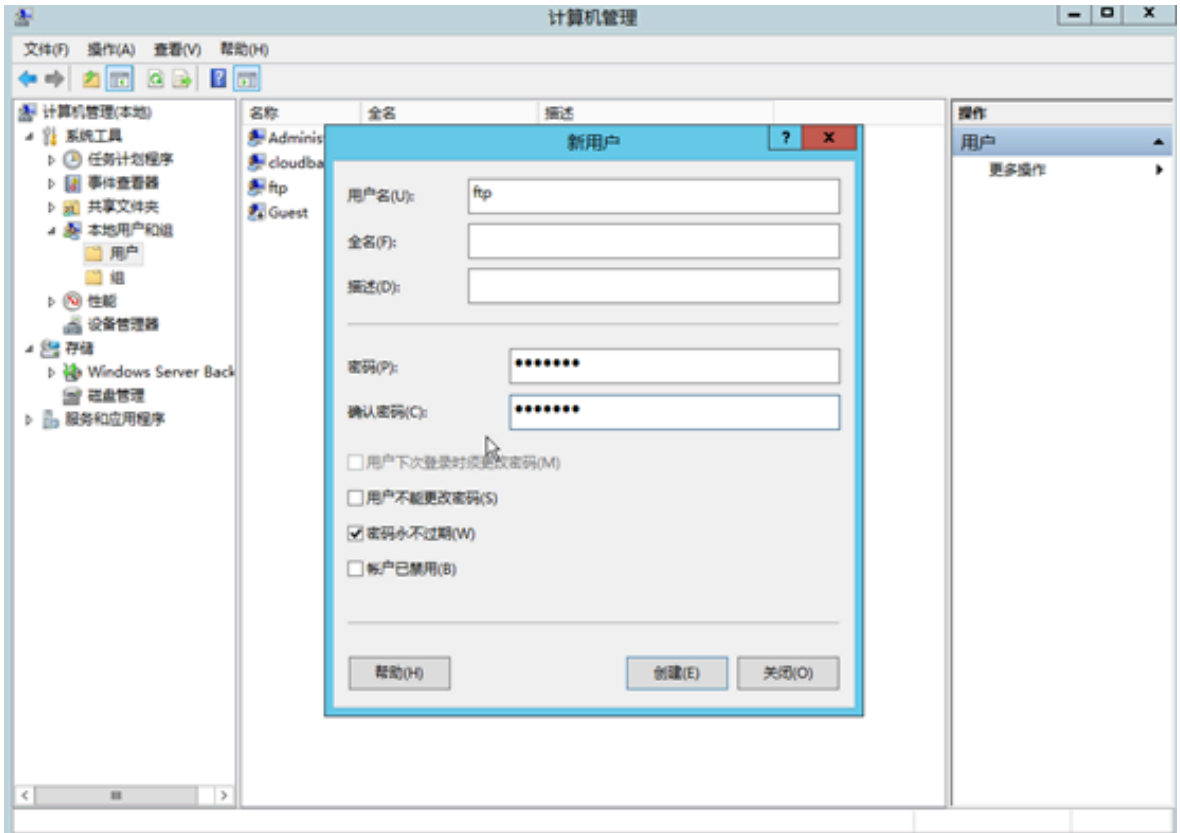


- b. 选择“系统工具 > 本地用户和组 > 用户”，在右侧空白处右击，并选择“新用户”。



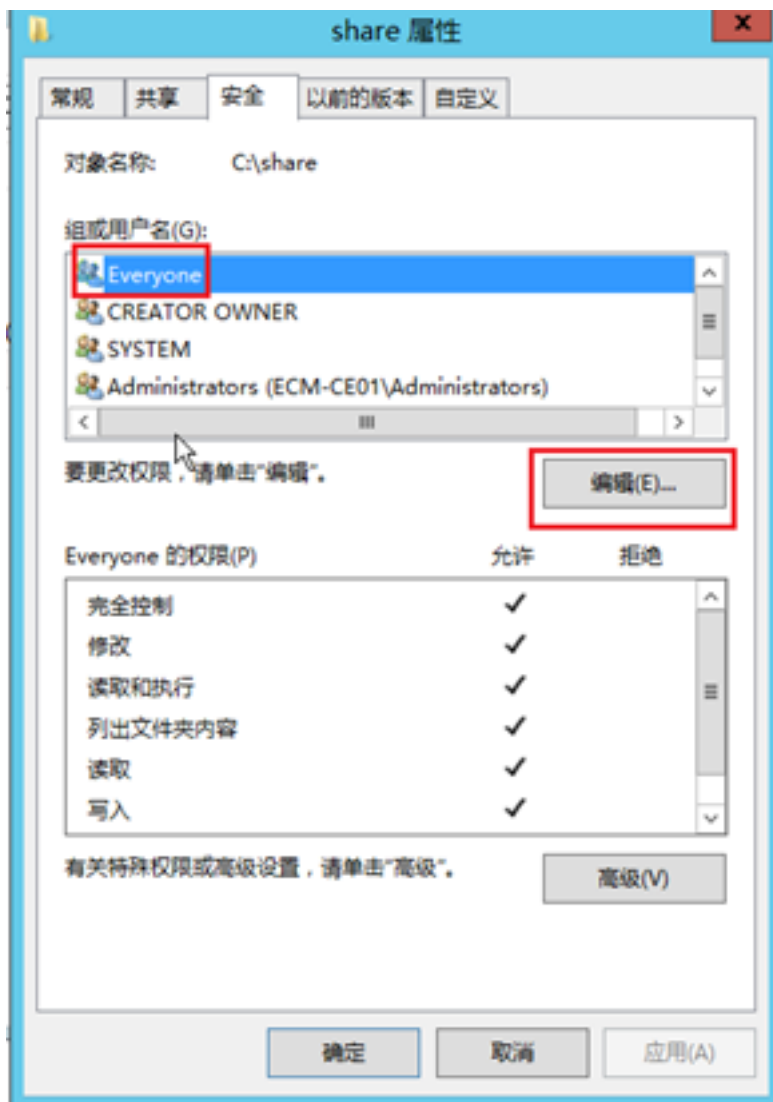
- c. 设置“用户名”和“密码”，此处用户名以“ftp”为例。

最佳实践



最佳实践

3. 设置共享文件权限。需要在FTP站点为共享给用户的文件夹设置访问及修改等权限。
 - a. 在服务器上创建一个供FTP使用的文件夹，选择文件夹，并单击右键选择“属性”。此处以“share”文件夹为例。
 - b. 在“安全”页签，选择“Everyone”，单击“编辑”。如果没有“Everyone”用户可以直接选择，需要先进行添加。
 - c. 选择“Everyone”，然后根据需要，选择“Everyone”的权限，并单击“确定”。此处以允许所有权限为例。



最佳实践

4. 配置FTP站点。

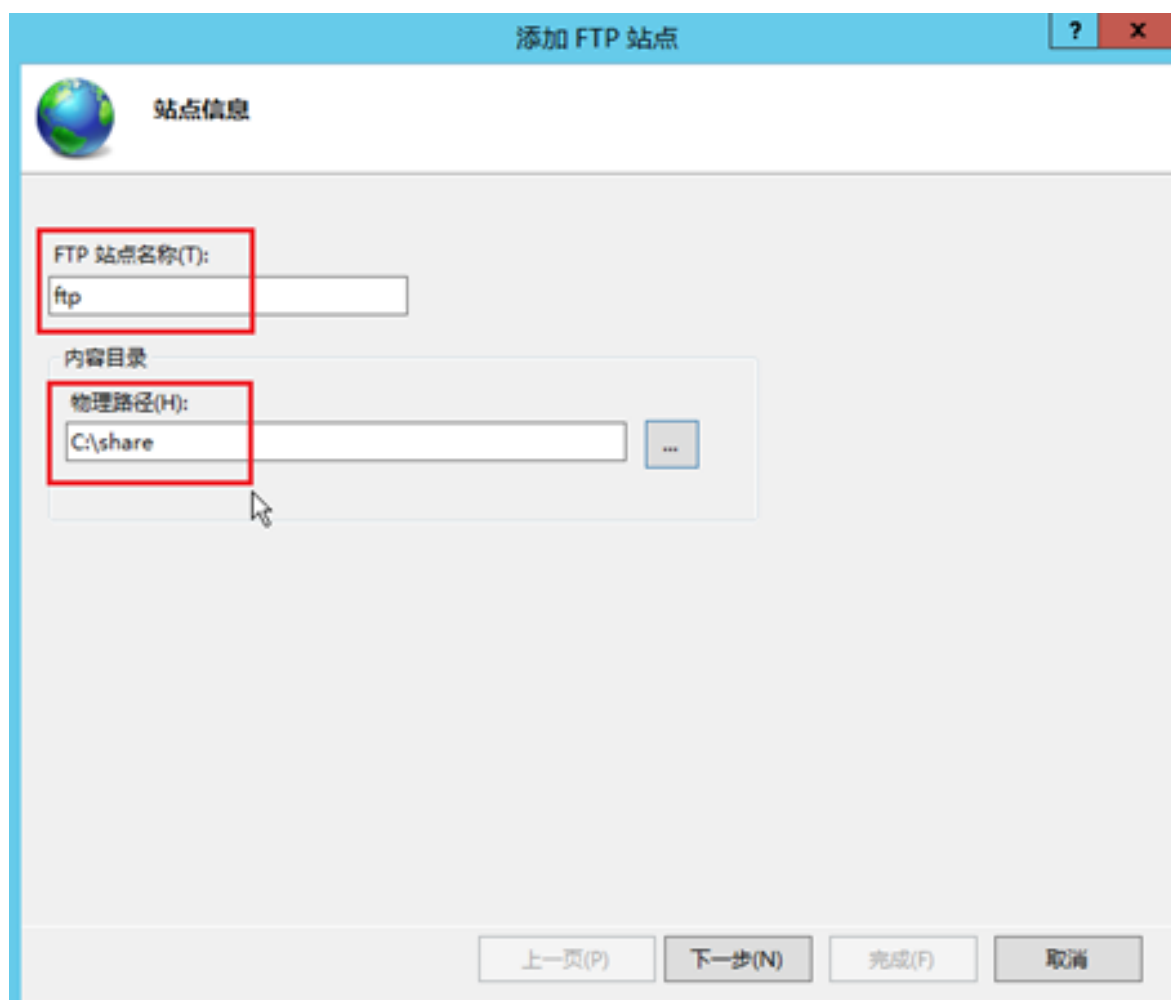
- a. 在“服务器管理器”中，选择“仪表板 > 工具 > Internet Information Services (IIS)管理器”。



- b. 选择“网站”并单击右键，然后选择“添加FTP站点”。



- c. 在弹出的窗口中，填写FTP站点名称及共享文件夹的物理路径，然后单击“下一步”。此处站点名称以“ftp”为例。



添加 FTP 站点

站点信息

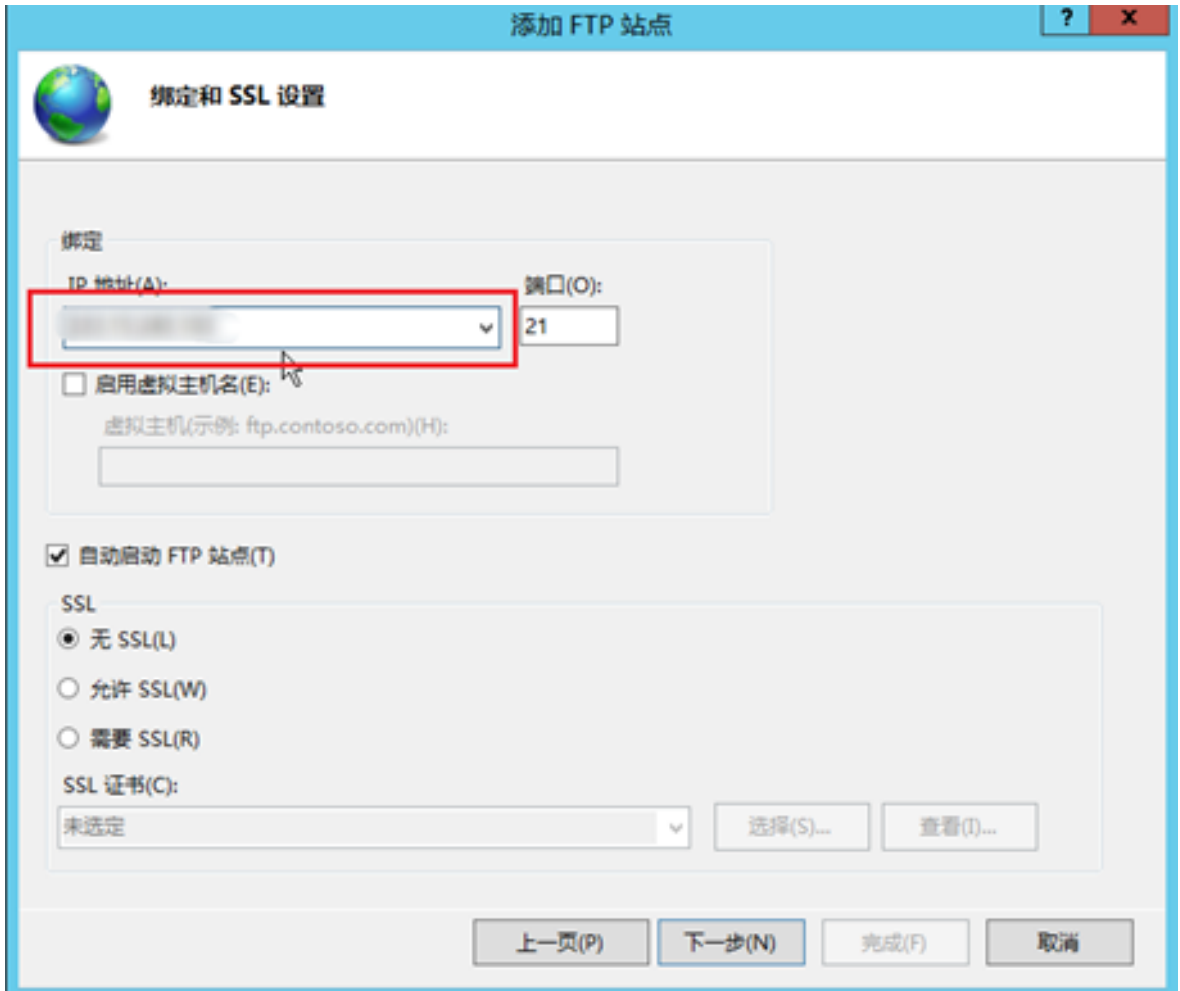
FTP 站点名称(T):
ftp

内容目录
物理路径(H):
C:\share

上一页(P) 下一步(N) 完成(F) 取消

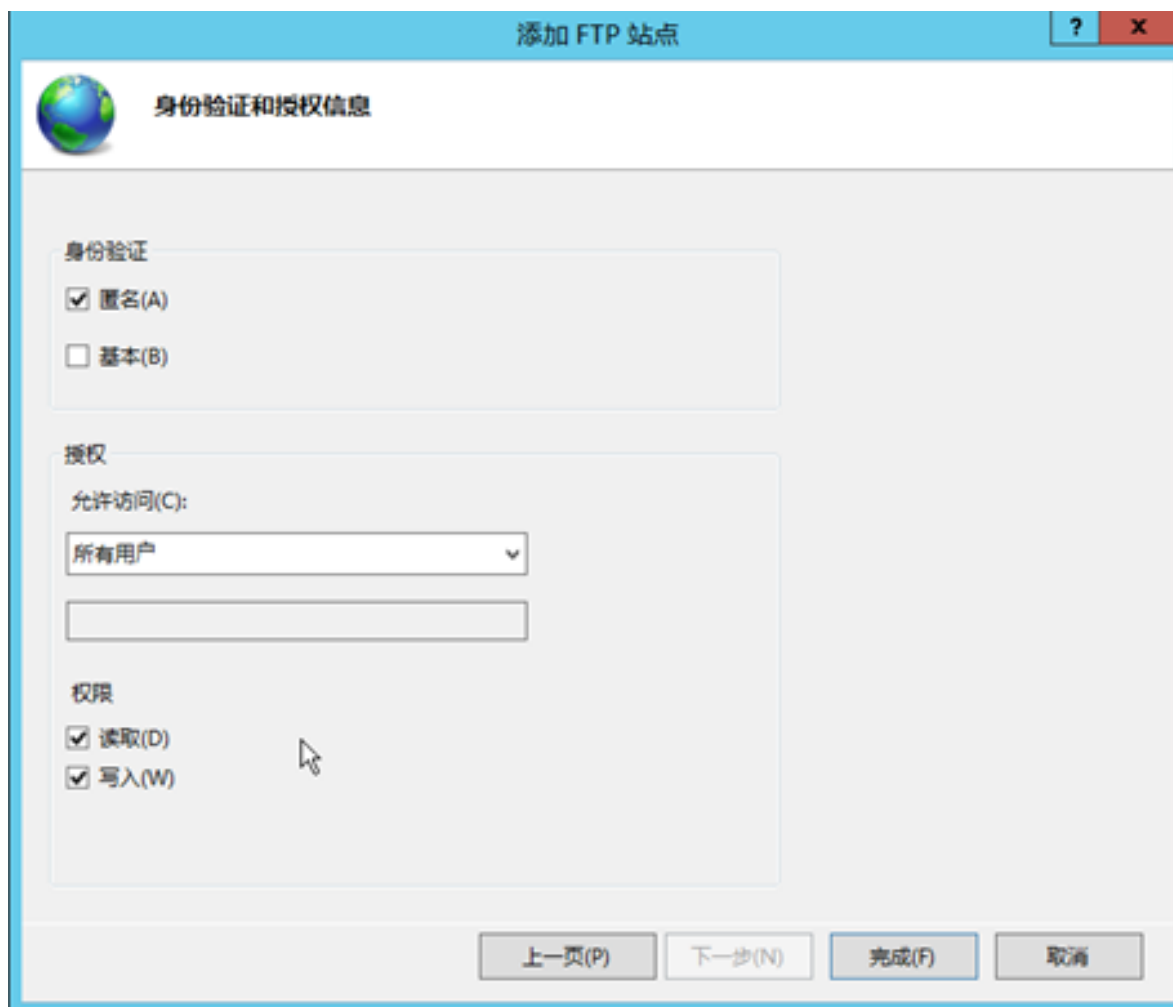
- d. 输入该弹性云主机的公网IP地址以及端口号，并设置SSL，单击“下一步”。

- 端口号默认为21，也可自行设置。
- SSL根据需要进行设置。
 - 无：不需要SSL加密。
 - 允许：允许FTP服务器与客户端的非SSL和SSL连接。
 - 需要：需要对FTP服务器和客户端之间的通信进行SSL加密。



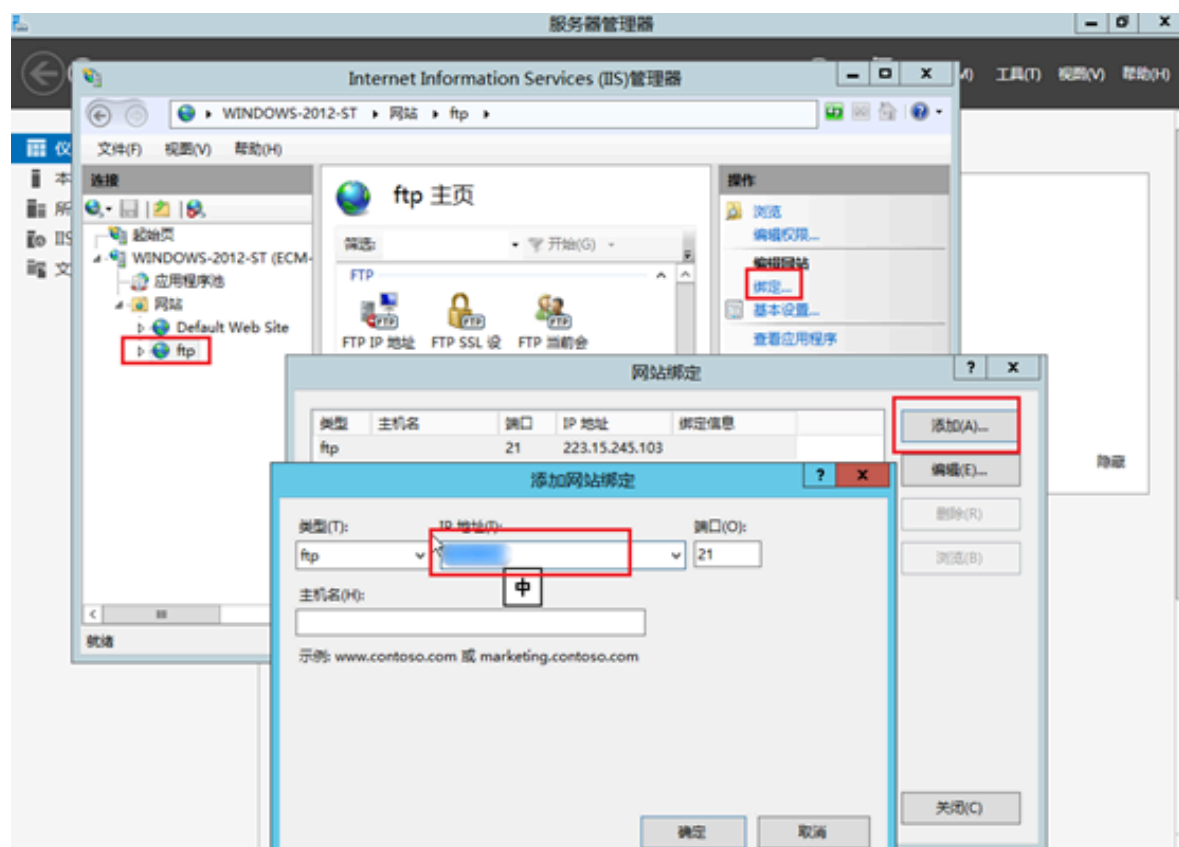
e. 设置身份认证和授权信息，并单击“完成”。

- 身份认证
 - 匿名：允许任何仅提供用户名“anonymous”或“ftp”的用户访问内容。
 - 基本：需要用户提供有效用户名和密码才能访问内容。但是基本身份验证通过网络传输密码时不加密，因此建议在确认客户端和FTP服务器之间的网络连接安全时使用此身份验证方法。
- 授权
 - 允许访问
 - 所有用户：所有用户均可访问相应内容。
 - 匿名用户：匿名用户可访问相应内容。
 - 指定角色或用户组：仅指定的角色或用户组的成员才能访问相应内容。如果选择此项，需要在下面输入框中输入指定的角色或用户组。
 - 指定用户：仅指定的用户才能访问相应内容。如果选择此项，需要在下面输入框中输入指定的用户。
 - 权限：选择经过授权的用户的“读取”和“写入”权限。



- f. 绑定弹性云主机的公网IP。选择“网站”，选中创建的FTP站点，单击“绑定”；在弹出的“网站绑定”窗口单击“添加”，然后在弹出的窗口中添加弹性云主机的私网IP地址，并单击“确定”。

最佳实践

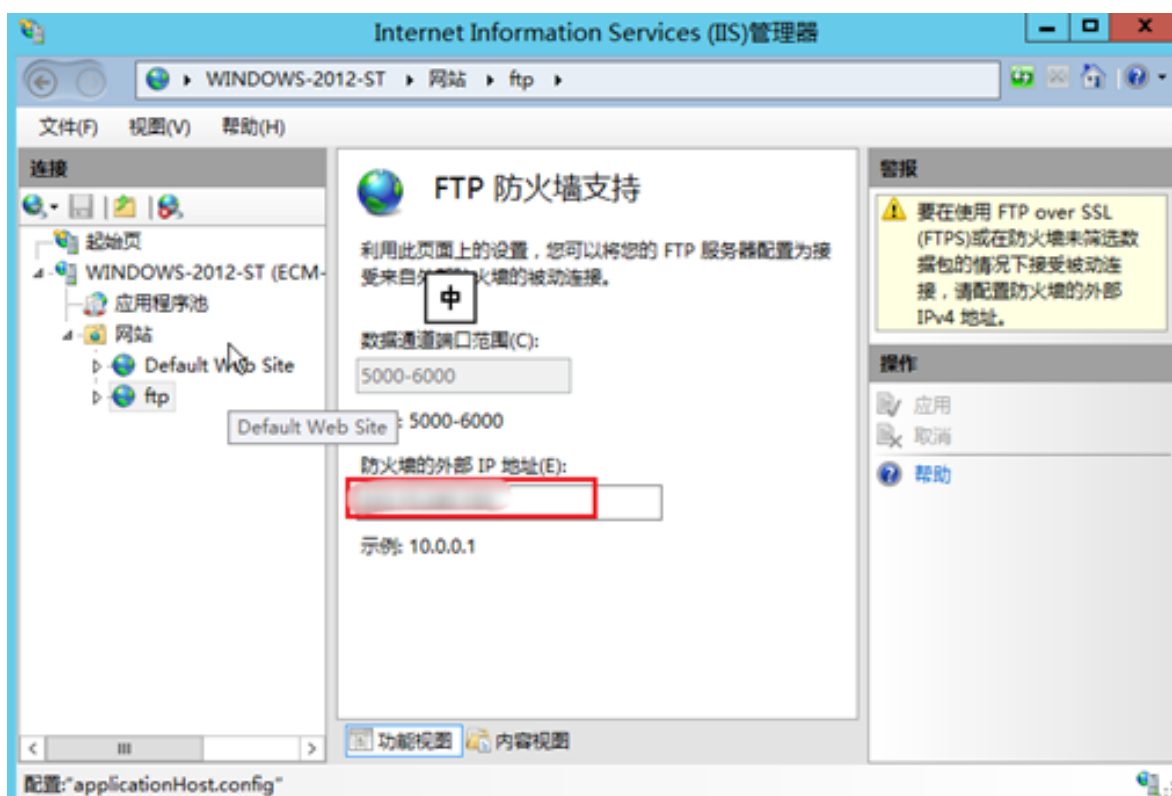


5. 配置FTP防火墙支持。

- 如果需要使用FTP服务器的被动模式，则需要配置FTP防火墙支持。
- 如果天翼云上的服务器需要通过公网IP地址访问天翼云上的实例搭建的FTP服务器时，需要配置FTP服务器的被动模式。
 - 双击“FTP防火墙支持”，打开FTP防火墙支持的配置界面。



- 配置相关参数，数据通道端口范围：指定用于被动连接的端口范围。可指定的有效端口范围为1025-65535，请根据实际需求进行设置，此处配置5000-6000，防火墙的外部IP地址：输入该弹性云主机的公网IP地址，并单击“应用”。



- 重启云主机使防火墙配置生效。

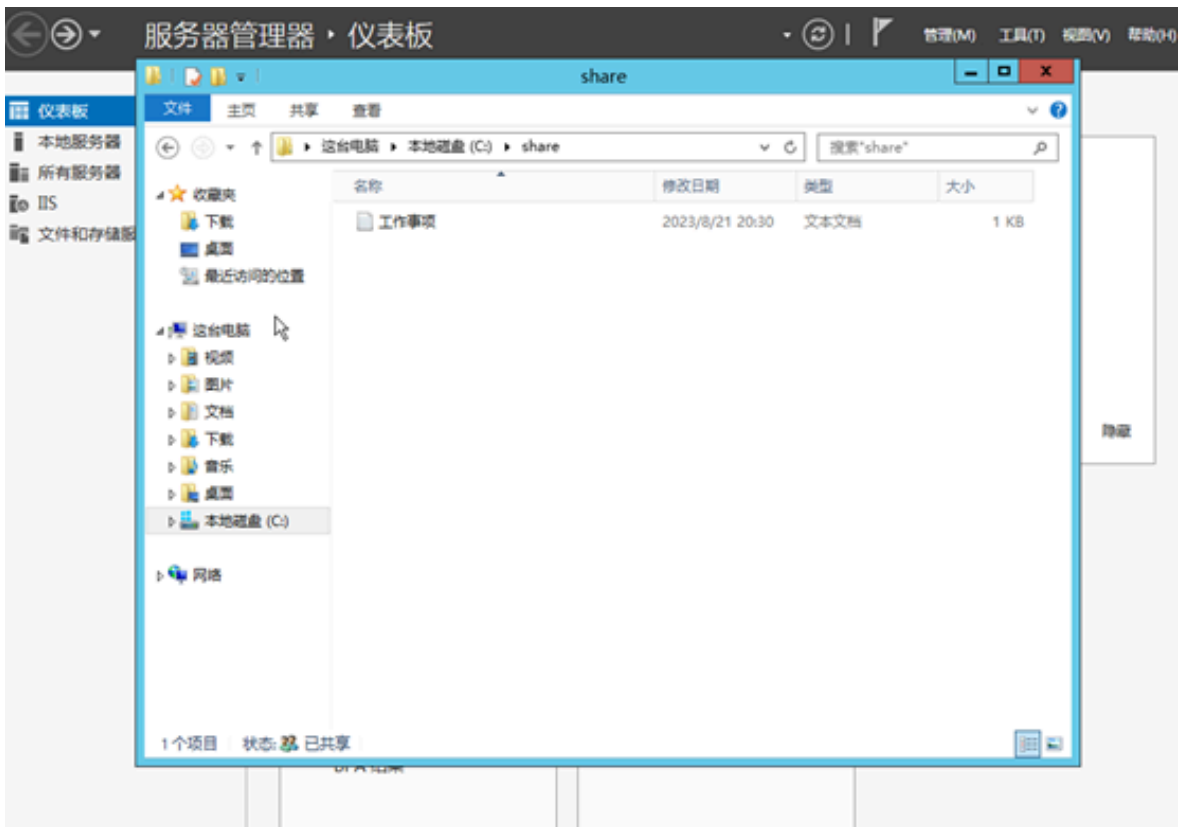
6. 配置安全组规则及防火墙策略。搭建好FTP站点后，需要在弹性云主机安全组的入方向添加一条放行FTP端口的规则，如果配置了“FTP防火墙支持”，需要在安全组中同时放行FTP站点使用的端口和FTP防火墙使用的数据通道端口。服务器防火墙默认放行TCP的21端口用于FTP服务。如果选用其他端口，需要在防火墙中添加一条放行此端口的入站规则。

FTP模式	方向	协议	端口	源地址
主动模式	入方向	TCP	20端口和21端口	0.0.0.0/0

最佳实践

FTP模式	方向	协议	端口	源地址
被动模式	入方向	TCP	21端口和1024~65535间的端口（根据上文配置，此处5000~6000）	0.0.0.0/0

7. 客户端进行连接测试。打开客户端的计算机，在路径栏输入“ftp://FTP服务器IP地址:FTP端口”（如果不填端口则默认访问21端口）。弹出输入用户名和密码的对话框表示配置成功，正确的输入用户名和密码后，即可对FTP文件进行相应权限的操作，share文件夹下创建工作事项.txt文件，在本地访问共享文件夹可以查看并下载。



搭建FTP

Windows云主机

添加IIS以及FTP服务角色

1. 登录弹性云主机。
2. 选择“开始 > 服务器管理器”。

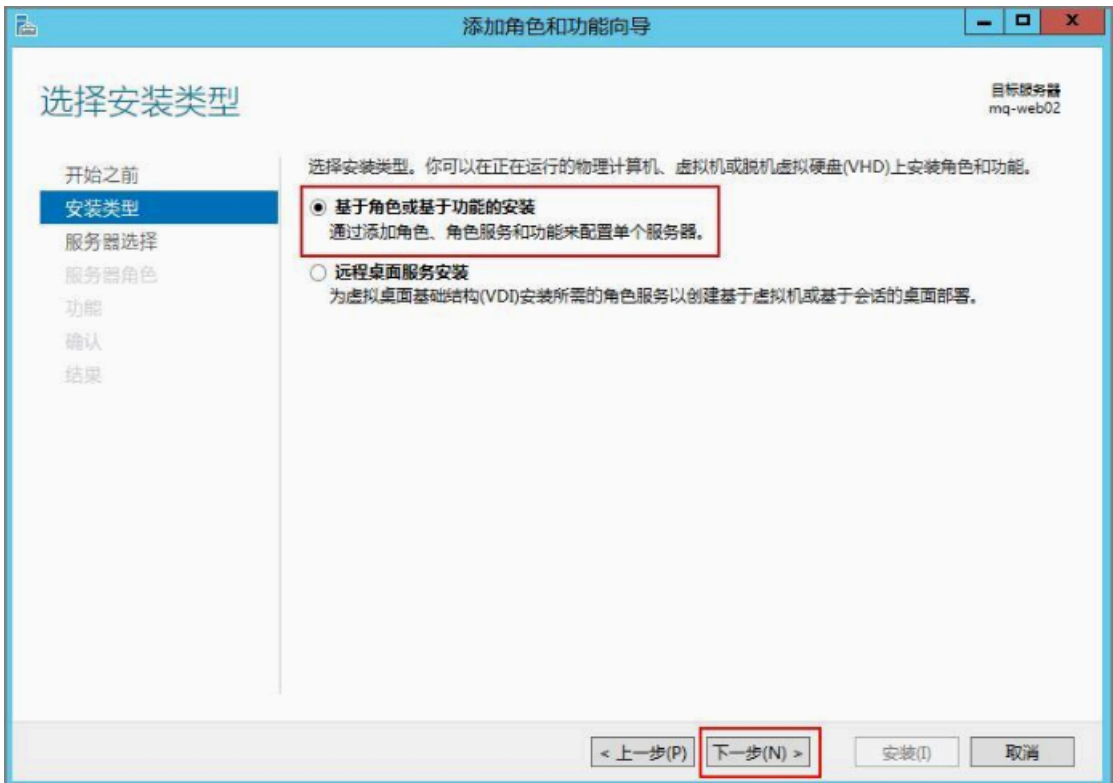
最佳实践

3. 单击“添加角色和功能”。



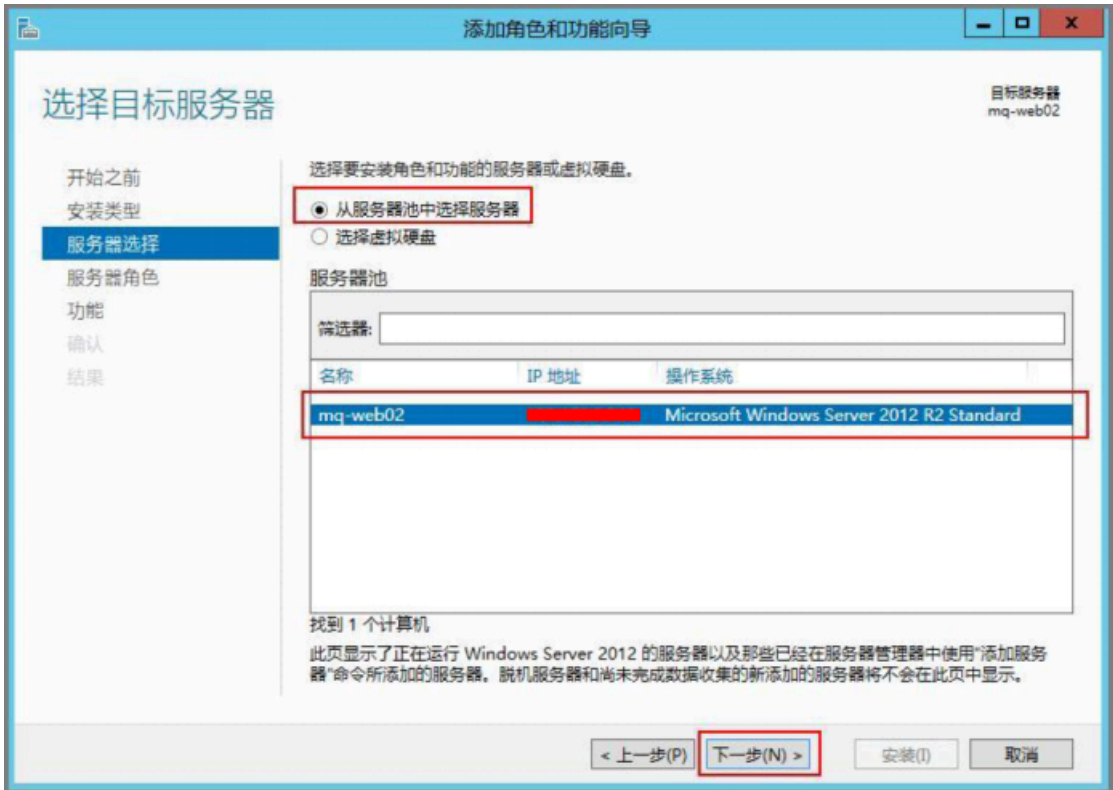
4. 在弹出的“开始之前”对话框中，单击“下一步”。

5. 选择“基于角色或基于功能的安装”，单击“下一步”。



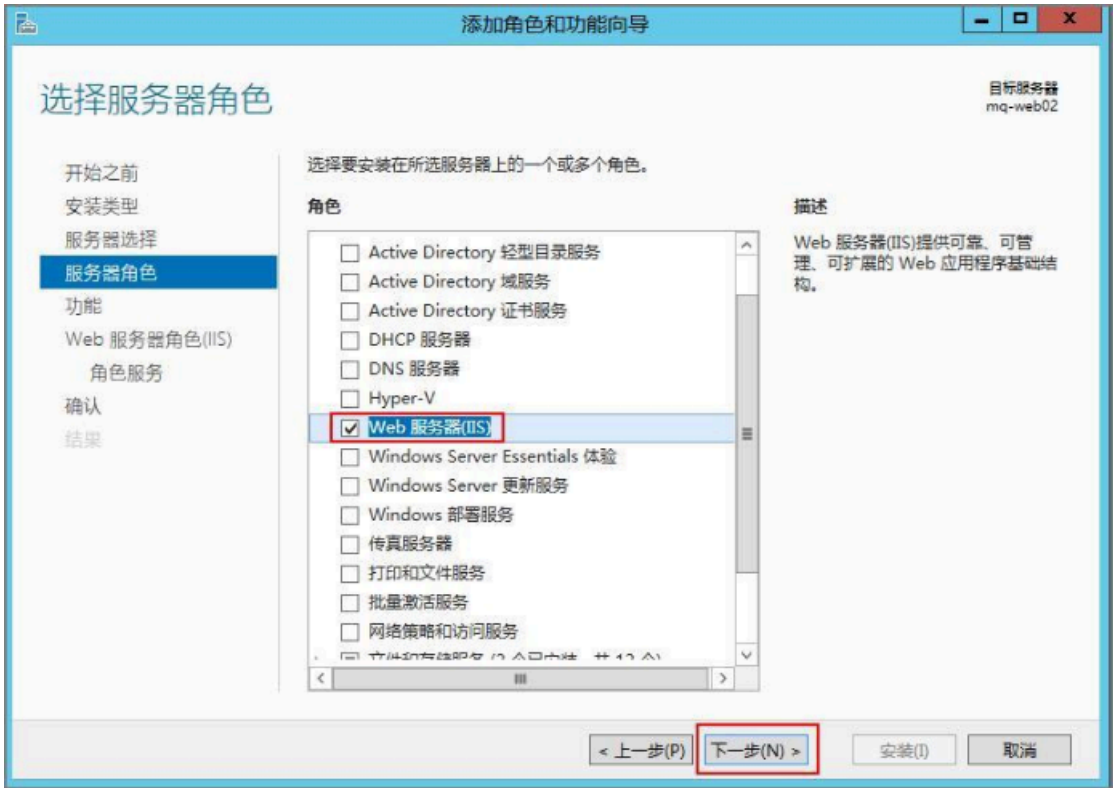
最佳实践

6. 选择需要部署FTP的服务器，单击“下一步”。



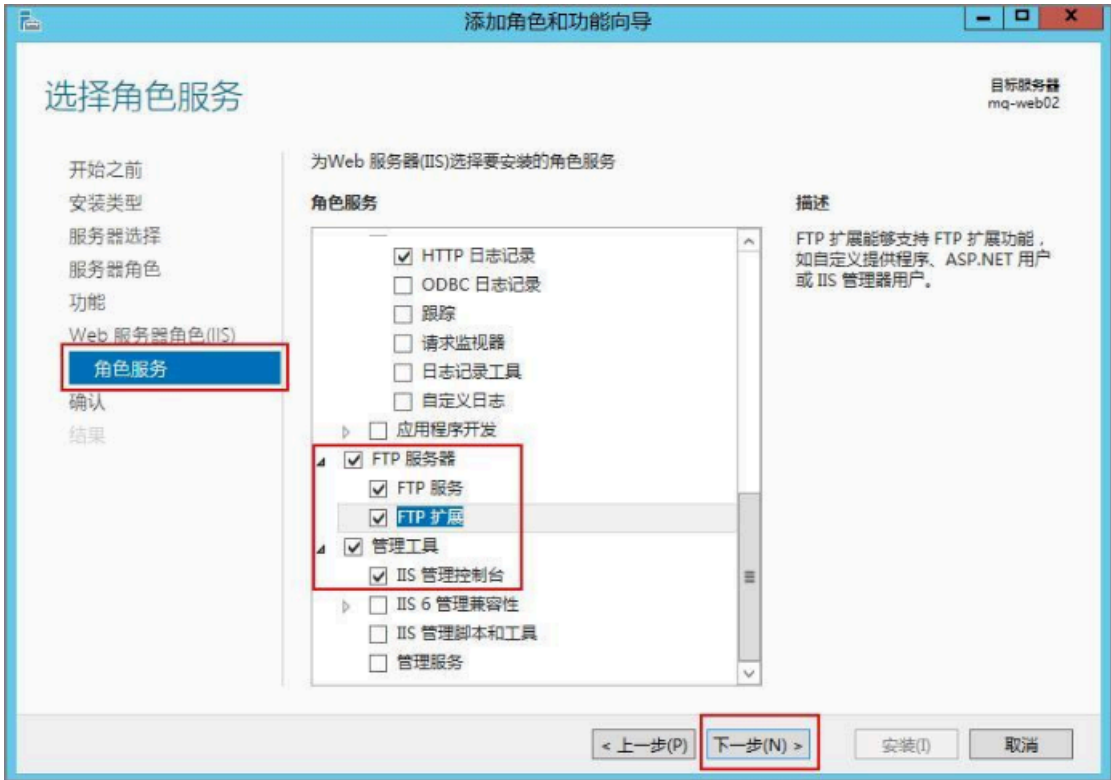
最佳实践

7. 选择“Web服务器（IIS）”，并在弹出的对话框中单击“添加功能”，然后单击“下一步”。



8. 连续单击“下一步”，到“角色服务”页面。

9. 选择“FTP服务器”以及“IIS管理控制台”，单击“下一步”。



10. 单击“安装”，开始部署服务角色。

11. 安装完成后，单击“关闭”。

创建FTP用户名及密码

创建Windows用户名和密码，用于FTP使用。如果您希望匿名用户可以访问，可以不创建FTP用户和密码。

最佳实践

1. 在“服务器管理器”中，选择“仪表板 > 工具 > 计算机管理”。



2. 选择“系统工具 > 本地用户和组 > 用户”，在右侧空白处右击，并选择“新用户”。

3. 设置“用户名”和“密码”，此处用户名以“ftpadmin”为例。

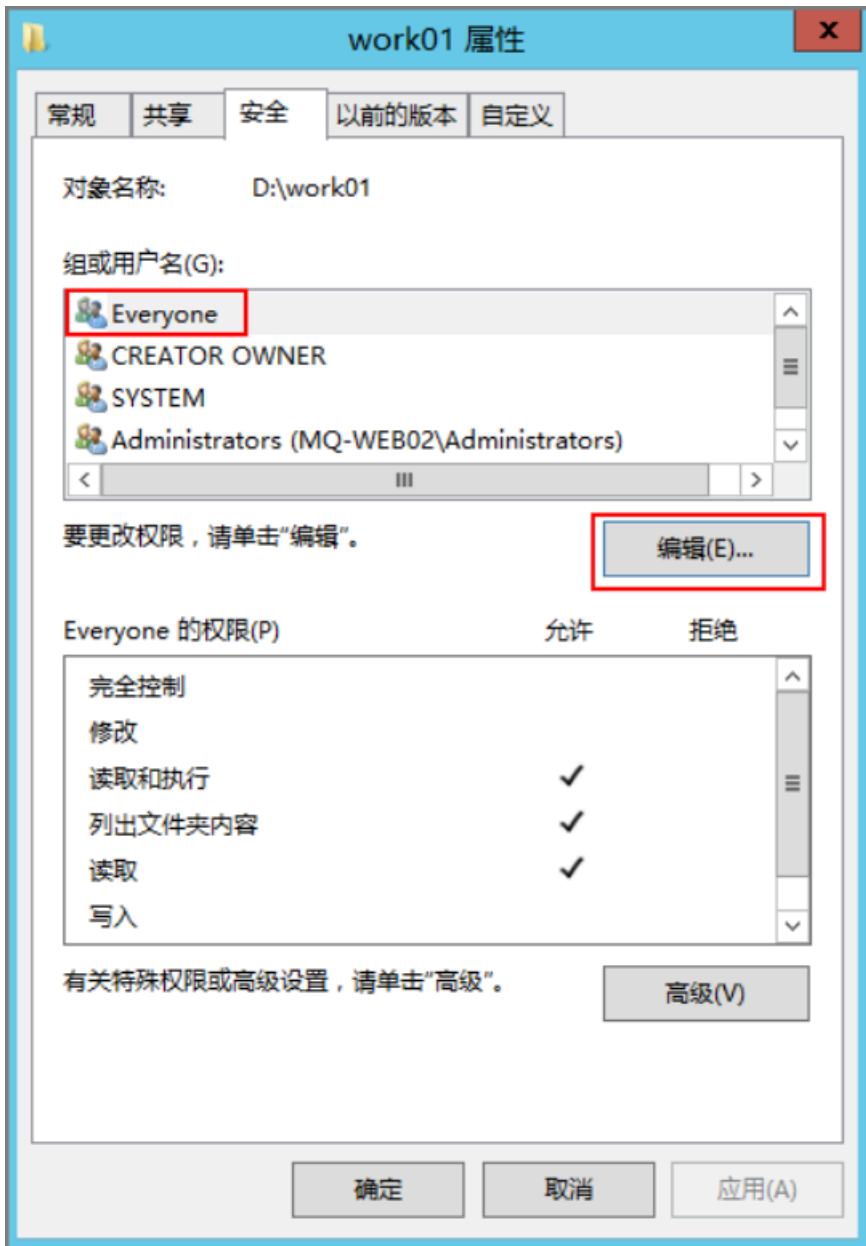
设置共享文件的权限

需要在FTP站点为共享给用户的文件夹设置访问及修改等权限。

1. 在服务器上创建一个供FTP使用的文件夹，选择文件夹，并单击右键选择“属性”。此处以“work01”文件夹为例。

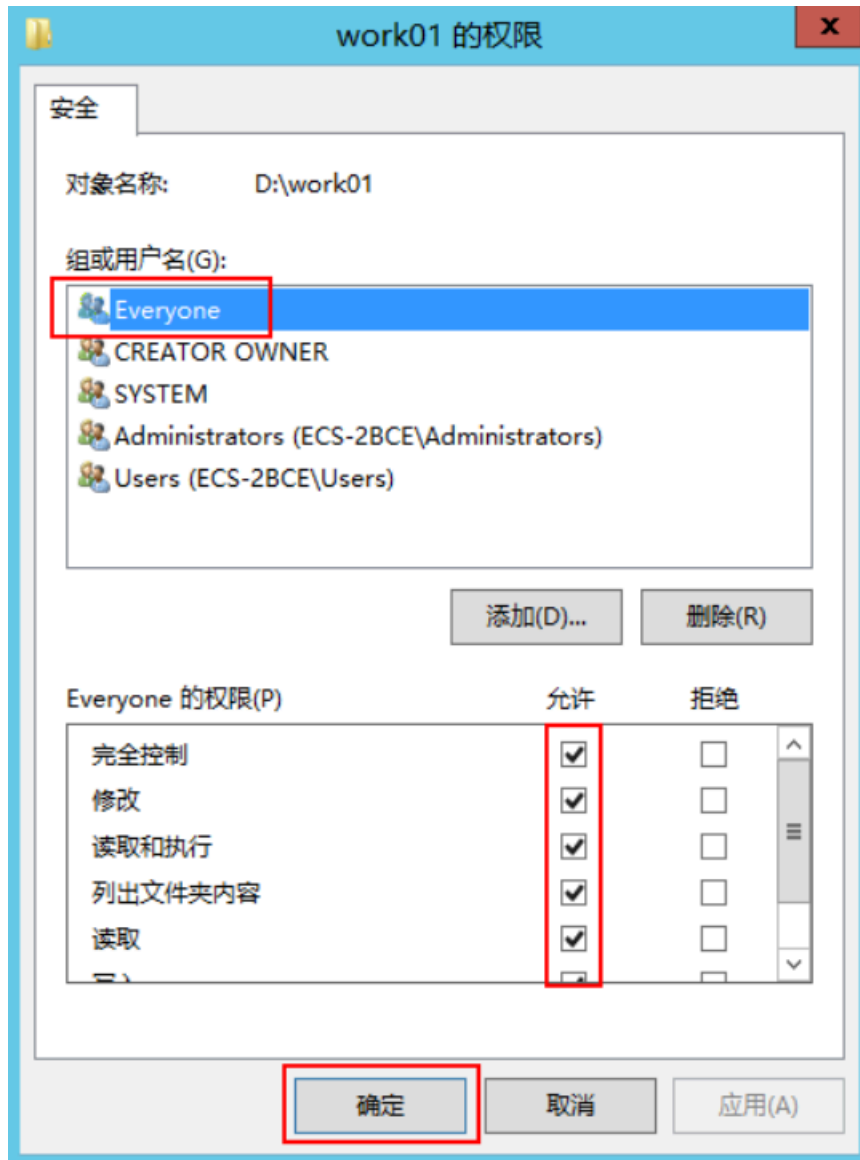
最佳实践

- 在“安全”页签，选择“Everyone”，单击“编辑”。如果没有“Everyone”用户可以直接选择，需要先进行添加。



最佳实践

3. 选择“Everyone”，然后根据需要，选择“Everyone”的权限，并单击“确定”。此处以允许所有权限为例。



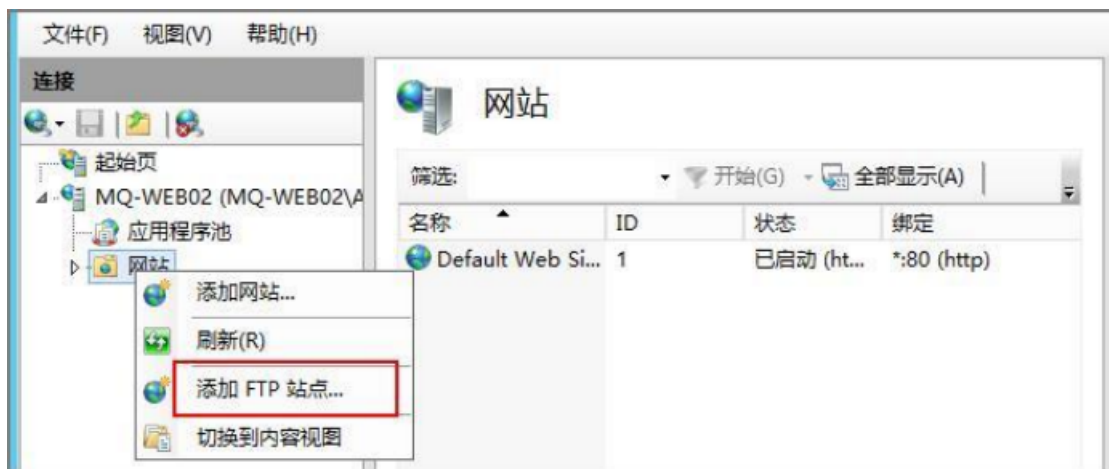
最佳实践

添加及设置FTP站点

1. 在“服务器管理器”中，选择“仪表板 > 工具 > Internet Information Services (IIS)管理器”。

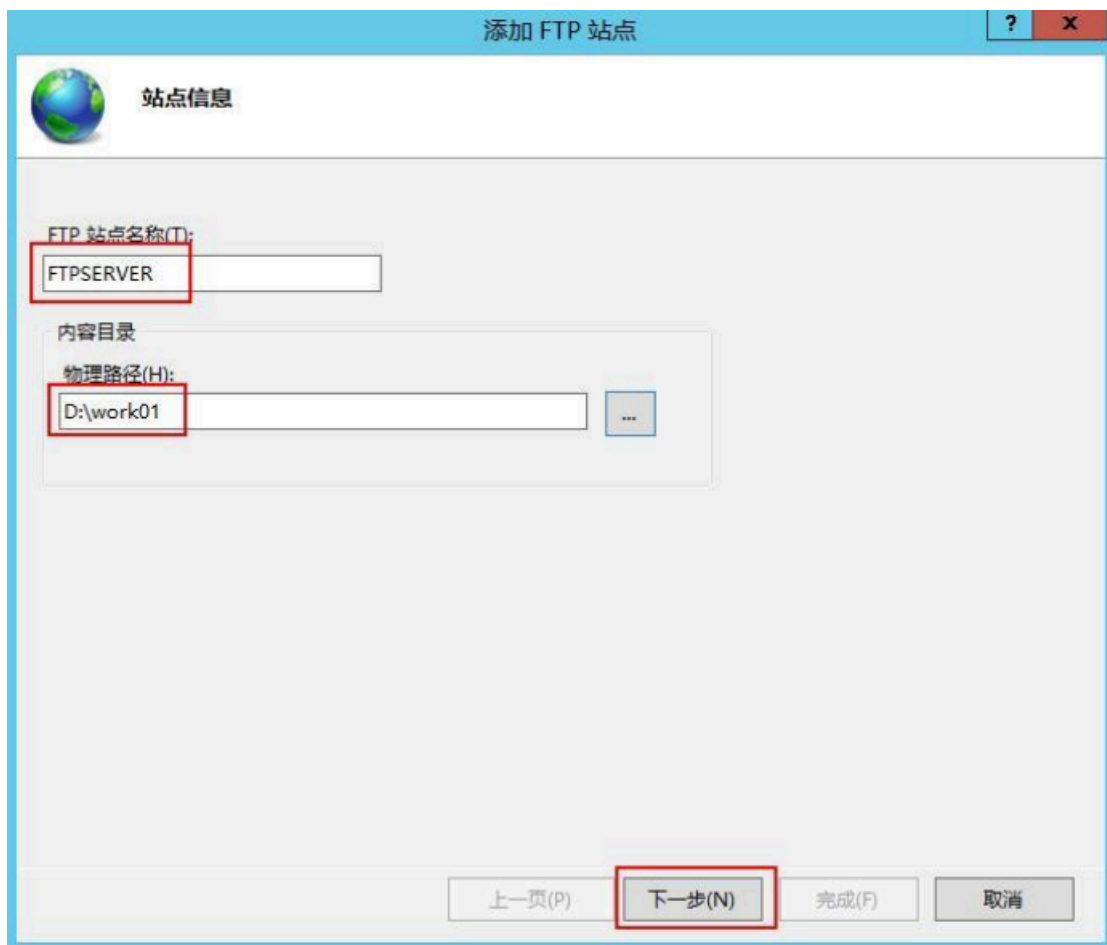


2. 选择“网站”并单击右键，然后选择“添加FTP站点”。



最佳实践

3. 在弹出的窗口中，填写FTP站点名称及共享文件夹的物理路径，然后单击“下一步”。此处站点名称以“FTPSERVER”为例。



4. 输入该弹性云主机的公网IP地址以及端口号，并设置SSL，单击“下一步”。端口号默认为21，也可自行设置。

最佳实践

5. SSL根据需要进行设置。

- 无：不需要SSL加密。
- 允许：允许FTP服务器与客户端的非SSL和SSL连接。
- 需要：需要对FTP服务器和客户端之间的通信进行SSL加密。

说明

当SSL选择“允许”和“需要”时，需要选择SSL证书。可以选择已有的SSL证书，也可以制作一个SSL证书，具体制作证书的方法请参见制作服务器证书。

The screenshot shows the '添加 FTP 站点' (Add FTP Site) wizard window. The title bar includes a help icon, a close icon, and the text '添加 FTP 站点'. The main window title is '绑定和 SSL 设置' (Binding and SSL Settings). The window contains the following elements:

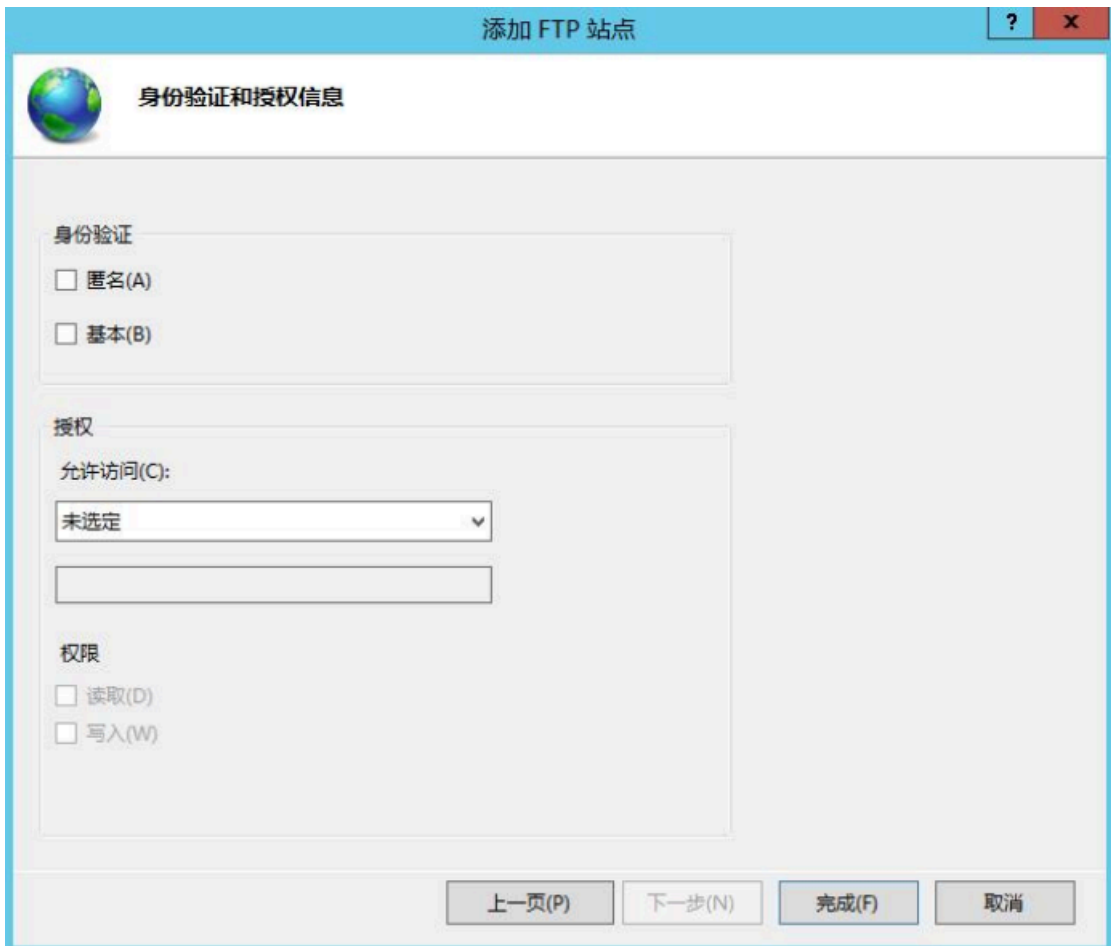
- 绑定 (Binding):**
 - IP 地址(A):** A dropdown menu with a red bar.
 - 端口(O):** A text box containing '21'.
 - 启用虚拟主机名(E):** A checkbox that is unchecked.
 - 虚拟主机(示例: ftp.contoso.com)(H):** A text box.
- 自动启动 FTP 站点(T)** (Automatically start FTP site): A checked checkbox.
- SSL (SSL):**
 - 无 SSL(L)** (No SSL): Unselected.
 - 允许 SSL(W)** (Allow SSL): Selected.
 - 需要 SSL(R)** (Require SSL): Unselected.
 - SSL 证书(C):** A dropdown menu with a red bar.
 - 选择(S)...** (Select...): A button.
 - 查看(I)...** (View...): A button.

At the bottom of the window, there are four buttons: '上一页(P)' (Previous page), '下一步(N)' (Next page), '完成(F)' (Finish), and '取消' (Cancel). The '下一步(N)' button is highlighted with a red rectangle.

最佳实践

6. 设置身份认证和授权信息，并单击“完成”。

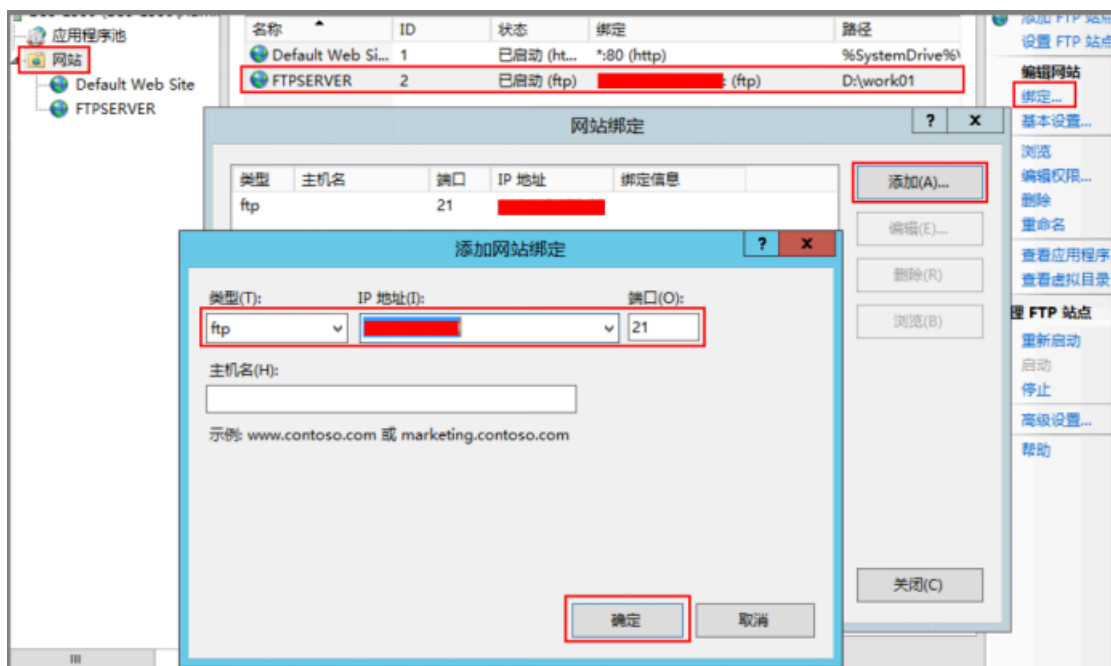
- 身份认证
 - 匿名：允许任何仅提供用户名“anonymous”或“ftp”的用户访问内容。
 - 基本：需要用户提供有效用户名和密码才能访问内容。但是基本身份验证通过网络传输密码时不加密，因此建议在确认客户端和FTP服务器之间的网络连接安全时使用此身份验证方法。
- 授权
 - 允许访问
 - 权限：选择经过授权的用户的“读取”和“写入”权限。
 - 所有用户：所有用户均可访问相应内容。
 - 匿名用户：匿名用户可访问相应内容。
 - 指定角色或用户组：仅指定的角色或用户组的成员才能访问相应内容。如果选择此项，需要在下面输入框中输入指定的角色或用户组。
 - 指定用户：仅指定的用户才能访问相应内容。如果选择此项，需要在下面输入框中输入指定的用户。



The screenshot shows a Windows-style dialog box titled "添加 FTP 站点" (Add FTP Site). The active tab is "身份验证和授权信息" (Authentication and Authorization Information). The dialog is divided into three sections: "身份验证" (Authentication), "授权" (Authorization), and "权限" (Permissions). In the "身份验证" section, the "匿名(A)" (Anonymous) radio button is selected. In the "授权" section, the "允许访问(C):" (Allow access to:) dropdown menu is set to "未选定" (None selected). Below the dropdown is an empty text input field. In the "权限" section, the "读取(D)" (Read) and "写入(W)" (Write) radio buttons are unselected. At the bottom of the dialog, there are four buttons: "上一页(P)" (Previous page), "下一步(N)" (Next page), "完成(F)" (Finish), and "取消" (Cancel).

最佳实践

7. 绑定弹性云主机的私网IP。选择“网站”，选中创建的FTP站点，单击“绑定”；在弹出的“网站绑定”窗口单击“添加”，然后在弹出的窗口中添加弹性云主机的私网IP地址，并单击“确定”。



(可选) 配置FTP防火墙支持

如果需要使用FTP服务器的被动模式，则需要配置FTP防火墙支持。

如果天翼云上的服务器需要通过公网IP地址访问天翼云上的实例搭建的FTP服务器时，需要配置FTP服务器的被动模式。

最佳实践

1. 双击“FTP防火墙支持”，打开FTP防火墙支持的配置界面。

FTP 防火墙支持

利用此页面上的设置，您可以将您的 FTP 服务器配置为接受来自外部防火墙的被动连接。

数据通道端口范围(C):

示例: 5000-6000

防火墙的外部 IP 地址(E):

示例: 10.0.0.1

警报

⚠ 要在使用 FTP over SSL (FTPS)或在防火墙未筛选数据包的情况下接受被动连接，请配置防火墙的外部 IPv4 地址。

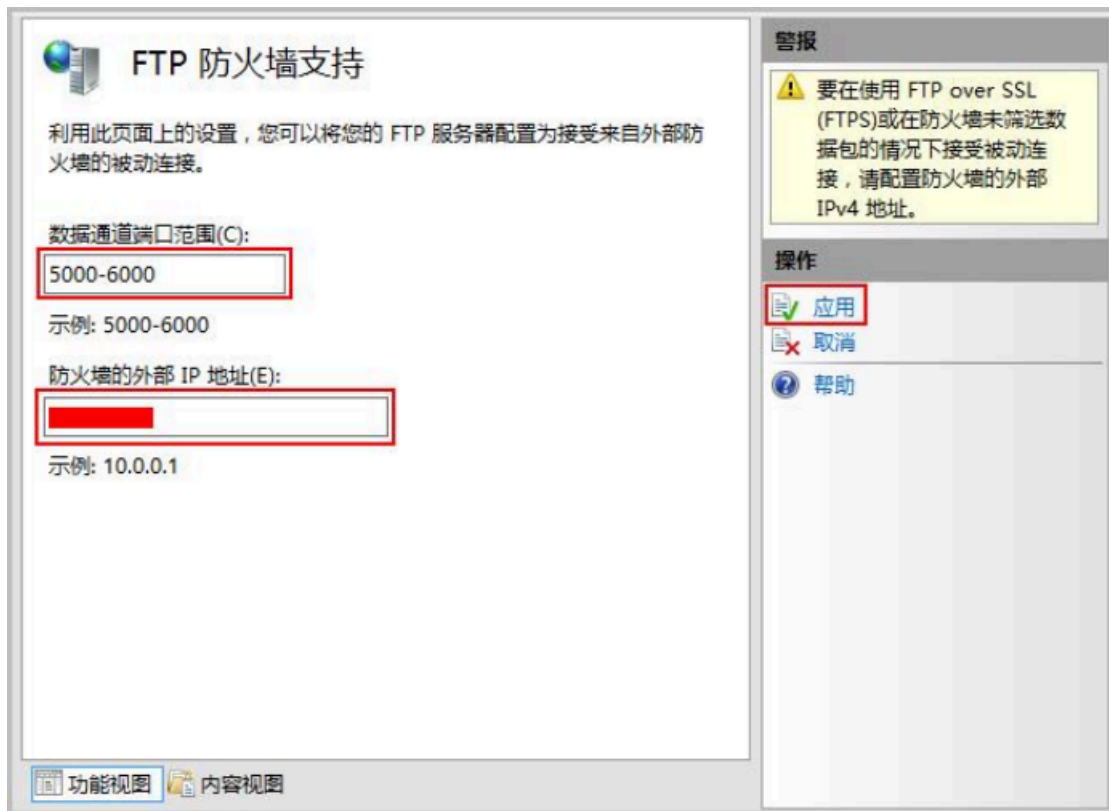
操作

功能视图 内容视图

2. 配置相关参数，并单击“应用”。

数据通道端口范围：指定用于被动连接的端口范围。可指定的有效端口范围为1025-65535。请根据实际需求进行设置。

防火墙的外部IP地址：输入该弹性云主机的公网IP地址。



3. 重启云主机使防火墙配置生效。

设置安全组及防火墙

搭建好FTP站点后，需要在弹性云主机安全组的入方向添加一条放行FTP端口的规则，具体步骤参见[虚拟私有云-安全组-添加安全组规则](#)。

如果配置了“FTP防火墙支持”，需要在安全组中同时放行FTP站点使用的端口和FTP防火墙使用的数据通道端口。

服务器防火墙默认放行TCP的21端口用于FTP服务。如果选用其他端口，需要在防火墙中添加一条放行此端口的入站规则。

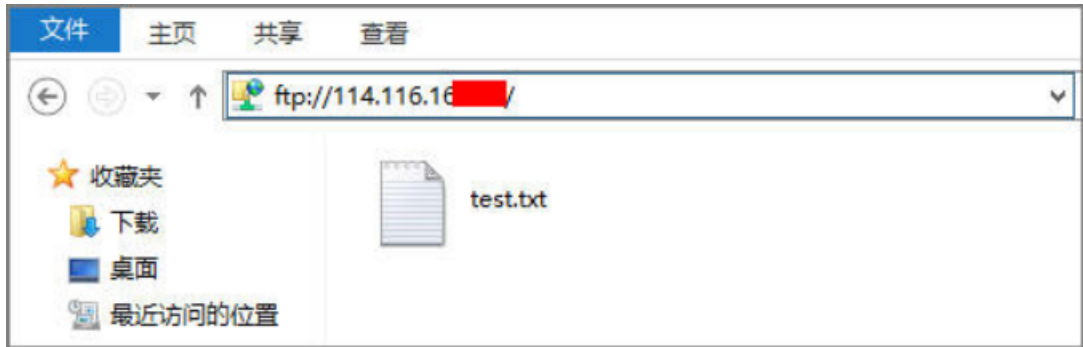
客户端测试

打开客户端的计算机，在路径栏输入“ftp://FTP服务器IP地址:FTP端口”（如果不填端口则默认访问21端口）。弹出输入用户名和密码的对话框表示配置成功，正确的输入用户名和密码后，即可对FTP文件进行相应权限的操作。

最佳实践

说明

如果没有配置“FTP防火墙支持”，客户端使用此方法访问FTP站点时，需要对IE浏览器进行设置，才能打开FTP的文件夹。打开IE浏览器，选择“设置 > Internet 选项 > 高级”。勾选“启用FTP文件夹视图”，取消勾选“使用被动FTP”。



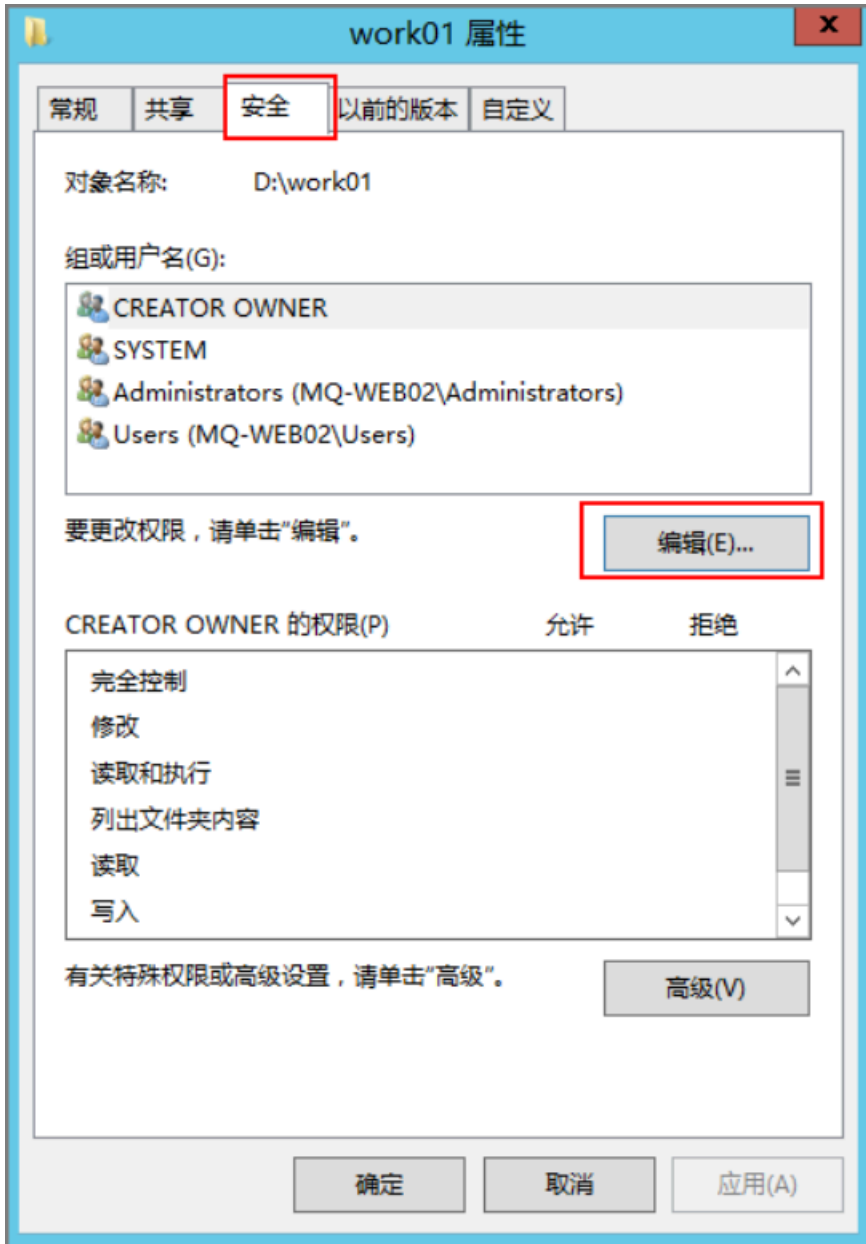
FAQ

使用Windows系统搭建FTP站点的更多信息请参见微软官方文档。

如果设置文件夹的属性时，没有“Everyone”用户可直接选择，可按照如下步骤添加“Everyone”用户。

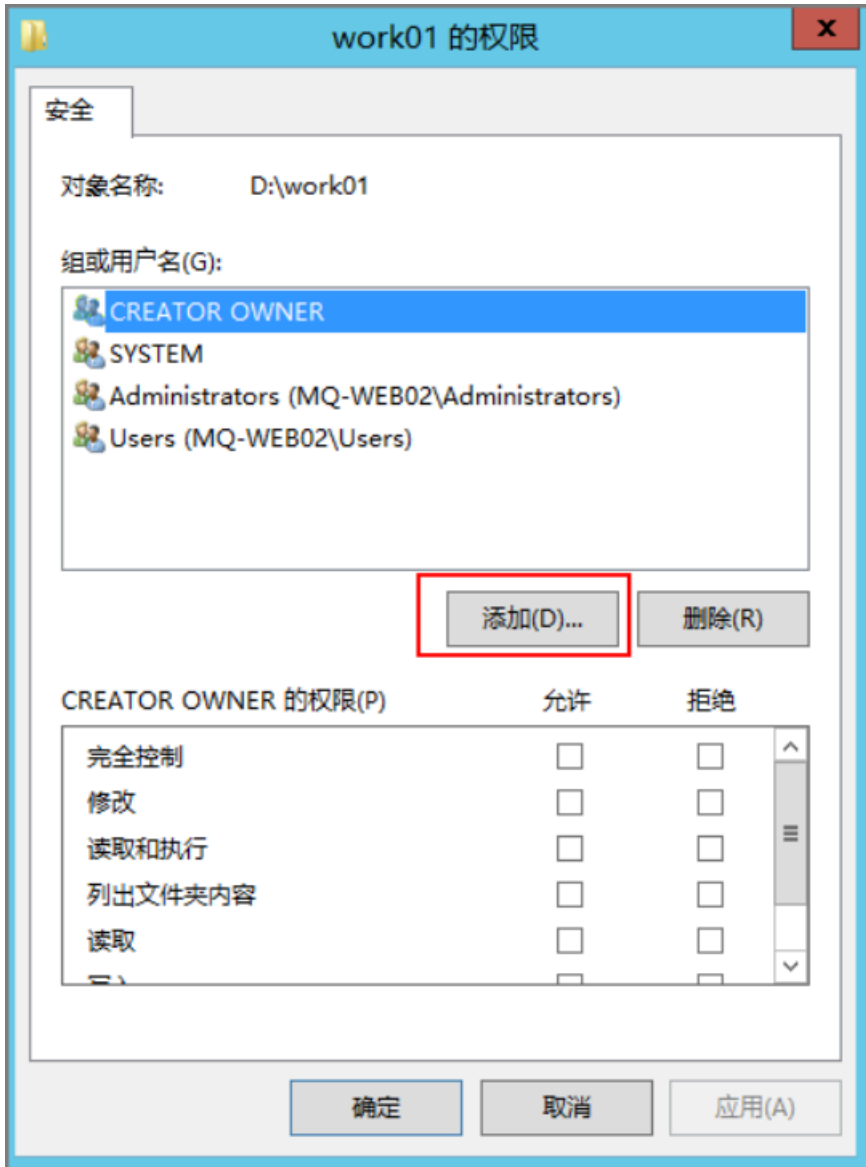
最佳实践

1. 在“安全”页签，单击“编辑”。



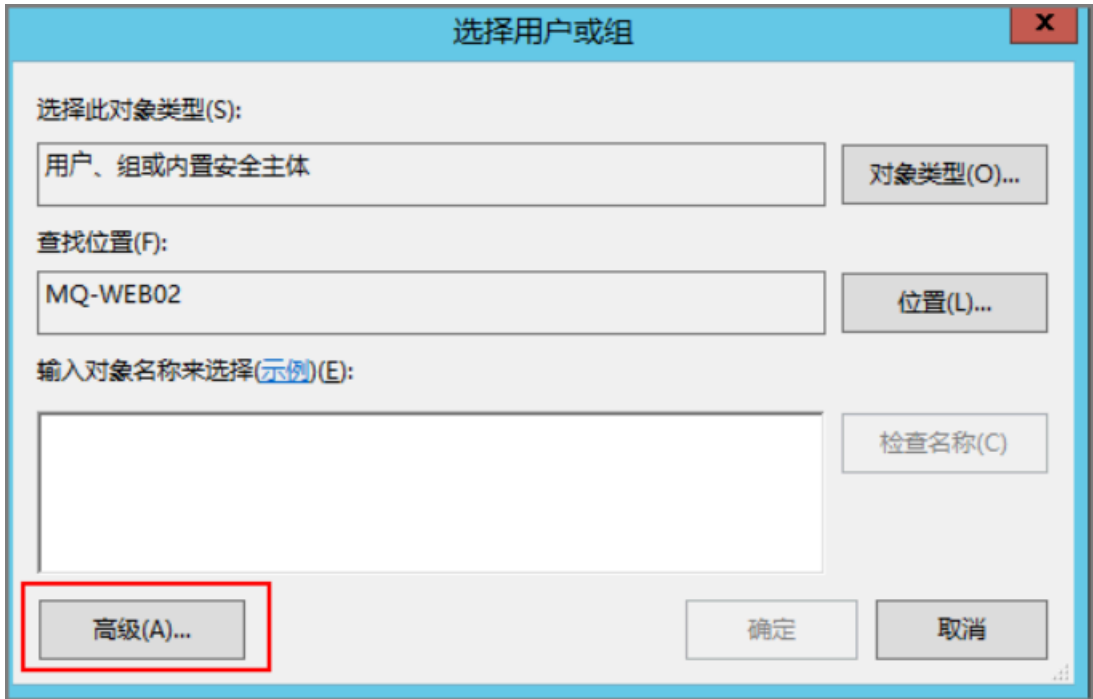
最佳实践

2. 在新弹出的窗口中，单击“添加”。



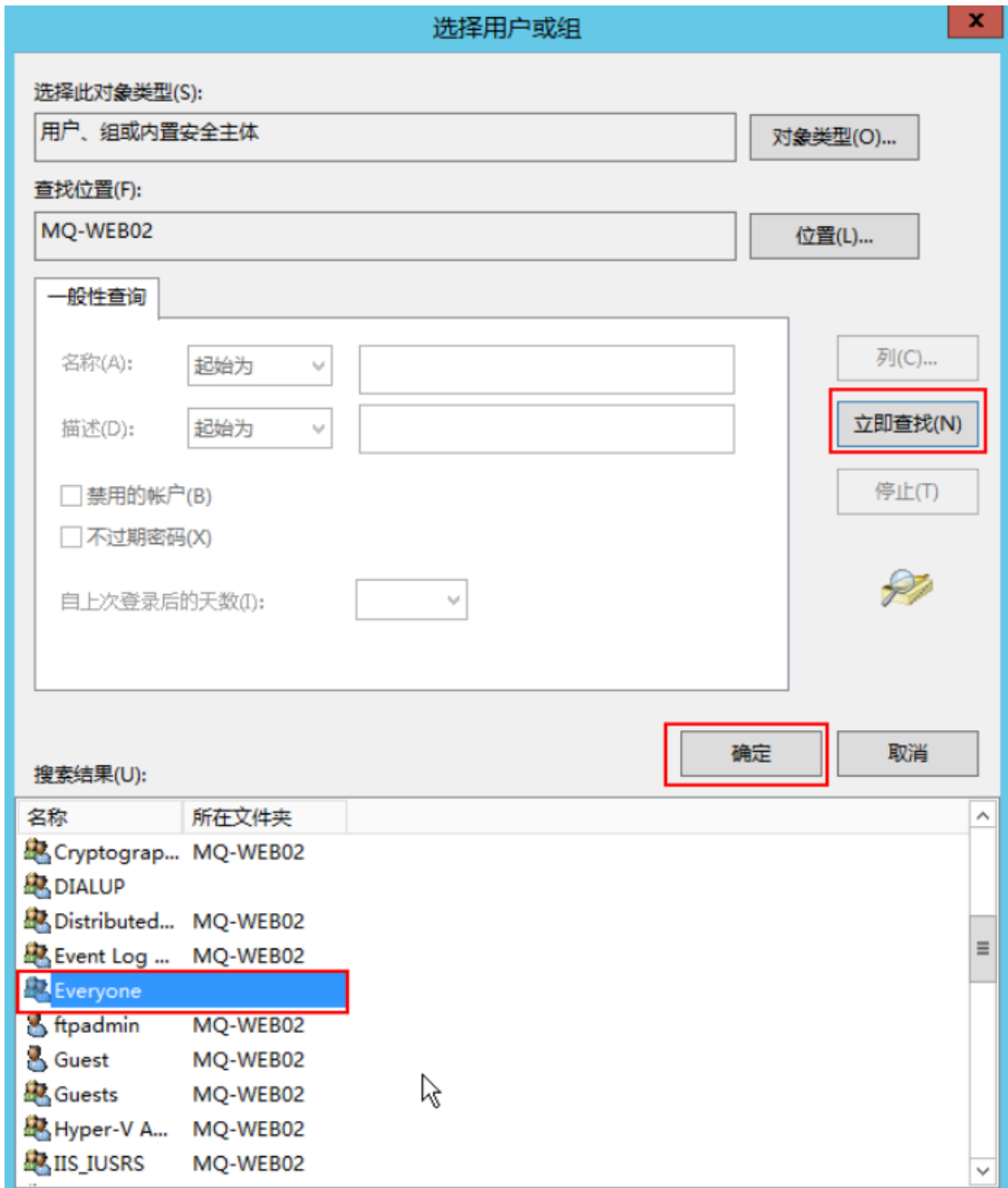
最佳实践

3. 在新弹出的窗口中，单击“高级”。



最佳实践

4. 在新弹出的窗口中，单击“立即查找(N)”，然后在搜索结果中选择“Everyone”并单击“确定”。



5. 单击“确定”，返回权限窗口。
6. 单击“确定”，完成添加。

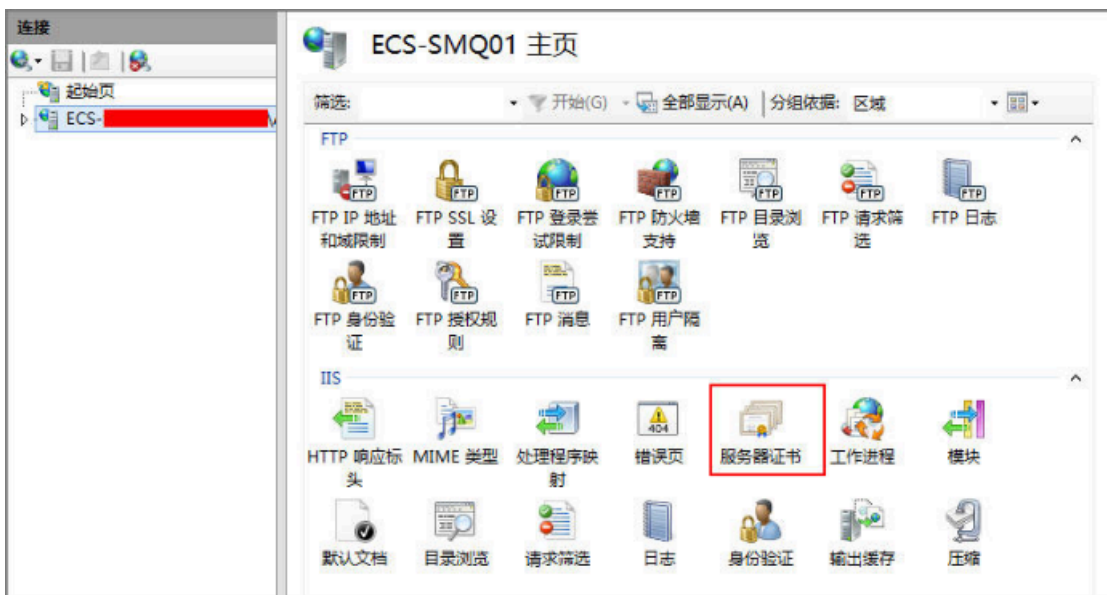
最佳实践

7. 制作服务器证书。

- a. 在“服务器管理器”中，选择“仪表板 > 工具 > Internet Information Services (IIS)管理器”。



- b. 在左侧列表单击服务器，然后在服务器主页“IIS”区域，双击“服务器证书”，进入“服务器证书”页面。



- c. 单击“创建自签名证书”。
- d. 输入证书的名称，并选择证书存储类型，单击“确定”。
- e. 创建成功后会在服务器证书界面显示已经创建的证书。

Linux云主机

安装vsftpd

1. 登录弹性云主机。

2. 执行以下命令安装vsftpd。

```
yum install -y vsftpd
```

回显如下类似信息时，表示软件安装成功。

```
Dependencies Resolved
-----
Package      arch          Version      Repository    Size
-----
Installing:
vsftpd       x86_64        3.0.2-22.e17 base          169 k
-----
Transaction Summary
-----
Install 1 Package

Total download size: 169 k
Installed size: 248 k
Downloading packages:
vsftpd-3.0.2-22.e17.x86_64.rpm                | 169 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : vsftpd-3.0.2-22.e17.x86_64                1/1
  Verifying  : vsftpd-3.0.2-22.e17.x86_64                1/1
Installed:
vsftpd.x86_64 0:3.0.2-22.e17
```

3. 执行以下命令设置FTP服务开机自启动。

```
systemctl enable vsftpd.service
```

4. 执行以下命令启动FTP服务。

```
systemctl start vsftpd.service
```

5. 执行以下命令查看FTP服务端口。

```
netstat -antup | grep ftp
```

回显如下类似信息。

```
tcp6      0      0 :::21                :::*                LISTEN      11836/vsftpd
```

配置vsftpd

vsftpd安装后默认开启了匿名FTP的功能，使用匿名FTP，用户无需输入用户名密码即可登录FTP服务器，但没有权限修改或上传文件。用户如果试图使用Linux操作系统中的账号登录服务器，将会被vsftpd拒绝，但可以在vsftpd里配置用户账号和密码登录。以下操作以在vsftpd里配置用户账号和密码登录FTP服务器为例。

1. 执行以下命令创建“ftpadmin”用户。此处“ftpadmin”用户为示例，可根据实际情况创建相应的用户。

```
useradd ftpadmin
```

2. 执行以下命令并按照提示设置“ftpadmin”用户密码。

```
passwd ftpadmin
```

3. 执行以下命令创建供FTP使用的文件目录，此处以“/var/ftp/work01”为例。

```
mkdir /var/ftp/work01
```

4. 执行以下命令将创建的文件目录所有者改为用于登录FTP的本地用户。

```
chown -R ftpadmin:ftpadmin /var/ftp/work01
```

5. 修改“vsftpd.conf”配置文件。

- a. 执行以下命令打开配置文件“vsftpd.conf”。

```
vi /etc/vsftpd/vsftpd.conf
```

- b. 按i键进入编辑模式。

修改打开的“vsftpd.conf”文件。可根据实际需求选择将FTP配置为主动模式或者被动模式。如果天翼云上的服务器需要通过公网IP地址访问天翼云上的实例搭建的FTP服务器时，需要将FTP服务器配置为被动模式。

主动模式需要配置的参数如下：

#设置以下参数，不允许匿名登录FTP服务器，允许本地用户登录FTP服务器，并指定FTP本地用户使用的文件目录。

```
anonymous_enable=NO #不允许匿名登录FTP服务器
```

```
local_enable=YES #允许本地用户登录FTP服务器
```

```
local_root=/var/ftp/work01 #FTP本地用户使用的文件目录
```

#设置以下参数，限制用户只能访问自身的主目录。

```
chroot_local_user=YES #所有用户都被限制在其主目录
```

```
chroot_list_enable=YES #启用例外用户名单
```

```
chroot_list_file=/etc/vsftpd/chroot_list #例外用户名单
```

```
allow_writeable_chroot=YES
```

被动模式除了需要配置主动模式所需的所有参数外，还需要配置的参数如下：

#设置以下参数，配置FTP支持被动模式。并指定FTP服务器的公网IP地址，以及可供访问的端口范围，端口范围请根据实际环境进行设置。

```
listen=YES
```

```
listen_ipv6=NO
```

```
pasv_address=xx.xx.xx.xx #FTP服务器的公网IP地址
```

```
pasv_min_port=3000 #被动模式下的最小端口
```

```
pasv_max_port=3100 #被动模式下的最大端口
```

- c. 按Esc键退出编辑模式，并输入:wq保存后退出

6. 在“/etc/vsftpd/”目录下创建“chroot_list”文件。

```
touch chroot_list
```

“chroot_list”文件是限制在主目录下的例外用户名单。如果需要设置某个用户不受只可以访问其主目录的限制，可将对应的用户名写入该文件。如果没有例外也必须要有“chroot_list”文件，内容可为空。

7. 执行以下命令重启vsftpd服务使配置生效。

```
service vsftpd restart
```

设置安全组

搭建好FTP站点后，需要在弹性云主机安全组的入方向添加一条放行FTP端口的规则，具体步骤参见[虚拟私有云-安全组-添加安全组规则](#)。

根据FTP的不同模式需要放通的端口如下：

- FTP为主动模式时：端口21。
- FTP为被动模式时：端口21，以及配置文件“/etc/vsftpd/vsftpd.conf”中参数“pasv_min_port”和“pasv_max_port”之间的所有端口。

客户端测试

打开客户端的计算机，在路径栏输入“ftp://FTP服务器IP地址:FTP端口”（如果不填端口则默认访问21端口）。弹出输入用户名和密码的对话框表示配置成功，正确的输入用户名和密码后，即可对FTP文件进行相应权限的操作。

说明

如果FTP服务器配置为主动模式，客户端使用此方法访问FTP站点时，需要对IE浏览器进行设置，才能打开FTP的文件夹。打开IE浏览器，选择“工具 > Internet 选项 > 高级”。勾选“启用FTP文件夹视图”，取消勾选“使用被动FTP”。

使用浏览器访问 FTP 服务器出错时，建议您清除浏览器缓存后再尝试。

快速构建FTP站点（Linux）

Linux实例手动搭建FTP站点具体操作步骤

1. 安装vsftpd
2. 配置vsftpd
3. 设置安全组
4. 客户端测试

示例环境

- 实例类型：s3.large.2 | 2核 | 4G 通用型云主机
- 所在区域：山西
- 系统盘：40GB
- 操作系统：CentOS7.2 64位
- 公网弹性IP带宽：1Mbps

操作步骤

1. 安装vsftpd

- a. 登录弹性云主机实例。
- b. 执行以下命令安装：

```
vsftpd, yum install -y vsftpd
```

回显如下类似信息时，表示软件安装成功。

```
[root@ecm-c9e7 ~]# yum install -y vsftpd
Loaded plugins: fastestmirror
Determining fastest mirrors
base                               | 3.6 kB | 00:00:00
epel                                 | 4.7 kB | 00:00:00
vsftpd                               | 2.9 kB | 00:00:00
updates                              | 2.9 kB | 00:00:00
(1/7): epel/updates/primary_db       | 90 kB | 00:00:00
(2/7): base/updates/primary_db       | 123 kB | 00:00:00
(3/7): epel/updates/primary_db       | 1.9 MB | 00:00:00
(4/7): epel/updates/primary_db       | 7.0 MB | 00:00:00
(5/7): base/updates/primary_db       | 5.1 MB | 00:00:00
(6/7): extras/updates/primary_db     | 250 kB | 00:00:00
(7/7): updates/updates/primary_db    | 22 MB | 00:00:02
Resolving Dependencies
--> Running transaction check
--> Package vsftpd.x86_64 0:3.0.2-29.el7_9 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
Package Arch Version Repository Size
================================================================================
Installing:
vsftpd x86_64 3.0.2-29.el7_9 updates 173 k
Transaction Summary
-----
Install 1 Package

Total download size: 173 k
Installed size: 303 k
Downloading packages:
vsftpd-3.0.2-29.el7_9.x86_64.rpm | 173 kB | 00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
Warning: RPMDB altered outside of yum.
Installing : vsftpd-3.0.2-29.el7_9.x86_64 | 1/1
Verifying : vsftpd-3.0.2-29.el7_9.x86_64 | 1/1
Installed:
vsftpd.x86_64 0:3.0.2-29.el7_9
Complete!
```

- c. 执行以下命令设置FTP服务开机自启动。

```
systemctl enable vsftpd.service
```

- d. 执行以下命令启动FTP服务。

```
systemctl start vsftpd.service
```

- e. 执行以下命令查看FTP服务端口。

```
netstat -antup | grep ftp
```

回显如下类似信息。

```
[root@ecm-c9e7 ~]# netstat -antup | grep ftp
tcp6      0      0  ::::21                :::*          LISTEN    3503/vsftpd
```

2. 配置vsftpd

vsftpd安装后默认开启了匿名FTP的功能，使用匿名FTP，用户无需输入用户名密码即可登录FTP服务器，但没有权限修改或上传文件。用户如果试图使用Linux操作系统中的账号登录服务器，将会被vsftpd拒绝，但

最佳实践

可以在vsftpd里配置用户账号和密码登录。以下操作以在vsftpd 里配置用户账号和密码登录FTP服务器为例。

- a. 执行以下命令创建“ftptest”用户，此处“ftptest”用户为示例，可根据实际情况创建相应的用户。

```
useradd ftptest
```

- b. 执行以下命令并按照提示设置“ftptest”用户密码passwd ftptest，输入密码即可。

```
[root@ecm-c9e7 ~]# useradd ftptest
[root@ecm-c9e7 ~]# passwd ftptest
Changing password for user ftptest.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

- c. 执行以下命令创建供FTP使用的文件目录，此处以“/var/ftp/test”为例。

```
mkdir /var/ftp/test
```

- d. 执行以下命令将创建的文件目录所有者改为用于登录FTP的本地用户。

```
chown -R ftptest:ftptest /var/ftp/test
```

- e. 修改“vsftpd.conf”配置文件。

- 执行以下命令打开配置文件“vsftpd.conf”。

```
vi /etc/vsftpd/vsftpd.conf
```

- 按i键进入编辑模式。
- 修改打开的“vsftpd.conf”文件。

```
anonymous_enable=YES #开启匿名用户访问。默认已开启。
```

```
write_enable=YES #开放服务器的写权限（若要上传，必须开启）。默认已开启。
```

```
anon_umask=022 #设置匿名用户所上传数据的权限掩码（反掩码）。默认已开启。
```

```
anon_upload_enable=YES #允许匿名用户上传文件。默认已注释，需取消注释。
```

```
anon_mkdir_write_enable=YES #允许匿名用户创建（上传）目录。默认已注释，需取消注释。
```

```
anon_other_write_enable =YES #允许删除、重命名、覆盖等操作。需添加。
```

最佳实践

```
# capabilities.
#
# All anonymous FTP? (Beware - allowed by default if you comment this out).
anonymous_enable=YES
# Uncomment this to allow local users to log in.
# when setroot is enforcing check for SE bool ftp_home_dir
local_enable=YES
# Uncomment this to enable any form of FTP write command.
write_enable=YES
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftp'd's)
local_umask=022
# Uncomment this to allow the anonymous FTP user to upload files. This only
# has an effect if the above global write enable is activated. Also, you will
# obviously need to create a directory writable by the FTP user.
# when SELinux is enforcing check for SE bool allow_ftpd_anon_write, allow_ftpd_full_access
anon_upload_enable=YES
# Uncomment this if you want the anonymous FTP user to be able to create
# new directories.
anon_mkdir_write_enable=YES
# Activate directory messages - messages given to remote users when they
# go into a certain directory.
dirmessage_enable=YES
#
# Activate logging of uploads/downloads.
xferlog_enable=YES
#
# Make sure PORT transfer connections originate from port 20 (ftp-data).
connect_from_port_20=YES
#
# Don't other write enable -YES.
#
/etc/vsftpd/vsftpd.conf* 131L, 5328C
44.1 8%
```

- f. 按Esc键退出编辑模式，并输入:wq保存后退出。
- g. 执行以下命令重启vsftpd服务使配置生效。

systemctl restart vsftpd.service

- h. 关闭防火墙和增强型安全功能。

systemctl stop firewalld

setenforce 0

3. 设置安全组

搭建好FTP站点后，需要配置弹性云主机的安全组规则，入方向添加一条放行FTP端口的规则，设置安全组规则如下表。

FTP模式	方向	协议	端口	源地址
主动模式	入方向	TCP	20端口和21端口	0.0.0.0/0
被动模式	入方向	TCP	21端口和配置文件“/etc/vsftpd/vsftpd.conf”中参数“pasv_min_port”和“pasv_max_port”之间的所有端口	0.0.0.0/0

最佳实践

4. 客户端测试

- 在Windows系统打开开始菜单，输入 cmd 命令打开命令提示符。
- 建立ftp连接：ftp XX.XX.XX.XX（虚拟机公网ip地址）。
- 输入配置好的用户名和密码，反馈如下表示访问成功。

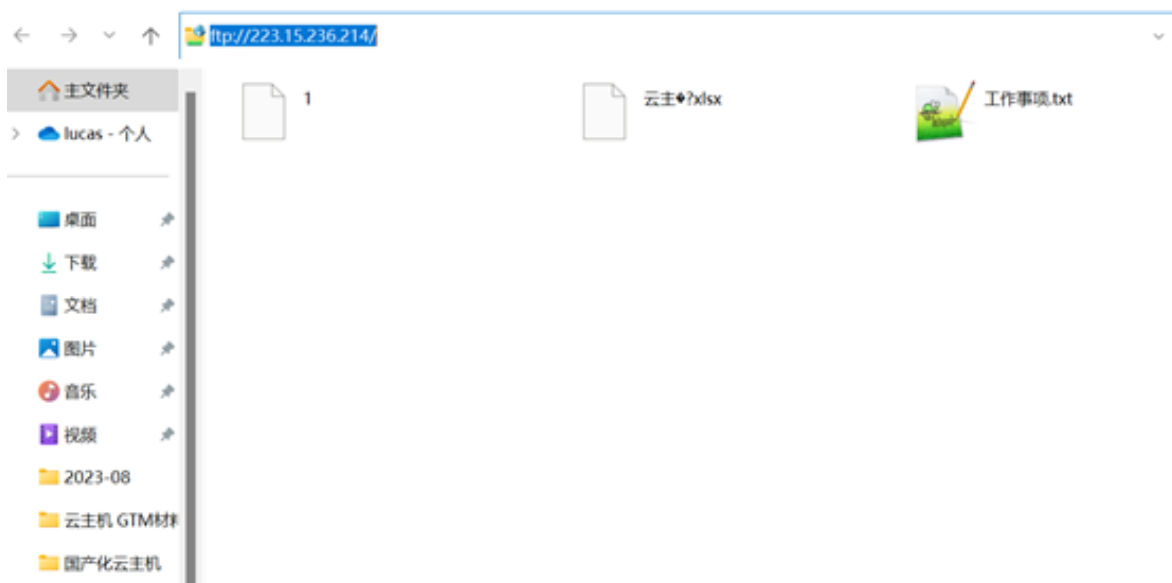
```
命令提示符 - ftp 223.15.236.2 x +
Microsoft Windows [版本 10.0.22621.2134]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\Ctyun>ftp 223.15.236.214
连接到 223.15.236.214.
220 (vsFTPd 3.0.2)
200 Always in UTF8 mode.
用户(223.15.236.214:(none)): ftptest
331 Please specify the password.
密码:
230 Login successful.
ftp>
```

- ftp> pwd #匿名访问ftp的根目录为Linux系统的/var/ftp/目录。
- ftp> ls #查看当前目录。

```
ftp> pwd
257 "/home/ftptest"
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
226 Directory send OK.
ftp>
```

- 打开客户端的计算机，在路径栏输入“ftp://FTP服务器IP地址:FTP端口”（如果不填端口则默认访问21端口）。弹出输入用户名和密码的对话框表示配置成功，正确的输入用户名和密码后，即可对FTP文件进行相应权限的操作。



在linux云主机上文件目录下一致。

```
[root@ecm-c9e7 ftpadmin]# ls
1 云主???xlsx 工作事项.txt
[root@ecm-c9e7 ftpadmin]# █
```

手工部署RabbitMQ (CentOS 7.4)

本文介绍了如何在天翼云上使用弹性云主机的Linux实例部署RabbitMQ。RabbitMQ是采用Erlang语言实现AMQP (Advanced Message Queuing Protocol, 高级消息队列协议) 的消息中间件, 它最初起源于金融系统, 用于在分布式系统中存储转发消息。RabbitMQ凭借其高可靠、易扩展、高可用及丰富的功能特性成为目前非常热门的一款消息中间件。

前提条件

弹性云主机所在安全组添加了如下表所示的安全组规则, 具体步骤参见为安全组添加安全组规则。

方向	类型	协议	端口/范围	源地址
入方向	IPv4	TCP	5672	0.0.0.0/0
入方向	IPv4	TCP	15672	0.0.0.0/0

操作步骤

1. 安装相关依赖包和perl。

- 登录弹性云主机。
- 执行以下命令, 安装相关依赖包。

```
yum -y install make gcc gcc-c++ m4 ncurses-devel openssl-devel unixODBC-devel
```

- 执行如下命令, 安装perl。

```
yum install perl
```

2. 安装erlang。

- 执行如下命令，下载erlang安装包。

```
wget http://erlang.org/download/otp_src_19.3.tar.gz
```

- 执行如下命令，解压缩安装包。

```
tar xzf otp_src_19.3.tar.gz
```

解压后生成一个“otp_src_19.3”文件夹。

- 执行如下命令，创建文件夹“erlang”。

```
mkdir /usr/local/erlang
```

- 执行如下命令，进入解压后生成的文件夹“otp_src_19.3”。

```
cd otp_src_19.3
```

- 执行如下命令，检查系统是否符合安装要求。

```
./configure --prefix=/usr/local/erlang --without-javac
```

- 执行如下命令，编译并安装Erlang。

```
make && make install
```

- 执行如下命令，配置erlang环境变量。
 - 执行如下命令打开配置文件"profile”。

```
vi /etc/profile
```

- 按i键进入编辑模式。
- 修改打开的“profile”文件，将如下内容添加到文件末尾。

```
export PATH=$PATH:/usr/local/erlang/bin
```

- 按Esc键退出编辑模式，并输入:wq保存后退出。
- 执行如下命令，使环境变量生效。

```
source /etc/profile
```

- 执行如下命令，检查安装结果。

```
erl -version
```

回显类似如下信息，说明erlang安装成功。

```
[root@ecs-rabbitmq ~]# erl -version
Erlang (ASYNC_THREADS, HIPE) (BEAM) emulator version 8.3
```

3. 安装RabbitMQ

- 执行如下命令，进入用户主目录。

```
cd
```

- 执行如下命令，下载RabbitMQ安装包。

```
wget https://www.rabbitmq.com/releases/rabbitmq-server/v3.6.9/rabbitmq-server-generic-unix-3.6.9.tar.xz
```

- 执行如下命令，解压RabbitMQ安装包。

```
tar xvJf rabbitmq-server-generic-unix-3.6.9.tar.xz
```

- 执行如下命令，将解压目录移至“/usr/local/rabbitmq”。

```
mv rabbitmq_server-3.6.9 /usr/local/rabbitmq
```

- 执行如下命令，配置RabbitMQ环境变量。
- 执行如下命令打开配置文件"profile”。

```
vi /etc/profile
```

- 按i键进入编辑模式。
- 修改打开的“profile”文件，将如下内容添加到文件末尾。

```
export PATH=$PATH:/usr/local/rabbitmq/sbin
```

- 按Esc键退出编辑模式，并输入:wq保存后退出。
- 执行如下命令，使环境变量生效。

```
source /etc/profile
```

4. 执行如下命令，启用RabbitMQ的web管理界面。

```
rabbitmq-plugins enable rabbitmq_management
```

回显类似如下信息：

```
[root@ecs-rabbitmq ~]# rabbitmq-plugins enable rabbitmq_management
The following plugins have been enabled:
amqp_client
cowlib
cowboy
rabbitmq_web_dispatch
rabbitmq_management_agent
rabbitmq_management
Applying plugin configuration to rabbit@ecs-rabbitmq... started 6 plugins.
```

5. 执行如下命令，创建一个新用户。

```
rabbitmqctl add_user 用户名 密码
```

命令示例：

```
rabbitmqctl add_user root 123456
```

最佳实践

6. 执行如下命令，设置用户为管理员。

```
rabbitmqctl set_user_tags 用户名 administrator
```

命令示例：

```
rabbitmqctl set_user_tags root administrator
```

7. 执行如下命令，赋予用户所有权限。

```
rabbitmqctl set_permissions -p / 用户名 '.*'.*
```

命令示例：

```
rabbitmqctl set_permissions -p / root '.*'.*
```

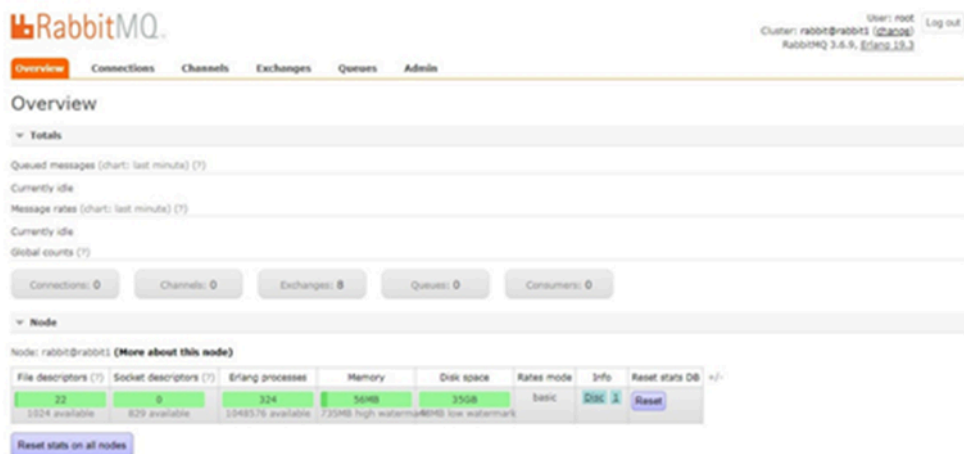
8. 执行如下命令，在后台启动RabbitMQ。

```
rabbitmq-server -detached
```

9. 使用浏览器访问 “http://弹性IP:15672”，显示如下页面，说明RabbitMQ安装成功。



10. 输入步骤5创建的用户名和密码后点击“Login”，进入RabbitMQ管理界面。



在天翼云上配置NTP服务器

该NTP服务器仅限于在天翼云控制台上购买的弹性云主机安装使用。该功能使用的前提是该节点具备内网DNS功能。

如需在购买的弹性云主机上安装NTP服务器，您可以选择使用天翼云控制台自己提供的NTP服务器，也可以选择其他NTP服务器。安装NTP服务器的操作相同，本节以天翼云控制台提供的NTP服务器为例，介绍在弹性云主机上安装NTP服务器的操作。

背景信息

使用天翼云提供的NTP服务器时，需和天翼云DNS服务器配套使用。天翼云NTP服务器配置请参考：[内网yum源及NTP配置](#)。

天翼云参考以下时钟服务器域名：

- cn.pool.ntp.org
- asia.pool.ntp.org
- pool.ntp.org
- hk.pool.ntp.org

Linux操作系统

1. 检查弹性云主机的DNS服务器域名是否正确。

- 登录Linux弹性云主机。
- 执行以下命令，打开resolv.conf文件。

```
vi /etc/resolv.conf
```

- 查看文件中nameserver的值是否和提供的DNS服务器域名相同。
 - 是，执行3。
 - 否，执行2。

2. (可选) 给弹性云主机配置DNS服务器。

- 登录Linux弹性云主机。
- 执行以下命令，编辑resolv.conf文件。

```
vi /etc/resolv.conf
```

- 添加如下语句，配置DNS服务器。
nameserver DNS服务器的IP地址（具体见内网DNS控制台提示）。

3. 给弹性云主机配置NTP服务器。

- 登录Linux弹性云主机。
- 执行以下命令，编辑ntp.conf文件。

```
vi /etc/ntp.conf
```

- 添加以下语句，配置NTP服务器。

```
server NTP服务器域名
```

示例：

```
server ntp.ctyun.cn
```

- 执行以下命令，系统重新启动时启动服务。
 - 对于SUSE操作系统：

```
service ntp restart
```

- 对于Euler、CentOS操作系统：

```
systemctl restart ntpd
```

请根据弹性云主机实际使用的操作系统，以及实际ntp服务名，选择具体的命令进行重启。

- 执行以下命令，检查与上层NTP服务器是否时钟同步成功。

```
ntpq -p
```

如果在您所配置的时钟服务器所在行的前面出现” * “，则表示系统已成功同步到您当前配置的时钟服务器。

首次进行NTP时钟同步需要一定时间，请等待几分钟再进行检查。

请根据弹性云主机实际使用的操作系统，以及实际ntp服务名，选择具体的命令设置NTP服务的开机自启动。

- 如果是SUSE操作系统：

```
chkconfig ntp on
```

- 如果是Euler、CentOS操作系统：

```
chkconfig ntpd on
```

Windows操作系统

以Windows 2016操作系统为例：

1. 以用户名Administrator，登录Windows弹性云主机。

2. 打开本地连接。
 - 在任务栏的右下角，单击网络连接的图标。
 - 单击“打开网络和共享中心”。
 - 在左侧导航栏，单击“更改适配器设置”。

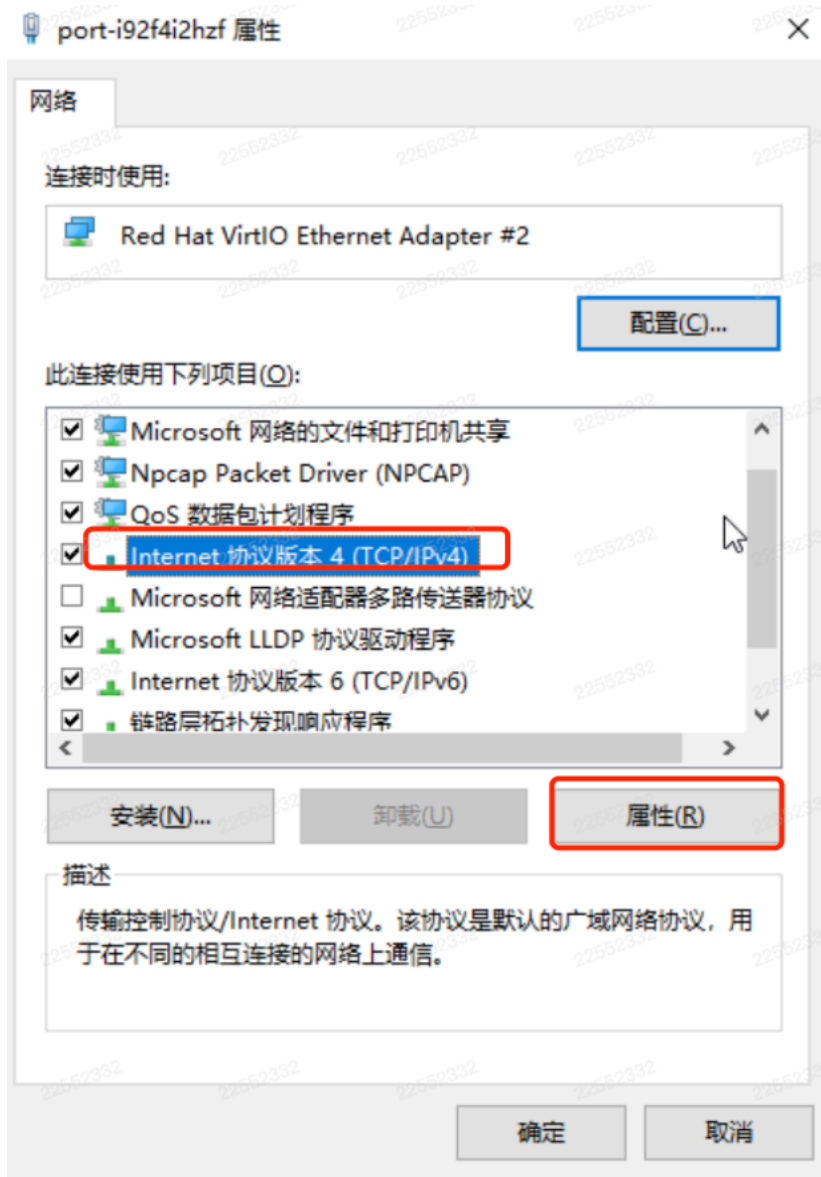
最佳实践

3. 给弹性云主机配置DNS服务器。

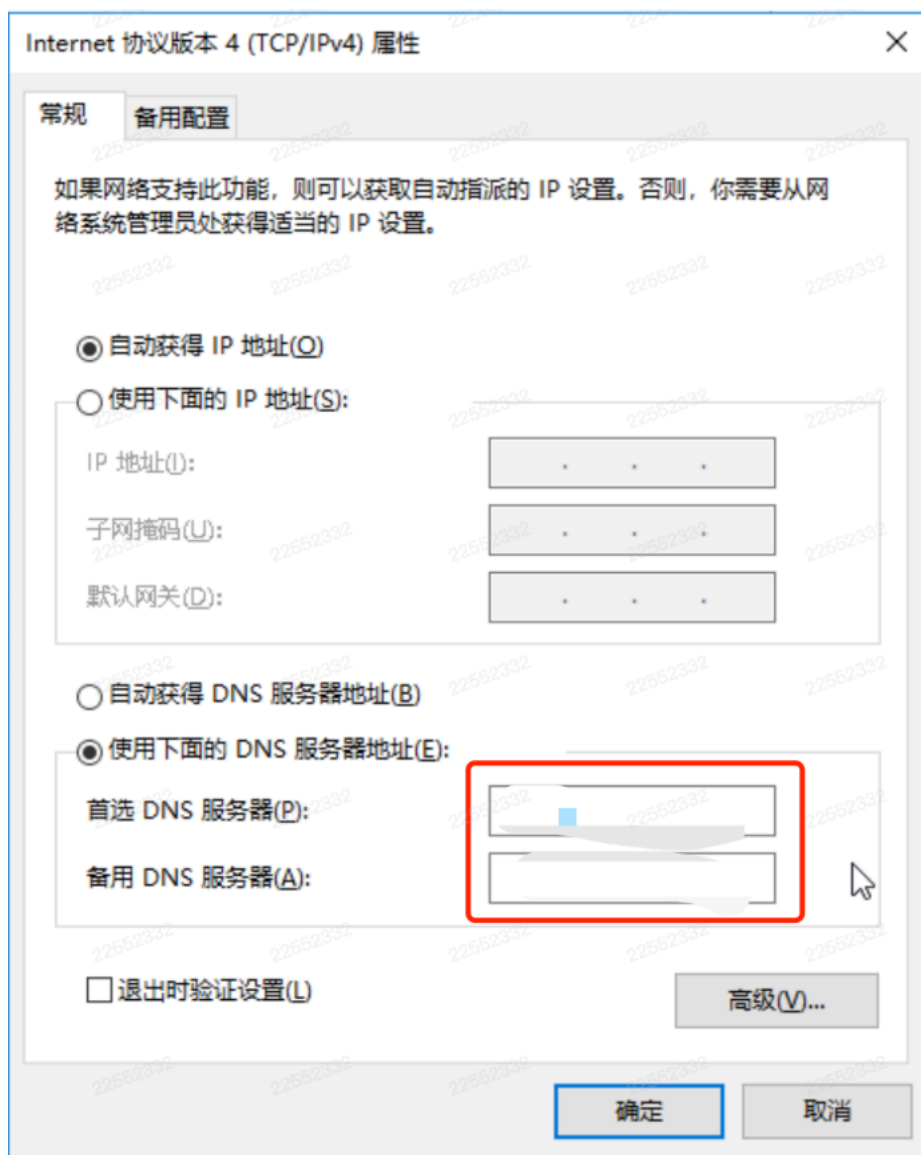
- 双击“本地连接”。
- 单击左下角的“属性”，如下图所示。



- 选择“Internet 协议版本 (TCP/IPv4)”，并单击“属性”，如下图所示。



- 选择“使用下面的DNS服务器域名”，并根据界面提示填写DNS服务器的IP地址，如下图所示。



4. 给弹性云主机配置NTP服务器。

- 使用快捷键“Win+R”，打开“运行”窗口。
- 在“打开”栏，输入“regedit”，并单击“确定”。打开注册表编辑器。
- 修改如下注册表信息：
 - 打开文件“HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpClient”，修改“Enabled”的“数值数据”为“1”，表示该设备为NTP服务器。
 - 打开文件“HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpServer”，修改“Enabled”的“数值数据”为“0”，关闭NTP服务器功能。
 - 打开文件“HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\Parameters”，设置“NtpServer”的“数值数据”为“ntp.ctyun.cn”，设置“TYPE”的“数值数据”为“NTP”。
 - 打开文件“HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpClient”，设置“SpecialPollInterval”的“数值数据”为“60”，“基数”为“十进制”，表示时钟同步周期为60s。
 - 打开文件“HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\W32Time\config”，设置“MaxPosPhaseCorrection”和“MaxNegPhaseCorrection”的“数值数据”为“ffffffff”，“基数”为“十六进制”。
- 使用快捷键“Win+R”，打开“运行”窗口。
- 在“打开”栏，输入“services.msc”，并单击“确定”。打开“服务”窗口。
- 查找名称为“Windows Time”的服务，并设置“启动类型”为“自动”，从NTP服务中同步时间。
- 使用快捷键“Win+R”，打开“运行”窗口。
- 依次执行以下命令，重启Windows Time服务。

```
net stop w32time
```

```
net start w32time
```

- 手动修改客户端时间，使得客户端时间与服务器时间不同。一分钟后，检查客户端时间与服务器时间是否一致，如果一致，则表示时间同步配置成功。

后续处理

弹性云主机重启后，DNS配置会被重置为VPC子网的DNS服务器地址。因此，建议在重启弹性云主机前，查询该弹性云主机的VPC子网DNS配置是否与目标DNS配置一致，如果不一致，请先修改弹性云主机VPC子网的DNS服务器地址。

手工搭建Ghost博客（Ubuntu 20.04）

Ghost是基于Node.js开发的一款免费的非常出色的开源博客平台，具有即时预览、极致简约、支持多用户等特性，本文将指导用户在Ubuntu20.04操作系统上的云主机实例上部署Ghost博客。

前置条件

使用本文的操作步骤，弹性云主机实例必须满足以下前置：

- 实例需要分配公网IP或绑定弹性公网IP。
- 操作系统：Ubuntu 16.04、Ubuntu 18.04、Ubuntu 20.04、Ubuntu 22.04。
- 实例所在的安全组入方向安全规则放行22、80、4423、2368端口。

创建新用户

由于Ghost官方不推荐使用root用户直接进行操作。因此需要重新创建新的用户，并且为其配置权限。

1. 执行以下命令，创建新用户。以创建test用户为例子。

```
adduser test
```

具体配置如下：

```
root@ecm-2d20-Ghost:~# adduser test
Adding user `test' ...
Adding new group `test' (1000) ...
Adding new user `test' (1000) with group `test' ...
Creating home directory `/home/test' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for test
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
```

2. 执行以下命令，将新创建的用户添加到组。

```
usermod -a -G sudo test
```

3. 执行以下命令，切换到test用户。

```
su - test
```

安装gcc和g++

1. 执行以下命令安装常用开发编译工具。

```
sudo apt-get install build-essential
```

2. 执行如下命令，安装gcc。

```
apt-get install gcc
```

3. 执行如下命令查看gcc版本。

```
gcc --version
```

回显信息：

```
root@ecm-da5b:~# gcc --version
gcc (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0
Copyright (C) 2019 Free Software Foundation, Inc.
```

4. 执行以下命令安装g++。

```
sudo apt-get install g++
```

5. 使用以下命令查看g++版本。

```
g++ --version
```

回显信息：

```
root@ecm-da5b:~# g++ --version
g++ (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0
Copyright (C) 2019 Free Software Foundation, Inc.
```

安装Node.js

在安装Ghost前，需要先安装Ghost所需要的Node.js。

1. 执行以下命令，进行Node.js安装源的下载。

```
curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
```

2. 执行以下命令，进行Node.js的安装。

```
sudo apt-get install -y nodejs
```

3. 在Node.js安装完成后，执行以下命令查看Node.js版本和npm版本。

```
node -v
```

```
npm -v
```

回显信息：

```
test@ecm-2d20-Ghost:~$ node -v
v16.20.2
test@ecm-2d20-Ghost:~$ npm -v
8.19.4
```

安装Nginx

部署Ghost博客，需要安装Nginx作为HTTP服务器。

1. 执行以下命令，进行Nginx安装。

```
sudo apt-get update
```

```
sudo apt-get install nginx
```

2. 安装完成后，执行以下命令查看Nginx版本。

```
nginx -v
```

回显如下：

```
test@ecm-2d20-Ghost:~$ nginx -v
nginx version: nginx/1.18.0 (Ubuntu)
```

3. 验证Nginx是否正常启动。

浏览器输入公网IP地址或者域名进行访问Nginx, 出现如下页面, 则说明Nginx安装成功。

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

4. Nginx配置信息修改（可选）。

输入以下命令修改配置文件。

```
vim /etc/nginx/nginx.conf
```

输入以下内容：

```
server {  
    listen 8001; //监听端口号 本文选择监听端口为8001, 避免出现80端口冲突问题  
    server_name 171.111.xxx.xxx; #这里写你的域名或者ip地址  
    location / {  
        proxy_set_header    X-Real-IP $remote_addr;  
        proxy_set_header    Host      $http_host;  
        proxy_pass            http://127.0.0.1:2368;  
    }  
}
```

安装MySQL

1. 执行以下命令, 安装MySQL。

```
sudo apt-get install -y mysql-server
```

2. 输入以下命令查看MySQL版本。

```
mysql -V
```

回显如下：

```
root@ecm-da5b:~# mysql -V  
mysql Ver 8.0.34-0ubuntu0.20.04.1 for Linux on x86_64 ((Ubuntu))
```

3. 打开MySQL控制台

通常，在未执行任何配置的情况下在 Ubuntu 上全新安装 MySQL 后，访问服务器的用户将使用身份验证套接字 (auth_socket) 插件进行身份验证。

auth_socket 的使用会阻碍服务器使用密码对用户进行身份验证。它不仅会引发安全问题，而且还会使用户无法使用外部程序访问数据库。因此我们需要将身份验证方法从 auth_socket 更改为使用 mysql_native_password(密码验证)。输入以下命令，打开MySQL控制台。

```
sudo mysql
```

4. 设置root用户密码。输入以下命令将MySQL身份验证方式从 auth_socket 更改为使用 mysql_native_password (密码验证)。

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY '输入root账号的密码'
```

5. 执行以下命令，创建Ghost使用的数据库。

本文以ghost为例。

```
create database ghost
```

6. 输入以下命令查看数据库创建情况。

```
show databases
```

回显信息：

```
mysql> show databases;
```

```
+-----+
| Database      |
+-----+
| ghost         |
| information_schema |
| mysql         |
| performance_schema |
| sys           |
+-----+
5 rows in set (0.00 sec)
```

7. 执行以下命令，退出MySQL。

```
exit
```

8. 避免数据库可能存在的中文乱码问题，可执行以下命令设置MySQL编码。

```
sudo vi /etc/my.cnf
```

输入以下内容，之后保存并退出。

```
[client]
default-character-set=utf8
[mysql]
default-character-set=utf8
[mysqld]
```

```
character-set-server=utf8  
collation-server=utf8_general_ci
```

执行以下命令, 重启MySQL生效。

```
sudo /usr/sbin/service mysql restart
```

安装Ghost

1. 输入以下命令创建Ghost-CLI命令工具, 快速配置Ghost。

```
sudo npm install ghost-cli@latest -g
```

2. 依次执行以下命令, 创建Ghost安装目录并进授权。

```
sudo mkdir -p /var/www/ghost
```

```
sudo chown test:test /var/www/ghost //配置权限
```

```
sudo chmod 775 /var/www/ghost
```

3. 执行以下命令, 进入创建的文件夹, 安装Ghost。

```
cd /var/www/ghost
```

最佳实践

ghost install

```
✓ Checking system Node.js version - found v16.20.2
✓ Checking current folder permissions
✓ Checking memory availability
✓ Checking free space
✓ Checking for latest Ghost version
✓ Setting up install directory
✓ Downloading and installing Ghost v5.60.0
✓ Finishing install process
Enter your blog URL: http://171.111
Enter your MySQL hostname: localhost
Enter your MySQL username: root
Enter your MySQL password: [hidden]
Enter your Ghost database name: ghost
✓ Configuring Ghost
✓ Setting up instance
+ sudo useradd --system --user-group ghost
? Sudo Password [hidden]
+ sudo chown -R ghost:ghost /var/www/ghost/content
✓ Setting up "ghost" system user
? Do you wish to set up "ghost" mysql user? Yes
✓ Setting up "ghost" mysql user
? Do you wish to set up Nginx? Yes
+ sudo mv /tmp/171-111-192-209/171-111-192-209.conf /etc/nginx/sites-available/171.111.192.209.conf
+ sudo ln -sf /etc/nginx/sites-available/171.111.192.209.conf /etc/nginx/sites-enabled/171.111.192.209.conf
+ sudo nginx -s reload
✓ Setting up Nginx
SSL certs cannot be generated for IP addresses, skipping
? Setting up SSL [skipped]
? Do you wish to set up Systemd? Yes
+ sudo mv /tmp/171-111-192-209/ghost_171-111-192-209.service /lib/systemd/system/ghost_171-111-192-209.service
+ sudo systemctl daemon-reload
✓ Setting up Systemd
+ sudo systemctl is-active ghost 171-111-192-209
? Do you want to start ghost? Yes
+ sudo systemctl start ghost 171-111-192-209
+ sudo systemctl is-enabled ghost 171-111-192-209
+ sudo systemctl enable ghost_171-111-192-209 --quiet
✓ Starting Ghost

ghost uses direct mail by default. To set up an alternative email method read our docs at https://ghost.org/docs/config/#mail
-----
ghost was installed successfully! To complete setup of your publication, visit:
http://171.111 ghost/
```

一些配置参数如下：

Enter your blog URL: 请输入已解析的域名或者公网IP，请输入 http://(您的域名或者公网IP)。

Enter your MySQL hostname: 请输入数据库连接地址，请输入 localhost 后按 Enter。

Enter your MySQL username: 请输入数据库用户名，请输入 root 后按 Enter。

Enter your MySQL password: 请输入数据库密码，请输入在设置root账户中已设置的密码后按 Enter。

Enter your database name: 请输入 Ghost 使用的数据库的名称，请输入在创建数据库中已创建的ghost后按 Enter。

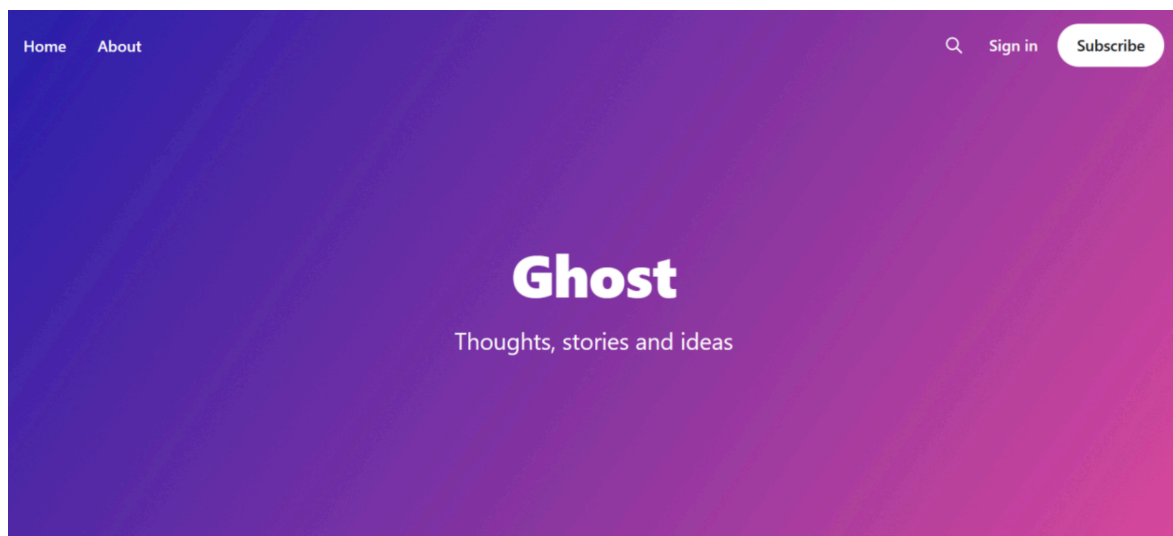
之后会有一些参数的配置，可按照需要进行设置，本文均输入yes。

4. 测试访问Ghost博客。

访问Ghost前台首页。

在任何具有访问公网能力的主机中，打开浏览器输入云主机实例的公网IP地址或者域名，即可访问Ghost前台首页。

例如：<http://公网IP地址或者域名>



如果想要进入管理员后台的话只需要在域名或者公网IP地址的后面加上/ghost即可进入，第一次进入需要进行账号注册，然后便能进入后台。

例如：<https://公网IP或者域名/ghost>



快速部署SD-WAN vCPE最佳实践

镜像描述

预装天翼云 SD-WAN智能网关软件版vCPE，支持客户快速部署入云服务。推荐配置：CPU 核数≥2，内存≥4G、挂载弹性IP。仅对ECS的主网卡，并且选择使用自动分配IPv4地址，挂载弹性IP场景使用。

配置要求

本镜像推荐配置：vCPU≥2核，内存≥4G，系统盘≥40G。

免责声明

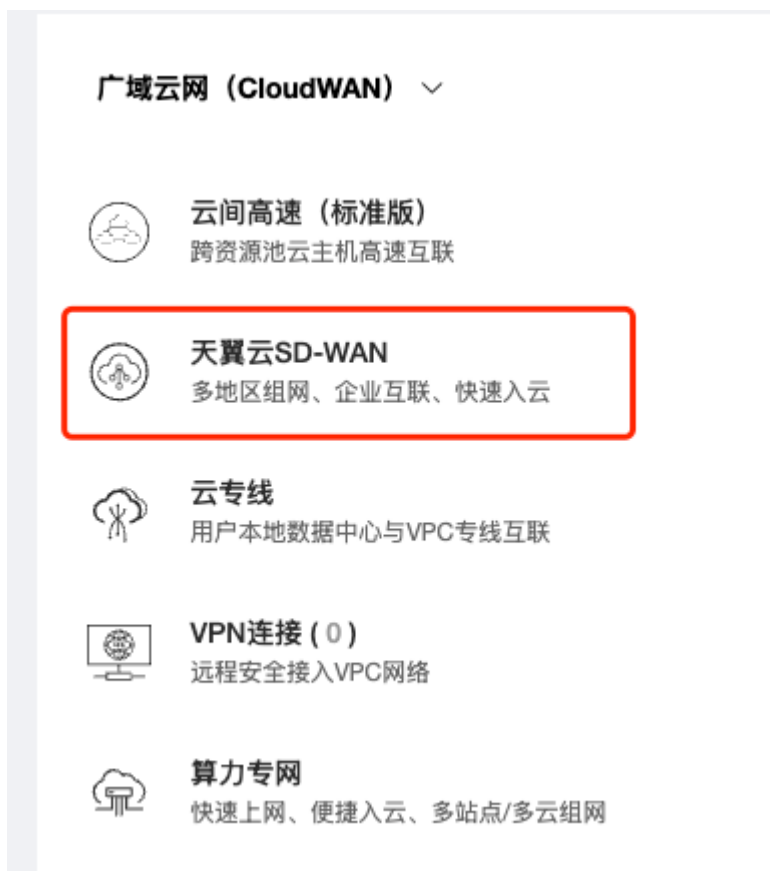
本公司发布的镜像作为SDWAN虚拟智能网关设备提供入云服务，仅供用户参考和使用。本公司不对因不当使用该镜像而导致的任何损害承担法律责任，用户应当确保其使用行为符合相关法律法规，并遵守相应协议。

实践指南

智能网关软件版（vCPE）是SD-WAN智能网关的软件镜像版，使用vCPE无需购买物理设备，vCPE作为镜像部署在云主机中，使云主机作为一个虚拟智能网关设备为用户提供更加灵活、便捷的入云服务。开通并部署vCPE后，用户可以通过SD-WAN智能网关控制台管理vCPE，并对其进行网络配置，将用户的网络接入云资源池。本文将为您介绍如何在云主机中部署vCPE镜像。

步骤一：开通SD-WAN 智能网关vCPE实例

1. 登录天翼云，点击页面右上方“控制中心”，进入控制台。
2. 单击管理控制台左上角的📍选择区域。
3. 依次选择“广域云网（CloudWAN）”，单击“天翼云SD-WAN”，进入天翼云SD-WAN总览页面。



4. 单击“智能网关”，在智能网关列表页面，单击“创建智能网关”按钮，进入创建智能网关页面。
5. 根据页面提示配置参数，关键参数信息如下：

参数	描述	样例
实例名称	长度为1-64字符，以大小写字母或中文开头，可包含数字、“.”、“_”、“-”	Test-01
组网方案	可选高性能核心网、互联网直连	高性能核心网
区域	所属位置信息	中国内地
基础带宽	带宽大小	5Mbps
是否购买设备	可选是、否	是
网关形态	可选择硬件版、软件版	软件版
设备类型	vCPE	vCPE

最佳实践

参数	描述	样例
使用方式	可选择单机、双机备份	单机
购买数量	购买智能网关的个数	1
购买时长	按需购买	1个月
企业项目	企业项目是一种云资源管理方式，企业项目管理服务提供统一的云资源按项目管理，以及项目内的资源管理、成员管理	default

6. 提交订单，确认订单配置信息，勾选协议，确认下单并支付。

步骤二：开通云服务器

使用镜像市场-"标准VCPE镜像-CTyunOS2.0.1-X86 64位"镜像开通云主机。云主机配置要求：配置要求2c4g以上，推荐计算型云主机，网卡使用自动分配内网IPv4地址，并有用弹性IP，带宽大小根据需要设置。

1. 登录天翼云，点击页面右上方“控制中心”，进入控制台。
2. 单击管理控制台左上角的📍选择区域。
3. 点击计算-弹性云主机进入创建云主机页面。
4. 单击右上角创建云主机进入云主机购买页面。
5. 进行基础配置。
 - a. 根据业务需求配置“计费模式”、“地域”、“企业项目”、“虚拟私有云”、“实例名称”、“主机名称”等。
 - b. 选择规格。此处选择"CPU架构"为"X86计算"、分类可以选"通用型"或"计算型"，推荐选用计算型。规格为2c4G以上。
 - c. 选择镜像。“镜像类型”选择“云镜像市场”，在云镜像市场中选择预置了vCPE的"标准VCPE镜像-CTyunOS2.0.1-X86 64位"镜像。
 - d. 设置云盘类型和大小。此处选择"系统盘"为"通用型SSD"，40GB。
6. 网络配置

设置网络，此处"网卡"选择“单网卡”，设置“自动分配内网地址”；"安全组"根据客户需求自行配置；"弹性IP"用于连通SDWAN POP接入点，带宽根据客户网络需求进行配置。

7. 高级配置

设置高级配置，包括"登录方式"、“云主机组”、“用户数据”。用户数据需配置为vCPE实例的SN号。vCPE SN号信息可通过“天翼云SD-WAN控制台->目标智能网关详情页->基本信息模块”查看，操作步骤可参考：[查看基本信息](#)。

用户数据填写格式如下：

其中，CTCEXXXXXXXXSN根据客户订购的vCPE实例SN号进行替换。

```
#!/bin/bash
set_sn CTCEXXXXXXXXSN
```

8. 确认配置信息并支付。

步骤三：激活vCPE实例

1. 登录控制中心。
2. 在控制中心页面左上角点击📍，选择区域。
3. 依次选择“广域网(CloudWAN)”，单击“天翼云SD-WAN”，进入天翼云SD-WAN总览页面。
4. 选择“智能网关”，单击目标智能网关vCPE实例“操作”列的“更多-激活”。
5. 根据页面提示，配置私网网络地址。
6. 单击“确定”，开始激活操作。
7. 激活成功后，业务状态变为“已激活”。

步骤四：绑定SD-WAN网络实例

1. 登录天翼云SD-WAN控制台。
2. 选择“智能网关”，单击目标智能网关实例“操作”列的“绑定网络实例”。
3. 根据页面提示，选择要加入的天翼云SD-WAN实例。
4. 单击“确定”，完成智能网关实例绑定天翼云SD-WAN实例。

步骤五：创建互联

1. 登录天翼云SD-WAN控制台。
2. 选择“天翼云SD-WAN”，单击上述已经绑定的目标SD-WAN实例名称。
3. 进入“设备互联”页签，单击“添加设备互联”。
4. 根据页面提示，选择要建立互联关系的两个智能网关实例。
5. 单击“确定”，完成在两个智能网关设备之间建联。

互联链路建立成功后，您可通过对端智能网关实例的下挂设备访问vCPE所在VPC下的云主机，测试是否连通。

基于弹性云主机的视频处理应用加速实践

镜像描述

当前镜像处于公测阶段，您可在华东1资源池试用。如果您希望在其他资源池试用该镜像，可以提交工单申请。

Intel® iVTAL 是一种用于视频处理加速的软件库，利用英特尔® 处理器中的硬件加速性能，以提供快速而优化的视频处理解决方案。通过使用 Intel® iVTAL 库，开发人员可以轻松地在其应用程序中集成视频处理功能。

视频处理专用镜像是一款集成intel视频处理调优工具iVTAL的专用的视频处理镜像，镜像内置视频抽帧工具ivtalDecoder和视频转码工具FFmpeg(intel版)，支持常见的音频和视频格式。专用视频处理镜像专为提升视频转码/解码/编码任务的速度和效率而生，告别通用镜像的平庸性能。

配置要求

推荐规格：c7.4xlarge.2或者c8.4xlarge.2

推荐配置：vCPU≥16核，内存≥32G

免责声明

免责声明：本公司发布的视频处理专用镜像，集成的转码和解码工具均来自第三方或者开源社区，仅供用户参考和使用。本公司不对因不当使用该镜像而导致的任何损害承担法律责任，用户应当确保其使用行为符合相关法律法规，并遵守相应协议。

实践指南

视频转码工具（英特尔版）

视频转码工具介绍

转码是将视频码流转换成另一个视频码流的过程。使用转码功能可以实现将视频转换成流畅、标清、高清以及超清等输出，从而用户可以选择合适的视频播放。视频转码最常用的三个软件工具是FFmpeg, x264, x265。在FFmpeg(version release/4.4), x264(version 0.164), x265(version 3.6)这三个工具的基础上, intel采用MVReuse算法策略实施了进一步的优化, 能够在几乎不影响视频质量·压缩率的情况下, 显著提升转码速度。

视频转码工具命令参数说明

运行一个最简单的视频转码所需要的参数如下：

- y 覆盖输出文件
- i 指定输入视频
- c:v 指定视频编码器, libx265转码相比于libx264能提供更好的压缩率, 但速度也会变慢
- crf 指定视频压缩力度, 越小压缩率越低质量越好, 越大压缩率越高质量越差
- mvreuse 2表示开启intel优化, 如果命令中没有-mvreuse 2则默认不开启英特尔优化, 此时转码效果与使用官方的FFmpeg(version release/4.4), x264(version 0.164), x265(version 3.6)效果相同
- psnr 显示转码后的质量分数

指定输出视频文件

运行示例1

#执行命令：

```
ffmpeg -y -i movie.mp4 -c:v libx264 -mvreuse 2 -crf 22 -psnr out.mp4
```

#命令解释：

ffmpeg	使用视频转码工具（英特尔版）
-y	想要默认覆盖输出文件
-i movie.mp4	输入1920x1080 24fps的电影场景视频
-c:v libx264	使用libx264编码器
-mvreuse 2	开启intel优化
-crf 22	控制压缩力度为22
-psnr	想看到转码后的质量分数
out.mp4	指定输出视频的名字

执行结果如图

最佳实践

```
frame= 734 fps=125 q=-1.0 LPSNR=Y:inf U:inf V:inf *:inf size= 22955kB time=00:00:30.45 bitrate=6173.9kbts/s speed= 5.2x
video:22946kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 0.036821%
[libx264 @ 0x2fa8940] frame I:33 Avg QP:20.43 size: 66543 PSNR Mean Y:50.10 U:54.69 V:55.47 Avg:51.20 Global:51.11
[libx264 @ 0x2fa8940] frame P:322 Avg QP:23.34 size: 37969 PSNR Mean Y:48.14 U:52.27 V:52.63 Avg:49.14 Global:48.96
[libx264 @ 0x2fa8940] frame B:379 Avg QP:24.35 size: 23944 PSNR Mean Y:46.26 U:51.22 V:51.76 Avg:47.41 Global:47.28
[libx264 @ 0x2fa8940] consecutive B-frames: 27.7% 6.8% 11.0% 54.5% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0% 0.0%
[libx264 @ 0x2fa8940] mb I I16..4: 15.6% 78.6% 5.9%
[libx264 @ 0x2fa8940] mb P I16..4: 11.5% 40.6% 1.4% P16..4: 38.8% 0.4% 0.1% 0.0% 0.0% skip: 7.2%
[libx264 @ 0x2fa8940] mb B I16..4: 1.9% 7.8% 0.3% B16..8: 49.9% 0.4% 0.1% direct: 5.9% skip:33.8% L0:46.1% L1:48.5% BI:
[libx264 @ 0x2fa8940] 8x8 transform intra:76.5% inter:89.1%
[libx264 @ 0x2fa8940] coded y,uvDC,uvAC intra: 51.5% 59.0% 10.7% inter: 24.5% 27.9% 0.2%
[libx264 @ 0x2fa8940] i16 v,h,dc,p: 19% 27% 5% 49%
[libx264 @ 0x2fa8940] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 24% 22% 14% 5% 7% 7% 7% 7%
[libx264 @ 0x2fa8940] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 28% 22% 10% 5% 9% 9% 7% 6% 3%
[libx264 @ 0x2fa8940] i8c dc,h,v,p: 50% 24% 22% 5%
[libx264 @ 0x2fa8940] ref P L0: 75.5% 14.0% 9.1% 1.5%
[libx264 @ 0x2fa8940] ref B L0: 90.4% 7.4% 2.2%
[libx264 @ 0x2fa8940] ref B L1: 97.5% 2.5%
[libx264 @ 0x2fa8940] PSNR Mean Y:47.261 U:51.834 V:52.306 Avg:48.337 Global:48.077 kb/s:6146.21
```

#日志解释:

- fps=125 视频转码实时吐帧量，用来衡量视频转码速度指标
- avg=48.337 视频转码后质量分数，用来衡量视频转码质量指标
- kb/s=6146.21 视频转码后平均码率，用来衡量视频转码压缩率指标

运行示例2

#执行命令:

```
ffmpeg -y -i movie.mp4 -c:v libx265 -mvreuse 2 -crf 28 -psnr out.mp4
```

#命令解释:

- ffmpeg 使用视频转码工具（英特尔版）
- y 想要默认覆盖输出文件
- i movie.mp4 输入1920x1080 24fps的电影场景视频
- c:v libx265 使用libx265编码器
- mvreuse 2 开启intel优化
- crf 28 控制压缩力度为28
- psnr 想看到转码后的质量分数
- out.mp4 指定输出视频的名字

执行结果如图

```
frame= 734 fps= 58 q=31.9 LPSNR=Y:inf U:inf V:inf *:inf size= 7835kB time=00:00:30.45 bitrate=2107.3kbts/s speed=2.41x
video:7824kB audio:0kB subtitle:0kB other streams:0kB global headers:2kB muxing overhead: 0.143564%
x265 [info]: frame I: 22, Avg QP:27.39 kb/s: 5315.23 PSNR Mean: Y:46.070 U:48.877 V:49.316
x265 [info]: frame P: 330, Avg QP:29.61 kb/s: 2793.41 PSNR Mean: Y:44.145 U:47.427 V:47.679
x265 [info]: frame B: 382, Avg QP:32.36 kb/s: 1306.11 PSNR Mean: Y:42.371 U:46.736 V:47.204
x265 [info]: Weighted P-Frames: Y:1.5% UV:1.5%
encoded 734 frames in 12.55s (58.51 fps), 2094.95 kb/s, Avg QP:30.97, Global PSNR: 44.283
```

#日志解释:

- 58.51fps 视频转码实时吐帧量，用来衡量视频转码速度指标
- PSNR:44.283 视频转码后质量分数，用来衡量视频转码质量指标
- 2094.95kb/s 视频转码后平均码率，用来衡量视频转码压缩率指标

最佳实践

视频抽帧工具（英特尔版）

视频抽帧工具介绍

视频抽帧可以从原始视频中提取关键帧，用于后续AI模型的内容分析、检索或建模。它还能显著降低数据量，加速视频处理流程，如智能分析、转码或特征提取等任务。视频抽帧最常用的软件工具是FFmpeg。参考FFmpeg(version release/6.1), intel实施了视频包预取和自适应线程优化，显著提升抽帧速度。

说明

Decoder功能仅针对6.1版本的ffmpeg且仅针对每400帧抽取1帧的情况下有效率提升。

抽帧工具ivtalDecoder转码参数说明

-mode 0 -fi 指定抽帧间隔

-thread 指定线程数量

指定输入视频

指定输出抽帧后像素文件

运行示例1

执行命令

```
ivtalDecoder -mode 0 -fi 400 -thread 1 movie.mp4 movie-ivtal2.yuv
```

#命令解释:

ivtalDecoder 使用视频抽帧工具（英特尔版）

-mode 0 -fi 400 指定每隔400帧抽一帧

-thread 1 用一个线程

movie.mp4 输入1920x1080 24fps的电影场景视频

movie-ivtal2.yuv 指定输出像素文件的名字

执行结果如图

```
[root@angyin ~]# time ivtalDecoder -mode 0 -fi 400 -thread 1 movie.mp4 movie-ivtal2.yuv

do videosouce function
do getfirstframe
YUV File deleted successfully

[input] movie.mp4
[output] movie-ivtal2.yuv

run decoder
[mode] 0
[fi] 400
save yuv: 2
real 0m0.713s
user 0m0.671s
sys 0m0.039s
```

#日志解释:

save yuv=2 一共抽出来多少帧

real=0.713 一共耗时多少

最佳实践

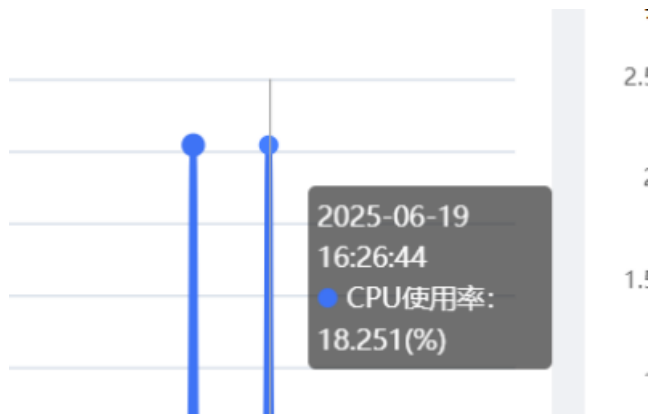
性能提升说明

下面是在规格c8.4xlarge.2云主机下的视频转码测试结果对比的数据。

- 优化前

```
ffmpeg -i movie.mp4 -c:v libx264 -crf 22 -psnr out.mp4
```

```
Stream #0:0 -> #0:0 (h264 (native) -> h264 (libx264))
Press [q] to stop, [?] for help
[libx264 @ 0x37aedc0] --psnr used with psy on: results will be invalid!
[libx264 @ 0x37aedc0] --tune psnr should be used if attempting to benchmark psnr!
[libx264 @ 0x37aedc0] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2 AVX FMA3 BMI2 AVX2 AVX512
[libx264 @ 0x37aedc0] profile High, level 4.0, 4:2:0, 8-bit
[libx264 @ 0x37aedc0] 264 - core 164 r3072 315af1b - H.264/MPEG-4 AVC codec - Copyleft 2003-2021 - http://www.videolan.org/v264.html - options: cabac=1 ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00 mixed_ref=1 me_range=16 chroma_me=1 trellis=1 8x8dct=1 cqm=0 deadzone=21,11 fast_pskip=1 chroma_qp_offset=-2 threads=24 lookahead_threads=4 sliced_threads=0 nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3 b_pyramid=2 b_adapt=1 b_bias=0 direct=1 weightb=1 op_en_gop=0 weightp=2 keyint=250 keyint_min=24 scenecut=40 intra_refresh=0 rc_lookahead=40 rc=crf mbtree=1 crf=22.0 qcomp=0.68 qpumi=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to 'out.mp4':
  Metadata:
    major_brand      : isom
    minor_version    : 512
    compatible_brands: isomiso2avc1mp41
    encoder          : Lavf58.76.100
  Stream #0:0(eng): Video: h264 (avc1 / 0x31637661), yuv420p(tv, bt709, progressive), 1920x1080, q=2-31, 24 fps, 12288 tbn (default)
  Metadata:
    handler_name     : ?Mainconcept Video Media Handler
    vendor_id        : [0][0][0][0]
    encoder          : Lavc58.134.100 libx264
  Side data:
    cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: N/A
frame= 734 fps= 55 q=1.0 LPSNR=Y:inf U:inf *:inf size= 23794kB time=00:00:30.45 bitrate=6399.5kbts/s speed= 2.3x
video:23785kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 0.036472%
[libx264 @ 0x37aedc0] frame I:30      Avg QP:20.27 size: 70459 PSNR Mean Y:50.00 U:54.84 V:55.63 Avg:51.21 Global:51.10
[libx264 @ 0x37aedc0] frame P:344    Avg QP:23.18 size: 39675 PSNR Mean Y:47.04 U:52.22 V:52.65 Avg:48.89 Global:48.61
[libx264 @ 0x37aedc0] frame B:360    Avg QP:24.18 size: 23870 PSNR Mean Y:46.42 U:51.14 V:51.59 Avg:47.52 Global:47.36
[libx264 @ 0x37aedc0] consecutive B-frames: 24.3% 27.5% 18.6% 37.6%
[libx264 @ 0x37aedc0] mb I  I16..4: 14.4% 79.1%  6.5%
[libx264 @ 0x37aedc0] mb P  I16..4: 11.2% 39.5%  1.5% P16..4: 32.9%  6.5%  1.4%  0.8%  0.8%  skip: 6.9%
[libx264 @ 0x37aedc0] mb B  I16..4:  2.0%  7.9%  0.3% B16..8: 42.2%  7.5%  0.9% direct: 5.3% skip:33.8% L0:47.9% L1:47.7% BI: 4.4%
[libx264 @ 0x37aedc0] 8x8 transform intra:76.3% inter:89.1%
[libx264 @ 0x37aedc0] coded y,uvDC,uvAC intra: 51.8% 50.8% 10.6% inter: 23.0% 20.3% 0.2%
[libx264 @ 0x37aedc0] i16 v,h,dc,p: 17% 29%  5% 49%
[libx264 @ 0x37aedc0] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 24% 22% 14%  5%  7%  7%  7%  7%
[libx264 @ 0x37aedc0] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 20% 23% 11%  5%  9%  9%  7%  6%
[libx264 @ 0x37aedc0] i8c dc,h,v,p: 49% 24% 22%  5%
[libx264 @ 0x37aedc0] Weighted P-Frames: Y:0.9% UV:0.9%
[libx264 @ 0x37aedc0] ref P L0: 66.0% 13.3% 13.8%  6.9%  0.0%
[libx264 @ 0x37aedc0] ref B L0: 89.8%  8.1%  2.1%
[libx264 @ 0x37aedc0] ref B L1: 90.3%  1.7%
[libx264 @ 0x37aedc0] PSNR Mean Y:47.237 U:51.800 V:52.251 Avg:48.314 Global:48.018 kb/s:6370.85
Profile:main-1997 testmode=0 flags=0 -i movie.mp4 -c:v libx264 -crf 22 -psnr out.mp4
```



最佳实践

- 优化后

```
ffmpeg -i movie.mp4 -c:v libx264 -mvreuse 2 -crf 22 -psnr out.mp4
```

```
[[libx264 @ 0x38a7ac0] using cpu capabilities: MMX SSE2Fast SSSE3 SSE4.2 AVX FMA3 BMI2 AVX2 AVX512
[[libx264 @ 0x38a7ac0] profile High, level 4.0, 4:2:0, 8-bit
[[libx264 @ 0x38a7ac0] 264 - core 164 r3072 315af1b - H.264/MPEG-4 AVC codec - Copyleft 2003-2021 - http://www.videolan.org/x264.html - options: cabac=1 ref=4 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00 mixed_ref=1 me_range=16 chroma_me=0 trellis=1 8x8dct=1 cqm=0 deadzone=21,11 fast_pskip=1 chroma_qp_offset=-2 threads=24 lookahead_threads=4 sliced_threads=0 nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=16 b_pyramid=2 b_adapt=0 b_bias=0 direct=1 weightb=0 open_gop=0 weightp=0 keyint=250 keyint_min=24 scenecut=40 intra_refresh=0 rc_lookahead=40 rc=crf mbtree=1 crf=22.0 qcomp=0.68 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to 'out.mp4':
  Metadata:
    major_brand      : isom
    minor_version    : 512
    compatible_brands: isomiso2avc1mp41
    encoder          : Lavf58.76.100
  Stream #0:0(eng): Video: h264 (avc1 / 0x31637661), yuv420p(tv, bt709, progressive), 1920x1080, q=2-31, 24 fps, 12288 tbn (default)
  Metadata:
    handler_name     : ?Mainconcept Video Media Handler
    vendor_id        : [0][0][0][0]
    encoder          : Lavc58.134.100 libx264
  Side data:
    cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: N/A
frame= 734 fps= 85 q=-1.0 LPSNR=Y:inf U:inf V:inf W:inf size= 22955kB time=00:00:30.45 bitrate=6173.9kbits/s speed=3.54x
video:22946kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 0.036821%
[[libx264 @ 0x38a7ac0] frame I:33 Avg QP:20.43 size: 66543 PSNR Mean Y:50.10 U:54.69 V:55.47 Avg:51.20 Global:51.11
[[libx264 @ 0x38a7ac0] frame P:322 Avg QP:23.34 size: 37969 PSNR Mean Y:48.14 U:52.27 V:52.63 Avg:49.14 Global:48.96
[[libx264 @ 0x38a7ac0] frame B:379 Avg QP:24.35 size: 23944 PSNR Mean Y:46.26 U:51.22 V:51.76 Avg:47.41 Global:47.28
[[libx264 @ 0x38a7ac0] consecutive B-frames: 27.7% 6.8% 11.8% 54.5% 0.8% 0.8% 0.8% 0.8% 0.8% 0.8% 0.8% 0.8% 0.8% 0.8% 0.8%
[[libx264 @ 0x38a7ac0] mb I 116.4: 15.6% 78.6% 5.9%
[[libx264 @ 0x38a7ac0] mb P 116.4: 11.5% 40.6% 1.4% P16.4: 38.8% 0.4% 0.1% 0.8% 0.8% skip: 7.2%
[[libx264 @ 0x38a7ac0] mb B 116.4: 1.9% 7.8% 0.3% B16.8: 49.9% 0.4% 0.1% direct: 5.9% skip:33.8% L0:46.1% L1:48.5% B1: 5.3%
[[libx264 @ 0x38a7ac0] 8x8 transform intra:76.5% inter:89.1%
[[libx264 @ 0x38a7ac0] coded y,u,v0C,u0C intra: 51.5% 59.0% 10.7% inter: 24.5% 27.9% 0.2%
[[libx264 @ 0x38a7ac0] i16 v,h,dc,p: 19% 27% 5% 49%
[[libx264 @ 0x38a7ac0] i8 v,h,dc,ddl,ddr,vr,hd,vl,hu: 24% 22% 14% 5% 7% 7% 7% 7% 7%
[[libx264 @ 0x38a7ac0] i4 v,h,dc,ddl,ddr,vr,hd,vl,hu: 28% 22% 10% 5% 9% 9% 7% 6% 3%
[[libx264 @ 0x38a7ac0] i8c dc,h,v,p: 58% 24% 22% 5%
[[libx264 @ 0x38a7ac0] ref P L0: 75.5% 14.0% 9.1% 1.5%
[[libx264 @ 0x38a7ac0] ref B L0: 98.4% 7.4% 2.2%
[[libx264 @ 0x38a7ac0] ref B L1: 97.5% 2.5%
[[libx264 @ 0x38a7ac0] PSNR Mean Y:47.261 U:51.834 V:52.306 Avg:48.337 Global:48.077 kb/s:6146.21
[root@ecm-1997 testmovie]# ffmpeg -y -i movie.mp4 -c:v libx264 -mvreuse 2 -crf 22 -psnr out.mp4
```

过程中cpu利用率为11%。



指标	优化前	优化后
PSNR (dB)	48.3	48.3
Bitrate (kbps)	6399.5	6173.9
FPS (帧/秒)	55	85
CPU利用率 (%)	18%	11%

优化后相比优化前，cpu利用率明显下降，视频转码fps提升55%。

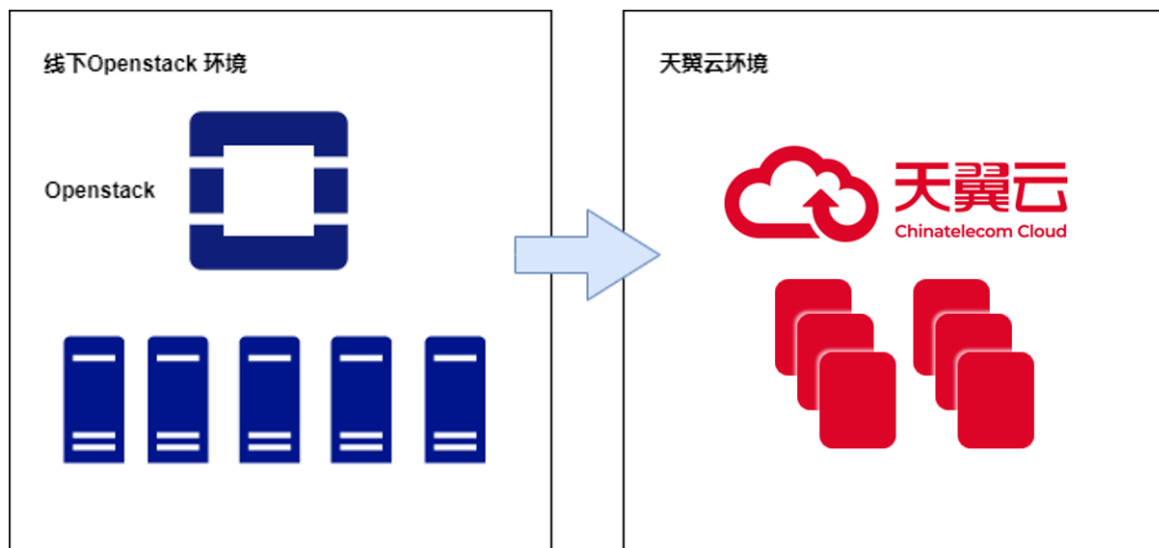
搭建环境相关最佳实践

线下OpenStack环境导入公有云环境

操作场景

在线下环境中，很多用户使用OpenStack构建云环境。为了降低成本，部分用户选择将线下OpenStack环境迁移至云服务商。此次场景将模拟OpenStack线下环境的云主机迁移到天翼云上，利用天翼云提供的服务提高资源运维管理能力，缩小成本。

架构迁移图示：



操作步骤

本文将介绍如何使用VMWare在本地搭建一套OpenStack环境，以模拟迁移和导入的过程。

模拟环境如下：

- OS: CentOS Linux release 7.9.2009
- VMWare Workstation 16.1.0
- 模拟流程包含如下步骤：
 - 安装OpenStack环境
 - 创建云主机
 - 导出云主机
 - 导入天翼云
 - 创建云主机（天翼云环境）

安装OpenStack环境

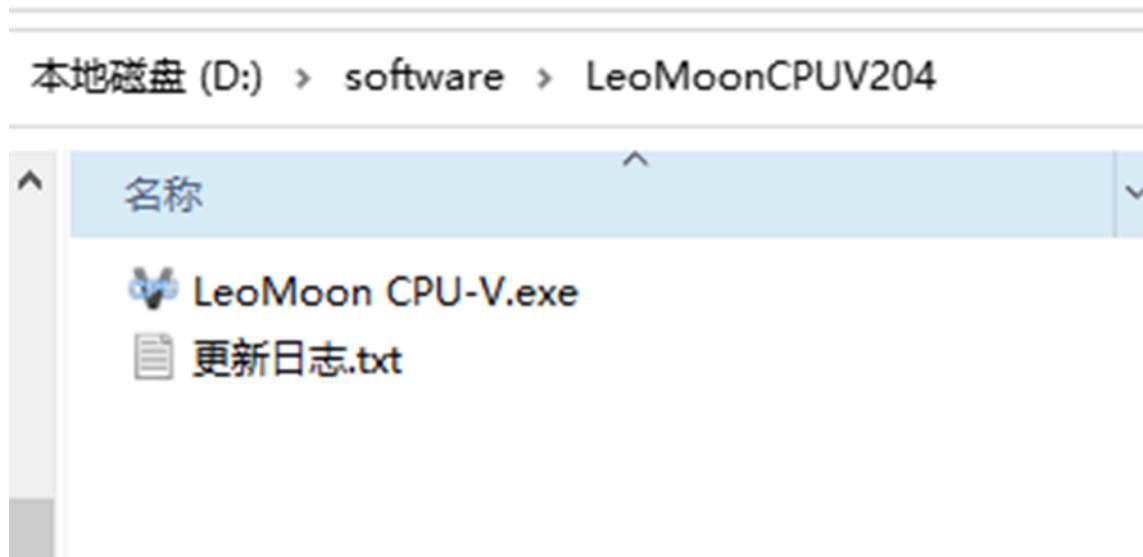
OpenStack能够创建云主机，我们需要保障如下要求符合：

最佳实践

- CPU支持虚拟化
- 内存至少8G以上

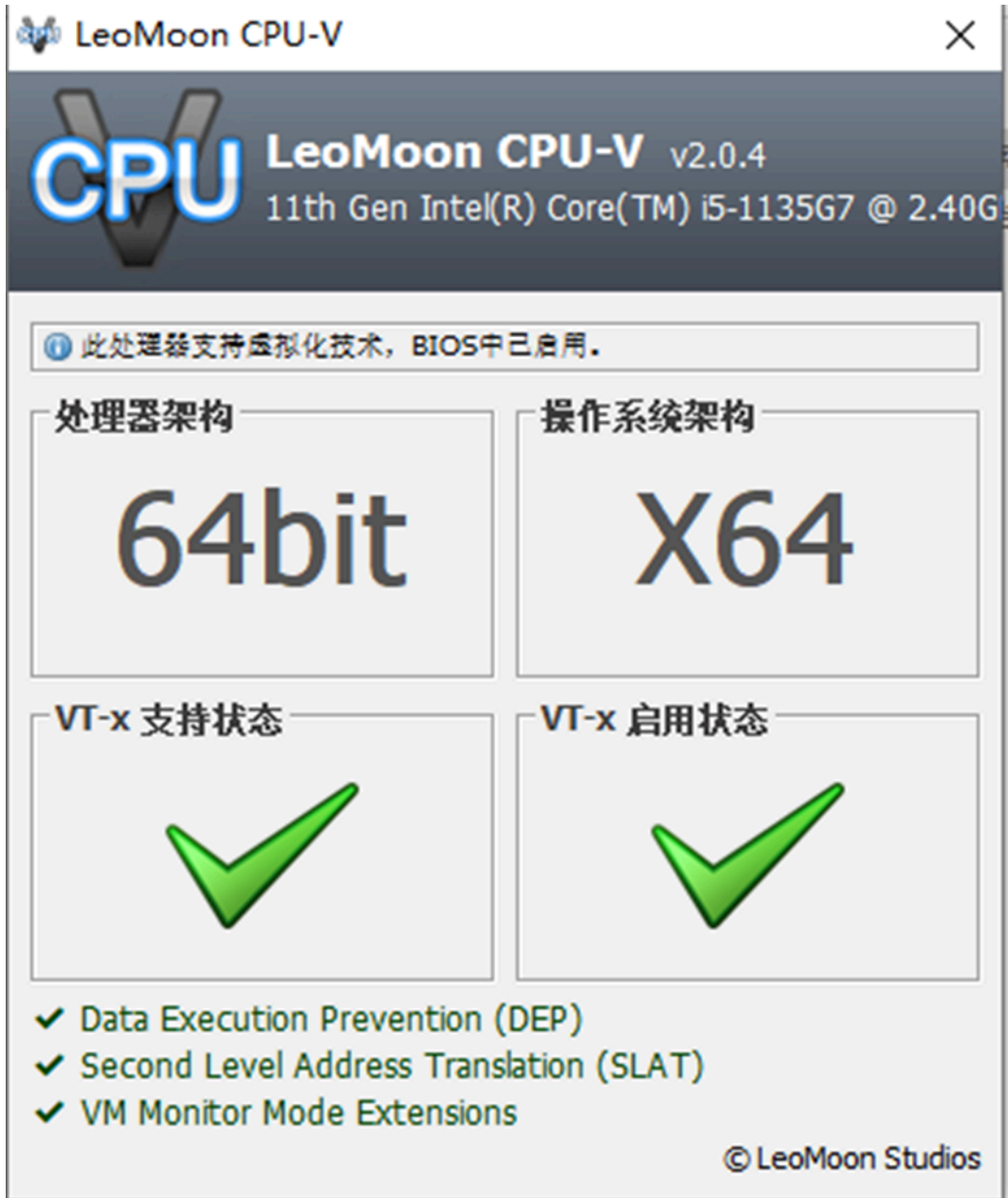
步骤1：检查CPU是否支持虚拟化

下载LeoMoon CPU-V，并双击打开。



最佳实践

步骤2: 如果结果如下图所示, 表明已经支持虚拟化



步骤3: 在虚拟机中安装OpenStack (all-in-one)

本次使用packstack自动安装rocky版本, 准备工作如下:

1. 执行以下命令, 禁止使用NetworkManager, 配置静态IP。

```
systemctl stop NetworkManager && systemctl disable NetworkManager
```

最佳实践

```
vim /etc/sysconfig/network-scripts/ifcfg-ens33
```

在配置文件中修改IP如下图后保存。

```
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="statics"
IPADDR=192.168.43.128
NETMASK=255.255.255.0
GATEWAY=192.168.43.2
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="ens33"
UUID="5a41ef4c-e3fd-4ed9-86cc-bc374c496ef0"
DEVICE="ens33"
ONBOOT="yes"
```

2. 执行以下命令，修改主机名。

```
hostname set-hostname controller
```

3. 如下图所示，将主机名与IP映射写入hosts。

```
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.43.128 controller
```

4. 执行以下命令，禁用firewalld。

```
hostname set-hostname controller
systemctl stop firewalld && systemctl disable firewalld
setenforce 0
```

最佳实践

5. 修改selinux配置文件，将SELINUX设置为disabled。

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#   enforcing - SELinux security policy is enforced.
#   permissive - SELinux prints warnings instead of enforcing.
#   disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#   targeted - Targeted processes are protected,
#   minimum - Modification of targeted policy. Only selected processes are protected.
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

6. 执行以下命令，配置yum源。

```
[root@controller yum.repos.d(keystone_admin)]# cat CentOS-Base.repo
[base]
name=CentOS-$releasever - Base - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/$releasever/os/$basearch/
        http://mirrors.aliyuncs.com/centos/$releasever/os/$basearch/
        http://mirrors.cloud.aliyuncs.com/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7

#released updates
[updates]
name=CentOS-$releasever - Updates - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/$releasever/updates/$basearch/
        http://mirrors.aliyuncs.com/centos/$releasever/updates/$basearch/
        http://mirrors.cloud.aliyuncs.com/centos/$releasever/updates/$basearch/
gpgcheck=1
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7

#additional packages that may be useful
[extras]
name=CentOS-$releasever - Extras - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/$releasever/extras/$basearch/
        http://mirrors.aliyuncs.com/centos/$releasever/extras/$basearch/
        http://mirrors.cloud.aliyuncs.com/centos/$releasever/extras/$basearch/
gpgcheck=1
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7

#additional packages that extend functionality of existing packages
[centosplus]
```

最佳实践

```
name=CentOS- $\$$ releasever - Plus - mirrors.aliyun.com
failovermethod=priority
baseurl=http://mirrors.aliyun.com/centos/ $\$$ releasever/centosplus/ $\$$ basearch/
    http://mirrors.aliyuncs.com/centos/ $\$$ releasever/centosplus/ $\$$ basearch/
    http://mirrors.cloud.aliyuncs.com/centos/ $\$$ releasever/centosplus/ $\$$ basearch/
gpgcheck=1
enabled=0
gpgkey=http://mirrors.aliyun.com/centos/RPM-GPG-KEY-CentOS-7

#contrib - packages by Centos Users
[contrib]
name=CentOS- $\$$ releasever - Contrib - mirrors.aliyun.com

[root@controller yum.repos.d(keystone_admin)]# cat epel.repo
[epel]
name=Extra Packages for Enterprise Linux 7 -  $\$$ basearch
# It is much more secure to use the metalink, but if you wish to use a local mirror
# place its address here.
#baseurl=http://download.example/pub/epel/7/ $\$$ basearch
metalink=https://mirrors.fedoraproject.org/metalink?repo=epel-7&arch= $\$$ basearch&infra=
 $\$$ infra&content= $\$$ contentdir
failovermethod=priority
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-7

[epel-debuginfo]
name=Extra Packages for Enterprise Linux 7 -  $\$$ basearch - Debug
# It is much more secure to use the metalink, but if you wish to use a local mirror
# place its address here.
#baseurl=http://download.example/pub/epel/7/ $\$$ basearch/debug
metalink=https://mirrors.fedoraproject.org/metalink?repo=epel-debug-7&arch=
 $\$$ basearch&infra= $\$$ infra&content= $\$$ contentdir
failovermethod=priority
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-7
gpgcheck=1

[epel-source]
name=Extra Packages for Enterprise Linux 7 -  $\$$ basearch - Source
# It is much more secure to use the metalink, but if you wish to use a local mirror
# place it's address here.
#baseurl=http://download.example/pub/epel/7/source/tree/
metalink=https://mirrors.fedoraproject.org/metalink?repo=epel-source-7&arch=
 $\$$ basearch&infra= $\$$ infra&content= $\$$ contentdir
failovermethod=priority
```

```
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-7
gpgcheck=1
```

```
[root@controller yum.repos.d(keystone_admin)]# cat CentOS-OpenStack-rocky.repo
[centotack-rocky]
name=openstack-rocky
baseurl=https://mirrors.aliyun.com/centos/7/cloud/x86_64/openstack-rocky/
enabled=1
gpgcheck=0
[qume-kvm]
name=qemu-kvm
baseurl=https://mirrors.aliyun.com/centos/7/virt/x86_64/kvm-common/
enabled=1
gpgcheck=0
```

7. 执行以下命令，安装packstack。

```
#安装leatherman
yum -y install leatherman
#安装packstack
yum -y install openstack-utils openstack-packstack
```

8. 执行以下命令，生成应答文件并编辑配置文件。

```
packstack --gen-answer-file=/root/openstack.ini
vim openstack.ini
#除修改以下所列项外，其他项不保持变（密码字段，可根据需要自行修改，这里全部配置为了123456）
CONFIG_DEFAULT_PASSWORD=123456
CONFIG_AODH_INSTALL=n
CONFIG_MARIADB_USER=root
CONFIG_MARIADB_PW=123456
CONFIG_KEYSTONE_DB_PW=123456
CONFIG_KEYSTONE_ADMIN_EMAIL=root@localhost
CONFIG_KEYSTONE_ADMIN_USERNAME=admin
CONFIG_KEYSTONE_ADMIN_PW=123456
CONFIG_NEUTRON_ML2_TYPE_DRIVERS=vxlan,flat
CONFIG_NEUTRON_ML2_TENANT_NETWORK_TYPES=vxlan
CONFIG_NEUTRON_ML2_MECHANISM_DRIVERS=openvswitch
CONFIG_NEUTRON_L2_AGENT=openvswitch
CONFIG_NEUTRON_OVS_BRIDGE_MAPPINGS=extnet:br-ex
CONFIG_NEUTRON_OVS_BRIDGE_IFACES=br-ex:eth0
CONFIG_NEUTRON_OVS_EXTERNAL_PHYSNET=extnet
CONFIG_PROVISION_DEMO=n
```

9. 执行以下命令，通过应答文件进行安装。

```
packstack --answer-file=/root/openstack.ini
```

最佳实践

10. 打开浏览器，输入http://IP，本示例为http://192.168.43.128/，登录OpenStack管理页面。



11. 执行以下命令，导入镜像。

```
openstack image create "centos" --file centos.img --disk-format qcow2 --container-format bare --public
```

12. 执行以下命令，创建网络。

```
openstack create net vnet
openstack network create --provider-network-type=flat --provider-physical-network=extnet --external --share ext_net
```

13. 执行以下命令，创建子网。

```
openstack subnet create subnet1 --network int --subnet-range 172.16.1.0/24 --gateway 172.16.1.1
openstack subnet create ex --network ext_net --gateway 192.168.101.254 --allocation-pool start=192.168.101.200,end=192.168.101.205 --subnet-range 192.168.101.0/24
```

14. 执行以下命令，创建路由。

```
openstack router create route
openstack router add subnet route ex
openstack router add subnet route subnet1
```

15. 执行以下命令，创建规格。

```
openstack flavor create --vcpus 1 --ram 512 --disk 10 test1
```

16. 执行以下命令，创建密钥。

```
openstack keypair create cy
```

最佳实践

17. 执行以下命令，创建虚拟机。

```
openstack server create --flavor test1 --image centos --nic net-id=cc9d7710-d82e-4109-9412-56ae02a18d7d --security-group default --key-name cy test
```

18. 执行以下命令，连接虚拟机并上传qemu-ga、cloudinit以及virtio组件。

下载路径在“镜像服务”>“私有镜像”>“创建私有镜像”页面中连接下载。

创建私有镜像

镜像类型: 系统盘镜像 | 数据盘镜像

镜像源: 云主机 | 镜像文件 | 云主机快照

1. 创建镜像前，请先认真阅读《导入私有镜像用户操作指南》并确保按照文档中步骤制作私有镜像文件。
2. 目前支持RAW、qcow2、vmdk、vhd格式的镜像文件的上传，如果想要尽快创建镜像，建议使用qcow2格式镜像，其他类型文件请做格式转换，格式转换工具下载：《格式转换工具》。
3. Linux检测工具下载：《镜像检测工具》、《cloud-init》、《virtio》、《qga》。
4. Windows镜像工具下载：《cloudbase-init》、《virtio》、《qga》。

镜像文件地址:

链接必须是当前资源池对象存储（原生版）I型产品中镜像的URL地址；如未上传过镜像文件，请先去对象存储（原生版）I型产品中上传镜像文件
镜像文件链接可以到对应的对象存储桶里面文件详情页查看或复制

操作系统: Windows | 2008 standard

系统架构: x86_64

取消 | 下一步

本地磁盘 (D:) > software > Linux系统驱动合集 > Linux系统驱动合集

在Linux系统驱动合集中搜索

名称	修改日期	类型	大小
configure-cloudinit.tar.gz	2020/7/7 17:10	WinRAR 压缩文件管理器	3 KB
image_check.sh	2020/7/9 17:46	SH 源文件	23 KB
linux安装qemu-ga.docx	2020/7/7 17:11	Microsoft Word 文档	1,242 KB
linux安装virtio驱动.docx	2020/7/7 17:03	Microsoft Word 文档	390 KB
qga-linux.tar.gz	2020/7/7 13:57	WinRAR 压缩文件管理器	1,252 KB
镜像检测工具使用说明.docx	2020/7/9 17:46	Microsoft Word 文档	47 KB
转换镜像格式.docx	2020/7/7 14:27	Microsoft Word 文档	15 KB
自定义镜像cloud-init安装文档.docx	2020/7/9 17:46	Microsoft Word 文档	30 KB

类型: Microsoft Word 文档
作者: 赵云飞
大小: 20.4 KB

根据说明文档安装后关闭虚拟机。

19. 执行以下命令，将云主机转为image。

```
openstack server image create 1b1d3411-b755-4980-9da6-db01d8331600 --name cytesting
```

20. 执行以下命令，将image保存到本地（此时的镜像为raw格式）。

```
openstack server image create 1b1d3411-b755-4980-9da6-db01d8331600 --name cytesting
```

21. 执行以下命令，将镜像转换为qcow2格式（防止镜像过大，上传到桶失败）。

```
qemu-img convert -f raw -o qcow2 cytesting centos.qcow2
```

最佳实践

22. 将镜像上传到天翼云。

进入对象存储原生版，创建桶后，上传镜像到桶中（镜像较大建议通过oss api上传）。



任务列表

⚠️ 刷新或关闭浏览器会中断当前的上传任务，并清除统计数据

删除

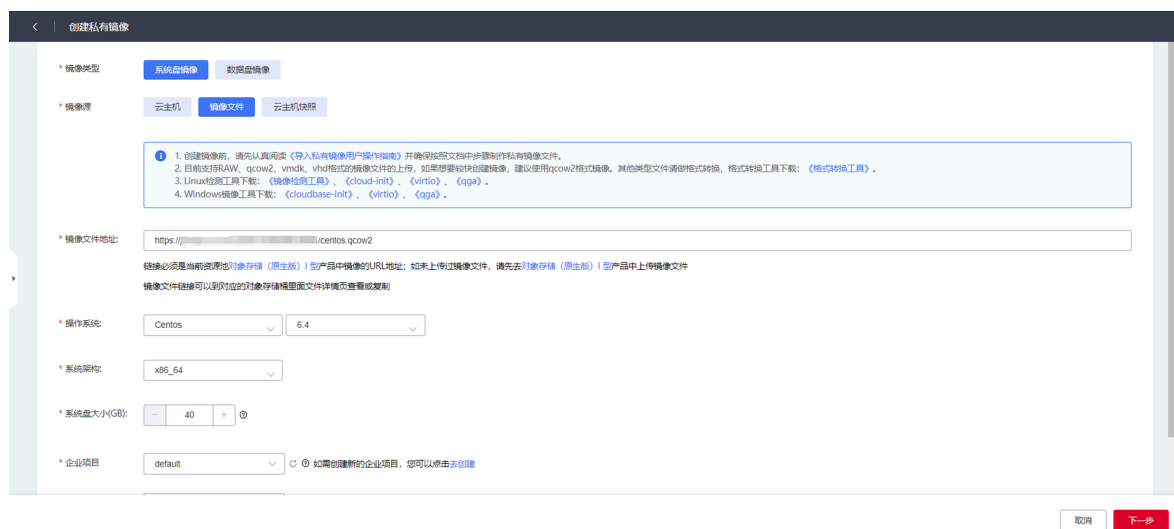
请输入内容

<input type="checkbox"/>	文件名称	读写权限	存储类型	更新时间	文件大小	状态	进度	操作
<input type="checkbox"/>	centos...	私有	标准存储	2023-09...	--	合并文件...	100%	删除

23. 创建私有镜像。

通过“镜像服务”>“私有镜像”>“创建私有镜像”，选择镜像文件并填写对象的地址，然后转为私有镜像。

其中，地址可以在对象存储控制台中的“更多”>“复制URL”选项中复制获得。



24. 创建成功后，“弹性云主机”>“创建云主机”，镜像处选择私有镜像。创建完成后可进入云主机查看数据完整性。

基于Tomcat构建Java web环境（CentOS 7.4）

应用场景

Tomcat是一个被广泛使用的Java Web应用服务器。本文介绍了在天翼云弹性云主机上部署Java Web环境的操作步骤。首先需要下载部署Java Web环境所需的安装包，并将安装包上传至云主机，然后设置弹性云主机安全组规则，再安装并配置相关软件，完成开发环境的配置。

本文使用的云主机以天翼云CentOS 7.4 64bit操作系统云主机为例。

最佳实践

方案架构

图1 基于Tomcat构建Java web环境示意图

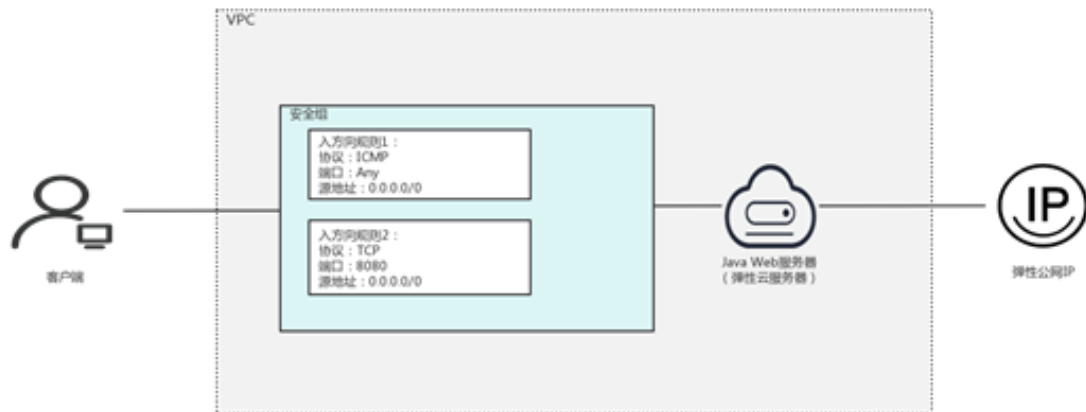


表1 资源和成本规划

资源	资源说明	成本说明
虚拟私有云VPC	VPC网段: 192.168.0.0/16	免费
虚拟私有云子网	可用区: 可用区1 子网网段: 192.168.0.0/24	免费
安全组	入方向规则1: <ul style="list-style-type: none">协议/应用: ICMP端口: Any源地址: 0.0.0.0/0 入方向规则2: <ul style="list-style-type: none">协议/应用: TCPTCP端口: 8080源地址: 0.0.0.0/0	免费
弹性云主机	计费模式: 包年/包月 可用区: 可用区1 规格: s6.large.2 镜像: CentOS 7.4 64bit 系统盘: 40G 弹性公网IP: 现在购买 公网带宽: 按流量计费 带宽大小: 5Mbit/s	ECS涉及以下几项费用: 云主机、云硬盘、弹性公网IP。具体的计费方式及标准请参考 计费说明 。

最佳实践

资源	资源说明	成本说明
jdk	Java开发工具软件 获取方 式： http://www.oracle.com/technetwork/java/javase/downloads	-
tomcat	一款开源的Web应用服务器 获取方 式： http://tomcat.apache.org/download-80.cgi	-
PuTTY	跨平台远程访问工具，用于在软件安装过程中，Windows系统上访问云主机 获取方式： https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html	-
WinSCP	跨平台文件传输工具，用于Windows系统和Linux系统间传输文件 获取方式： http://winscp.net/	-

说明

表1中为jdk和tomcat官方获取地址，您还可以参考其他开源镜像地址获取安装包。

实施步骤（手动）

安装前准备

- 创建弹性云主机，且弹性云主机已绑定弹性公网IP。
- 为了更好的获取和更新系统和软件，建议您更新镜像源为天翼云镜像源，详细操作，请参见[如何使用天翼云提供的内网yum源](#)。
- 登录弹性云主机，执行如下命令，新建jdk目录和tomcat目录。

```
cd /home/  
mkdir webDemo  
cd webDemo/  
mkdir jdk  
mkdir tomcat
```

- 您可以选择将安装包下载至本地后使用文件传输工具将安装包上传至云主机。或者选择使用wget命令直接下载安装包至云主机。
 - 方法一：使用文件传输工具上传安装包至云主机。
 - 使用WinSCP工具上传jdk软件包至云主机jdk文件夹。
 - 使用WinSCP工具上传tomcat软件包至云主机tomcat文件夹。
 - 方法二：使用wget命令直接下载安装包至云主机。
 - 执行如下命令，进入jdk目录。

```
cd /home/webDemo/jdk
```

- 执行如下命令，下载jdk软件包。软件包下载地址请参考表1查询jdk下载地址，或者使用其他开源镜像地址获取安装包。

wget jdk软件包下载地址

例如：单击开源镜像地址，查询当前可用版本，以v8.5.93版本安装包为例，执行以下命令。

```
wget
```

```
https://dlcdn.apache.org/tomcat/tomcat-8/v8.5.93/bin/apache-tomcat-8.5.93.tar.gz --no-check-certificate
```

- 执行如下命令，进入tomcat目录。

```
cd /home/webDemo/tomcat
```

- 执行如下命令，下载tomcat软件包。

请参考表1查询tomcat下载地址，或者使用其他开源镜像地址获取安装包。

```
wget
```

```
https://dlcdn.apache.org/tomcat/tomcat-x/vx.x.xx/bin/apache-tomcat-x.x.xx.tar.gz
```

例如：单击开源镜像地址，查询当前可用版本，以v8.5.93版本安装包为例，执行以下命令。

```
wget
```

```
https://dlcdn.apache.org/tomcat/tomcat-8/v8.5.93/bin/apache-tomcat-8.5.93.tar.gz --no-check-certificate
```

设置弹性云主机安全组规则

1. 单击弹性云主机名称，查看弹性云主机详情，在弹性云主机详情页面，选择“安全组”。
2. 在“安全组”界面，进入安全组详情界面。
3. 在安全组详情界面，单击“添加规则”，弹出添加规则窗口。

最佳实践

4. 根据界面提示配置安全组规则。部署JavaWeb环境需为弹性云主机添加两个安全组规则。
 - a. 为云主机添加ICMP安全组规则。如果云主机默认设置是禁止ICMP规则，当ping弹性服务器IP时会显示超时。因此首先为云主机添加ICMP规则。

图3 添加ICMP规则

添加规则 ✕

* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略: 允许 拒绝

* 协议

* 类型

* 源地址: IP 安全组

?

描述

注意

天翼云云主机默认设置是放行ICMP规则。

- b. 为云主机添加web项目分配端口的访问规则，以8080 端口为例。

图4 添加8080端口

添加规则



* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略: 允许 拒绝

* 协议

* 端口范围

* 源地址: IP 安全组

?

描述

确定

取消

安装jdk

1. 执行如下命令，进入jdk目录。

```
cd /home/webDemo/jdk
```

2. 解压jdk安装包到jdk目录下。

```
tar -xvf jdk-17_linux-x64_bin.tar.gz -C /home/webDemo/jdk/
```

3. 配置环境变量。

```
vim /etc/profile
```

4. 在底部添加以下内容。

```
#set java environment
JAVA_HOME=/home/webDemo/jdk/jdk-17.0.x
JRE_HOME=$JAVA_HOME
PATH=JAVA_HOME/bin:JAVAHOME/bin:PATH
CLASSPATH=.:JAVA_HOME/lib/dt.jar:JAVAHOME/lib/dt.jar:JRE_HOME/lib/tools.jar
```

最佳实践

```
export JAVA_HOME JRE_HOME PATH CLASSPATH
```

说明

“jdk-17.0.x”表示jdk安装包的具体版本，实际值需要从步骤2的返回值中获取。
例如：jdk-17.0.8。

5. 执行以下命令保存并退出。

```
:wq
```

6. 执行以下命令使/etc/profile里的配置生效。

```
source /etc/profile
```

7. 验证安装。

```
java -version
```

回显信息如下所示，验证安装jdk成功。

```
[root@ecm-c9e7 ~]# java -version
java version "17.0.8" 2023-07-18 LTS
Java(TM) SE Runtime Environment (build 17.0.8+9-LTS-211)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.8+9-LTS-211, mixed mode, sharing)
[root@ecm-c9e7 ~]#
```

安装tomcat

1. 执行如下命令，进入tomcat目录。

```
cd /home/webDemo/tomcat
```

2. 解压tomcat安装包到tomcat目录下。

```
tar -xvf apache-tomcat-x.x.xx.tar.gz -C /home/webDemo/tomcat/
```

例如：以v8.5.93版本安装包为例，执行以下命令。

```
tar -xvf apache-tomcat-8.5.93.tar.gz -C /home/webDemo/tomcat/
```

3. 进入tomcat的bin目录，执行以下命令安装tomcat。

```
cd /home/webDemo/tomcat/apache-tomcat-x.x.xx/
cd bin/
```

例如：以v8.5.93版本安装包为例，执行以下命令。

```
cd /home/webDemo/tomcat/apache-tomcat-8.5.93/
cd bin/
```

4. 执行如下命令编辑setclasspath.sh脚本。

```
vi setclasspath.sh
```

并在setclasspath.sh脚本底部添加以下内容。

请根据资源和成本规划中jdk的版本号替换如下内容中的jdk版本号。

```
export JAVA_HOME=/home/webDemo/jdk/jdk-17.0.8
```

```
export JRE_HOME=$JAVA_HOME
```

5. 执行如下命令保存后退出。

```
!wq
```

6. 可输入以下命令启动tomcat。

```
./startup.sh
```

7. 执行如下命令查看tomcat进程。

```
ps -ef | grep tomcat
```

若返回如下图所示，表示tomcat启动成功。

图5 查看tomcat进程

```
[root@ecs-c9e7 ~]# ps -ef | grep tomcat
root      13077      1  0 19:50 ?        00:00:14 /home/webDemo/jdk/jdk-17.0.8/bin/java -Djava.util.logging.config.file=/home/webDemo/tomcat/apache-tomcat-8.5.93/conf/logging.properties -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Djdk.tls.spnego.algorithms=2048 -Djava.protocol.handler.pkgs=org.apache.catalina.webresources -Dorg.apache.catalina.security.SecurityListener.IMASK=0021 -Dignore.endorsed.dirs=classpath:/home/webDemo/tomcat/apache-tomcat-8.5.93/bin/bootstrap.jar:/home/webDemo/tomcat/apache-tomcat-8.5.93/bin/tomcat-juli.jar -Dcatalina.base=/home/webDemo/tomcat/apache-tomcat-8.5.93 -Dcatalina.home=/home/webDemo/tomcat/apache-tomcat-8.5.93 -Djava.io.tmpdir=/home/webDemo/tomcat/apache-tomcat-8.5.93/temp org.apache.catalina.startup.Bootstrap start
root      25538  25360  0  22:06 pts/0    00:00:00 grep --color=auto tomcat
```

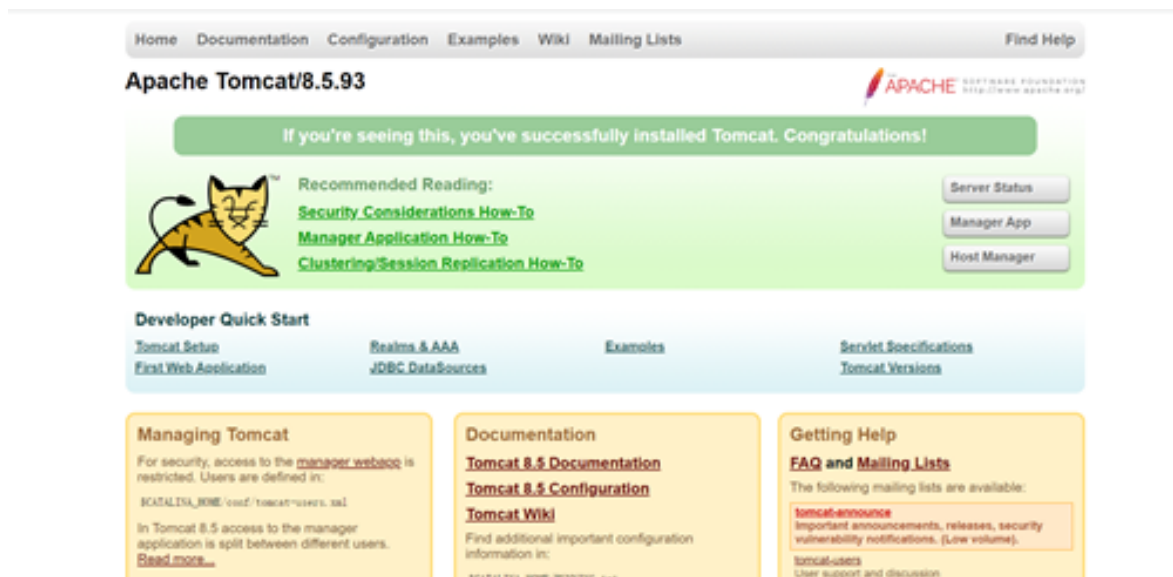
验证Java Web环境搭建完成

浏览器输入以下内容：

http:// 云主机弹性公网IP:8080

如果界面跳转至默认的Tomcat界面，证明Java Web环境搭建完成，可以在公网访问云主机的8080端口。

图6 访问云主机的8080端口



手动部署Java Web

Tomcat是一个被广泛使用的Java Web应用云主机。本文介绍了在天翼云弹性云主机上部署Java Web环境的操作步骤。首先需要下载部署Java Web环境所需的安装包，并将安装包上传至云主机，然后设置弹性云主机安全组规则，再安装并配置相关软件，完成开发环境的配置。

适用对象：本文档适用于使用天翼云弹性云主机部署Java Web环境的用户。

相关软件及工具

软件包名称	获取方式
jdk	http://www.oracle.com/technetwork/java/javase/downloads
tomcat	http://tomcat.apache.org/download-80.cgi

说明

上表中为jdk和tomcat软件包官方获取地址，您还可以参考其他开源镜像地址获取安装包。

工具名称	说明	获取方式
PuTTY	跨平台远程访问工具。用于在软件安装过程中在Windows系统上访问云主机。	https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
WinSCP	跨平台文件传输工具。用于在Windows系统和Linux系统间传输文件。	http://winscp.net/

说明

上表中为PuTTY和WinSCP工具包官方获取地址，您还可以参考其他开源镜像地址获取安装包。

必备事项

1. 创建弹性云主机，且弹性云主机已绑定弹性IP。
2. 登录弹性云主机，执行如下命令，新建jdk目录。

```
cd /home/  
mkdir webDemo  
cd webDemo/  
mkdir jdk
```

3. 登录弹性云主机，执行如下命令，新建tomcat目录。

```
cd webDemo/  
mkdir tomcat
```

4. 您可以选择将安装包下载至本地后使用文件传输工具将安装包上传至云主机。或者选择使用wget命令直接下载安装包至云主机。

- 方法一

- 使用文件传输工具上传安装包至云主机。
- 使用WinSCP工具上传jdk软件包至云主机jdk文件夹。
- 使用WinSCP工具上传tomcat软件包至云主机tomcat文件夹。

- 方法二

- 使用wget命令直接下载安装包至云主机。
 - 执行如下命令，进入jdk目录。

```
cd /home/webDemo/jdk
```

- 执行如下命令，下载jdk软件包。

```
wget https://download.oracle.com/java/17/latest/jdk-17_linux-x64_bin.tar.gz
```

- 执行如下命令，进入tomcat目录。

```
cd /home/webDemo/tomcat
```

- 执行如下命令，下载tomcat软件包。

```
wget https://mirrors.tuna.tsinghua.edu.cn/apache/tomcat/tomcat-8/v8.5.88/bin/apache-tomcat-8.5.88.tar.gz --no-check-certificate
```

说明

本文使用的云主机以天翼云CentOS 7.3 64bit操作系统云主机为例。

- *JDK软件包以jdk-17_linux-x64_bin.tar.gz安装包为例。
- Tomcat以apache-tomcat-8.5.78.tar.gz安装包为例。
- 如果当前操作步骤中下载链接过期，您可以参考表1查询其他版本jdk和tomca下载地址，或者使用其他开源镜像地址获取安装包。

设置弹性云主机安全组规则

安全组规则设置步骤

1. 单击弹性云主机名称，查看弹性云主机详情，在弹性云主机详情页面，选择“安全组”。
2. 在“安全组”界面，单击“更改安全组规则”，进入安全组详情界面。
3. 在安全组详情界面，单击“添加规则”，弹出添加规则窗口。
4. 根据界面提示配置安全组规则。

添加ICMP安全组规则

如果云主机默认设置是禁止ICMP规则，当ping弹性云主机IP时会显示超时。因此首先为云主机添加ICMP规则。

为云主机添加web项目分配端口的访问规则，以8080 端口为例：

最佳实践

添加规则



* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略： 允许 拒绝

* 协议

* 类型

* 源地址： IP 安全组



描述

确定

取消

说明

- 入方向：指从外部访问安全组规则下的弹性云主机。
- 出方向：指安全组规则下的弹性云主机访问安全组外的实例。
- 端口范围：安全组规则的端口范围，取值范围为：1~65535，Any代表所有端口。

其中0.0.0.0/0表示所有IPv4地址，::/0表示所有IPv6地址。

WEB服务端口默认使用得端口为：80（http）、443（https），用户也可选择将Web云主机配置为在非标准端口上运行，如8080，此时需要配置对应的安全组端口。

安装JDK

- 解压jdk安装包到jdk目录下。

```
tar -xvf jdk-8u231-linux-x64.tar.gz -C /home/webDemo/jdk/
```

- 配置环境变量。

```
vi /etc/profile
```

- 在底部添加以下内容。

```
#set java environment
export JAVA_HOME=/home/webDemo/jdk/jdk1.8.0_231
export JRE_HOME=/home/webDemo/jdk/jdk1.8.0_231/jre
export CLASSPATH=.:JAVA_HOME/lib/dt.jar:JAVA_HOME/lib/dt.jar:JRE_HOME/lib/tools.jar
export PATH=JAVA_HOME/bin:JAVA_HOME/bin: $PATH
```

- 执行以下命令保存并退出。

```
:wq
```

- 执行以下命令使/etc/profile里的配置生效。

```
source /etc/profile
```

- 验证安装。

```
java -version
```

回显信息如下所示验证安装jdk成功。

```
[root@ecs-c525-web ~]#
java -version
java version "1.8.0_231"
Java(TM) SE Runtime Environment (build 1.8.0_231-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.231-b11, mixed mode)
```

安装tomcat

- 解压tomcat安装包到tomcat目录下。

```
tar -xvf apache-tomcat-8.5.47.tar.gz -C /home/webDemo/tomcat/
```

- 进入tomcat的bin目录，执行以下命令安装tomcat。

```
cd /home/webDemo/tomcat/apache-tomcat-8.5.47/
cd bin/
```

- 执行如下命令编辑setclasspath.sh 脚本。

```
vi setclasspath.sh
```

并在setclasspath.sh 脚本添加以下内容。

```
export JAVA_HOME=/home/webDemo/jdk/jdk-17.0.3
export JRE_HOME=$JAVA_HOME
```

- 保存后退出，可输入以下命令启动tomcat。

```
./startup.sh
```

最佳实践

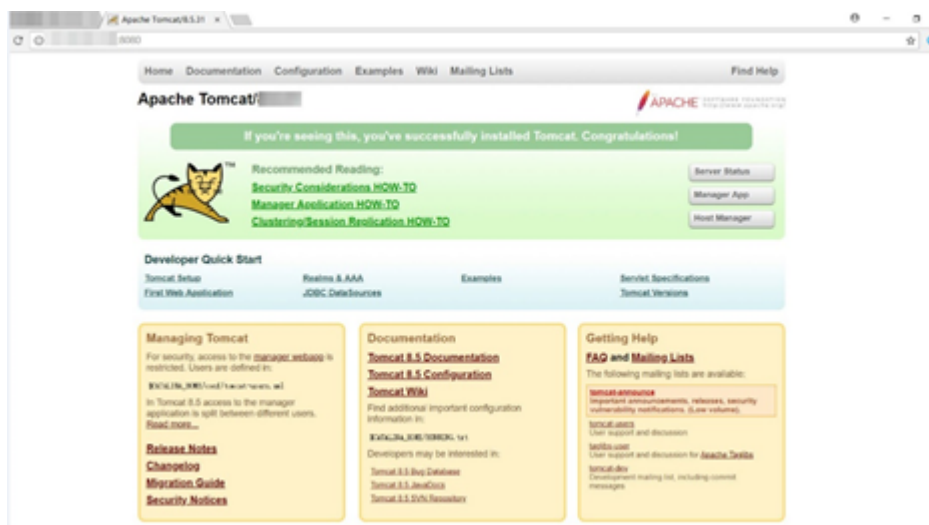
验证Java Web环境搭建完成

- 在浏览器输入以下内容:

http://云主机弹性IP:8080

如果界面跳转至默认的Tomcat界面,证明Java Web环境搭建完成。我们就可以在公网访问云主机的8080端口了。

访问云主机的8080端口如图所示:



注意

天翼云云主机80/8080/443/8443端口需备案完成后才可开通使用,除上述四个端口,云主机其余端口均为开通状态,直接调用即可。但域名+IP+端口号如能通过互联网访问,则属于未备案,只做内部访问无妨。

80/8080/443/8443端口会在备案成功后一个工作日内开通,开通后会电话联系网站负责人。

手动搭建LNMP环境

本文主要介绍了在天翼云上如何使用弹性云主机的Linux实例手工搭建LNMP平台的web环境。该指导具体操作以CentOS 7.2 64位操作系统为例。

操作流程

- 安装nginx。
- 安装MySQL。
- 安装PHP。
- 浏览器访问测试。

前提条件

- 弹性云主机已绑定弹性IP。
- 弹性云主机所在安全组添加了如下表所示的安全组规则,具体步骤参见为安全组添加安全组规则。

最佳实践

方向	协议/应用	端口/范围	源地址
入方向	HTTP(80)	80	0.0.0.0/0

操作步骤

安装nginx。

1. 登录弹性云主机。
2. 执行以下命令，下载对应当前系统版本的nginx包。

```
wget http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-centos-7-0.el7ngx.noarch.rpm
```

3. 执行以下命令，建立nginx的yum仓库。

```
rpm -ivh nginx-release-centos-7-0.el7ngx.noarch.rpm
```

4. 执行以下命令，安装nginx。

```
yum -y install nginx
```

5. 执行以下命令，启动nginx并设置开机启动。

```
systemctl start nginx  
systemctl enable nginx
```

6. 使用浏览器访问“http://服务器IP地址”，显示如下页面，说明nginx安装成功。

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

安装MySQL

1. 依次执行以下命令，安装MySQL。

```
rpm -Uvh http://dev.mysql.com/get/mysql57-community-release-el7-8.noarch.rpm  
yum -y install mysql-community-server
```

2. 依次执行以下命令，启动MySQL服务并设置开机自启动。

```
systemctl start mysqld  
systemctl enable mysqld
```

3. 执行以下命令，获取安装MySQL时自动设置的root用户密码。

```
grep 'temporary password' /var/log/mysqld.log
```

回显如下类似信息。

```
2018-08-29T07:27:37.541944Z 1
```

```
[Note] A temporary password is generated for root@localhost: 2YY?3uHUA?Ys
```

4. 执行以下命令，并按照回显提示信息进行操作，加固MySQL。

```
mysql_secure_installation
```

```
Securing the MySQL server deployment.
```

```
Enter password for user root: #输入上一步骤中获取的安装MySQL时自动设置的root用户密码
```

```
The existing password for the user account root has expired. Please set a new password.
```

```
New password: #设置新的root用户密码
```

```
Re-enter new password: #再次输入密码
```

```
The 'validate_password' plugin is installed on the server.
```

```
The subsequent steps will run with the existing configuration of the plugin.
```

```
Using existing password for root.
```

```
Estimated strength of the password: 100
```

```
Change the password for root ? ((Press y|Y for Yes, any other key for No): N #是否更改root用户密码，输入N
```

```
... skipping.
```

```
By default, a MySQL installation has an anonymous user, allowing anyone to log into MySQL without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.
```

```
Remove anonymous users? (Press y|Y for Yes, any other key for No): Y #是否删除匿名用户，输入Y  
Success.
```

```
Normally, root should only be allowed to connect from
```

```
'localhost'. This ensures that someone cannot guess at the root password from the network.
```

```
Disallow root login remotely? (Press y|Y for Yes, any other key for No): Y #禁止root远程登录，输入Y
```

```
Success.
```

```
By default, MySQL comes with a database named
```

```
'test' that anyone can access. This is also intended only for testing, and should be removed before moving
```

```
Remove test database and access to it? (Press y|Y for Yes, any other key for No): Y #是否删除test库和对它的访问权限，输入Y
```

```
-Dropping test database...
```

```
Success.
```

```
-Removing privileges on test database...
```

```
Success.
```

```
Reloading the privilege tables will ensure that all changes made so far will take effect immediately.
```

Reload privilege tables now? (Press y|Y for Yes, any other key for No): Y #是否重新加载授权表, 输入Y

Success.

All done!

安装PHP

1. 依次执行以下命令, 安装PHP 7和一些所需的PHP扩展。

```
rpm -Uvh https://mirror.webtatic.com/yum/el7/epel-release.rpmrpm -
Uvh https://mirror.webtatic.com/
yum/el7/webtatic-release.rpm
yum -y install php70w-tidy php70w-common php70w-devel php70w-pdo php70w-mysql php70w-
gd php70w-ldap php70w-mbstring php70w-mcrypt php70w-fpm
```

2. 执行以下命令, 验证PHP的安装版本。

```
php -v
```

回显如下类似信息:

```
PHP 7.0.31 (cli) (built: Jul 20 2018 08:55:22) ( NTS )
Copyright (c) 1997-2017 The PHP Group
Zend Engine v3.0.0, Copyright (c) 1998-2017 Zend Technologies
```

3. 执行以下命令, 启动PHP服务并设置开机自启动。

```
systemctl start php-fpm
systemctl enable php-fpm
```

4. 修改nginx配置文件以支持PHP。

执行以下命令打开配置文件“default.conf”。

```
vim /etc/nginx/conf.d/default.conf
```

按i键进入编辑模式。

修改打开的“default.conf”文件。

在所支持的主页面格式中添加php格式的主页, 如下所示:

```
location / {
root /usr/share/nginx/html;
index index.php index.html index.htm;
}
```

取消如下内容的注释, 并设置字体加粗部分为nginx的默认路径, 如下所示:

```
location ~ \.php$ {

root html;

fastcgi_pass 127.0.0.1:9000;
```

最佳实践

```
fastcgi_index index.php;

fastcgi_param SCRIPT_FILENAME /usr/share/nginx/html$fastcgi_script_name;

include    fastcgi_params;

}
```

按Esc键退出编辑模式，并输入 :wq保存后退出。

执行以下命令，重新载入nginx的配置文件。

```
service nginx reload
```

浏览器访问测试

1. /usr/share/nginx/html/目录下创建“info.php”的测试页面。

a. 执行以下命令创建并打开“info.php”的测试文件。


```
vim /usr/share/nginx/html/info.php
```

b. 按i键进入编辑模式。

c. 修改打开的“info.php”文件，将如下内容写入文件。

d. 按Esc键退出编辑模式，并输入:wq保存后退出。

2. 使用浏览器访问 <http://服务器IP地址/info.php>，显示如下页面，说明搭建环境成功。



PHP Version 7.0.31	
System	Linux ecs-5d3f.novalocal 3.10.0-693.11.1.el7.x86_64 #1 SMP Mon Dec 4 23:52:40 UTC 2017 x86_64
Build Date	Jul 20 2018 08:57:28
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/bc2.ini, /etc/php.d/calendar.ini, /etc/php.d/cctype.ini, /etc/php.d/curl.ini, /etc/php.d/exception.ini, /etc/php.d/fileinfo.ini, /etc/php.d/filter.ini, /etc/php.d/gd.ini, /etc/php.d/gettext.ini, /etc/php.d/gmp.ini, /etc/php.d/iconv.ini, /etc/php.d/json.ini, /etc/php.d/libxml.ini, /etc/php.d/mcrypt.ini, /etc/php.d/mysqli.ini, /etc/php.d/openssl.ini, /etc/php.d/pdo_mysql.ini, /etc/php.d/pdo_sqlite.ini, /etc/php.d/phar.ini, /etc/php.d/shmop.ini, /etc/php.d/simplexml.ini, /etc/php.d/sockets.ini, /etc/php.d/sqlite3.ini, /etc/php.d/tidy.ini, /etc/php.d/tokenizer.ini, /etc/php.d/xml.ini, /etc/php.d/zip.ini
PHP API	20151012
PHP Extension	20151012

网络相关最佳实践

网络性能测试的最佳实践

iperf可用于测试两个计算机之间的网络带宽和延迟。它是一个开源项目，提供了用于Windows、Linux和其他操作系统的版本。通过在两台计算机之间运行iPerf服务器和客户端，您可以测量它们之间的网络性能。iPerf客户端发送数据流到服务器，并且服务器测量接收到的数据量和速度，支持在TCP和UDP协议上运行。

本文将详细介绍如何使用 iperf3 工具，测试弹性云主机间网络性能。内容主要包括“测试前准备”、“TCP 带宽测试”、“UDP PPS 测试”。

安装使用

```
[root@ss-daliu1 ~]# sudo yum install iperf3
```

```
[root@ss-daliu2 ~]# sudo yum install iperf3
```

参数说明

iperf常用参数中文说明：（1）-s, -server: iperf服务器模式，默认启动的监听端口为5201, eg: iperf -s （2）-c, -client host: iperf客户端模式, host是server端地址, eg: iperf -c 222.35.11.23 （3）-i, -interval: 指定每次报告之间的时间间隔, 单位为秒, eg: iperf3 -c 192.168.12.168 -i 2 （4）-p, -port: 指定服务器端监听的端口或客户端所连接的端口, 默认是5001端口。（5）-u, -udp: 表示采用UDP协议发送报文, 不带该参数表示采用TCP协议。（6）-l, -len: 设置读写缓冲区的长度, (default 128 KB for TCP, dynamic or 1 for UDP)。通常测试 PPS 的时候该值为16, 测试BPS时该值为1400。（7）-b, -bandwidth [K|M|G]: 指定UDP模式使用的带宽, 单位bits/sec, 默认值是1 Mb/s。（8）-t, -time: 指定数据传输的总时间, 即在指定的时间内, 重复发送指定长度的数据包。默认10秒。（9）-A: CPU亲和性, 可以将具体的iperf3进程绑定对应编号的逻辑CPU, 避免iperf进程在不同的CPU间调度。

通用参数 (Server端和Client端共用): (1) -f, -format [k|m|g|K|M|G]: 指定带宽输出单位, “[k|m|g|K|M|G]” 分别表示以Kb, Mb, Gb, KBytes, MBytes, GBytes显示输出结果, 默认Mb, eg: iperf3 -c 192.168.12.168 -f M (2) -p, -port: 指定服务器端监听的端口或客户端所连接的端口, 默认是5001端口。（3）-i, -interval: 指定每次报告之间的时间间隔, 单位为秒, eg: iperf3 -c 192.168.12.168 -i 2 （4）-F: 指定文件作为数据流进行带宽测试。例如: iperf3 -c 192.168.12.168 -F web-ixdba.tar.gz

Server端专用参数：（1）-s, -server: iperf服务器模式, 默认启动的监听端口为5201, eg: iperf -s （2）-c, -client host: 如果iperf运行在服务器模式, 并且用-c参数指定一个主机, 那么iperf将只接受指定主机的连接。此参数不能工作于UDP模式。（3）-D: Unix平台下将Iperf作为后台守护进程运行。在Win32平台下, Iperf将作为服务运行。（4）-R: 卸载Iperf服务(仅用于Windows)。（5）-o: 重定向输出到指定文件(仅用于Windows)。（6）-P, -parallel: 服务器关闭之前保持的连接数。默认是0, 这意味着永远接受连接。

Client端专用参数：（1）-c, -client host: iperf客户端模式, host是server端地址, eg: iperf -c 222.35.11.23 （2）-u, -udp: 表示采用UDP协议发送报文, 不带该参数表示采用TCP协议。（3）-b, -bandwidth [K|M|G]: 指定UDP模式使用的带宽, 单位bits/sec, 默认值是1 Mbit/sec。（4）-t, -time: 指定数据传输的总时间, 即在指定的时间内, 重复发送指定长度的数据包。默认10秒。（5）-l, -len: 设置读写缓冲区的长度, (default 128 KB for TCP, dynamic or 1 for UDP)。通常测试 PPS 的时候该值为16, 测试BPS时该值为1400。（6）-n, -num [K|M|G]: 指定传输数据包的字节数, 例如: iperf3 -c 192.168.12.168 -n 100M （7）-P, -parallel: 指定客户端与服务端之间使用的线程数。默认是1个线程。需要客户端与服务端同时使用此参数。（8）-w, -window: 指定套接字缓冲区大小, 在TCP方式下, 此设置为TCP窗口的大小。在UDP方式下, 此设置为接受UDP数据包的缓冲区大小, 用来限制可以接收数据包的最大值 （9）-B, -bind: 用来绑定一个主机地址或接口, 这个参数仅用于具有多个网络接口的主机。在UDP模式下, 此参数用于绑定和加入一个多播组。

（10）-M, -mss: 设置TCP最大信息段的值 （11）-N, -nodelay: 设置TCP无延时 （12）-V: 绑定一个IPv6地址。（13）-d, -dualtest: 运行双测试模式。将使服务器端反向连接到客户端, 使用-L参数中指定的端口（或默认使用客户端连接到服务器端的端口）。使用参数-r以运行交互模式。（14）-L, -listenport: 指定服务端反向连接到客户端时使用的端口。默认使用客户端连接至服务端的端口。（15）-r, -tradeoff: 往复测试模式。当客户端到服务器端的测试结束时, 服务器端反向连接至客户端。当客户端连接终止时, 反向连接随即开始。如果需要同时进行双向测试, 请尝试-d参数。

最佳实践

一对一场景

测试准备

类型	数量	镜像	规格
被测机（被测机：被压力测试网络性能的弹性云服务器，可作为 iperf3 测试中的 client 端（发送端））	1台	CTyunOS 23.01（推荐）	4C8G（推荐）
辅助云服务器（作为 iperf3 测试中的 server 端（接收端））	1台	CTyunOS 23.01（推荐）	4C8G（推荐）

准备两台处于不同计算节点的弹性云主机，网络可达，分别安装iperf，添加对应的安全组规则、防护墙规则放行。

测试TCP吞吐量

1. TCP发送带宽。

```
# Server端开启iperf的服务器模式，指定TCP端口：10000，安全组规则入方向放行TCP 10000端口  
[root@ss-daliu2 ~]# iperf3 -s -i 1 -p 10000
```

```
-----  
Server listening on 10000 (test #1)
```

```
# Client端启动iperf的客户端模式，连接服务端。ip为server端地址  
[root@ss-daliu1 ~]# iperf3 -c 192.168.0.15 -i 1 -t 60 -p 10000  
Connecting to host 192.168.0.15, port 10000  
[ 5] local 192.168.0.8 port 37720 connected to 192.168.0.15 port 10000  
[ID] Interval      Transfer  Bitrate    Retr  Cwnd  
[ 5] 0.00-1.00 sec   332 MBytes 2.78 Gbits/sec 4394  167 KBytes  
...  
...  
...  
[ 5] 52.00-53.00 sec   180 MBytes 1.51 Gbits/sec 6546  130 KBytes  
[ 5] 53.00-54.00 sec   166 MBytes 1.39 Gbits/sec 5237  180 KBytes  
[ 5] 54.00-55.00 sec   166 MBytes 1.40 Gbits/sec 5505  19.8 KBytes  
[ 5] 55.00-56.00 sec   207 MBytes 1.74 Gbits/sec 7651  12.7 KBytes  
[ 5] 56.00-57.00 sec   167 MBytes 1.40 Gbits/sec 5822  314 KBytes  
[ 5] 57.00-58.00 sec   168 MBytes 1.41 Gbits/sec 5952  58.0 KBytes  
[ 5] 58.00-59.00 sec   178 MBytes 1.49 Gbits/sec 6940  372 KBytes  
[ 5] 59.00-60.00 sec   205 MBytes 1.72 Gbits/sec 6756  2.83 KBytes
```

```
-----  
[ID] Interval      Transfer  Bitrate    Retr  
[ 5] 0.00-60.00 sec  10.7 GBytes 1.53 Gbits/sec 343419          sender
```

```
[ 5] 0.00-60.00 sec 10.7 GBytes 1.53 Gbits/sec receiver
```

结果说明：

Interval表示时间间隔。

Transfer表示时间间隔里面传输的数据量。

Bitrate是时间间隔里的传输速率。

2. TCP接收带宽。

```
# Server端开启iperf的服务器模式，指定TCP端口：10000，安全组规则入方向放行TCP 10000端口  
[root@ss-daliu2 ~]# iperf3 -s -i 1 -p 10000
```

```
-----  
Server listening on 10000 (test #1)
```

```
# Client端启动iperf的客户端模式，连接服务端。ip为server端地址  
[root@ss-daliu1 ~]# iperf3 -c 192.168.0.15 -i 1 -t 60 -p 10000 -R  
Connecting to host 192.168.0.15, port 10000  
Reverse mode, remote host 192.168.0.15 is sending  
[ 5] local 192.168.0.8 port 38316 connected to 192.168.0.15 port 10000  
[ ID] Interval      Transfer  Bitrate  
[ 5] 0.00-1.00 sec  336 MBytes 2.81 Gbits/sec  
[ 5] 1.00-2.00 sec  172 MBytes 1.45 Gbits/sec  
...  
...  
...  
[ 5] 59.00-60.00 sec  179 MBytes 1.50 Gbits/sec
```

```
-----  
[ ID] Interval      Transfer  Bitrate  Retr          sender  
[ 5] 0.00-60.00 sec  10.7 GBytes 1.53 Gbits/sec 360910  
[ 5] 0.00-60.00 sec  10.7 GBytes 1.53 Gbits/sec receiver
```

结果说明：

Interval表示时间间隔。

Transfer表示时间间隔里面传输的数据量。

Bitrate是时间间隔里的传输速率。

3. TCP上下行带宽测试（双向传输）。

```
# Server端开启iperf的服务器模式，指定TCP端口：10000，安全组规则入方向放行TCP 10000端口  
[root@ss-daliu2 ~]# iperf3 -s -i 1 -p 10000
```

```
-----  
Server listening on 10000 (test #1)
```

```
# Client端启动iperf的客户端模式，连接服务端。ip为server端地址  
[root@ss-daliu1 ~]# iperf3 -c 192.168.0.15 -i 1 -d -t 60 -p 10000  
...  
Thread FD 5 stopped  
Sender threads stopped
```

最佳实践

```
tcpi_snd_cwnd 335 tcpi_snd_mss 1448 tcpi_rtt 1054
send_results
{
  "cpu_util_total": 4.7314293702543742,
  "cpu_util_user": 0.0694572749595791,
  "cpu_util_system": 4.6619720952947947,
  "sender_has_retransmits": 1,
  "congestion_used": "cubic",
  "streams": [{
    "id": 1,
    "bytes": 11446517760,
    "retransmits": 354575,
    "jitter": 0,
    "errors": 0,
    "omitted_errors": 0,
    "packets": 0,
    "omitted_packets": 0,
    "start_time": 0,
    "end_time": 60.00129
  }]
}
get_results
{
  "cpu_util_total": 3.3835424728514676,
  "cpu_util_user": 0.14802798323701713,
  "cpu_util_system": 3.2355161562396457,
  "sender_has_retransmits": -1,
  "congestion_used": "cubic",
  "streams": [{
    "id": 1,
    "bytes": 11442323456,
    "retransmits": -1,
    "jitter": 0,
    "errors": 0,
    "omitted_errors": 0,
    "packets": 0,
    "omitted_packets": 0,
    "start_time": 0,
    "end_time": 60.00149
  }]
}
interval_len 1.000265 bytes_transferred 176291840
interval forces keep
[ 5] 59.00-60.00 sec 168 MBytes 1.41 Gbits/sec 5895 474 KBytes
-----
[ ID] Interval      Transfer  Bitrate  Retr
```

最佳实践

```
[ 5] 0.00-60.00 sec 10.7 GBytes 1.53 Gbits/sec 354575      sender
[ 5] 0.00-60.00 sec 10.7 GBytes 1.53 Gbits/sec           receiver
Receiver threads stopped
```

iperf Done.

测试4线程TCP吞吐量, 如果没有指定发送方式, iperf客户端只会使用单线程。

```
$ iperf3 -c 172.16.0.9 -i 1 -P 4 -t 60 -p 10000
```

结果说明:

Interval表示时间间隔。

Transfer表示时间间隔里面传输的数据量。

Bitrate是时间间隔里的传输速率。

测试UDP吞吐量

带宽测试通常采用UDP模式, 因为能测出极限带宽、时延抖动、丢包率。在进行测试时, 首先以链路理论带宽作为数据发送速率进行测试, 例如, 从客户端到服务器之间的链路的理论带宽为100Mbps, 先用-b 100M进行测试, 然后根据测试结果(包括实际带宽, 时延抖动和丢包率), 再以实际带宽作为数据发送速率进行测试, 会发现时延抖动和丢包率比第一次好很多, 重复测试几次, 就能得出稳定的实际带宽。

Server端开启iperf的服务器模式, 指定UDP端口:

```
[root@ss-daliu2 ~]# iperf3 -s -i 1 -p 10001
```

```
-----
Server listening on 10001
-----
```

Client端启动iperf的客户端模式, 连接服务端

```
[root@ss-daliu1 ~]# iperf3 -u -c 192.168.0.15 -b 100m -t 60 -p 10001
```

```
Connecting to host 172.16.0.9, port 10001
```

```
[ 4] local 172.16.0.2 port 46011 connected to 172.16.0.9 port 10001
```

```
[ ID] Interval      Transfer  Bandwidth  Total Datagrams
```

```
[ 4] 0.00-1.00 sec 10.8 MBytes 90.3 Mbits/sec 7799
```

```
...
```

```
[ 4] 58.00-59.00 sec 11.9 MBytes 100 Mbits/sec 8640
```

```
[ 4] 59.00-60.00 sec 11.9 MBytes 99.9 Mbits/sec 8625
```

```
-----
[ ID] Interval      Transfer  Bitrate    Jitter  Lost/Total Datagrams
```

```
[ 4] 0.00-60.00 sec 714 MBytes 99.8 Mbits/sec 0.001 ms 133914/517118 (26%)
```

```
[ 4] Sent 517118 datagrams
```

iperf Done.

测试多线程UDP吞吐量

```
$ iperf3 -u -c 192.168.0.15 -b 5m -P 4 -t 60 -p 10001
```

进行上下行带宽测试(双向传输)

最佳实践

```
$ iperf3 -u -c 192.168.0.15 -b 100M -d -t 60 -p 10001
```

结果说明： jitter为抖动，在连续传输中的平滑平均值差。 Lost为丢包数量。 Total Datagrams为包数量。

iperf测试udp丢包严重：可以尝试增大收发包缓冲区。

```
echo 16777216 > /proc/sys/net/core/rmem_max
echo 16777216 > /proc/sys/net/core/rmem_default
echo 16777216 > /proc/sys/net/core/wmem_max
echo 16777216 > /proc/sys/net/core/wmem_default
```

多流打流测试方案（针对大规格云主机）

针对大规格云主机，单进程打流可能无法发挥出云主机的性能，可以参考下述多流打流测试方案。

虚机网络性能说明

虚机从同#个队列发出的报文，在底层只能被#个cpu转发。即虚机#个队列的发包性能 <= 虚拟交换机单核转发性能。

性能测试步骤

1. 确定两个#于性能测试的计算节点: A, B。
2. 在每个计算节点上分别创建2个虚机，虚机规格尽量大，虚机内核版本4.19及以上(建议使用CTyunOS 23.01)。
3. 4个测试虚机内安装iperf3包。

```
yum install iperf3
```

4. B计算节点上的两个虚机(B1, B2)分别起iperf服务端。

```
for i in {0..63}; do iperf3 -s -D -p $((5200 + $i)); done
```

5. A计算节点上两个虚机(A1, A2)起iperf客户端(A1给B1发包, A2给B2发包),

创建发包脚本:

```
#!/usr/bin/env bash
flow_per_thread=2
max_thread=64
run_time=60
udp_opts="-u"
# 默认用全力(100G)发送udp报文, 但可能存在丢包.
udp_band_Mb=100000
tcp_opts=""
use_tcp=1
usage() {
    echo '
usage:
    $0 -s <server-ip> [--pps] [-n <nprocess>] [-t <time>] [--tcp]
```

```
-s      Iperf server ip address.
```

```
--pps
```

Send small pkts for pps benchmark. By default send large pkts for bandwidth benchmark.

最佳实践

```
-n      Set max process number. If not set, tx queue number is used.
-t      time in seconds to transmit for, default 60 seconds.
--udp   Send udp instead of tcp(default). Udp is good for some special scenarios.
-b      Bandwidth to send for udp mode. i.e. 1M, 2G,
,
}
while [ $# -gt 0 ]; do
  case $1 in
    -s | --server)
      server="$2"
      shift
      ;;
    -n | --nprocess)
      max_thread="$2"
      shift
      ;;
    -t | --time)
      run_time="$2"
      shift
      ;;
    --pps)
      # 144 bytes
      tcp_opts="$tcp_opts -M 90"
      # 18(eth) + 28(ip) + 18(udp) = 64
      udp_opts="$udp_opts -l 18"
      ;;
    --udp)
      use_tcp=0
      ;;
    -b)
      udp_band="$2"
      _unit=$(echo $udp_band | grep -oiP '\d+\K[mg]')
      _num=$(echo $udp_band | grep -oP '\d+')
      if [ -z "$_unit" ] || [ -z "$_num" ];then
        echo "bad bandwidth"
        exit 1
      fi
      if [ ${_unit,,} == 'g' ];then
        _num=$(( _num*1024 ))
      fi
      udp_band_Mb=$_num

      ;;
    -h | --help)
      usage
      exit 0
      ;;
```

最佳实践

```
*)
    usage
    exit 1
;;
esac
shift
done

interface=$(ip route get $server |grep -oP "(?<=dev )\S+")
n_txq=$(ls /sys/class/net/$interface/queues/tx-*/xps_cpus | wc -l)
band_per_flow=$((udp_band_Mb/flow_per_thread/max_thread))
i=0
for fxps in `ls /sys/class/net/$interface/queues/tx-*/xps_cpus | sort`
do
    aff=$(cat $fxps)
    if ((i >= max_thread));then
        break
    fi
    no_zero_aff=$(echo "$aff" | tr -d '0xX')
    if [ -z "$no_zero_aff" ];then
        _aff=$((1 << $i))
        aff=$(printf "%x" $_aff)
        echo $aff > $fxps
    fi

    # Affinity on only one core
    suffix=""
    for (( j=${#aff}-1; j>=0; j-- )); do
        if [ "${aff:$j:1}" == "," ];then
            continue
        fi
        if [ "${aff:$j:1}" != "0" ];then
            aff_one="${aff:$j:1}${suffix}"
            break
        fi
        suffix+="0"
    done
    if [ -z "$aff_one" ];then
        echo "Failed to find txq affinity core"
        exit 1
    fi
    aff_one=`echo $aff_one | tr '13579bdf' '1' | tr '26ae' '2' | tr '4c' '4'

    if ((use_tcp > 0));then
        taskset $aff_one iperf3 -c $server $tcp_opts -P 2 -p $((5200+$i)) -t $run_time &
    else
```

最佳实践

```
taskset $aff_one iperf3 -c $server $udp_opts -b ${udp_band_Mb}M -P 2 -p $((5200+$i)) -t
$run_time &
fi
((i++))
done
wait
```

测试pps(144B/64B小包):

```
bash iperf-client.sh -s <Bl-ip> --pps -t 300
```

测试带宽(1500B大包):

```
bash iperf-client.sh -s <Bl-ip> -t 300
```

6. 打流的过程中虚拟交换机会根据流量做balance，所以我们流量测试应持续5分钟。等待底层balance到最佳状态，以得到较好的性能结果。

ECS安全组实践（入方向规则）

弹性云主机是天翼云提供的云服务器。安全组（Security Group）是用于设置云服务器实例的网络访问控制的虚拟防火墙，一般是作为流量访问白名单存在，以下是一些ECS安全组的最佳实践：

开放原则

- 默认拒绝所有流量：新建的安全组规则默认情况下拒绝所有入站和出站流量，这是一种良好的安全实践。
- 仅开放必要的端口和协议，避免使用0.0.0.0/0规则：根据应用需求，只开放必要的端口和协议，例如HTTP（80端口）、HTTPS（443端口）等。避免开放不必要的端口，以减少攻击面。

分组原则

- 根据应用需求进行分组：根据应用程序或服务的需求，可以将安全组规则进行分组。例如，将Web服务器相关的规则放在一个Web层安全组中，数据库服务器相关的规则放在另一个Database层安全组中。这样可以更好地管理和组织安全组规则，暴露不同的出入规则和权限。
- 避免过度复杂化：尽量避免创建过多的安全组，以免管理和维护变得复杂。根据实际情况合理划分安全组，保持简洁和易于管理。

授权原则

- 基于最小权限原则：为安全组授权时，应遵循最小权限原则，仅授予实例所需的访问权限。只开放必要的端口和IP地址范围，避免授权过度，减少潜在的安全风险。
- 限制访问来源：根据实际需求，限制访问来源的IP地址或IP地址段。仅允许特定的IP地址或IP地址段访问云主机，以增加安全性。

安全原则

- 定期审查和更新安全组规则：定期审查安全组规则，确保只有必要的端口和IP地址被允许访问。如果某些规则不再需要或存在安全风险，及时进行更新和删除。
- 启用日志记录和监控：开启安全组的日志记录功能，对网络流量进行审计和监控。及时发现潜在的安全问题，并采取相应的措施。
- 使用网络安全组合ACL：结合使用网络安全组合ACL（Access Control List），可以在更细粒度的网络层面控制流量，提供额外的安全层。

变更安全组规则

用户可以参照[配置安全组规则](#)来进行安全组规则的配置，如果要变更安全组规则，需要注意变更安全组规则可能会影响用户实例间的网络通信，通常我们会选择放行必要的实例，再执行安全组策略收紧变更来保证必要的网络通信。

- 将需要互通访问的实例加入到一个新建的安全组，再执行变更操作。
- 如果授权类型为安全组访问，则将需要互通访问的对端实例所绑定的安全组ID添加为授权对象。
- 如果授权类型为地址段访问，则将需要互通访问的对端实例内网IP添加为授权对象。

数据库相关最佳实践

下载并安装SQL Server

SQL Server各个版本的安装要求

SQL版本	安装要求
SQL Server 2016, SQL Server 2017	SQL Server 2016和2017：硬件和软件要求
SQL Server 2019	SQL Server 2019：硬件和软件要求
SQL Server 2022	SQL Server 2022：硬件和软件要求

目前，Microsoft官网仅提供SQL Server 2016，SQL Server 2017，SQL Server 2019，SQL Server 2022的下载。其余版本的SQL Server需要自行在网络上找资源下载。

SQL Server 2016 支持Windows Server 2012，Windows Server 2016，Windows Server 2019的操作系统，且不支持Windows Server 2022操作系统。更多的硬件和软件要求详情请查看上方表格链接。

SQL Server 2017支持Windows Server 2012，Windows Server 2016，Windows Server 2019的操作系统，更多的硬件和软件要求详情请查看上方表格链接。




SQL Server 2019支持Windows Server 2016，Windows Server 2019的操作系统，更多的硬件和软件要求详情请查看上方表格链接。

SQL Server 2022支持Windows Server 2016，Windows Server 2019的操作系统，更多的硬件和软件要求详情请查看上方表格链接。

Windows Server 2008的云主机请选用SQL Server 2014或更低的版本。

SQL Server 2016安装步骤

以下安装方法以SQL Server 2016安装为例，云主机操作系统为Windows Server 2012 数据中心版 R2 64位。

1. 点击下载好的“SQLServer2016-SSEI-Eval.exe”文件，进入SQL安装向导，进行安装操作。
2. 选择“自定义”安装，等待安装程序准备完毕。

3. 选择安装的语言，以及安装目录，点击“安装”。等待安装程序下载，下载完毕后自动进入SQL Server 安装中心。



4. 点击左侧的“安装”导航栏，选择第一个选项，即“全新SQL Server独立安装或向现有安装添加功能”。



5. 选择“指定可用版本”，采用免费镜像模式进行安装。（若要正式搭建Sharepoint，需要输入密钥安装SQL Server完整版本。）点击“下一步”。



6. 勾选“我接受许可条款和隐私声明”。点击“下一步”。



7. 如果需要Microsoft Update更新则勾选。不需要则不用勾选。点击“下一步”。



8. 勾选“全选”，选择安装全部功能后，取消勾选“针对外部数据的PolyBase查询服务”、R服务（数据库内）以及“R Server（独立）”功能。该云主机为域控制器，PolyBase无法在域控制器中使用，因此取消该选项。点击“下一步”。



9. 勾选“默认实例”。使用默认值来配置实例，不需要任何修改。点击“下一步”。



10. 设置SQL server Configuration。

修改“SQL Server 数据库引擎”的“账户名”为“NT AUTHORITY\NETWORK SERVICE”。

设置“SQL Server Analysis Services”为Windows Server添加AD，DHCP，DNS，IIS服务中设置的域账户和密码。详细步骤参见：[为Windows Server添加AD，DHCP，DNS，IIS服务](#)。

设置完毕之后点击“下一步”。



11. 勾选“Windows身份验证模式”，点击“添加当前用户”，将当前账号作为SQL Server的管理员账号。点击“下一步”。



12. 勾选“多维和数据挖掘模式”。点击“添加当前用户”，为当前账号添加Analysis Services的管理员权限。点击“下一步”。



13. 设置Reporting Services，在Reporting Services本机模式中，勾选“安装和配置”，在Reporting Services SharePoint集成模式下，勾选“仅安装”。单击“下一步”。



14. 点击“添加当前用户”，为当前账号添加Distributed Replay控制器的权限。点击“下一步”。



15. Distributed Replay客户端设施使用默认值，不修改。点击“下一步”。



16. 确认SQL Server配置，点击“安装”开始安装。



17. 检查功能状态是否全部成功，如果全部成功，点击“关闭”，SQL Server安装完成。



MySQL数据库自建，使用，问题排查最佳实践

创建ECS并自建mysql服务器

前提条件

- 1.已创建ECS虚拟机，请参考[指定ECS规格创建实例](#)创建MySQL客户端的弹性云服务器，如追求数据库IO读写性能，推荐使用本地盘云主机Ip3。
- 2.已下载MySQL客户端安装包，MySQL官网提供了针对不同操作系统的客户端安装包，单击[此处](#)下载8.0的最新版，单击[此处](#)下载其他归档版本，用户需根据自己的操作系统类型选择符合要求的安装包。

mysql部署

系统设置

确认系统及依赖库版本

本篇使用OS镜像版本为CTyunOS 22.06.3

```
[root@localhost ~]# cat /etc/ctyunos-release  
CTyunOS release 22.06.3
```

```
[root@localhost ~]# ldd --version  
ldd (GNU libc) 2.28  
Copyright (C) 2018 Free Software Foundation, Inc.  
This is free software; see the source for copying conditions. There is NO  
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  
Written by Roland McGrath and Ulrich Drepper.
```

确认mysql版本

二进制搭建，mysql 对应的 OS Version 必须和操作系统依赖库版本保持一致，找到对应版本进行下载。

MySQL Product Archives

MySQL Community Server (Archived Versions)

Please note that these are old versions. New releases will have recent bug fixes and features!
To download the latest release of MySQL Community Server, please visit [MySQL Downloads](#).

Product Version:
Operating System:
OS Version:

Generic Linux Minimal tarballs excludes debug binaries, and regular binaries are stripped

Compressed TAR Archive <small>(mysql-8.4.0-linux-glibc2.28-x86_64.tar.xz)</small>	Apr 10, 2024	458.9M	Download
Compressed TAR Archive, Test Suite <small>(mysql-test-8.4.0-linux-glibc2.28-x86_64.tar.xz)</small>	Apr 10, 2024	339.7M	Download
TAR <small>(mysql-8.4.0-linux-glibc2.28-x86_64.tar)</small>	Apr 10, 2024	819.7M	Download

We suggest that you use the [MD5 checksums](#) and [GnuPG signatures](#) to verify the integrity of the packages you download.

MySQL open source software is provided under the [GPL License](#).

防火墙及SELinux

```
[root@localhost ~]# systemctl disable firewalld --now #关闭并禁用防火墙
```

```
[root@localhost ~]# setenforce 0 #关闭selinux  
setenforce: SELinux is disabled
```

```
[root@localhost ~]# sed -i 's/^SELINUX=enforcing$/SELINUX=disabled/' /etc/selinux/config #配置持久化
```

创建用户及目录

```
[root@localhost ~]# groupadd mysql
```

```
[root@localhost ~]# useradd -r -g mysql -s /sbin/nologin mysql
```

```
[root@localhost ~]# mkdir -p /mysql/install/data
```

```
[root@localhost ~]# mkdir -p /mysql/install/tmp
```

```
[root@localhost ~]# mkdir -p /mysql/install/file
```

```
[root@localhost ~]# mkdir -p /mysql/install/log
```

创建MySQL

上传并解压

```
[root@localhost install]# pwd  
/mysql/install
```

最佳实践

```
[root@localhost install]# ls
data file log mysql-8.4.0-linux-glibc2.28-x86_64.tar.xz tmp

[root@localhost install]# tar -xf mysql-8.4.0-linux-glibc2.28-x86_64.tar.xz

[root@localhost install]# ln -s mysql-8.4.0-linux-glibc2.28-x86_64 mysql #创建软链接

[root@localhost install]# ll
total 459M
drwxr-xr-x 2 root root 6 Oct 8 17:04 data
drwxr-xr-x 2 root root 6 Oct 8 17:04 file
drwxr-xr-x 2 root root 6 Oct 8 17:04 log
lrwxrwxrwx 1 root root 34 Oct 8 17:08 mysql -> mysql-8.4.0-linux-glibc2.28-x86_64
drwxr-xr-x 9 root root 129 Oct 8 17:07 mysql-8.4.0-linux-glibc2.28-x86_64
-rw-r--r-- 1 root root 459M Oct 8 17:04 mysql-8.4.0-linux-glibc2.28-x86_64.tar.xz
drwxr-xr-x 2 root root 6 Oct 8 17:04 tmp
```

环境变量

```
[root@localhost install]# vi ~/.bash_profile

#PATH 增加路径 /mysql/install/mysql/bin
[root@localhost install]# cat ~/.bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin
PATH=$PATH:/mysql/install/mysql/bin:$HOME/bin
export PATH
```

```
[root@localhost install]# source ~/.bash_profile
```

初始化配置文件

```
#注意bind_address的IP地址和对应路径的修改
cat <<eof> /mysql/install/my.cnf
[mysql]
no-beep
prompt="\u@mydb \R:\m:\s [d]> "
#no-auto-rehash
auto-rehash
```

```
default-character-set=utf8
```

```
[mysqld]
```

```
lower_case_table_names=1
```

```
server-id=3306
```

```
port=3306
```

```
user = mysql
```

```
bind_address= 192.168.2.134
```

```
basedir=/mysql/install/mysql
```

```
datadir=/mysql/install/data
```

```
#socket = /mysql/install/mysql.sock
```

```
socket = /tmp/mysql.sock
```

```
pid-file = /mysql/install/mysql.pid
```

```
character-set-server=utf8
```

```
autocommit = 0
```

```
#skip_name_resolve = 1
```

```
max_connections = 800
```

```
max_connect_errors = 1000
```

```
default-storage-engine=INNODB
```

```
transaction_isolation = READ-COMMITTED
```

```
explicit_defaults_for_timestamp = 1
```

```
sort_buffer_size = 32M
```

```
join_buffer_size = 128M
```

```
tmp_table_size = 72M
```

```
max_allowed_packet = 16M
```

```
#sql_mode = "STRICT_TRANS_TABLES, NO_ENGINE_SUBSTITU
```

```
TION, NO_ZERO_DATE, NO_ZERO_IN_DATE, ERROR_FOR_DIVISION_BY_ZERO, NO_AUTO_CREATE_USER"
```

```
interactive_timeout = 1800
```

```
wait_timeout = 1800
```

```
read_buffer_size = 16M
```

```
read_rnd_buffer_size = 32M
```

```
#query_cache_type = 1
```

```
#query_cache_size=1M
```

```
table_open_cache=2000
```

```
thread_cache_size=768
```

```
myisam_max_sort_file_size=10G
```

```
myisam_sort_buffer_size=135M
```

```
key_buffer_size=32M
```

```
read_buffer_size=8M
```

```
read_rnd_buffer_size=4M
```

```
back_log=1024
```

```
#flush_time=0
```

```
open_files_limit=65536
```

```
table_definition_cache=1400
```

```
#binlog_row_event_max_size=8K
#sync_master_info=10000
#sync_relay_log=10000
#sync_relay_log_info=10000

log_output=FILE
general_log = 0
general_log_file=/mysql/install/log/hiri-general.err
slow_query_log = ON
slow_query_log_file=/mysql/install/log/hiri-query.err
long_query_time=10
log_error=/mysql/install/log/hiri-error.err

log_queries_not_using_indexes = 1
log_slow_admin_statements = 1
log_slow_slave_statements = 1
log_throttle_queries_not_using_indexes = 10
#expire_logs_days = 90
#binlog_expire_logs_seconds=2592000
binlog_expire_logs_seconds=604800
min_examined_row_limit = 100

log_bin=/mysql/install/log/hiri-binlog
log_bin_index=/mysql/install/log/hiri-binlog.index
binlog_format='ROW'
binlog_rows_query_log_events=on

#master_info_repository = TABLE
#relay_log_info_repository = TABLE
#log_bin = bin.log
#sync_binlog = 1
#gtid_mode = on
#enforce_gtid_consistency = 1
#log_slave_updates
#binlog_format = row
#relay_log = relay.log
#relay_log_recovery = 1
#binlog_gtid_simple_recovery = 1
#slave_skip_errors = ddl_exist_errors

innodb_io_capacity = 4000
innodb_io_capacity_max = 8000
innodb_buffer_pool_size = 500M
innodb_buffer_pool_instances = 8
innodb_buffer_pool_load_at_startup = 1
innodb_buffer_pool_dump_at_shutdown = 1
```

最佳实践

```
innodb_lru_scan_depth = 2000
innodb_lock_wait_timeout = 5
#innodb_flush_method = O_DIRECT

innodb_log_file_size = 200M
innodb_log_files_in_group = 2
innodb_log_buffer_size = 16M

#innodb_undo_logs = 128
innodb_undo_tablespaces = 3
innodb_undo_log_truncate = 1
innodb_max_undo_log_size = 2G

innodb_flush_neighbors = 1
innodb_purge_threads = 4
#innodb_large_prefix = 1
innodb_thread_concurrency = 64
innodb_print_all_deadlocks = 1
innodb_strict_mode = 1
innodb_sort_buffer_size = 64M
innodb_flush_log_at_trx_commit=1
innodb_autoextend_increment=64
innodb_concurrency_tickets=5000
innodb_old_blocks_time=1000
innodb_open_files=65536
innodb_stats_on_metadata=0
innodb_file_per_table=1
innodb_checksum_algorithm=0
innodb_data_file_path=ibdata1:200M;ibdata2:200M;ibdata3:200M:autoextend:max:5G
innodb_temp_data_file_path = ibtmp1:200M:autoextend:max:20G

innodb_buffer_pool_dump_pct = 40
innodb_page_cleaners = 4
innodb_purge_rseg_truncate_frequency = 128
binlog_gtid_simple_recovery=1
log_timestamps=system
#transaction_write_set_extraction=MURMUR32
#default_authentication_plugin=mysql_native_password #mysql 8.4版本之后不允许这样配置
了，使用mysql_native_password=ON可以正常使用
mysql_native_password=ON
#default_authentication_plugin=caching_sha2_password
EOF
```

```
[root@localhost install]# chown -R mysql:mysql /mysql #修改mysql目录的属组和属主
```

```
[root@localhost install]# ll
```

最佳实践

```
total 459M
drwxr-xr-x 2 mysql mysql  6 Oct  8 17:04 data
drwxr-xr-x 2 mysql mysql  6 Oct  8 17:04 file
drwxr-xr-x 2 mysql mysql  6 Oct  8 17:04 log
-rw-r--r-- 1 mysql mysql 3.5K Oct  8 17:13 my.cnf
lrwxrwxrwx 1 mysql mysql  34 Oct  8 17:08 mysql -> mysql-8.4.0-linux-glibc2.28-x86_64
drwxr-xr-x 9 mysql mysql 129 Oct  8 17:07 mysql-8.4.0-linux-glibc2.28-x86_64
-rw-r--r-- 1 mysql mysql 459M Oct  8 17:04 mysql-8.4.0-linux-glibc2.28-x86_64.tar.xz
drwxr-xr-x 2 mysql mysql  6 Oct  8 17:04 tmp
```

初始化数据库

```
[root@localhost install]# /mysql/install/mysql/bin/mysqld --defaults-
file=/mysql/install/my.cnf --initialize --user=mysql --basedir=/mysql/install/mysql --
datadir=/mysql/install/data
```

```
[root@localhost log]# grep 'temporary password' /mysql/install/log/hiri-error.err
2024-10-08T17:34:43.831698+08:00 6 [Note] [MY-010454]
[Server] A temporary password is generated for root@localhost: ,H4xd+oy0M>t
```

启库并修改密码

```
[root@localhost install]# /mysql/install/mysql/bin/mysqld_safe --defaults-
file=/mysql/install/my.cnf --datadir=/mysql/install/data --pid-file=/mysql/install/mysql.pid
&
```

```
[1] 2152
```

```
[root@localhost install]# 2024-10-08T11:26:28.216260Z mysqld_safe Logging to
'/mysql/install/log/hiri-error.err'.
2024-10-08T11:26:28.470158Z mysqld_safe Starting mysqld daemon with databases from
/mysql/install/data
```

```
[root@localhost install]# mysqladmin -uroot -p',H4xd+oy0M>t' password 'redhat' -S
/mysql/install/mysql.sock
```

```
mysqladmin: [Warning] Using a password on the command line interface can be insecure.
```

```
Warning: Since password will be sent to server in plain text, use ssl connection to ensure password safe
```

配置远程登录

```
[root@localhost install]# ln -sf /mysql/install/mysql.sock /tmp/mysql.sock
```

```
[root@localhost install]# mysql -uroot -predhat
```

```
mysql: [Warning] Using a password on the command line interface can be insecure.
```

```
Welcome to the MySQL monitor. Commands end with ; or \g.
```

```
Your MySQL connection id is 10
```

```
Server version: 8.4.0 MySQL Community Server - GPL
```

```
Copyright (c) 2000, 2024, Oracle and/or its affiliates.
```

```
Oracle is a registered trademark of Oracle Corporation and/or its
```

最佳实践

affiliates. Other names may be trademarks of their respective owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

```
mysql> show databases;
```

```
+-----+
| Database      |
+-----+
| information_schema |
| mysql         |
| performance_schema |
| sys          |
+-----+
4 rows in set (0.02 sec)
```

```
mysql> use mysql;
```

Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed

```
mysql> select Host,User from user;
```

```
+-----+-----+
| Host   | User       |
+-----+-----+
| localhost | mysql.infoschema |
| localhost | mysql.session  |
| localhost | mysql.sys      |
| localhost | root          |
+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> create user 'root'@'%' identified by 'redhat'; #创建root用户并设置密码为redhat
Query OK, 0 rows affected (0.09 sec)
```

```
mysql> grant all privileges on *.* to 'root'@'%' with grant option; #赋予root用户所有权限
Query OK, 0 rows affected (0.03 sec)
```

```
mysql> flush privileges; #刷新权限使之生效
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> select Host,User from user;
```

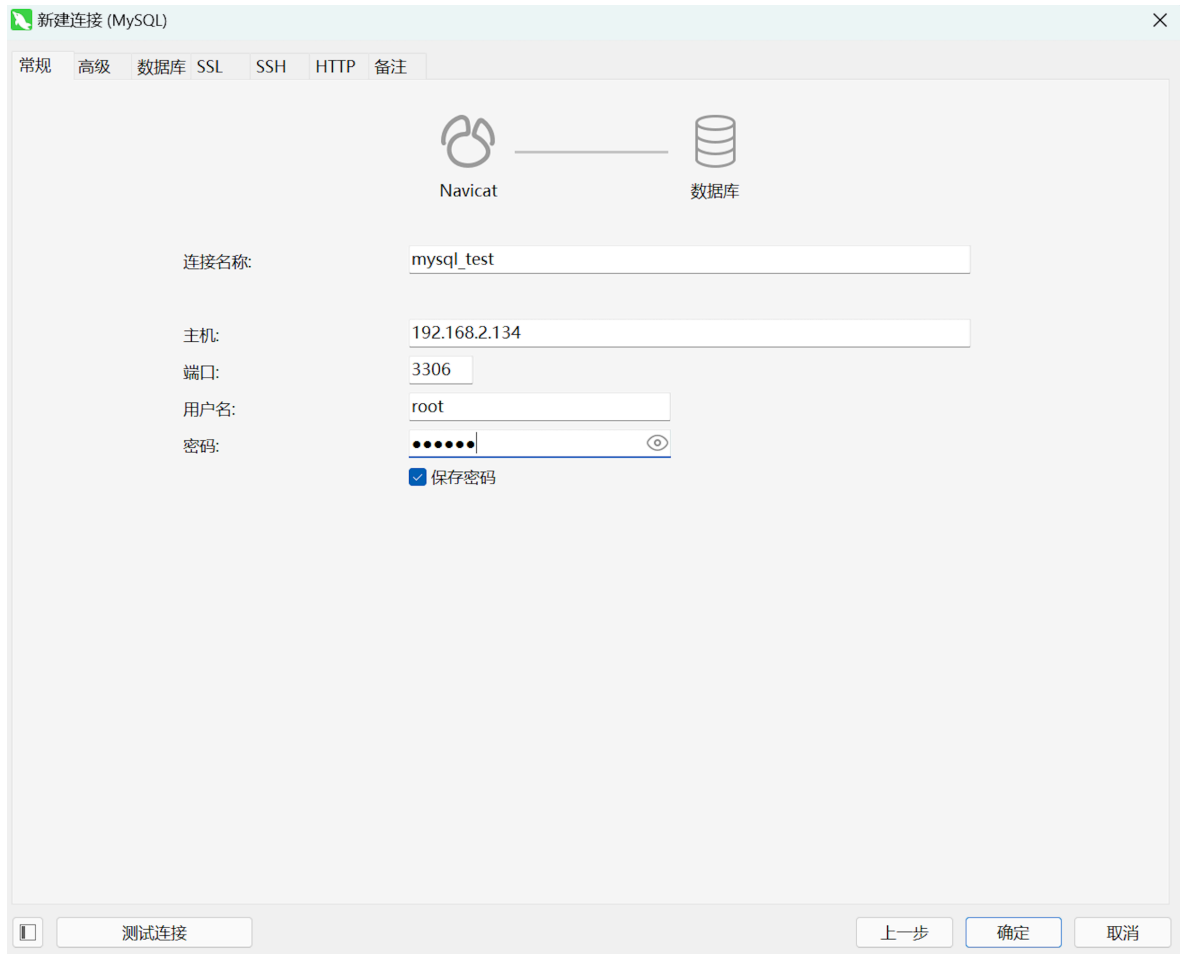
```
+-----+-----+
| Host   | User       |
+-----+-----+
| %      | root       |
| localhost | mysql.infoschema |
```

最佳实践

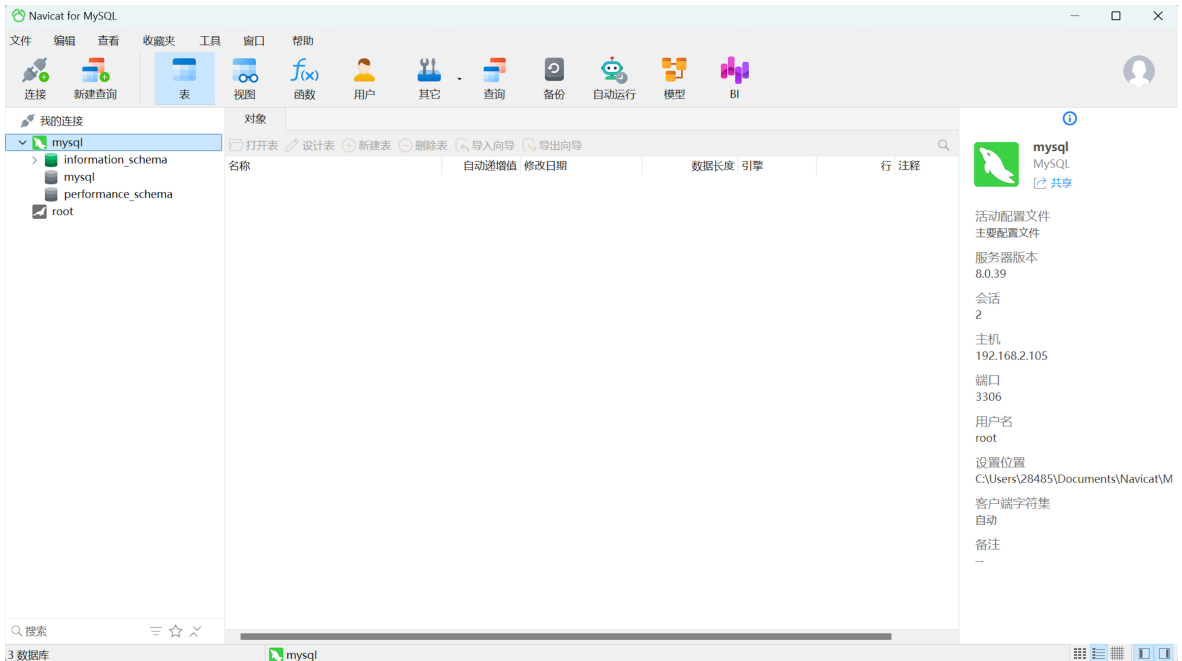
```
| localhost | mysql.session |
| localhost | mysql.sys      |
| localhost | root           |
+-----+-----+
5 rows in set (0.00 sec)
```

```
mysql> exit
Bye
```

连接测试



最佳实践



配置启动服务

在/mysql/install/mysql/support-files目录下，官方提供的有个脚本模板文件 mysql.server，可以拷贝一份起名叫mysql并修改脚本里面对应的路径和参数即可使用，这里已经改好的脚本内容如下：

```
[root@localhost support-files]# vim mysql
#!/bin/sh
# Copyright Abandoned 1996 TCX DataKonsult AB & Monty Program KB & Detron HB
# This file is public domain and comes with NO WARRANTY of any kind

# MySQL daemon start/stop script.

# Usually this is put in /etc/init.d (at least on machines SYSV R4 based
# systems) and linked to /etc/rc3.d/S99mysql and /etc/rc0.d/K01mysql.
# When this is done the mysql server will be started when the machine is
# started and shut down when the systems goes down.

# Comments to support chkconfig on RedHat Linux
# chkconfig: 2345 64 36
# description: A very fast and reliable SQL database engine.

# Comments to support LSB init script conventions
#### BEGIN INIT INFO
# Provides: mysql
# Required-Start: $local_fs $network $remote_fs
# Should-Start: ybind nscd ldap ntpd xntpd
# Required-Stop: $local_fs $network $remote_fs
```

最佳实践

```
# Default-Start: 2 3 4 5
# Default-Stop: 0 1 6
# Short-Description: start and stop MySQL
# Description: MySQL is a very fast and reliable SQL database engine.
#### END INIT INFO

# If you install MySQL on some other places than /usr/local/mysql, then you
# have to do one of the following things for this script to work:
#
# - Run this script from within the MySQL installation directory
# - Create a /etc/my.cnf file with the following information:
# [mysqld]
# basedir=<path-to-mysql-installation-directory>
# - Add the above to any other configuration file (for example ~/.my.ini)
# and copy my_print_defaults to /usr/bin
# - Add the path to the mysql-installation-directory to the basedir variable
# below.
#
# If you want to affect other MySQL variables, you should make your changes
# in the /etc/my.cnf, ~/.my.cnf or other MySQL configuration files.

# If you change base dir, you must also change datadir. These may get
# overwritten by settings in the MySQL configuration files.

basedir=/mysql/install/mysql
datadir=/mysql/install/data

# Default value, in seconds, afterwhich the script should timeout waiting
# for server start.
# Value here is overridden by value in my.cnf.
# 0 means don't wait at all
# Negative numbers mean to wait indefinitely
service_startup_timeout=900

# Lock directory for RedHat / SuSE.
lockdir='/var/lock/subsys'
lock_file_path="$lockdir/mysql"

# The following variables are only set for letting mysql.server find things.

# Set some defaults
mysqld_pid_file_path=/mysql/install/mysql.pid
if test -z "$basedir"
then
    basedir=/mysql/install/mysql
    bindir=/mysql/install/mysql/bin
    if test -z "$datadir"
```

```
then
    datadir=/mysql/install/data
fi
sbindir=/mysql/install/mysql/bin
libexecdir=/mysql/install/mysql/bin
else
bindir="$basedir/bin"
if test -z "$datadir"
then
    datadir="$basedir/data"
fi
sbindir="$basedir/sbin"
libexecdir="$basedir/libexec"
fi

# datadir_set is used to determine if datadir was set (and so should be
# *not* set inside of the --basedir= handler.)
datadir_set=

#
# Use LSB init script functions for printing messages, if possible
#
lsb_functions="/lib/lsb/init-functions"
if test -f $lsb_functions ; then
    . $lsb_functions
else
    log_success_msg()
    {
        echo " SUCCESS! $@"
    }
    log_failure_msg()
    {
        echo " ERROR! $@"
    }
fi

PATH="/sbin:/usr/sbin:/bin:/usr/bin:$basedir/bin"
export PATH

mode=$1 # start or stop

[ $# -ge 1 ] && shift

other_args="$*" # uncommon, but needed when called from an RPM upgrade action
# Expected: "--skip-networking --skip-grant-tables"
# They are not checked here, intentionally, as it is the responsibility
```

最佳实践

of the "spec" file author to give correct arguments only.

```
case `echo "testing\c", `echo -n testing` in
  *c*, -n*) echo_n= echo_c= ;;
  *c*, *) echo_n=-n echo_c= ;;
  *) echo_n= echo_c='\c' ;;
esac

parse_server_arguments() {
  for arg do
    case "$arg" in
      --basedir=*) basedir=`echo "$arg" | sed -e 's/^[^=]*=/'`
        bindir="$basedir/bin"
        if test -z "$datadir_set"; then
          datadir="$basedir/data"
        fi
        sbindir="$basedir/sbin"
        libexecdir="$basedir/libexec"
        ;;
      --datadir=*) datadir=`echo "$arg" | sed -e 's/^[^=]*=/'`
        datadir_set=1
        ;;
      --pid-file=*) mysqld_pid_file_path=`echo "$arg" | sed -e 's/^[^=]*=/'` ;;
      --service-startup-timeout=*) service_startup_timeout=`echo "$arg" | sed -e 's/^[^=]*=/'` ;;
    esac
  done
}

wait_for_pid () {
  verb="$1" # created | removed
  pid="$2" # process ID of the program operating on the pid-file
  pid_file_path="$3" # path to the PID file.

  i=0
  avoid_race_condition="by checking again"

  while test $i -ne $service_startup_timeout ; do

    case "$verb" in
      'created')
        # wait for a PID-file to pop into existence.
        test -s "$pid_file_path" && i=" " && break
        ;;
      'removed')
        # wait for this PID-file to disappear
        test ! -s "$pid_file_path" && i=" " && break
        ;;
    esac
  done
}
```

```
*)
  echo "wait_for_pid () usage: wait_for_pid created|removed pid pid_file_path"
  exit 1
;;
esac

# if server isn't running, then pid-file will never be updated
if test -n "$pid"; then
  if kill -0 "$pid" 2>/dev/null; then
    : # the server still runs
  else
    # The server may have exited between the last pid-file check and now.
    if test -n "$avoid_race_condition"; then
      avoid_race_condition=""
      continue # Check again.
    fi
  fi

  # there's nothing that will affect the file.
  log_failure_msg "The server quit without updating PID file ($pid_file_path)."
  return 1 # not waiting any more.
fi
fi

echo $echo_n ".$echo_c"
i=`expr $i + 1`
sleep 1

done

if test -z "$i"; then
  log_success_msg
  return 0
else
  log_failure_msg
  return 1
fi
}

# Get arguments from the my.cnf file,
# the only group, which is read from now on is [mysqld]
if test -x "$bindir/my_print_defaults"; then
  print_defaults="$bindir/my_print_defaults"
else
  # Try to find basedir in /etc/my.cnf
  conf=/mysql/install/my.cnf
  print_defaults=
  if test -r $conf
```

```

then
  subpat='^[^=]*basedir[^=]*=(.*)$'
  dirs=`sed -e "/$subpat/!d" -e 's/\\1/ $conf`
  for d in $dirs
  do
    d=`echo $d | sed -e 's/[ ]//g'`
    if test -x "$d/bin/my_print_defaults"
    then
      print_defaults="$d/bin/my_print_defaults"
      break
    fi
  done
fi

# Hope it's in the PATH ... but I doubt it
test -z "$print_defaults" && print_defaults="my_print_defaults"
fi

#
# Read defaults file from 'basedir'. If there is no defaults file there
# check if it's in the old (depricated) place (datadir) and read it from there
#

extra_args=""
if test -r "/mysql/install/my.cnf"
then
  extra_args="-e /mysql/install/my.cnf"
fi

parse_server_arguments ` $print_defaults
$extra_args mysql server mysql_server mysql.server `

#
# Set pid file if not given
#
if test -z "$mysqld_pid_file_path"
then
  mysqld_pid_file_path=$datadir/hostname`.pid
else
  case "$mysqld_pid_file_path" in
    /*) ;;
    *) mysqld_pid_file_path="$datadir/$mysqld_pid_file_path" ;;
  esac
fi

case "$mode" in
  'start')

```

最佳实践

```
# Start daemon

# Safeguard (relative paths, core dumps..)
cd $basedir

echo $echo_n "Starting MySQL"
if test -x $bindir/mysqld_safe
then
    # Give extra arguments to mysqld with the my.cnf file. This script
    # may be overwritten at next upgrade.
    $bindir/mysqld_safe --defaults-file=/mysql/install/my.cnf --datadir="$datadir" --pid-
file="$mysqld_pid_file_path" $other_args >/dev/null &
    wait_for_pid created "$!" "$mysqld_pid_file_path"; return_value=$?

    # Make lock for RedHat / SuSE
    if test -w "$lockdir"
    then
        touch "$lock_file_path"
    fi

    exit $return_value
else
    log_failure_msg "Couldn't find MySQL server ($bindir/mysqld_safe)"
fi
;;

'stop')
# Stop daemon. We use a signal here to avoid having to know the
# root password.

if test -s "$mysqld_pid_file_path"
then
    # signal mysqld_safe that it needs to stop
    touch "$mysqld_pid_file_path.shutdown"

    mysqld_pid=`cat "$mysqld_pid_file_path"`

    if (kill -0 $mysqld_pid 2>/dev/null)
    then
        echo $echo_n "Shutting down MySQL"
        kill $mysqld_pid
        # mysqld should remove the pid file when it exits, so wait for it.
        wait_for_pid removed "$mysqld_pid" "$mysqld_pid_file_path"; return_value=$?
    else
        log_failure_msg "MySQL server process #$mysqld_pid is not running!"
        rm "$mysqld_pid_file_path"
    fi
fi
```

```
# Delete lock for RedHat / SuSE
if test -f "$lock_file_path"
then
    rm -f "$lock_file_path"
fi
exit $return_value
else
    log_failure_msg "MySQL server PID file could not be found!"
fi
;;

'restart')
# Stop the service and regardless of whether it was
# running or not, start it again.
if $0 stop $other_args; then
    $0 start $other_args
else
    log_failure_msg "Failed to stop running server, so refusing to try to start."
    exit 1
fi
;;

'reload'|'force-reload')
if test -s "$mysqld_pid_file_path"; then
    read mysqld_pid < "$mysqld_pid_file_path"
    kill -HUP $mysqld_pid && log_success_msg "Reloading service MySQL"
    touch "$mysqld_pid_file_path"
else
    log_failure_msg "MySQL PID file could not be found!"
    exit 1
fi
;;

'status')
# First, check to see if pid file exists
if test -s "$mysqld_pid_file_path"; then
    read mysqld_pid < "$mysqld_pid_file_path"
    if kill -0 $mysqld_pid 2>/dev/null; then
        log_success_msg "MySQL running ($mysqld_pid)"
        exit 0
    else
        log_failure_msg "MySQL is not running, but PID file exists"
        exit 1
    fi
else
    # Try to find appropriate mysqld process
```

最佳实践

```
mysql_d_pid=`pidof $libexecdir/mysql_d`

# test if multiple pids exist
pid_count=`echo $mysql_d_pid | wc -w`
if test $pid_count -gt 1; then
    log_failure_msg "Multiple MySQL running but PID file could not be found ($mysql_d_pid)"
    exit 5
elif test -z $mysql_d_pid; then
    if test -f "$lock_file_path"; then
        log_failure_msg "MySQL is not running, but lock file ($lock_file_path) exists"
        exit 2
    fi
    log_failure_msg "MySQL is not running"
    exit 3
else
    log_failure_msg "MySQL is running but PID file could not be found"
    exit 4
fi
fi
;;
*)
# usage
basename=`basename "$0"`
echo "Usage: $basename {start|stop|restart|reload|force-reload|status}
[ MySQL server options ]"
    exit 1
;;
esac

exit 0

[root@localhost support-files]# pwd
/mysql/install/mysql/support-files

[root@localhost support-files]# chown mysql.mysql mysql

[root@localhost support-files]# chmod 755 mysql

[root@localhost support-files]# ll
total 32K
-rwxr-xr-x 1 mysql mysql 11K Oct  8 19:44 mysql
-rwxr-xr-x 1 mysql mysql 1.1K Apr 10 14:26 mysql_multi.server
-rw-r--r-- 1 mysql mysql 2.1K Apr 10 15:58 mysql-log-rotate
-rwxr-xr-x 1 mysql mysql 11K Apr 10 15:58 mysql.server
[root@localhost support-files]# ./mysql status
SUCCESS! MySQL running (3305)
```

最佳实践

```
[root@localhost support-files]# ./mysql stop
Shutting down MySQL.....2024-10-08T11:50:53.760574Z mysqld_safe mysqld from pid file
/mysql/install/mysql.pid ended
SUCCESS!
[1]+  Done                /mysql/install/mysql/bin/mysqld_safe --defaults-
file=/mysql/install/my.cnf --datadir=/mysql/install/data --pid-file=/mysql/install/mysql.pid
(wd: /mysql/install)
(wd now: /mysql/install/mysql/support-files)

[root@localhost support-files]# ./mysql start
Starting MySQL..... SUCCESS!

[root@localhost support-files]# ./mysql status
SUCCESS! MySQL running (4653)

使用systemd启动服务器(可选)

[root@localhost support-files]# ./mysql stop
Shutting down MySQL.. SUCCESS!

[root@localhost support-files]# cd /usr/lib/systemd/system

[root@localhost system]# touch mysqld.service

[root@localhost system]# chmod 644 mysqld.service

[root@localhost system]# vi mysqld.service
[root@localhost system]# cat mysqld.service
[Unit]
Description=MySQL Server
Documentation=man:mysqld(8)
Documentation=http://dev.mysql.com/doc/refman/en/using-systemd.html
After=network.target
After=syslog.target

[Install]
WantedBy=multi-user.target

[Service]
User=mysql
Group=mysql

# Have mysqld write its state to the systemd notify socket
Type=notify

# Disable service start and stop timeout logic of systemd for mysqld service.
TimeoutSec=0
```

最佳实践

```
# Start main service
ExecStart=/mysql/install/mysql/bin/mysqld --defaults-file=/mysql/install/my.cnf $MYSQLD_OPTS

# Use this to switch malloc implementation
EnvironmentFile=/etc/sysconfig/mysql

# Sets open_files_limit
LimitNOFILE = 10000

Restart=on-failure

RestartPreventExitStatus=1

# Set environment variable MYSQLD_PARENT_PID. This is required for restart.
Environment=MYSQLD_PARENT_PID=1

PrivateTmp=false

[root@localhost system]# systemctl enable mysqld.service
Created symlink /etc/systemd/system/multi-user.target.wants/mysqld.service
→ /usr/lib/systemd/system/mysqld.service.

[root@localhost system]# systemctl start mysqld.service

[root@localhost system]# systemctl status mysqld
● mysqld.service - MySQL Server
  Loaded: loaded (/usr/lib/systemd/system/mysqld.service; enabled; vendor preset: disabled)
  Active: active (running) since Tue 2024-10-08 19:53:18 CST; 1s ago
    Docs: man:mysqld(8)
          http://dev.mysql.com/doc/refman/en/using-systemd.html
  Main PID: 4794 (mysqld)
    Status: "Server is operational"
     Tasks: 39
  Memory: 422.6M
  CGroup: /system.slice/mysqld.service
          └─4794 /mysql/install/mysql/bin/mysqld --defaults-file=/mysql/install/my.cnf

Oct 08 19:53:10 localhost systemd[1]: Starting MySQL Server...
Oct 08 19:53:18 localhost systemd[1]: Started MySQL Server.

[root@localhost system]# systemctl stop mysqld

自此mysql服务器自建完成。
```

mysql数据库基础使用介绍

mysql常用命令

查看mysql中有哪些数据库:

mysql> show databases; #注意: 以分号结尾, 分号是英文的分号。

```
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| performance_schema |
+-----+
3 rows in set (0.02 sec)
```

选择使用某个数据库:

mysql> use mysql;

Reading table information for completion of table and column names

You can turn off this feature to get a quicker startup with -A

Database changed

查看某个数据库下有哪些表:

mysql> show tables;

```
+-----+
| Tables_in_mysql          |
+-----+
| columns_priv             |
| component                |
| db                      |
| default_roles            |
| engine_cost              |
| func                    |
| general_log              |
| global_grants            |
| gtid_executed            |
| help_category            |
| help_keyword             |
| help_relation            |
| help_topic               |
| innodb_index_stats       |
| innodb_table_stats       |
| ndb_binlog_index         |
| password_history         |
| plugin                   |
```

```

|procs_priv          |
|proxies_priv        |
|replication_asynchronous_connection_failover      |
|replication_asynchronous_connection_failover_managed|
|replication_group_configuration_version           |
|replication_group_member_actions                 |
|role_edges          |
|server_cost         |
|servers             |
|slave_master_info   |
|slave_relay_log_info|
|slave_worker_info   |
|slow_log            |
|tables_priv         |
|time_zone           |
|time_zone_leap_second|
|time_zone_name      |
|time_zone_transition|
|time_zone_transition_type|
|user                |
+-----+

```

创建数据库:

```
mysql> create database example;
Query OK, 1 row affected (0.01 sec)
```

```
mysql> show databases;
```

```

+-----+
|Database      |
+-----+
|example       |
|information_schema|
|mysql         |
|performance_schema|
+-----+

```

4 rows in set (0.00 sec)

查看mysql数据库的版本号:

```
mysql> select version();
```

```

+-----+
|version() |
+-----+
|8.4.0     |
+-----+

```

1 row in set (0.00 sec)

最佳实践

查看当前使用的是哪个数据库:

```
mysql> select database();
```

```
+-----+
```

```
| database() |
```

```
+-----+
```

```
| mysql |
```

```
+-----+
```

```
1 row in set (0.01 sec)
```

导入数据库表:

```
mysql> source D:\course\03-MySQL\document\test.sql #注意: 路径中不要有中文。
```

查看表的结构:

```
mysql> desc <table_name>; #describe缩写为: desc
```

简单查询select

```
mysql> select * from db;
```

```
+-----+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+
```

```
+-----+-----+-----+
```

```
| Host      | Db          | User      | Select_priv | Insert_priv | Update_priv | Delete_priv |  
| Create_priv | Drop_priv | Grant_priv | References_priv | Index_priv | Alter_priv | Create_  
tmp_table_priv | Lock_tables_priv | Create_view_priv | Show_view_priv | Create_routine_priv  
| Alter_routine_priv | Execute_priv | Event_priv | Trigger_priv |
```

```
+-----+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+
```

```
+-----+-----+-----+
```

```
| localhost | performance_schema | mysql.session | Y          | N          | N          | N          | N          | N          |  
| N          | N          | N          | N          | N          | N          | N          | N          | N          |  
| N          | N          | N          | N          |
```

```
+-----+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+
```

```
+-----+-----+-----+-----+
```

```
+-----+-----+-----+
```

```
1 row in set (0.00 sec)
```

#这种方式的缺点:

- 1、效率低 (他会先把*号转换为字段)
- 2、可读性差。

给查询的列起别名：使用as关键字起别名

```
mysql> select host,user as username from db; #as也可以省略
```

```
+-----+-----+
| host   | username |
+-----+-----+
| localhost | mysql.session |
+-----+-----+
1 row in set (0.00 sec)
```

条件查询

查询语法格式：select 字段1, 字段2, 字段3... from 表名 where 条件;

```
mysql> select host,user from user;
```

```
+-----+-----+
| host   | user      |
+-----+-----+
| %      | root      |
| localhost | mysql.infoschema |
| localhost | mysql.session |
| localhost | root      |
+-----+-----+
4 rows in set (0.00 sec)
```

```
mysql> select host,user username from user where user='mysql.session';
```

```
+-----+-----+
| host   | username |
+-----+-----+
| localhost | mysql.session |
+-----+-----+
1 row in set (0.00 sec)
```

表的创建

#创建一个学生表

```
create table t_student(
    no int,
    name varchar(32),
    sex char(1),
    age int(3),
    email varchar(255)
);
```

表的删除

#删除表：

```
drop table t_student;
// 当这张表不存在的时候会报错
```

最佳实践

```
// 如果这张表存在，删除
drop table if exists t_student;
```

插入数据insert

插入数据insert (DML)

语法格式: insert into 表名(字段名1, 字段名2, 字段名3...) values(值1, 值2, 值3);

```
mysql> create table t_user(number bigint(32), username varchar(32), sex char(1) default '男');
mysql> desc t_user;
```

```
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| number | bigint | YES  |     | NULL    |      |
| username | varchar(32) | YES  |     | NULL    |      |
| sex    | char(1) | YES  |     | 男      |      |
+-----+-----+-----+-----+-----+
```

```
mysql> insert into t_user (number, username, sex) values (1, 'user1', '男'), (2, 'user2', '女'),
(3, 'user3', '男');
```

Query OK, 3 rows affected (0.00 sec)

Records: 3 Duplicates: 0 Warnings: 0

```
mysql> select * from t_user;
```

```
+-----+-----+-----+
| number | username | sex |
+-----+-----+-----+
| 1 | user1 | 男 |
| 2 | user2 | 女 |
| 3 | user3 | 男 |
+-----+-----+-----+
```

3 rows in set (0.00 sec)

修改update

语法格式: update 表名 set 字段名1=值1, 字段名2=值2, 字段名3=值3... where 条件;

注意: 没有条件限制会导致所有数据全部更新!

```
update t_user set name = 'jack', birth = '2000-10-11' where id = 2;
```

```
+-----+-----+-----+-----+
| id | name  | birth   | create_time   |
+-----+-----+-----+-----+
| 1 | zhangsan | 1990-10-01 | 2020-03-18 15:49:50 |
| 2 | jack   | 2000-10-11 | 2020-03-18 15:51:23 |
+-----+-----+-----+-----+
```

```
update t_user set name = 'jack', birth = '2000-10-11', create_time = now() where id = 2;
```

更新所有：

```
update t_user set name = 'abc';
```

删除数据deleted

语法格式：delete from 表名 where 条件；

注意：没有条件，整张表的数据会全部删除！

```
delete from t_user where id = 2;
```

```
insert into t_user(id) values(2);
```

```
delete from t_user; // 删除所有
```

快速删除表中的数据：

//删除dept_bak表中的数据

```
delete from dept_bak;
```

```
mysql> select * from dept_bak;
```

```
Empty set (0.00 sec)
```

删除表操作：

```
drop table 表名; // 删除表。
```

mysql常见问题及解决思路

案例一：Too many connections（连接数过多，导致连接不上数据库，业务无法正常进行）

问题复现：

```
mysql> SHOW VARIABLES LIKE 'max_connections'; #将当前数据库允许最大连接数设置为1
```

```
+-----+-----+
```

```
| Variable_name | Value |
```

```
+-----+-----+
```

```
| max_connections | 800 |
```

```
+-----+-----+
```

```
1 row in set (0.03 sec)
```

```
mysql> set global max_connections=1;
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> SHOW VARIABLES LIKE 'max_connections';
```

```
+-----+-----+
```

```
| Variable_name | Value |
```

```
+-----+-----+
```

```
| max_connections | 1 |
```

```
+-----+-----+
```

```
1 row in set (0.00 sec)
```

```
[root@mysql ~]# mysql -uroot -predhat
mysql: [Warning] Using a password on the command line interface can be insecure.
ERROR 1040 (HY000): Too many connections #报错，太多连接
```

解决思路：

首先要考虑在我们 MySQL 数据库参数文件里面，对应的 `max_connections` 这个参数值是不是设置的太小了，导致客户端连接数超过了数据库所承受的最大值。该值默认大小是 151，我们可以根据实际情况进行调整。

对应解决办法：

```
set global max_connections=500
```

但这样调整会有隐患，因为我们无法确认数据库是否可以承担这么大的连接压力，就好比原来一个人只能吃一个馒头，但现在却非要让他吃 10 个，他肯定接受不了。反应到服务器上面，就有可能会出现宕机的可能。

所以这又反映出了，我们在新上线一个业务系统的时候，要做好压力测试。保证后期对数据库进行优化调整。

其次可以限制 InnoDB 的并发处理数量，如果 `innodb_thread_concurrency = 0`（这种代表不受限制）可以先改成 16 或是 64 看服务器压力。

如果非常大，可以先改的小一点让服务器的压力下来之后，然后再慢慢增大，根据自己的业务而定。

案例二 主从复制报错类型

```
Last_SQL_Errno: 1062 (从库与主库数据冲突)
```

```
Last_Errno: 1062
```

```
Last_Error: Could not execute Write_rows event on table test.t;
```

```
Duplicate entry '4' for key 'PRIMARY',
```

```
Error_code: 1062; handler error HA_ERR_FOUND_DUPP_KEY;
```

```
the event's master log mysql-bin.000014, end_log_pos 1505
```

针对这个报错，我们首先要考虑是不是在从库中误操作导致的。结果发现，我们在从库中进行了一条针对有主键表的 SQL 语句的插入，导致主库再插入相同 sql 的时候，主从状态出现异常。发生主键冲突的报错。解决方法：在确保主从数据一致性的前提下，可以在从库进行错误跳过。一般使用 `percona-toolkit` 中的 `pt-slave-restart` 进行。

在从库完成如下操作：

```
[root@zs bin]# ./pt-slave-restart -uroot -proot123
```

之后最好在从库中开启 `read_only` 参数，禁止在从库进行写入操作。

```
Last_IO_Errno: 1593 (server-id冲突)
```

```
Last_IO_Error:
```

```
Fatal error: The slave I/O thread stops because master and slave have equal MySQL server ids;
```

these ids must be different for replication to work

(or the `-replicate-same-server-id` option must be used on slave but this

does not always make sense; please check the manual before using it)

这个报错出现之后，就能一目了然看到两台机器的 `server-id` 是一样的。

在搭建主从复制的过程中，我们要确保两台机器的 `server-id` 是唯一的。这里再强调一下 `server-id` 的命名规则（服务器 ip 地址的最后一位+本 MySQL 服务的端口号）。

解决方法：

在主从两台机器上设置不同的 `server-id`。

```
Last_SQL_Errno: 1032 (从库少数据，主库更新的时候，从库报错)
Last_SQL_Error:
```

```
Could not execute Update_rows event on table test.t; Can't find record
in 't', Error_code: 1032; handler error HA_ERR_KEY_NOT_FOUND; the
event's master log mysql-bin.000014, end_log_pos 1708
```

解决问题的办法：

根据报错信息，我们可以获取到报错日志和 `position` 号，然后就能找到主库执行的哪条 `sql`，导致的主从报错。

在主库执行：

```
/usr/local/mysql/bin/mysqlbinlog -no-defaults -v -v --output=decode-rows /data/mysql/mysql-
bin.000014 |grep -A 10 1708 > 1.log
```

```
cat 1.log
```

```
#170720 14:20:15 server id 3 end_log_pos 1708 CRC32 0x97b6bdec Update_rows: table id 113 flags: STMT_END
```

```
### UPDATE `test`.`t`
```

```
### WHERE
```

```
### @1=4 /* INT meta=0 nullable=0 is_null=0 */
```

```
### @2=' dd' /* VARSTRING(60) meta=60 nullable=1 is_null=0 */
```

```
### SET
```

```
### @1=4 /* INT meta=0 nullable=0 is_null=0 */
```

```
### @2=' ddd' /* VARSTRING(60) meta=60 nullable=1 is_null=0 */
```

```
# at 1708
```

```
#170720 14:20:15 server id 3 end_log_pos 1739 CRC32 0xecaf1922 Xid = 654
```

```
COMMIT/*!*/;
```

```
DELIMITER ;
```

```
# End of log file
```

```
ROLLBACK /* added by mysqlbinlog */;
```

```
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
```

```
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=0*/;
```

获取到SQL语句之后，就可以在从库反向执行SQL语句。把从库缺少的SQL语句补全，解决报错信息。

在从库依次执行：

```
mysql> insert into t (b) values ( 'ddd' );
```

```
Query OK, 1 row affected (0.01 sec)
```

```
mysql> stop slave;
```

```
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> exit
```

```
Bye
```

```
[root@node4 bin]# ./pt-slave-restart -uroot -proot123
```

案例三 MySQL安装过程中的报错

```
[root@zs data]# /usr/local/mysql/bin/mysqld_safe - defaults-file=/etc/my.cnf &[1] 3758
```

```
[root@zs data]# 170720 14:41:24 mysqld_safe Logging to '/data/mysql/error.log' .
```

```
170720 14:41:24 mysqld_safe Starting mysqld daemon with databases from /data/mysql170720
```

```
14:41:25 mysqld_safe mysqld from pid file /data/mysql/node4.pid ended
```

```
170720 14:41:24 mysqld_safe Starting mysqld daemon with databases from /data/mysql2017-07-20
```

```
14:41:25 0 [Warning] TIMESTAMP with implicit DEFAULT value is deprecated.
```

```
Please use - explicit_defaults_for_timestamp server option
```

最佳实践

(see documentation for more details)./usr/local/mysql/bin/mysqld:

File '/data/mysql/mysql-bin.index' not found (Errcode: 13 - Permission denied)

2017-07-20 14:41:25 4388 [ERROR] Aborting

解决思路:

遇到这样的报错信息，我们要学会时时去关注错误日志 error log 里面的内容。看见了关键的报错点 Permission denied，证明当前 MySQL 数据库的数据目录没有权限。

解决方法:

```
[root@zs data]# chown mysql:mysql -R mysql
```

```
[root@zs data]# /usr/local/mysql/bin/mysqld_safe - defaults-file=/etc/my.cnf &
```

```
[1] 4402
```

```
[root@zs data]# 170720 14:45:56 mysqld_safe Logging to '/data/mysql/error.log' .
```

```
170720 14:45:56 mysqld_safe Starting mysqld daemon with databases from /data/mysql
```

如何避免这类问题，个人建议在安装 MySQL 初始化的时候，一定加上 `- user=mysql`，这样就可以避免权限问题。

```
./mysql_install_db - basedir=/usr/local/mysql/ - datadir=/data/mysql/ - defaults-  
file=/etc/my.cnf - user=mysql
```

案例四 数据库密码忘记的问题

```
[root@zs ~]# mysql -uroot -p
```

Enter password:

```
ERROR 1045 (28000): Access denied for user 'root' @ 'localhost' (using password: YES)
```

```
[root@zs ~]# mysql -uroot -p
```

Enter password:

```
ERROR 1045 (28000): Access denied for user 'root' @ 'localhost' (using password: YES)
```

解决思路:

目前是进入不了数据库的情况，所以我们要考虑是不是可以跳过权限。因为在数据库中，MySQL 数据库中 user 表记录着我们用户的信息。**解决方法：**启动 MySQL 数据库的过程中，可以这样执行：

```
/usr/local/mysql/bin/mysqld_safe - defaults-file=/etc/my.cnf - skip-grant-tables &
```

最佳实践

这样启动，就可以不用输入密码，直接进入 MySQL 数据库了。然后在修改你自己想要改的 root 密码即可。

```
update mysql.user set password=password( 'root123') where user=' root' ;
```

案例五 truncate 删除数据，导致自动清空自增 ID，前端返回报错 not found这个问题的出现，就要考虑下 truncate 和 delete 的区别了，看下实验演练：

首先先创建一张表：

```
CREATE TABLE `t` (  
  `a` int(11) NOT NULL AUTO_INCREMENT,  
  `b` varchar(20) DEFAULT NULL,  
  PRIMARY KEY (`a`),  
  KEY `b` (`b`)  
) ENGINE=InnoDB AUTO_INCREMENT=300 DEFAULT CHARSET=utf8
```

插入三条数据：

```
mysql> insert into t (b) values ( 'aa' );
```

```
Query OK, 1 row affected (0.00 sec)
```

```
mysql> insert into t (b) values ( 'bb' );
```

```
Query OK, 1 row affected (0.00 sec)
```

```
mysql> insert into t (b) values ( 'cc' );
```

```
Query OK, 1 row affected (0.00 sec)
```

```
mysql> select * from t;
```

```
+-- - +---+
```

```
| a | b |
```

```
+-- - +---+
```

```
| 300 | aa |
```

```
| 301 | bb |
```

```
| 302 | cc |
```

```
+---+ +---+
```

3 rows in set (0.00 sec)

先用 delete 进行删除全表信息，再插入新值。

结果发现 truncate 把自增初始值重置了，自增属性从 1 开始记录了。当前端用主键 id 进行查询时，就会报没有这条数据的错误。

个人建议不要使用 truncate 对表进行删除操作，虽然可以回收表空间，但是会涉及自增属性问题。这些坑，我们不要轻易钻进去。

案例六 can't open file (errno:24)

有的时候，数据库跑得好好的，突然报不能打开数据库文件的错误了。解决思路：首先我们要先查看数据库的 error log。然后判断是表损坏，还是权限问题。还有可能磁盘空间不足导致的不能正常访问表；操作系统的限制也要关注下；用 perror 工具查看具体错误！

```
linux:/usr/local/mysql/bin # ./perror 24
```

OS error code 24: Too many open files

超出最大打开文件数限制！ulimit -n 查看系统的最大打开文件数是 65535，不可能超出！那必然是数据库的最大打开文件数超出限制！

在 MySQL 里查看最大打开文件数限制命令：show variables like 'open_files_limit'；

发现该数值过小，改为 2048，重启 MySQL，应用正常

处理方法：

```
repair table；
```

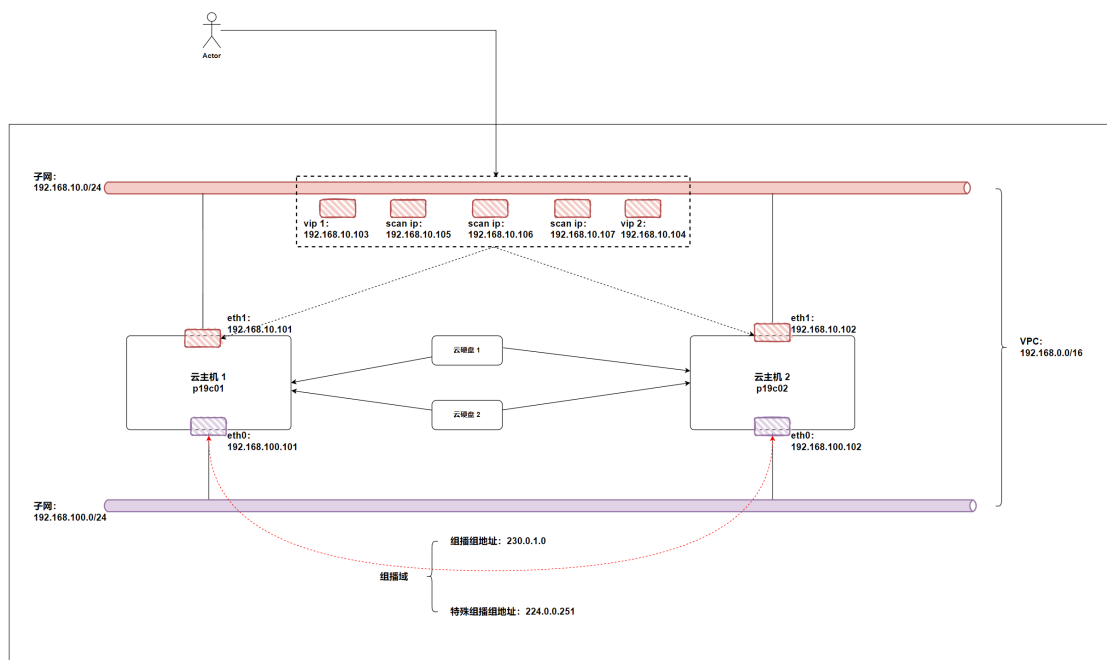
```
chown mysql 权限
```

```
清理磁盘中的垃圾数据
```

最佳实践

基于天翼云ECS搭建Oracle RAC集群

资源规划



系统规划

序号	节点主机名称	操作系统	虚拟机规格
1	p19c01	CentOS 7.964位	s7.2xlarge.2
2	p19c02	CentOS 7.964位	s7.2xlarge.2

网络规划

序号	节点主机名称	Private IP	Public IP	Virtual IP	Scan IP
1	p19c01	192.168.100.101	192.168.10.101	192.168.10.103	192.168.10.105 192.168.10.106 192.168.10.107
2	p19c02	192.168.100.102	192.168.10.102	192.168.10.104	192.168.10.105 192.168.10.106 192.168.10.107

存储规划

序号	角色	冗余模式	大小
1	裁决盘 OCR	EXTERNAL	100G

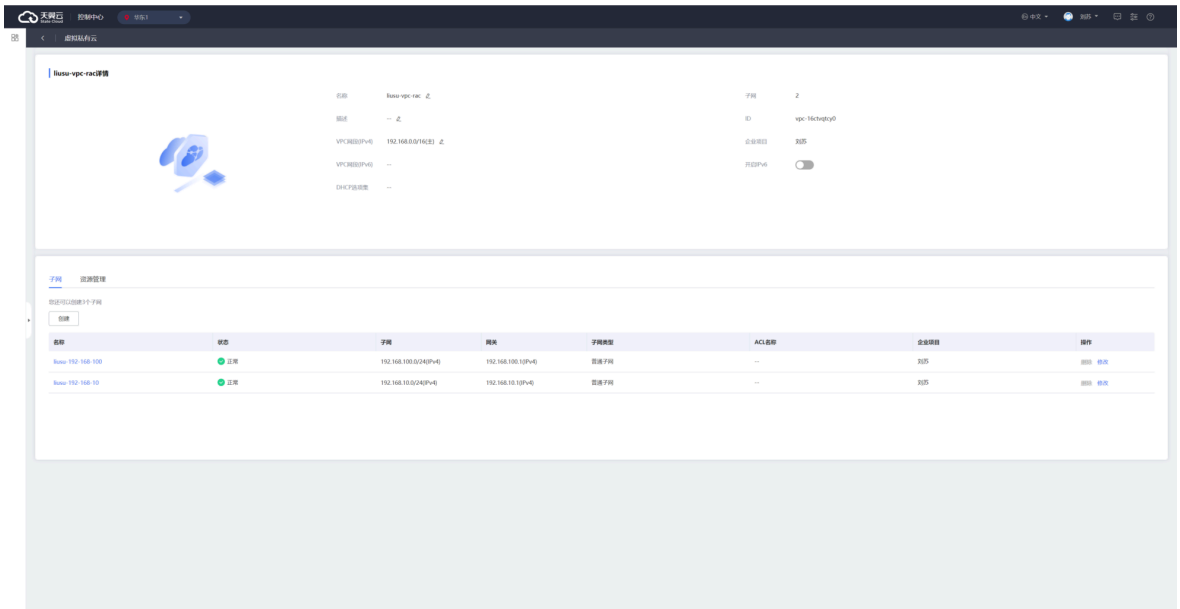
最佳实践

序号	角色	冗余模式	大小
2	数据盘 DATA	EXTERNAL	200G

资源创建

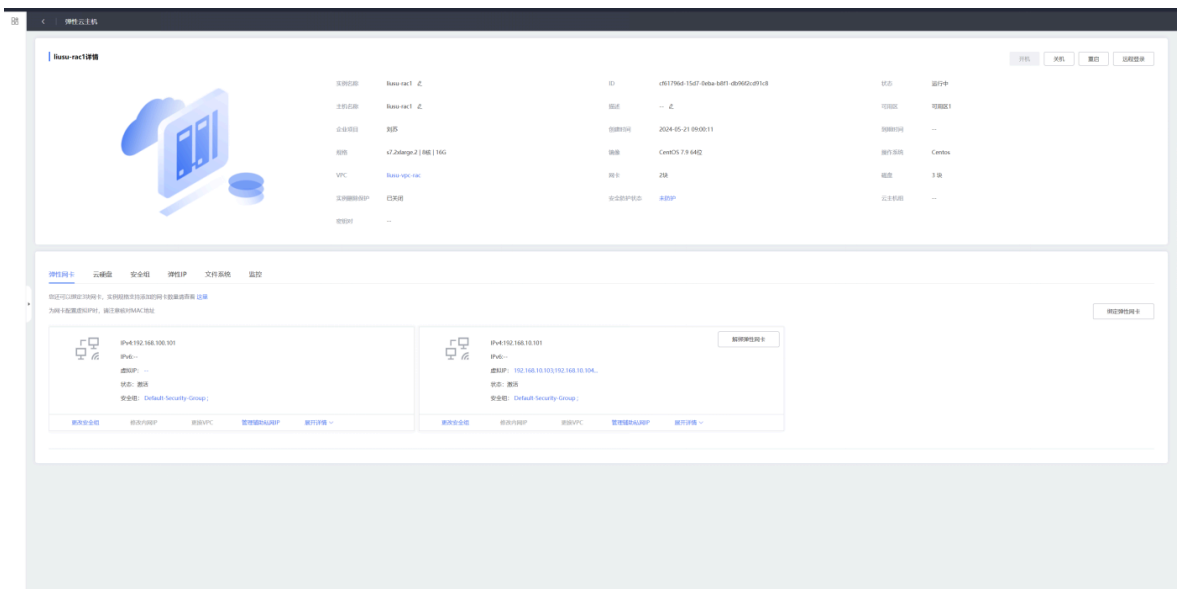
1. 创建VPC网络。

如下图所示为创建的VPC网络并创建了2个子网：



2. 创建云主机。

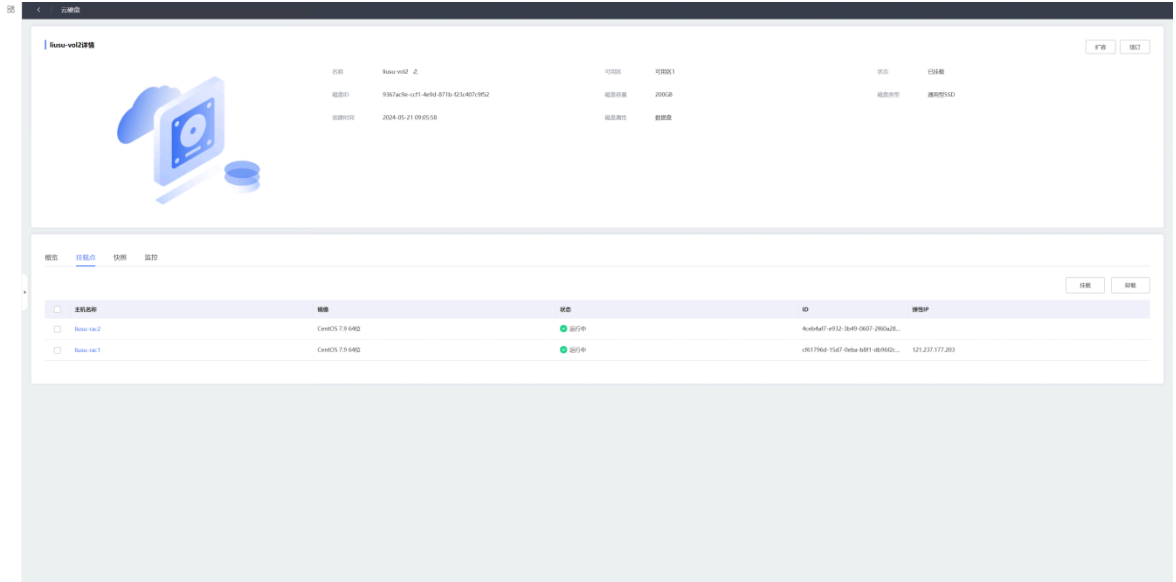
如下图所示为创建的云主机，共创建两台同规格的云主机：



最佳实践

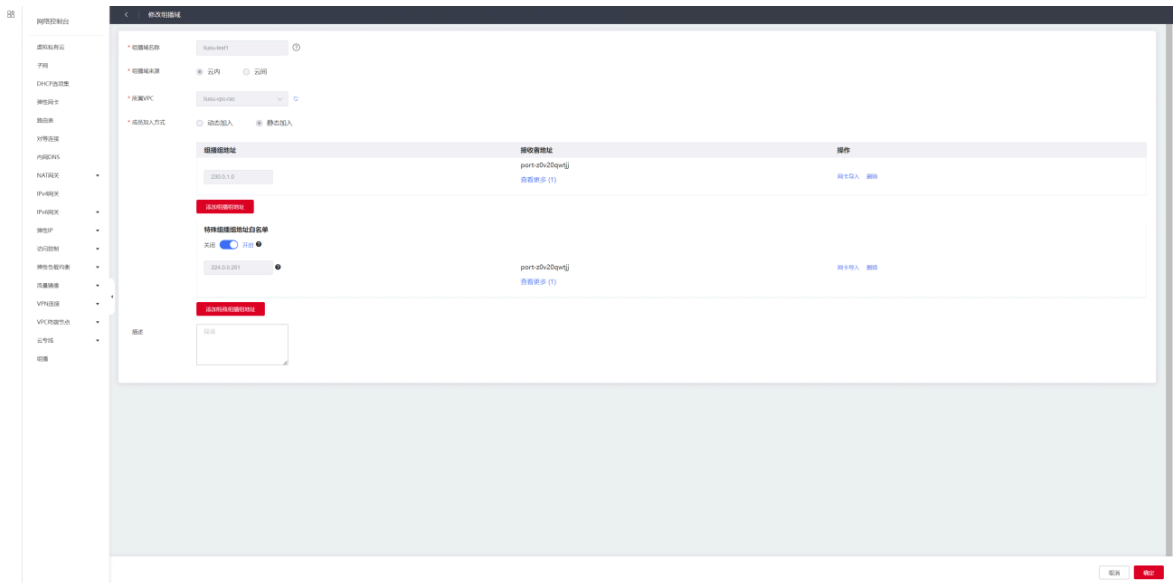
3. 创建共享盘。

创建 2 个共享盘云硬盘并挂载至 2 个云主机上：



4. 创建组播。

创建一个组播，所属 VPC 选择上述步骤创建的 VPC，并设置为静态加入方式，配置组播组地址：



节点配置

1. Host文件配置。

设置两个节点的主机名：

```
hostnamectl set-hostname p19c01
```

```
hostnamectl set-hostname p19c02
```

配置两个节点的/etc/hosts/文件：

```
192.168.10.101 p19c01
```

```
192.168.10.102 p19c02
```

```
192.168.100.101 p19c01-priv
```

```
192.168.100.102 p19c02-priv
```

```
192.168.10.103 p19c01-vip
```

```
192.168.10.104 p19c02-vip
```

```
192.168.10.105 p19c-scan
```

```
192.168.10.106 p19c-scan
```

```
192.168.10.107 p19c-scan
```

注意：两个节点均需执行。

2. 存储配置。

1) SCSI 磁盘模式场景

配置udev规则，步骤如下所示：

```
[root@p19c01 soft]# cat asm.sh
#!/bin/bash
for i in a b ;
do
echo "KERNEL=="sd*", ENV{DEVTYPE}=="disk", SUBSYSTEM=="block", PROGRAM=="
"/usr/lib/udev/scsi_id -g -u -d \${devnode}", RESULT==""/usr/lib/udev/scsi_id -g -u
/dev/sd${i}\", RUN+="/bin/sh -c 'mknod /dev/asmdisk${i} b \${major} \${minor}; chown grid:asmadmin
/dev/asmdisk${i}; chmod 0660 /dev/asmdisk${i}\""
done
[root@p19c01 soft]# cat /etc/udev/rules.d/99-oracle-asmdevices.rules
KERNEL=="sd*", ENV{DEVTYPE}=="disk", SUBSYSTEM=="block", PROGRAM=="/usr/lib/udev/scsi_id -
g -u -d \${devnode}", RESULT=="360000000000000000e000000002f0001", RUN+="/bin/sh -c 'mknod
/dev/asmdiska b \${major} \${minor}; chown grid:asmadmin /dev/asmdiska; chmod 0660 /dev/asmdiska"
KERNEL=="sd*", ENV{DEVTYPE}=="disk", SUBSYSTEM=="block", PROGRAM=="/usr/lib/udev/scsi_id -
g -u -d \${devnode}", RESULT=="360000000000000000e000000002b0001", RUN+="/bin/sh -c 'mknod
/dev/asmdiskb b \${major} \${minor}; chown grid:asmadmin /dev/asmdiskb; chmod 0660 /dev/asmdiskb"
[root@p19c01 soft]# udevadm control --reload
[root@p19c01 soft]# udevadm trigger
[root@p19c01 soft]#
[root@p19c01 soft]# ls -l /dev/ |grep asm
```

最佳实践

```
brw-rw---- 1 root root    8,  0 May 24 17:08 asmdiska
brw-rw---- 1 root root    8, 16 May 24 17:08 asmdiskb
[root@p19c01 soft]#
[root@p19c02 ~]# sh asm.sh
KERNEL=="sd*", ENV{DEVTYPE}=="disk", SUBSYSTEM=="block", PROGRAM=="usr/lib/udev/scsi_id -
g -u -d $devnode", RESULT=="360000000000000000e000000002f0001", RUN+="/bin/sh -c 'mknod
/dev/asmdiska b $major $minor; chown grid:asmadmin /dev/asmdiska; chmod 0660 /dev/asmdiska'"
KERNEL=="sd*", ENV{DEVTYPE}=="disk", SUBSYSTEM=="block", PROGRAM=="usr/lib/udev/scsi_id -
g -u -d $devnode", RESULT=="360000000000000000e000000002b0001", RUN+="/bin/sh -c 'mknod
/dev/asmdiskb b $major $minor; chown grid:asmadmin /dev/asmdiskb; chmod 0660 /dev/asmdiskb'"
[root@p19c02 ~]#
[root@p19c02 ~]#
[root@p19c02 ~]# vim /etc/udev/rules.d/99-oracle-asmdevices.rules
[root@p19c02 ~]#
[root@p19c02 ~]# udevadm control --reload
[root@p19c02 ~]# udevadm trigger
[root@p19c02 ~]#
[root@p19c02 ~]#
[root@p19c02 ~]# ls -l /dev |grep asm
brw-rw---- 1 root root    8,  0 May 24 17:09 asmdiska
brw-rw---- 1 root root    8, 16 May 24 17:09 asmdiskb
```

注意：两个节点均需执行。

2) VBD 磁盘模式场景。

```
[root@p19c01 ~]# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sr0  11:0  1 380K 0 rom
vda  253:0  0 100G 0 disk
└─vda1 253:1  0 100G 0 part /
vdb  253:16 0 100G 0 disk
vdc  253:32 0 200G 0 disk
[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]# cat /sys/class/block/vdb/serial
965fcb9e-4f9c-4aa0-9[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]# cat /sys/class/block/vdc/serial
9367ac9e-ccf1-4e9d-8[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]# vim /etc/udev/rules.d/
75-persistent-net-generator.rules 90-configure-multi-queue.rules 99-gen-net-conf.rules
```

最佳实践

```
[root@p19c01 ~]# vim /etc/udev/rules.d/99-oracle-asmdevices.rules
[root@p19c01 ~]#
[root@p19c01 ~]# cat /etc/udev/rules.d/99-oracle-asmdevices.rules
KERNEL=="vd*", ATTR{serial}=="965fcbe9-4f9c-4aa0-9", SYMLINK
+="asm_ocr", OWNER="grid", GROUP="asmadmin", MODE="0660"
KERNEL=="vd*", ATTR{serial}=="9367ac9e-ccf1-4e9d-8", SYMLINK
+="asm_data", OWNER="grid", GROUP="asmadmin", MODE="0660"
[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]# udevadm control --reload
[root@p19c01 ~]#
[root@p19c01 ~]# udevadm trigger
[root@p19c01 ~]#
[root@p19c01 ~]#
[root@p19c01 ~]# ls -l /dev/ |grep asm
lrwxrwxrwx 1 root root      3 May 27 15:30 asm_data -> vdc
lrwxrwxrwx 1 root root      3 May 27 15:30 asm_ocr -> vdb
[root@p19c02 ~]# lsblk
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sr0   11:0  1 380K  0 rom
vda   253:0  0 100G  0 disk
└─vda1 253:1  0 100G  0 part /
vdb   253:16  0 100G  0 disk
vdc   253:32  0 200G  0 disk
[root@p19c02 ~]#
[root@p19c02 ~]#
[root@p19c02 ~]# vim /etc/udev/rules.d/99-oracle-asmdevices.rules
[root@p19c02 ~]#
[root@p19c02 ~]#
[root@p19c02 ~]# udevadm control --reload
[root@p19c02 ~]#
[root@p19c02 ~]# udevadm trigger
[root@p19c02 ~]#
[root@p19c02 ~]#
[root@p19c02 ~]# ls -l /dev/ |grep asm
lrwxrwxrwx 1 root root      3 May 27 15:31 asm_data -> vdc
lrwxrwxrwx 1 root root      3 May 27 15:31 asm_ocr -> vdb
[root@p19c02 ~]#
[root@p19c02 ~]#
[root@p19c02 ~]#
[root@p19c02 ~]# cat /etc/udev/rules.d/99-oracle-asmdevices.rules
KERNEL=="vd*", ATTR{serial}=="965fcbe9-4f9c-4aa0-9", SYMLINK
+="asm_ocr", OWNER="grid", GROUP="asmadmin", MODE="0660"
KERNEL=="vd*", ATTR{serial}=="9367ac9e-ccf1-4e9d-8", SYMLINK
+="asm_data", OWNER="grid", GROUP="asmadmin", MODE="0660"
```

```
[root@p19c02 ~]#
```

注意：两个节点均需执行。

3. avahi-daemon 配置。

```
yum install -y avahi*
systemctl disable avahi-daemon.service
```

注意：两个节点均需执行。

4. 系统参数配置。

配置参数文件如下所示：

```
memTotal=$(grep MemTotal /proc/meminfo | awk '{print $2}')
totalMemory=$((memTotal / 2048))
shmall=$((memTotal / 4))
if [ $shmall -lt 2097152 ]; then
    shmall=2097152
fi
shmmax=$((memTotal * 1024 - 1))
if [ "$shmmax" -lt 4294967295 ]; then
    shmmax=4294967295
fi
cat <<EOF>>/etc/sysctl.conf
fs.aio-max-nr = 1048576
fs.file-max = 6815744
kernel.shmall = $shmall
kernel.shmmax = $shmmax
kernel.shmmni = 4096
kernel.sem = 250 32000 100 128
net.ipv4.ip_local_port_range = 9000 65500
net.core.rmem_default = 262144
net.core.rmem_max = 4194304
net.core.wmem_default = 262144
net.core.wmem_max = 1048576
net.ipv4.conf.eth0.rp_filter = 1
net.ipv4.conf.eth1.rp_filter = 2
EOF
```

生效系统参数：

```
sysctl -p
```

5. 系统资源限制配置。

配置limits.conf：

```
cat <<EOF>>/etc/security/limits.conf
oracle soft nofile 1024
oracle hard nofile 65536
```

```
oracle soft stack 10240
oracle hard stack 32768
oracle soft nproc 2047
oracle hard nproc 16384
oracle hard memlock 134217728
oracle soft memlock 134217728
```

```
grid soft nofile 1024
grid hard nofile 65536
grid soft stack 10240
grid hard stack 32768
grid soft nproc 2047
grid hard nproc 16384
EOF
```

配置pam.d/login:

```
cat <<EOF>>/etc/pam.d/login
session required pam_limits.so
session required /lib64/security/pam_limits.so
EOF
```

6. 用户及组、目录创建。

用户组创建:

```
/usr/sbin/groupadd -g 54321 oinstall
/usr/sbin/groupadd -g 54322 dba
/usr/sbin/groupadd -g 54323 oper
/usr/sbin/groupadd -g 54324 backupdba
/usr/sbin/groupadd -g 54325 dgdba
/usr/sbin/groupadd -g 54326 kmdba
/usr/sbin/groupadd -g 54327 asmdba
/usr/sbin/groupadd -g 54328 asmoper
/usr/sbin/groupadd -g 54329 asmadmin
/usr/sbin/groupadd -g 54330 racdba
```

用户grid/oracle创建:

```
/usr/sbin/useradd -u 11012 -g oinstall -G asmadmin, asmdba, asmoper, dba, racdba, oper grid
/usr/sbin/useradd -u 54321 -g oinstall -
G asmdba, dba, backupdba, dgdba, kmdba, racdba, oper oracle
echo "oracle" |passwd oracle --stdin
echo "oracle" |passwd grid --stdin
id grid
id oracle
udevadm control --reload-rules
udevadm trigger
```

```
ll /dev |grep asm
```

创建软件目录:

```
mkdir -p /u01/app/19.3.0/grid
mkdir -p /u01/app/grid
mkdir -p /u01/app/oracle/product/19.3.0/db
mkdir -p /u01/app/oraInventory
mkdir -p /backup
mkdir -p /home/oracle/scripts
chown -R oracle:oinstall /backup
chown -R oracle:oinstall /home/oracle/scripts
chown -R grid:oinstall /u01
chown -R grid:oinstall /u01/app/grid
chown -R grid:oinstall /u01/app/19.3.0/grid
chown -R grid:oinstall /u01/app/oraInventory
chown -R oracle:oinstall /u01/app/oracle
chmod -R 775 /u01
```

注意: 两个节点均需执行。

7. 环境变量配置。

用户grid环境变量:

```
cat <<EOF>>/home/grid/.bash_profile
#####OracleBegin#####
umask 022
export TMP=/tmp
export TMPDIR=$TMP
export NLS_LANG=AMERICAN_AMERICA.AL32UTF8
export ORACLE_BASE=/u01/app/grid
export ORACLE_HOME=/u01/app/19.3.0/grid
export ORACLE_TERM=xterm
export TNS_ADMIN=$ORACLE_HOME/network/admin
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib
export ORACLE_SID=+ASM1
export PATH=/usr/sbin:$PATH
export PATH=$ORACLE_HOME/bin:$ORACLE_HOME/OPatch:$PATH
alias sas='sqlplus / as sysasm'
export PS1="[ \whoami\ @\hostname\ :"\$PWD]\$ "
EOF
cat <<EOF>>/home/grid/.bash_profile
#####OracleBegin#####
umask 022
export TMP=/tmp
export TMPDIR=$TMP
export NLS_LANG=AMERICAN_AMERICA.AL32UTF8
export ORACLE_BASE=/u01/app/grid
```

最佳实践

```
export ORACLE_HOME=/u01/app/19.3.0/grid
export ORACLE_TERM=xterm
export TNS_ADMIN=\$ORACLE_HOME/network/admin
export LD_LIBRARY_PATH=\$ORACLE_HOME/lib/lib:usr/lib
export ORACLE_SID=+ASM2
export PATH=/usr/sbin:\$PATH
export PATH=\$ORACLE_HOME/bin:\$ORACLE_HOME/OPatch:\$PATH
alias sas='sqlplus / as sysasm'
export PS1="[ \whoami\ @ \hostname\ : \"\$PWD]\$ "
EOF
```

用户oracle环境变量:

```
cat <<EOF>>/home/oracle/.bash_profile
#####OracleBegin#####
umask 022
export TMP=/tmp
export TMPDIR=\$TMP
export NLS_LANG=AMERICAN_AMERICA.AL32UTF8
export ORACLE_BASE=/u01/app/oracle
export ORACLE_HOME=\$ORACLE_BASE/product/19.3.0/db
export ORACLE_HOSTNAME=p19c01
export ORACLE_TERM=xterm
export TNS_ADMIN=\$ORACLE_HOME/network/admin
export LD_LIBRARY_PATH=\$ORACLE_HOME/lib/lib:usr/lib
export ORACLE_SID=p19c01
export PATH=/usr/sbin:\$PATH
export PATH=\$ORACLE_HOME/bin:\$ORACLE_HOME/OPatch:\$PATH
alias sas='sqlplus / as sysdba'
export PS1="[ \whoami\ @ \hostname\ : \"\$PWD]\$ "
#####OracleEnd#####
EOF
cat <<EOF>>/home/oracle/.bash_profile
#####OracleBegin#####
umask 022
export TMP=/tmp
export TMPDIR=\$TMP
export NLS_LANG=AMERICAN_AMERICA.AL32UTF8
export ORACLE_BASE=/u01/app/oracle
export ORACLE_HOME=\$ORACLE_BASE/product/19.3.0/db
export ORACLE_HOSTNAME=p19c02
export ORACLE_TERM=xterm
export TNS_ADMIN=\$ORACLE_HOME/network/admin
export LD_LIBRARY_PATH=\$ORACLE_HOME/lib/lib:usr/lib
export ORACLE_SID=p19c02
export PATH=/usr/sbin:\$PATH
```

```
export PATH=$ORACLE_HOME/bin:$ORACLE_HOME/OPatch:$PATH
alias sas='sqlplus / as sysdba'
export PS1="[ \whoami \ @ \ hostname \ : \ $PWD ] $ "
#####OracleEnd#####
EOF
```

注意：两个节点均需执行。

8. SWAP 配置。

```
dd if=/dev/zero of=/swap bs=1M count=16390
mkswap /swap
swapon /swap
echo "/swap swap swap default 0 0" >> /etc/fstab
```

注意：两个节点均需执行。

9. 安装依赖包。

```
yum groupinstall -y "Server with GUI"
```

```
yum install -y bc \
binutils \
compat-libcap1 \
compat-libstdc++-33 \
gcc \
gcc-c++ \
elfutils-libelf \
elfutils-libelf-devel \
glibc \
glibc-devel \
ksh \
libaio \
libaio-devel \
libgcc \
libstdc++ \
libstdc++-devel \
libxcb \
libX11 \
libXau \
libXi \
libXtst \
libXrender \
libXrender-devel \
make \
net-tools \
nfs-utils \
smartmontools \
sysstat \
```

```
e2fsprogs \  
e2fsprogs-libs \  
fontconfig-devel \  
expect \  
unzip \  
openssh-clients \  
readline* \  
tigervnc* \  
psmisc
```

注意：两个节点均需执行。

10. 上传并解压安装包。

注意：只在第一个节点执行。

```
mkdir /soft
```

上传安装包到该目录下：

```
[root@p19c01 soft]# ls -lh  
total 5.6G  
-rw-r--r-- 1 root root 2.9G May 21 10:44 LINUX.X64_193000_db_home.zip  
-rw-r--r-- 1 root root 2.7G May 21 09:56 LINUX.X64_193000_grid_home.zip
```

解压grid安装包：

```
[root@p19c01 ~]# cd /soft  
[root@p19c01 soft]# ls -l  
total 5809468  
-rw-r--r-- 1 root root 3059705302 May 21 10:44 LINUX.X64_193000_db_home.zip  
-rw-r--r-- 1 root root 2889184573 May 21 09:56 LINUX.X64_193000_grid_home.zip  
[root@p19c01 soft]#  
[root@p19c01 soft]# chown -R grid:oinstall /soft  
[root@p19c01 soft]#  
[root@p19c01 soft]# su - grid -c "unzip -q /soft/LINUX.X64_193000_grid_home.zip -d  
/u01/app/19.3.0/grid/"
```

解压oracle安装包：

```
[root@p19c01 soft]# ls -lh  
total 5.6G  
-rw-r--r-- 1 grid oinstall 2.9G May 21 10:44 LINUX.X64_193000_db_home.zip  
-rw-r--r-- 1 grid oinstall 2.7G May 21 09:56 LINUX.X64_193000_grid_home.zip  
[root@p19c01 soft]#  
[root@p19c01 soft]# chown -R oracle:oinstall /soft  
[root@p19c01 soft]#
```

最佳实践

```
[root@p19c01 soft]# su - oracle -c "unzip -q /soft/LINUX.X64_193000_db_home.zip -d /u01/app/oracle/product/19.3.0/db/"
```

安装cvuqdisk（两个节点都需执行）：

```
[root@p19c01 soft]# cd /u01/app/19.3.0/grid/cv/rpm/
[root@p19c01 rpm]# ls -l
total 12
-rw-r--r-- 1 grid oinstall 11412 Mar 13 2019 cvuqdisk-1.0.10-1.rpm
[root@p19c01 rpm]# rpm -ivh cvuqdisk-1.0.10-1.rpm
Preparing... ##### [100%]
Using default group oinstall to install package
Updating / installing...
 1:cvuqdisk-1.0.10-1 ##### [100%]
[root@p19c01 rpm]# scp cvuqdisk-1.0.10-1.rpm root@p19c02:/root
[root@p19c01 rpm]#
[root@p19c01 rpm]# ssh p19c02
root@p19c02's password:
Last failed login: Thu May 23 14:22:55 CST 2024 from 183.81.169.238 on ssh:notty
There were 22 failed login attempts since the last successful login.
Last login: Thu May 23 09:12:17 2024 from 101.227.46.166
[root@p19c02 ~]# ls -lh
total 12K
-rw-r--r-- 1 root root 12K May 23 15:05 cvuqdisk-1.0.10-1.rpm
[root@p19c02 ~]# rpm -ivh cvuqdisk-1.0.10-1.rpm
Preparing... ##### [100%]
Using default group oinstall to install package
Updating / installing...
 1:cvuqdisk-1.0.10-1 ##### [100%]
```

安装Grid软件

配置grid用户vnc图形界面：

```
[root@p19c01 ~]# chown -R grid:oinstall /soft
[root@p19c01 ~]#
[root@p19c01 ~]# su - grid
Last login: Thu May 23 14:05:58 CST 2024 on pts/0
[grid@p19c01:/home/grid]$
[grid@p19c01:/home/grid]$ vncserver
```

You will require a password to access your desktops.

Password:

Verify:

Would you like to enter a view-only password (y/n)? n

A view-only password is not used

xauth: file /home/grid/.Xauthority does not exist

New 'p19c01:1 (grid)' desktop is p19c01:1

Creating default startup script /home/grid/.vnc/xstartup

Creating default config /home/grid/.vnc/config

Starting applications specified in /home/grid/.vnc/xstartup

Log file is /home/grid/.vnc/p19c01:1.log

```
[grid@p19c01:/home/grid]$
```

```
[grid@p19c01:/home/grid]$
```

```
[grid@p19c01:/home/grid]$ ps aux |grep vnc
```

```
grid 1090 1.4 0.2 286708 46392 pts/0 S1 15:39 0:00 /bin/Xvnc :1 -auth  
/home/grid/.Xauthority -desktop p19c01:1 (grid) -fp catalogue:/etc/X11/fontpath.d -  
geometry 1024x768 -httpd /usr/share/vnc/classes -pn -rfbauth /home/grid/.vnc/passwd -  
rfbport 5901 -rfbwait 30000
```

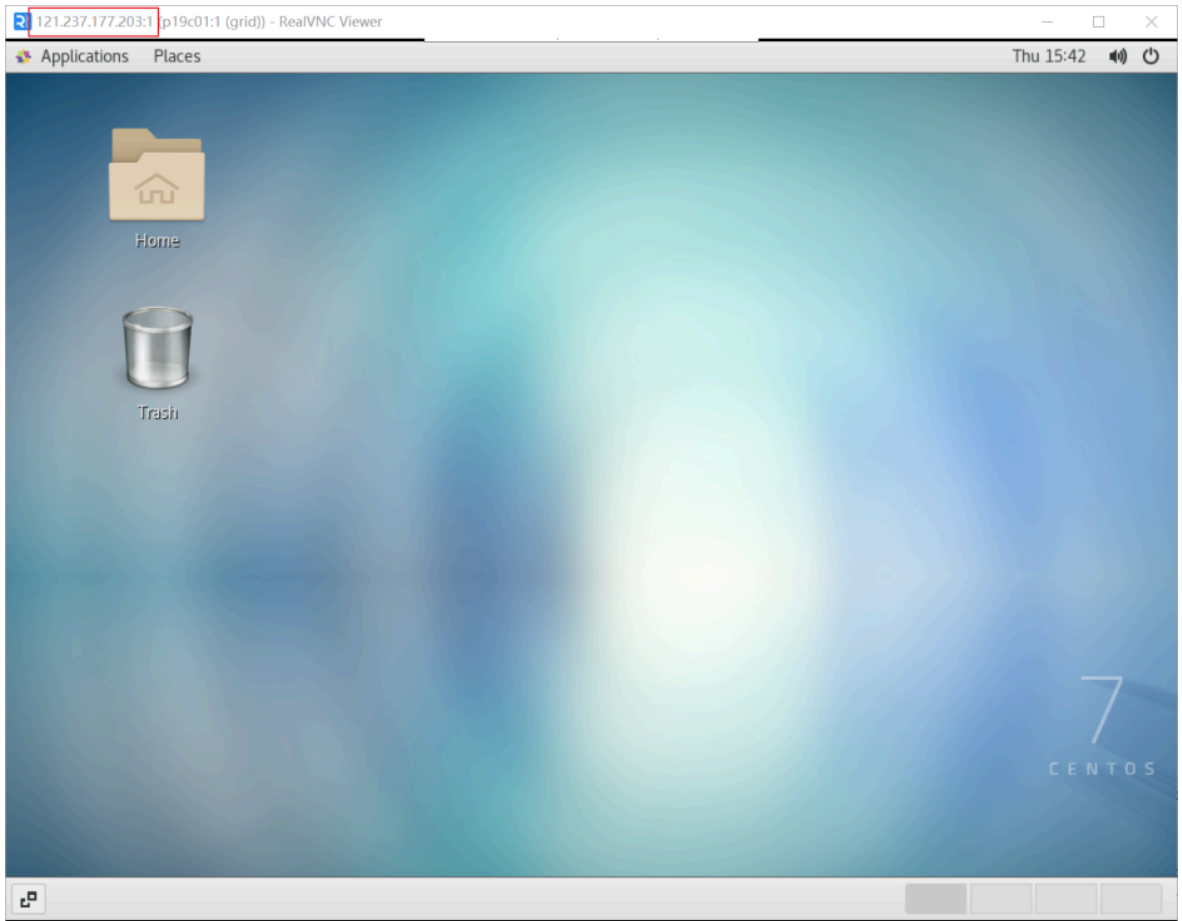
```
grid 1103 0.0 0.0 113288 1200 pts/0 S 15:39 0:00 /bin/sh /home/grid/.vnc/xstartup
```

```
grid 1898 0.0 0.0 112820 980 pts/0 S+ 15:39 0:00 grep --color=auto vnc
```

```
[grid@p19c01:/home/grid]$
```

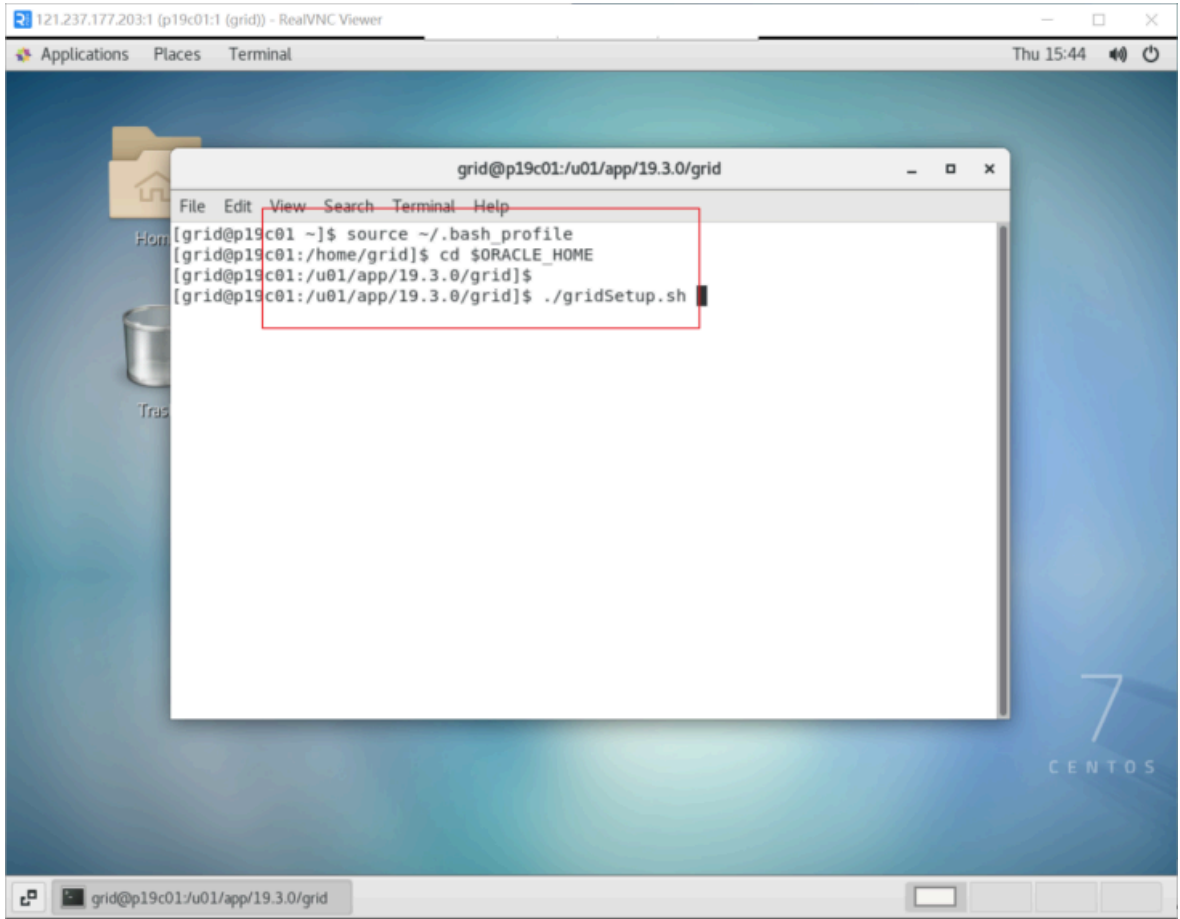
在vnc客户端通过VNC Viewer来访问，输入地址为节点1的弹性EIP加上一个端口1：

最佳实践



如下图所示，开始执行安装：

最佳实践



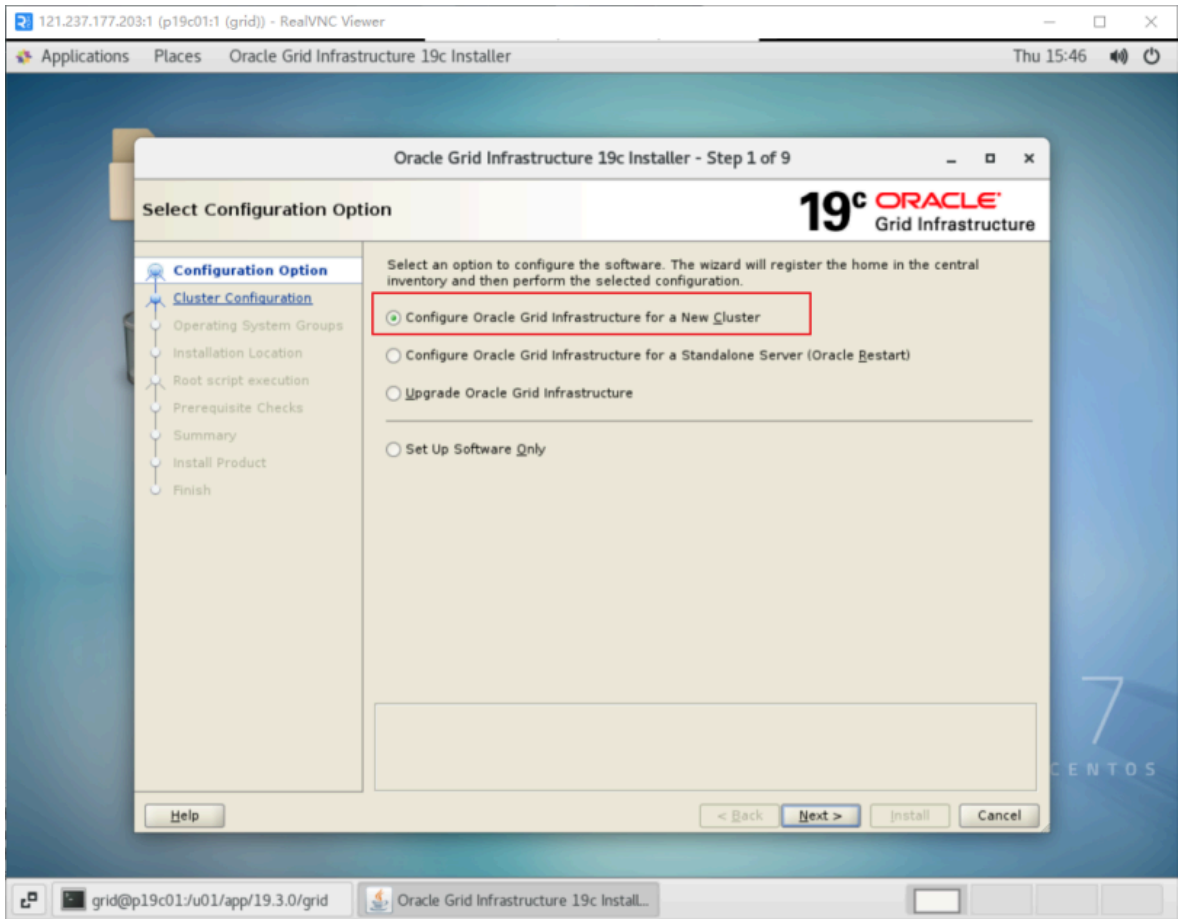
The screenshot shows a terminal window titled "grid@p19c01:/u01/app/19.3.0/grid" within a RealVNC Viewer. The terminal displays the following commands and their outputs:

```
grid@p19c01 ~]$ source ~/.bash_profile
grid@p19c01:/home/grid]$ cd $ORACLE_HOME
grid@p19c01:/u01/app/19.3.0/grid]$
grid@p19c01:/u01/app/19.3.0/grid]$ ./gridSetup.sh
```

The terminal window is overlaid on a desktop environment with a blue background and a "7 CENTOS" logo in the bottom right corner. The terminal window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help".

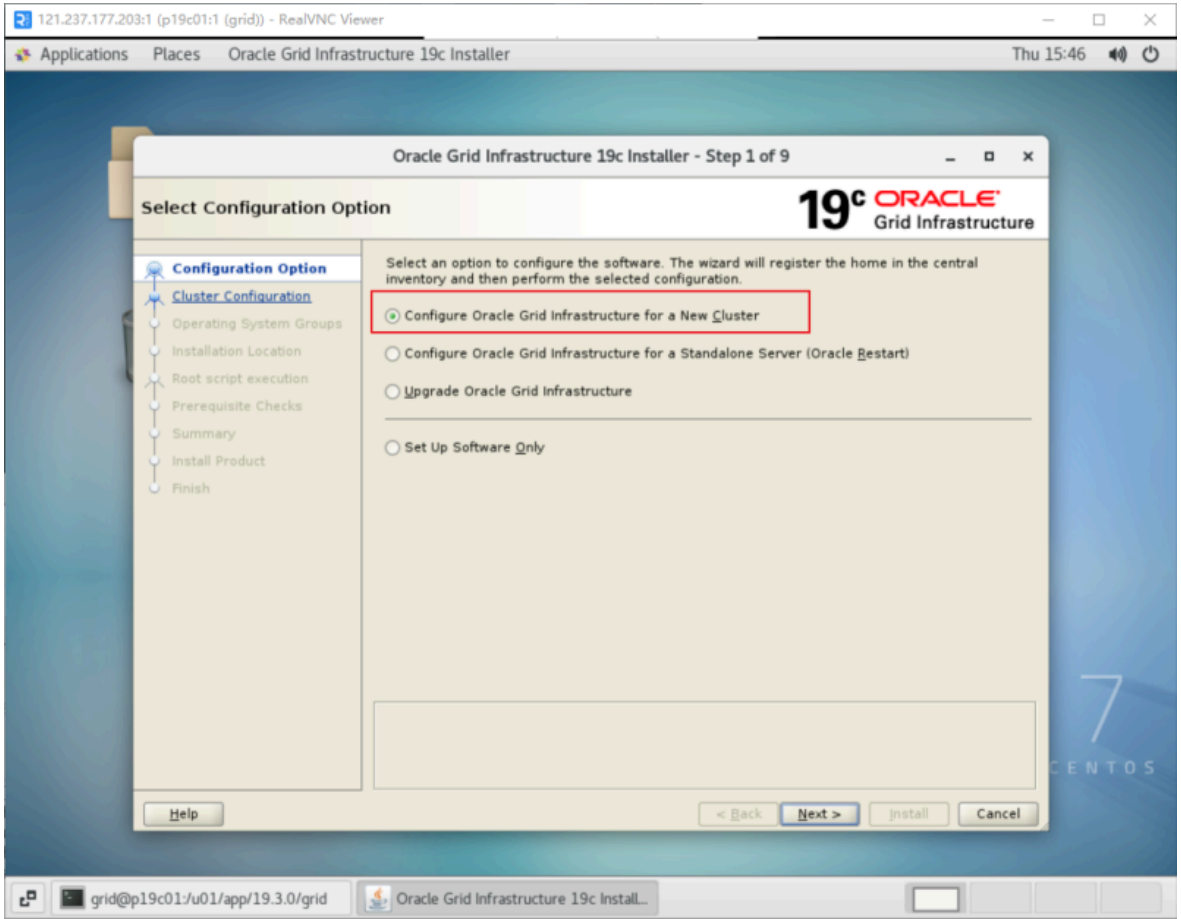
选择集群安装:

最佳实践



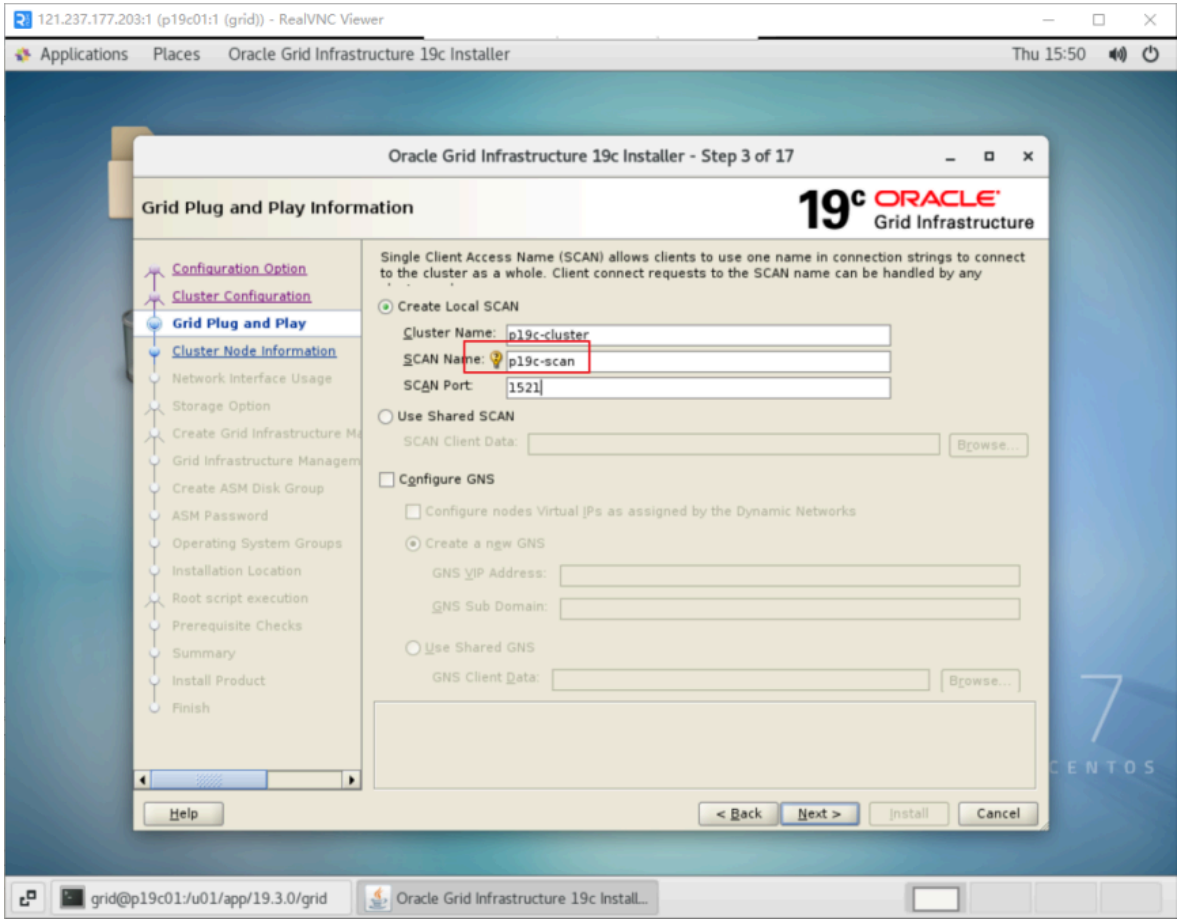
选择Standalone模式:

最佳实践



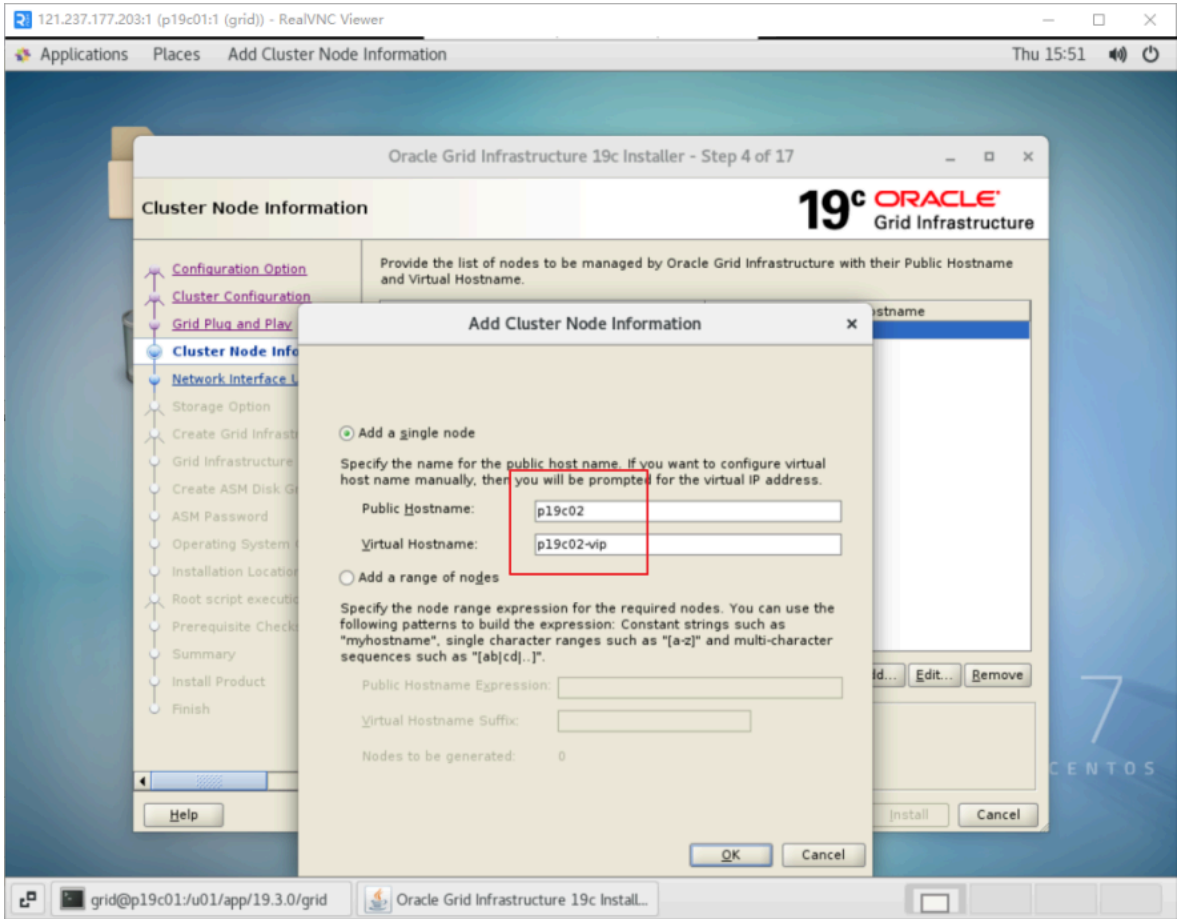
修改scan名称，与hosts文件配置scan名称保持一致：

最佳实践



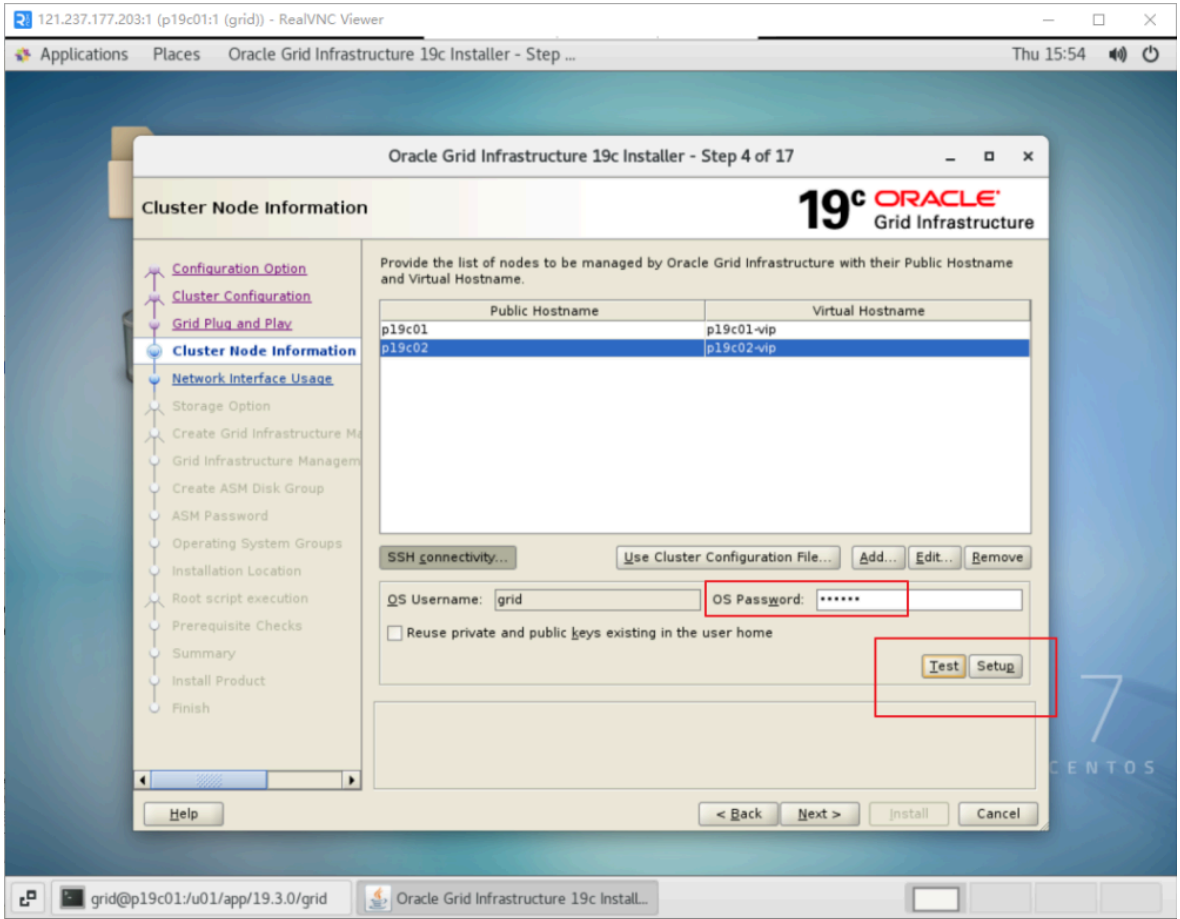
添加节点2信息，进行互信：

最佳实践



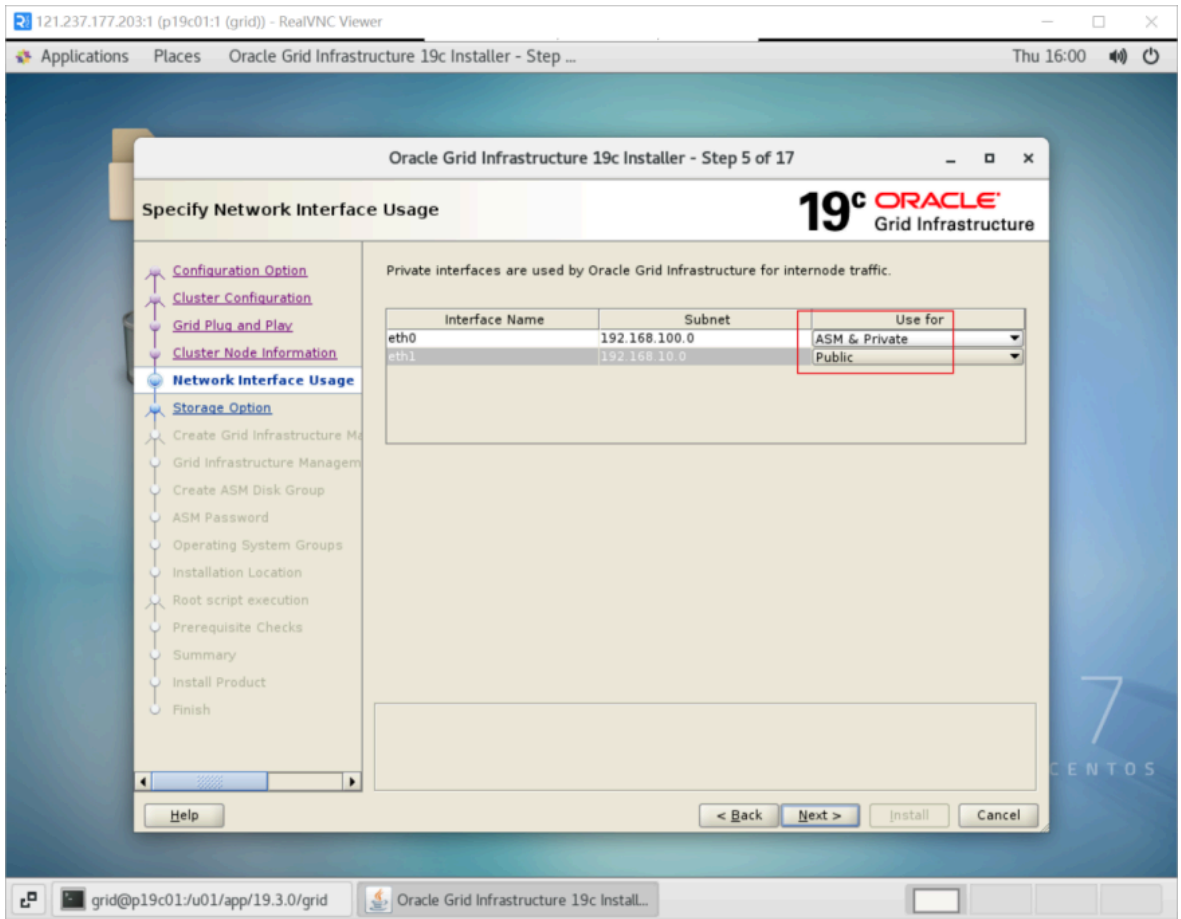
输入grid用户密码，创建用户时两节点必须保持一致。先执行Setup，再执行Test:

最佳实践



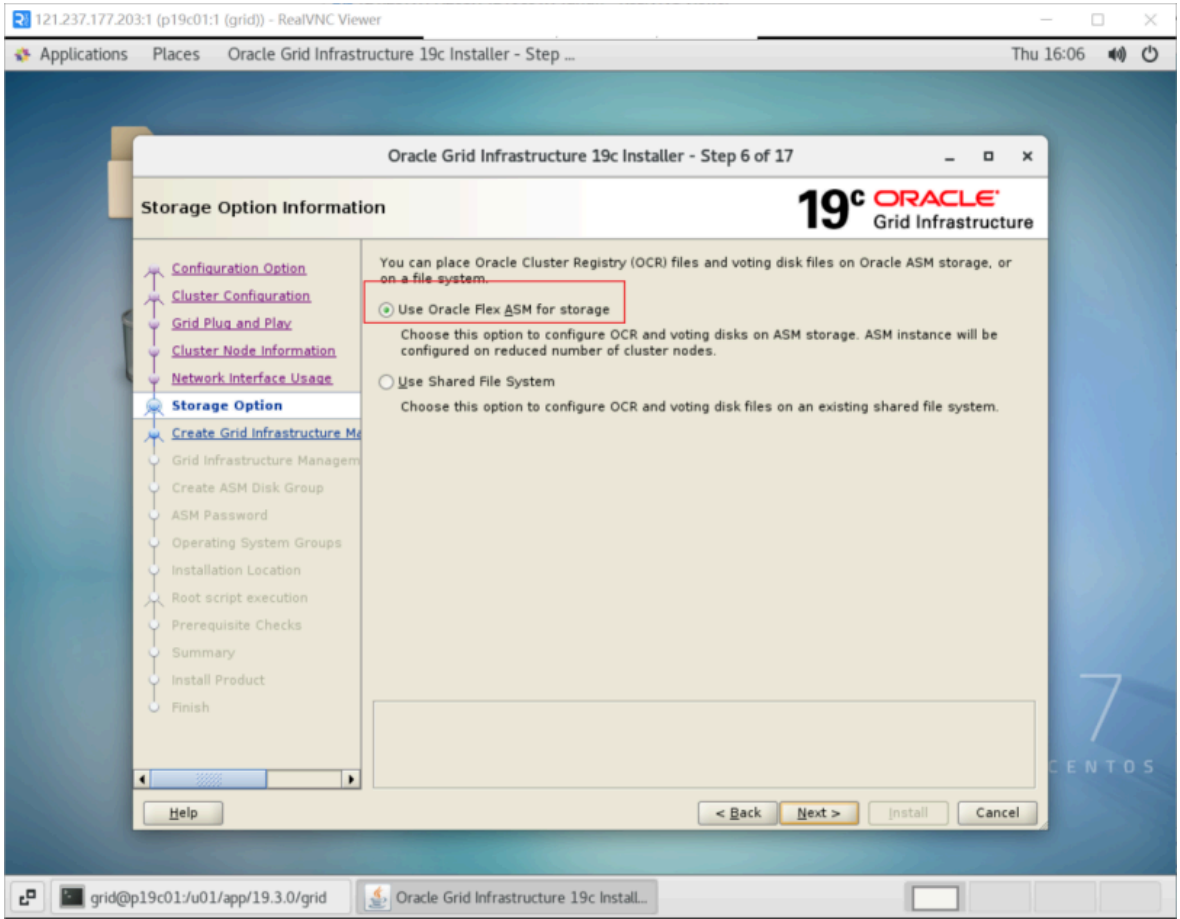
在网卡接口用途确认界面，修改私网网络用途为ASM&Private:

最佳实践



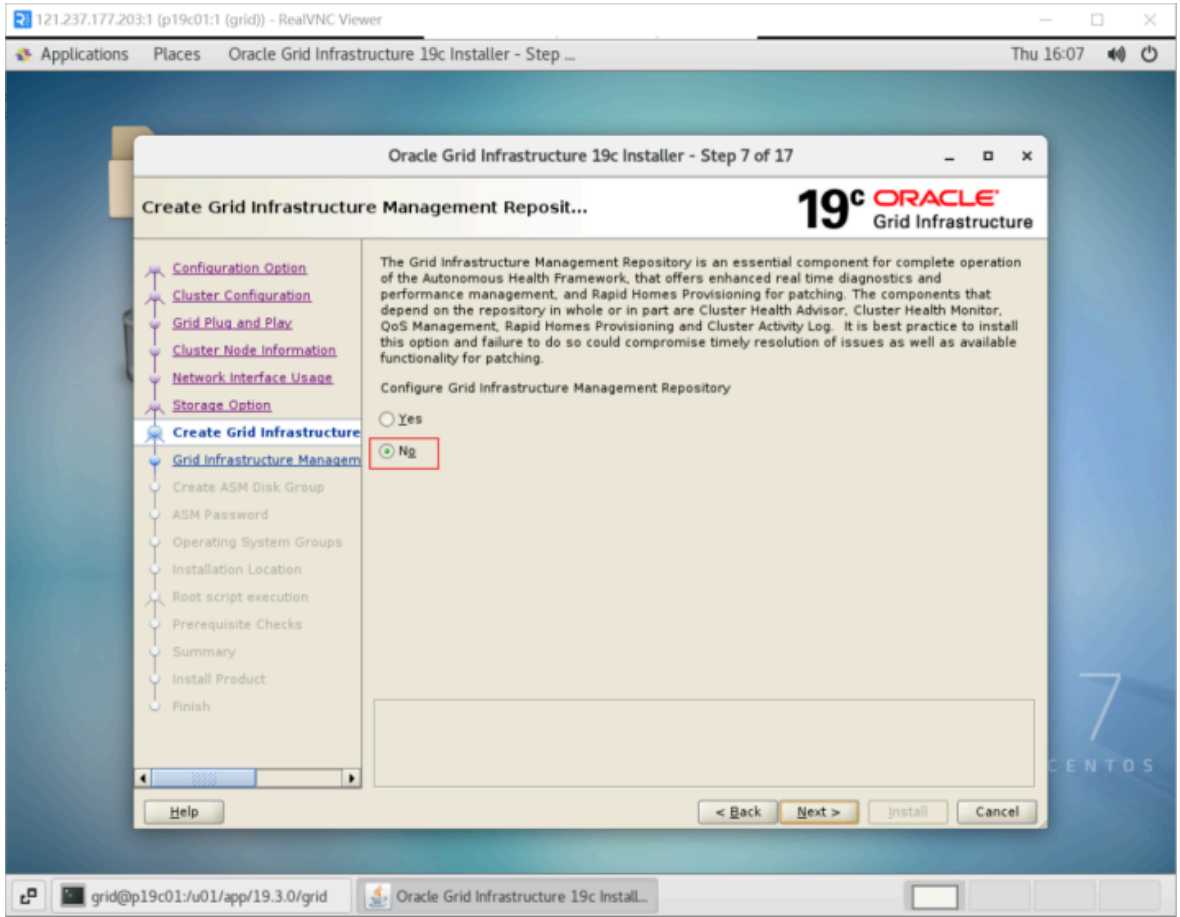
选择存储类型：

最佳实践



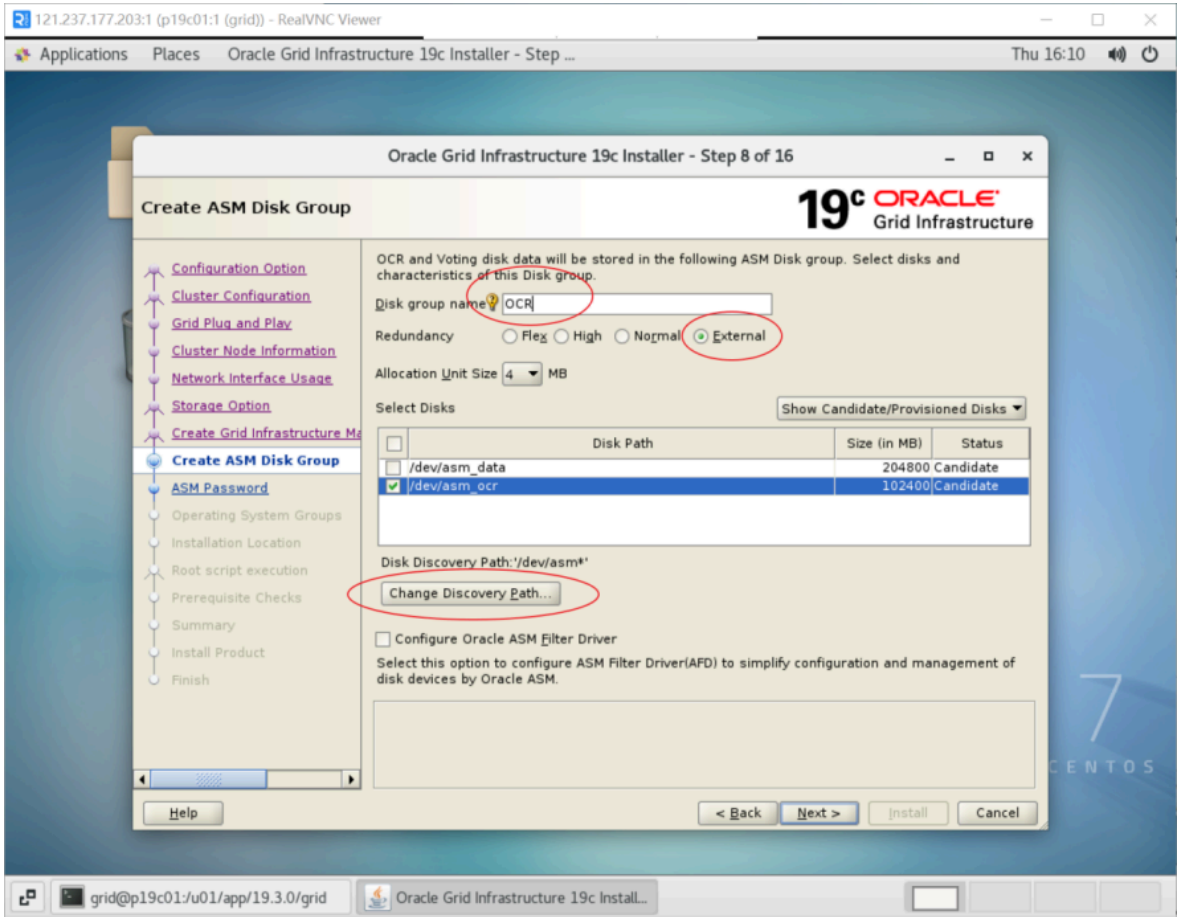
步骤GRIM采用默认不安装的选择:

最佳实践



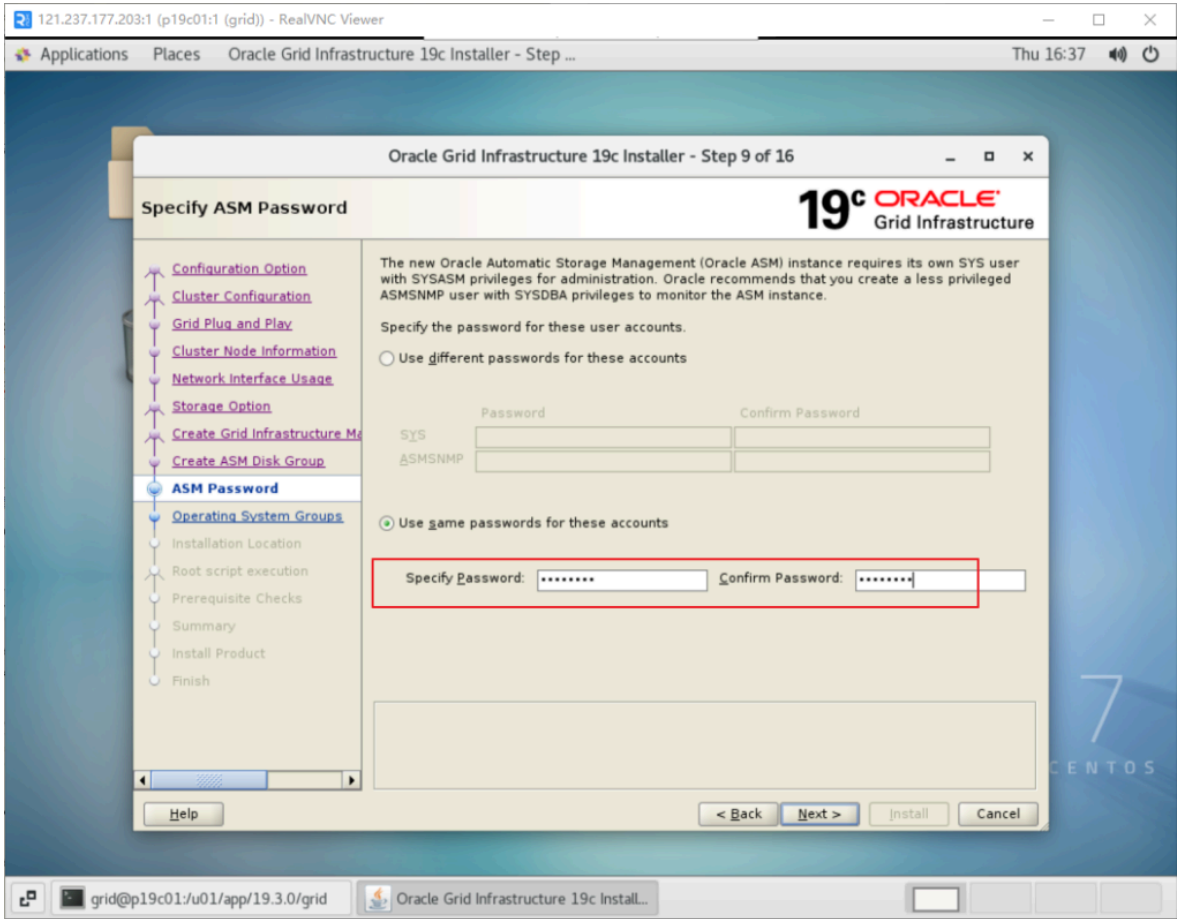
安装填写OCR盘，一块盘冗余External，目录选择udev绑定的路径：

最佳实践



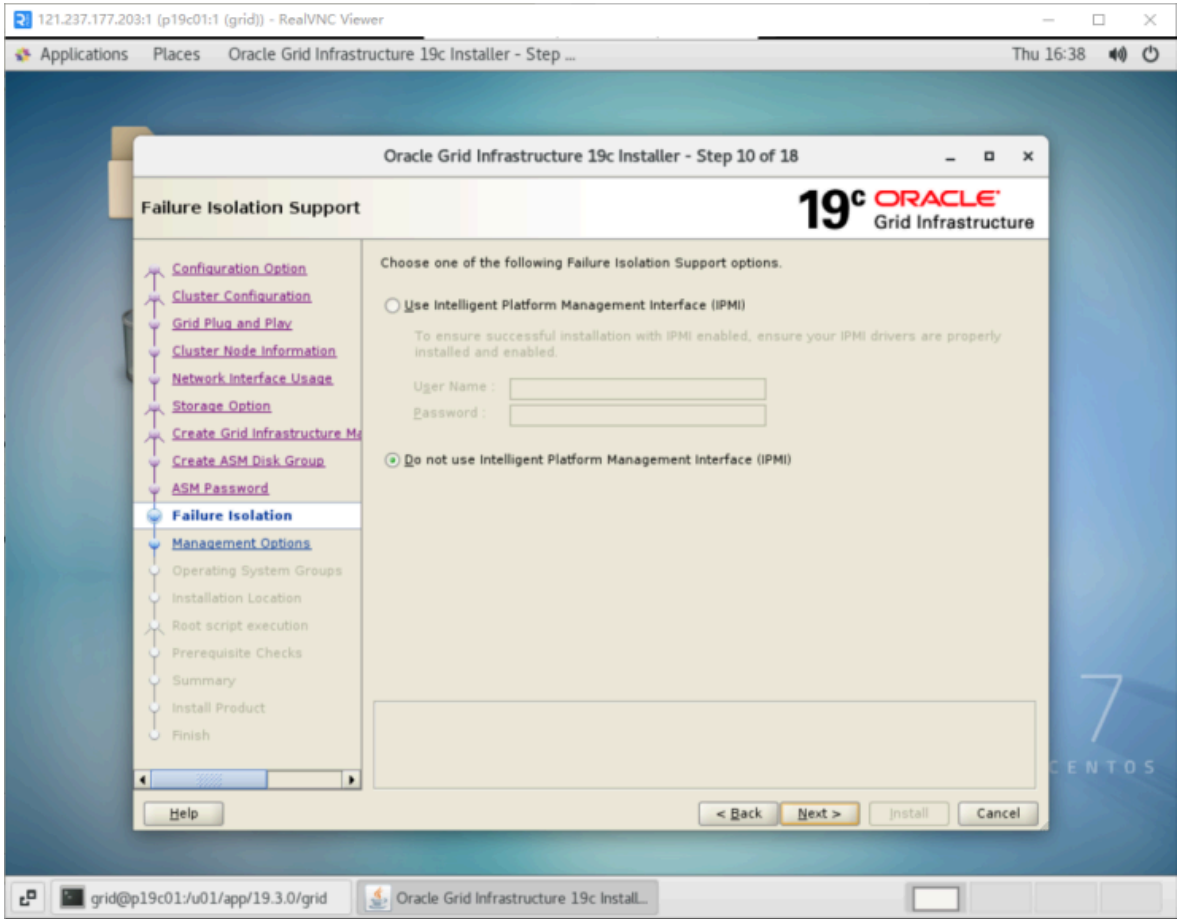
填写ASM密码:

最佳实践



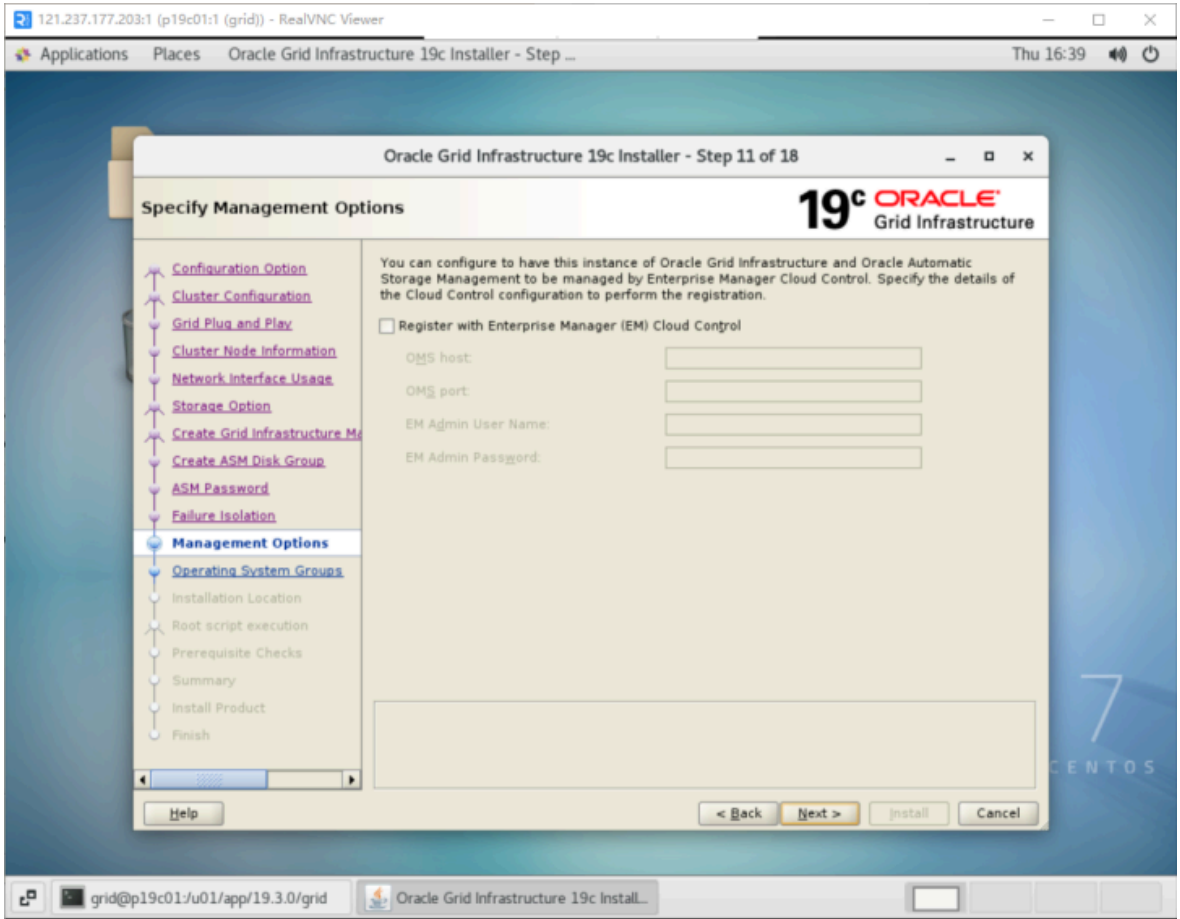
默认:

最佳实践



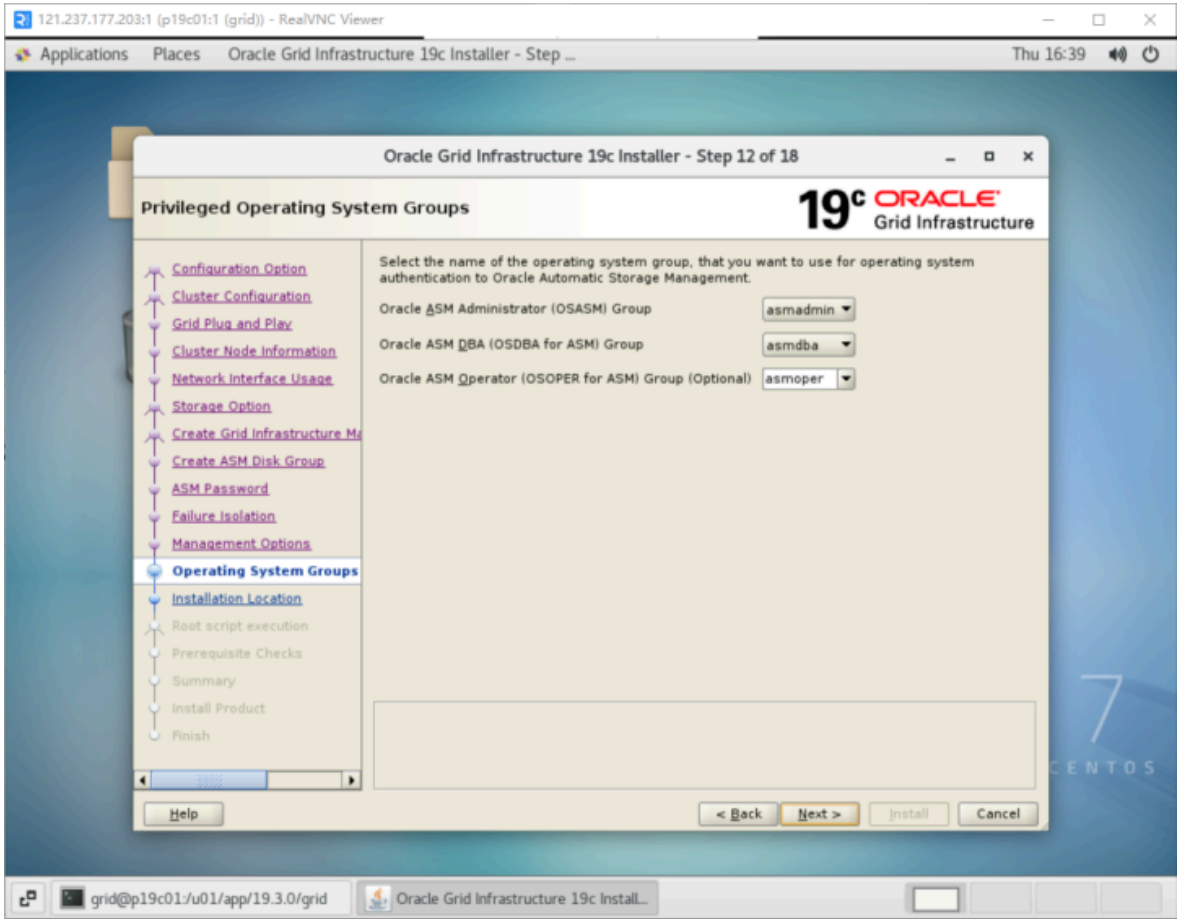
默认:

最佳实践



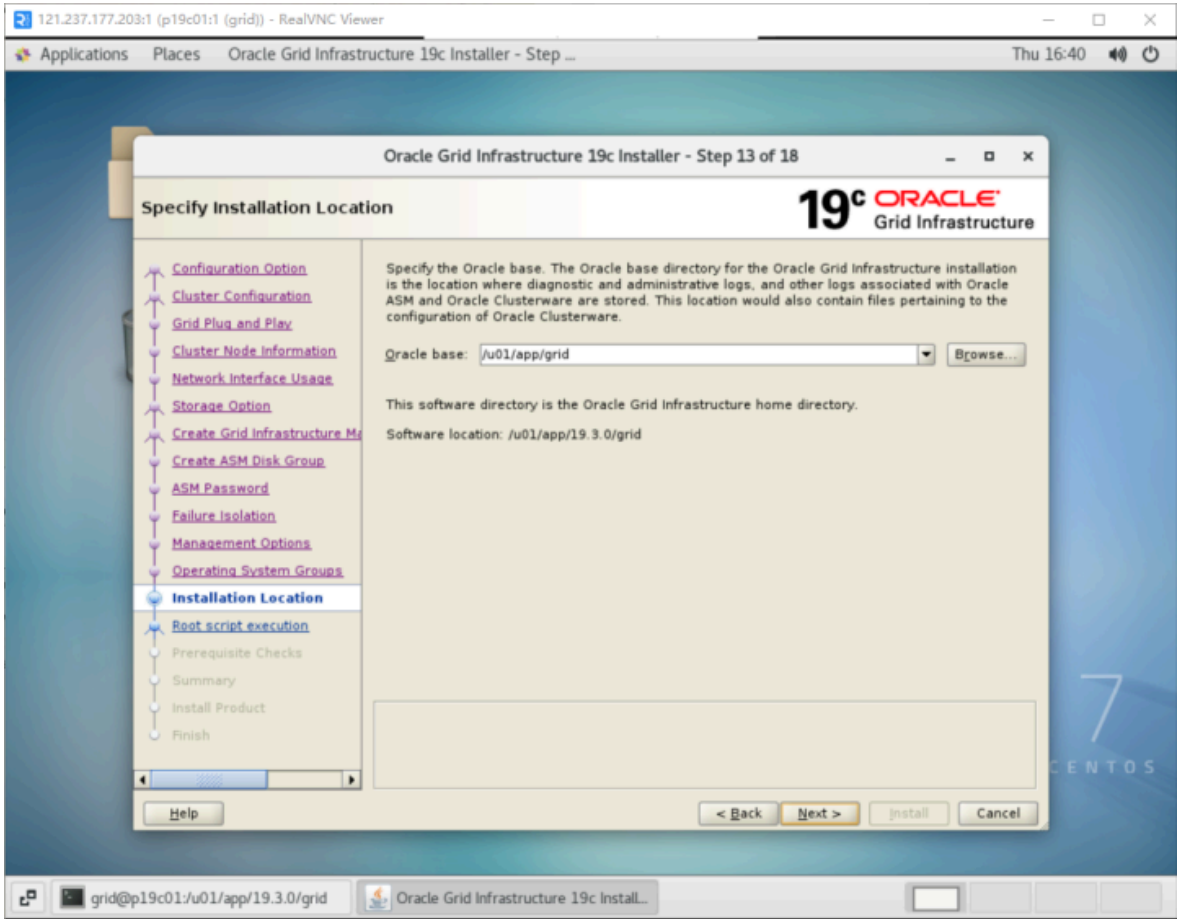
默认:

最佳实践



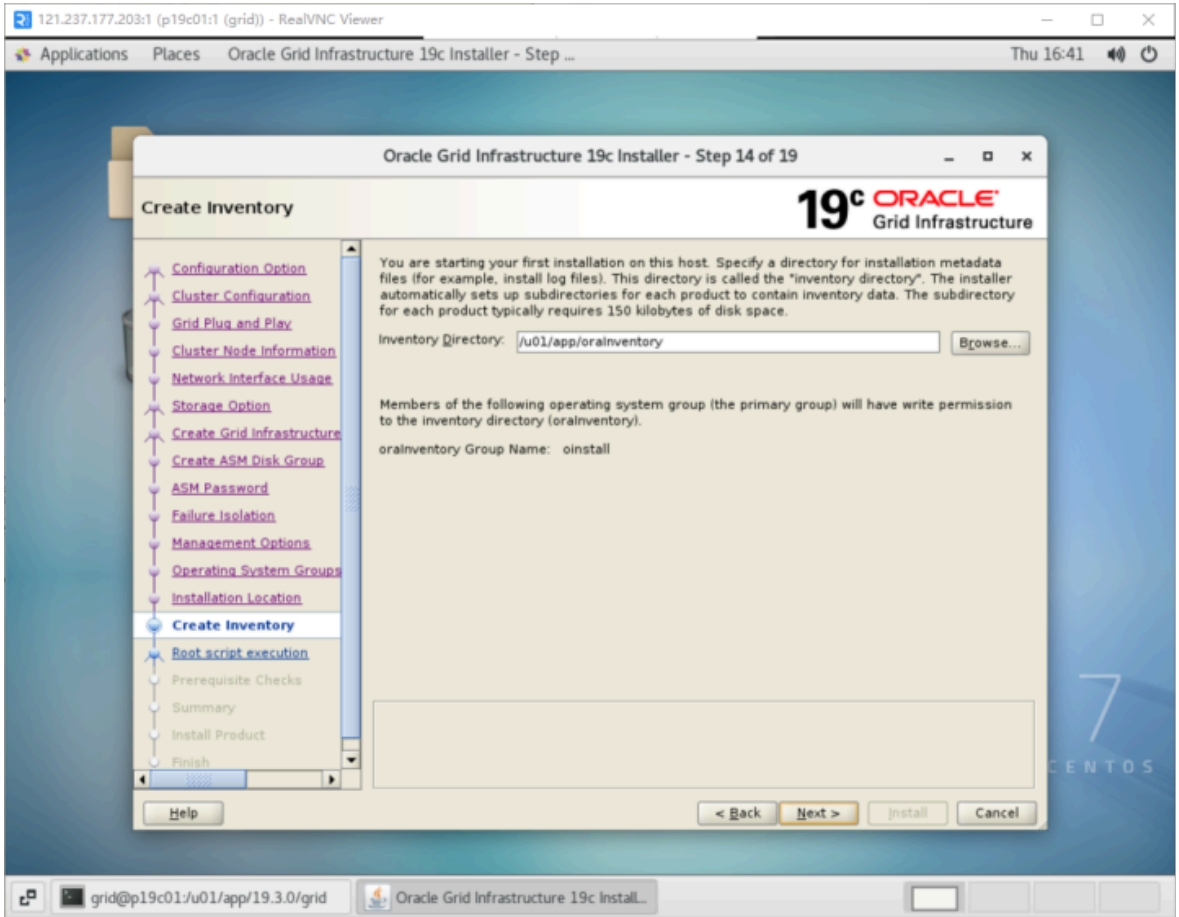
默认:

最佳实践



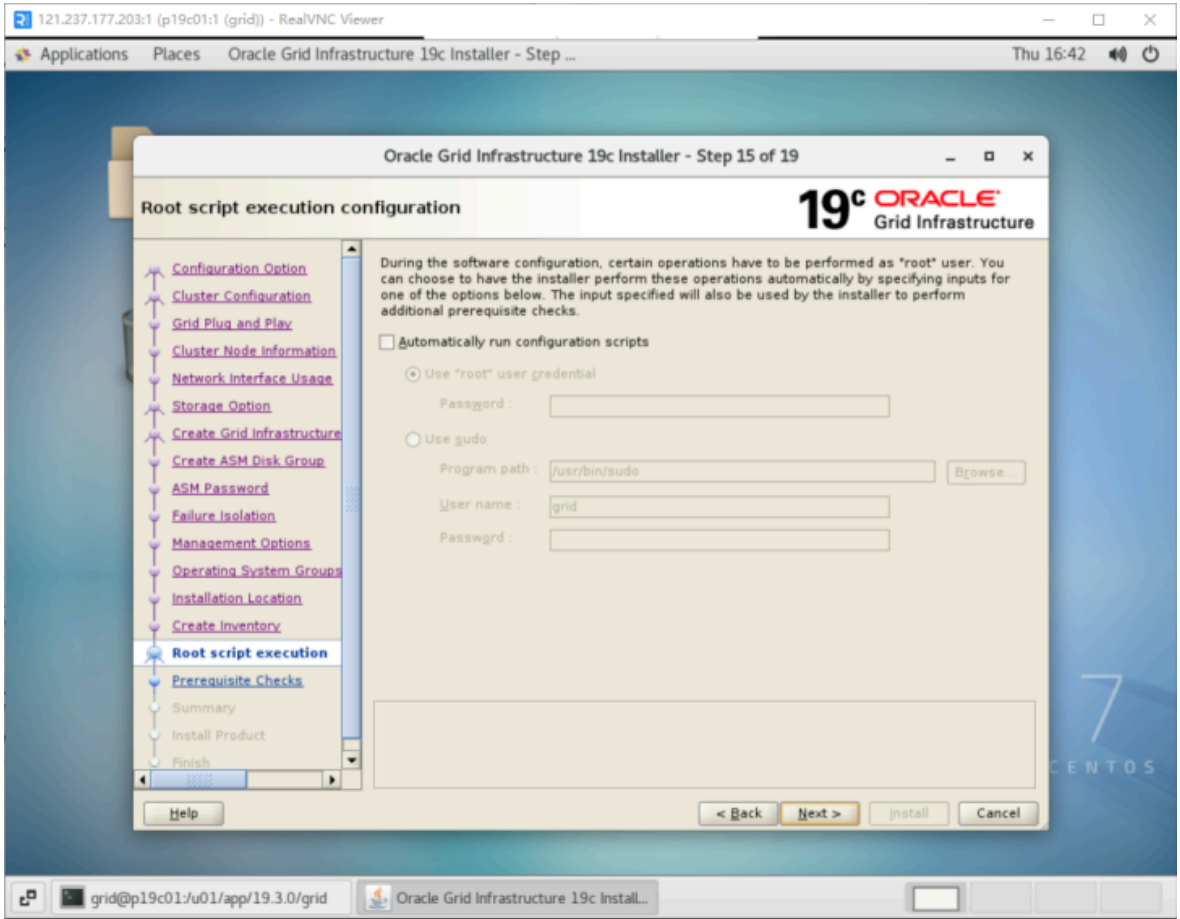
默认:

最佳实践



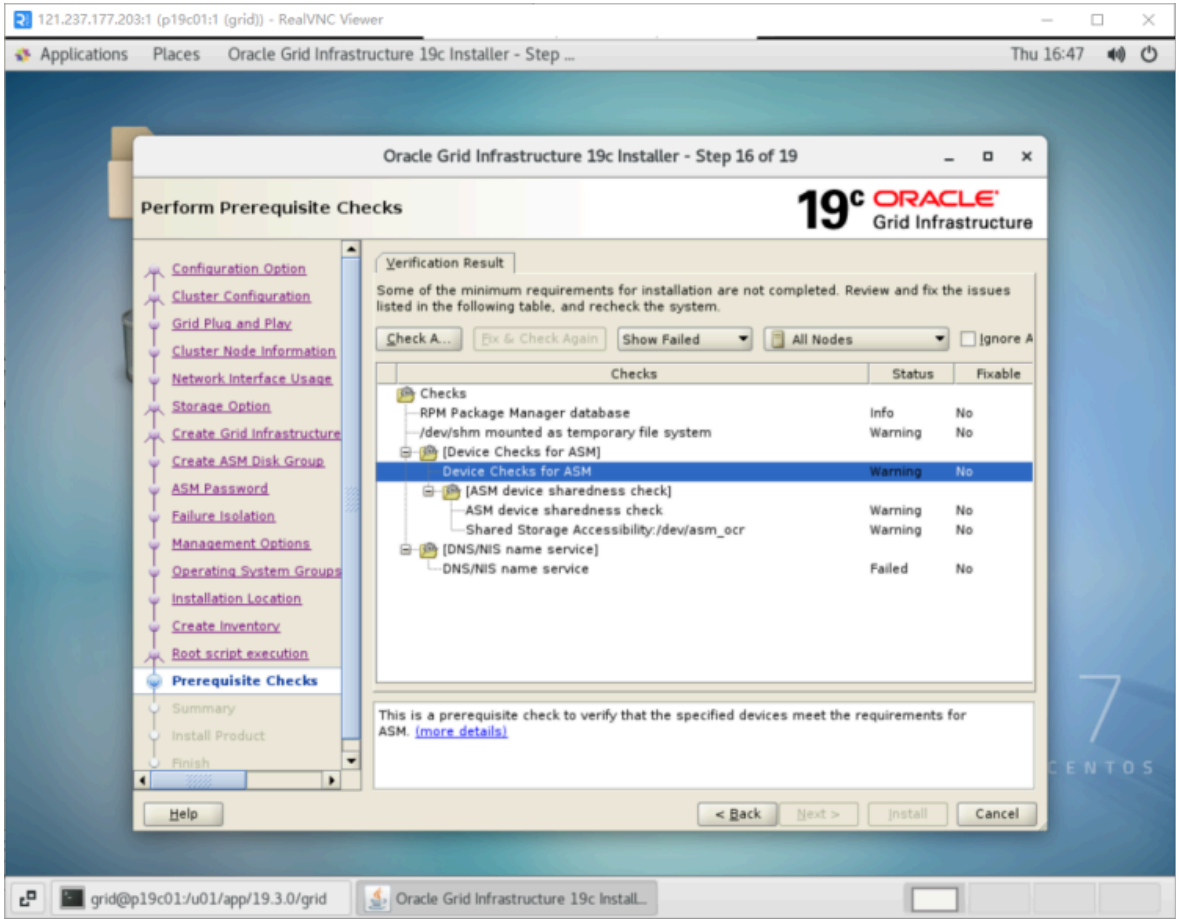
默认:

最佳实践



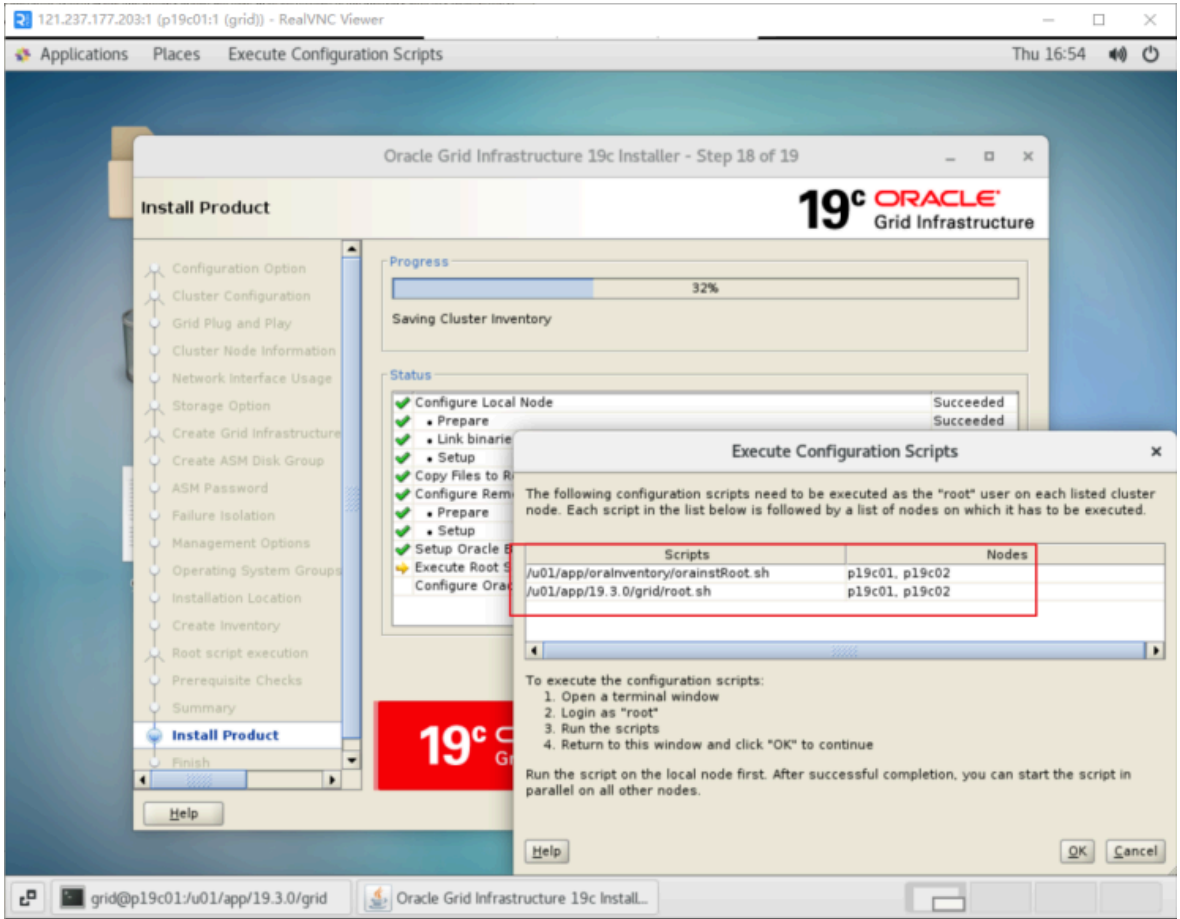
安装预检查结果:

最佳实践



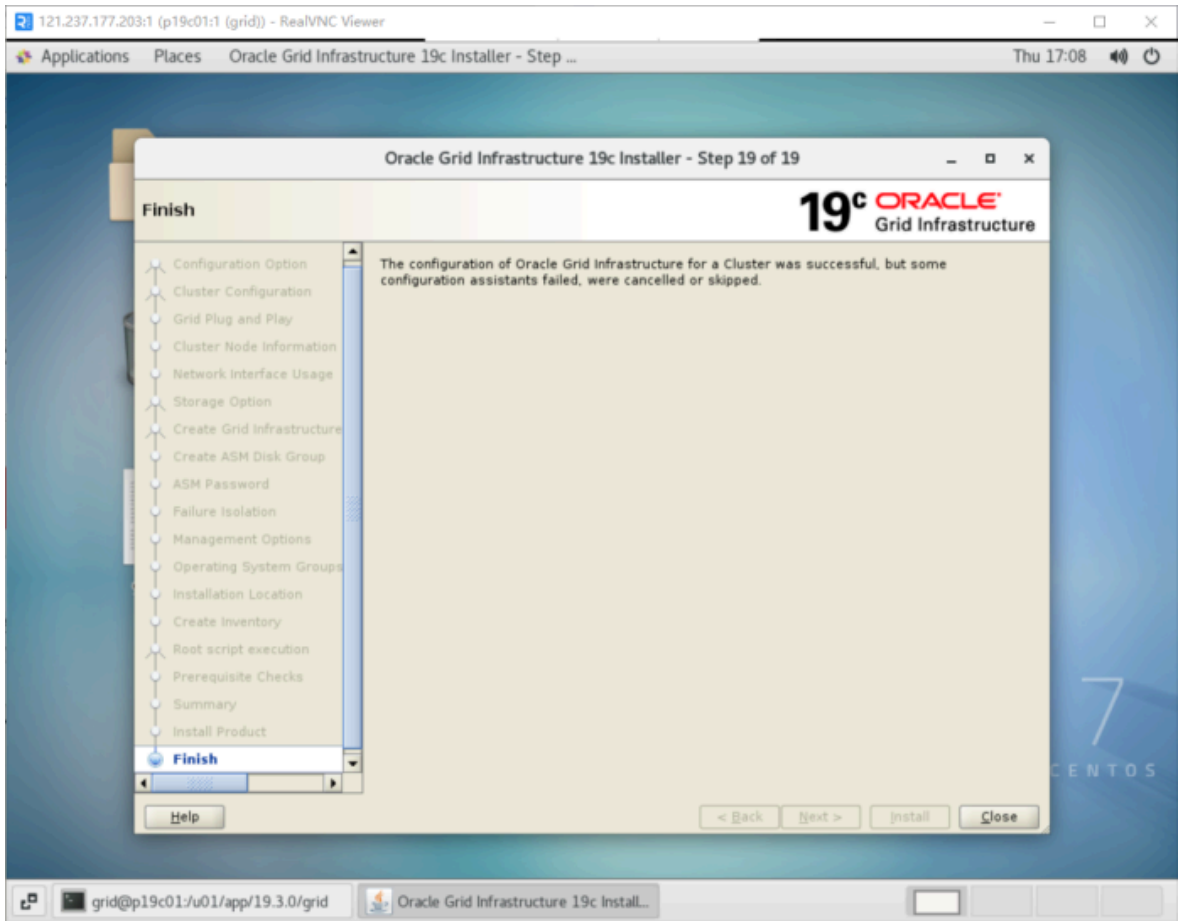
选择忽略按钮Ignore All，开始进行安装：

最佳实践



在两个节点上按顺序分别执行上述脚本：

最佳实践



这个时候查看服务状态发现只有一个节点是在线的：

```
[grid@p19c01:/home/grid]$ crsctl stat res -t
```

```
-----  
Name          Target State   Server          State details  
-----  
Local Resources  
-----  
ora.LISTENER.lsnr  
      ONLINE ONLINE    p19c01          STABLE  
      ONLINE ONLINE    p19c02          STABLE  
ora.chad  
      ONLINE ONLINE    p19c01          STABLE  
      ONLINE ONLINE    p19c02          STABLE  
ora.net1.network  
      ONLINE ONLINE    p19c01          STABLE  
      ONLINE ONLINE    p19c02          STABLE  
ora.ons  
      ONLINE ONLINE    p19c01          STABLE
```

最佳实践

ONLINE ONLINE p19c02 STABLE

Cluster Resources

```

ora.ASMNET1LSNR_ASM.lsnr(ora.asmgroup)
  1 ONLINE ONLINE p19c01 STABLE
  2 ONLINE ONLINE p19c02 STABLE
  3 OFFLINE OFFLINE STABLE
ora.LISTENER_SCAN1.lsnr
  1 ONLINE ONLINE p19c02 STABLE
ora.LISTENER_SCAN2.lsnr
  1 ONLINE ONLINE p19c01 STABLE
ora.LISTENER_SCAN3.lsnr
  1 ONLINE ONLINE p19c01 STABLE
ora.OCR.dg(ora.asmgroup)
  1 ONLINE ONLINE p19c01 STABLE
  2 OFFLINE OFFLINE STABLE
  3 OFFLINE OFFLINE STABLE
ora.asm(ora.asmgroup)
  1 ONLINE ONLINE p19c01 Started, STABLE
  2 ONLINE OFFLINE Instance Shutdown, ST
    ABLE
  3 OFFLINE OFFLINE STABLE
ora.asmnet1.asmnetwork(ora.asmgroup)
  1 ONLINE ONLINE p19c01 STABLE
  2 ONLINE ONLINE p19c02 STABLE
  3 OFFLINE OFFLINE STABLE
ora.cvu
  1 ONLINE ONLINE p19c01 STABLE
ora.p19c01.vip
  1 ONLINE ONLINE p19c01 STABLE
ora.p19c02.vip
  1 ONLINE ONLINE p19c02 STABLE
ora.qosmsserver
  1 ONLINE ONLINE p19c01 STABLE
ora.scan1.vip
  1 ONLINE ONLINE p19c02 STABLE
ora.scan2.vip
  1 ONLINE ONLINE p19c01 STABLE
ora.scan3.vip
  1 ONLINE ONLINE p19c01 STABLE

```

最佳实践

这是因为 2 个云主机节点的 169.254 网段不通，即 Oracle RAC 的 HAIP 不通。这里采用将 HAIP 禁用掉的方案来解决，分别在两个节点执行如下操作：

```
/u01/app/19.3.0/grid/bin/crsctl stop crs
```

```
/u01/app/19.3.0/grid/bin/crsctl start crs -excl -nocrs
```

```
/u01/app/19.3.0/grid/bin/crsctl modify res ora.cluster_interconnect.haip -attr "ENABLED=0" -init
```

```
/u01/app/19.3.0/grid/bin/crsctl modify res ora.asm -attr "START_DEPENDENCIES='hard(ora.cssd,ora.ctssd)pullup(ora.cssd,ora.ctssd)weak(ora.drivers.acfs)',STOP_DEPENDENCIES='hard(intermediate:ora.cssd)'" -init
```

```
/u01/app/19.3.0/grid/bin/crsctl stop crs
```

最后再重新两个节点的 crs，然后查看集群状态就正常了：

```
[grid@p19c01:/home/grid]$ crsctl stat res -t
```

Name	Target	State	Server	State details
------	--------	-------	--------	---------------

Local Resources

ora.LISTENER.lsnr				
	ONLINE	ONLINE	p19c01	STABLE
	ONLINE	ONLINE	p19c02	STABLE
ora.chad				
	ONLINE	ONLINE	p19c01	STABLE
	ONLINE	ONLINE	p19c02	STABLE
ora.net1.network				
	ONLINE	ONLINE	p19c01	STABLE
	ONLINE	ONLINE	p19c02	STABLE
ora.ons				
	ONLINE	ONLINE	p19c01	STABLE
	ONLINE	ONLINE	p19c02	STABLE

Cluster Resources

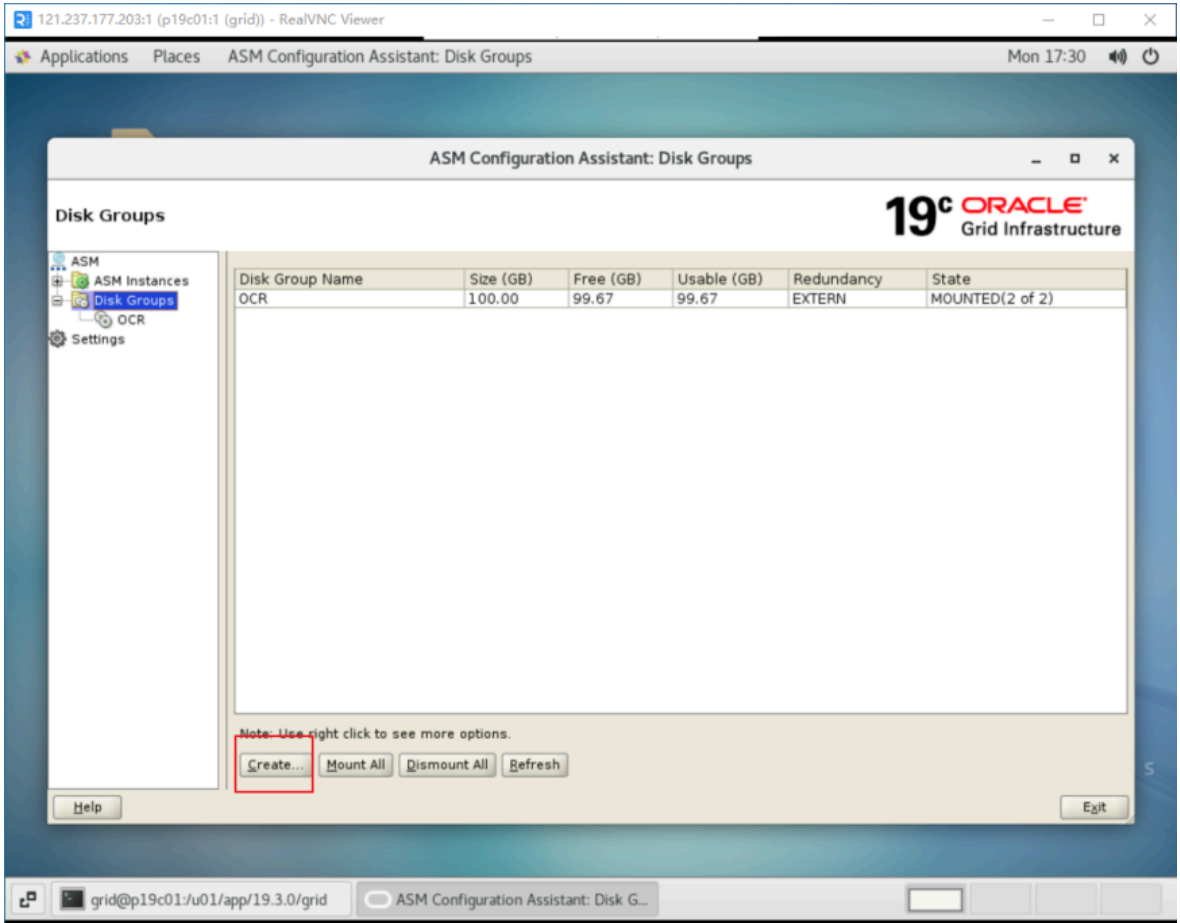
ora.ASMNET1LSNR_ASM.lsnr(ora.asmgrou)				
1	ONLINE	ONLINE	p19c01	STABLE
2	ONLINE	ONLINE	p19c02	STABLE
3	ONLINE	OFFLINE		STABLE
ora.LISTENER_SCAN1.lsnr				
1	ONLINE	ONLINE	p19c02	STABLE
ora.LISTENER_SCAN2.lsnr				
1	ONLINE	ONLINE	p19c01	STABLE

最佳实践

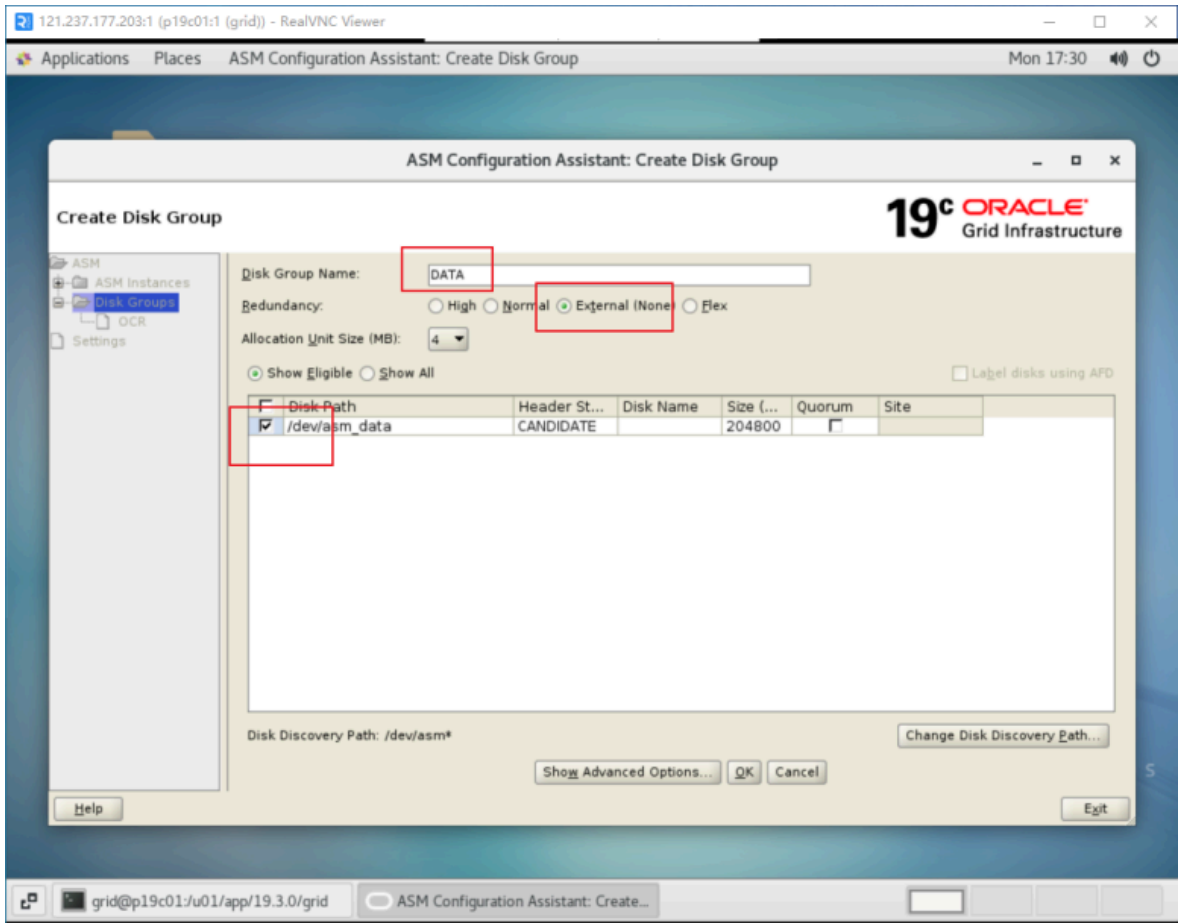
```
ora.LISTENER_SCAN3.lsnr
  1    ONLINE ONLINE    p19c01    STABLE
ora.OCR.dg(ora.asmgroup)
  1    ONLINE ONLINE    p19c01    STABLE
  2    ONLINE ONLINE    p19c02    STABLE
  3    OFFLINE OFFLINE          STABLE
ora.asm(ora.asmgroup)
  1    ONLINE ONLINE    p19c01    Started, STABLE
  2    ONLINE ONLINE    p19c02    Started, STABLE
  3    OFFLINE OFFLINE          STABLE
ora.asmnet1.asmnetwork(ora.asmgroup)
  1    ONLINE ONLINE    p19c01    STABLE
  2    ONLINE ONLINE    p19c02    STABLE
  3    OFFLINE OFFLINE          STABLE
ora.cvu
  1    ONLINE ONLINE    p19c01    STABLE
ora.p19c01.vip
  1    ONLINE ONLINE    p19c01    STABLE
ora.p19c02.vip
  1    ONLINE ONLINE    p19c02    STABLE
ora.qosmsserver
  1    ONLINE ONLINE    p19c01    STABLE
ora.scan1.vip
  1    ONLINE ONLINE    p19c02    STABLE
ora.scan2.vip
  1    ONLINE ONLINE    p19c01    STABLE
ora.scan3.vip
  1    ONLINE ONLINE    p19c01    STABLE
```

添加 ASM DATA 数据盘： 执行asmca弹出磁盘组管理界面：

最佳实践



最佳实践



查看添加成功:

```
[grid@p19c01:/home/grid]$ asmcmd lsdg
State Type Rebal Sector Logical_Sector Block AU Total_MB Free_MB Req_mir_free_MB
Usable_file_MB Offline_disks Voting_files Name
MOUNTED EXTERN N 512 512 4096 4194304 204800 204668 0 204668
0 N DATA/
MOUNTED EXTERN N 512 512 4096 4194304 102400 102064 0 102064
0 Y OCR/
```

安装Oracle软件

配置oracle用户vnc图形界面:

```
[root@p19c01 ~]# chown -R oracle:oinstall /soft
[root@p19c01 ~]# su - oracle
Last login: Mon May 27 15:54:16 CST 2024 on pts/0
[oracle@p19c01:/home/oracle]$
[oracle@p19c01:/home/oracle]$ vncserver
```

最佳实践

You will require a password to access your desktops.

Password:

Verify:

Would you like to enter a view-only password (y/n)? n

A view-only password is not used

xauth: file /home/oracle/.Xauthority does not exist

New 'p19c01:2 (oracle)' desktop is p19c01:2

Creating default startup script /home/oracle/.vnc/xstartup

Creating default config /home/oracle/.vnc/config

Starting applications specified in /home/oracle/.vnc/xstartup

Log file is /home/oracle/.vnc/p19c01:2.log

```
[oracle@p19c01:/home/oracle]$
```

```
[oracle@p19c01:/home/oracle]$ ps aux |grep vnc
```

```
oracle 1495 1.8 0.2 286620 46376 pts/0 S1 17:33 0:00 /bin/Xvnc :2 -auth  
/home/oracle/.Xauthority -desktop p19c01:2 (oracle) -fp catalogue:/etc/X11/fontpath.d -  
geometry 1024x768 -httpd /usr/share/vnc/classes -pn -rfbauth /home/oracle/.vnc/passwd -  
rfbport 5902 -rfbwait 30000
```

```
oracle 1523 0.0 0.0 113288 1200 pts/0 S 17:33 0:00 /bin/sh /home/oracle/.vnc/xstartup
```

```
oracle 2262 0.0 0.0 112820 984 pts/0 S+ 17:34 0:00 grep --color=auto vnc
```

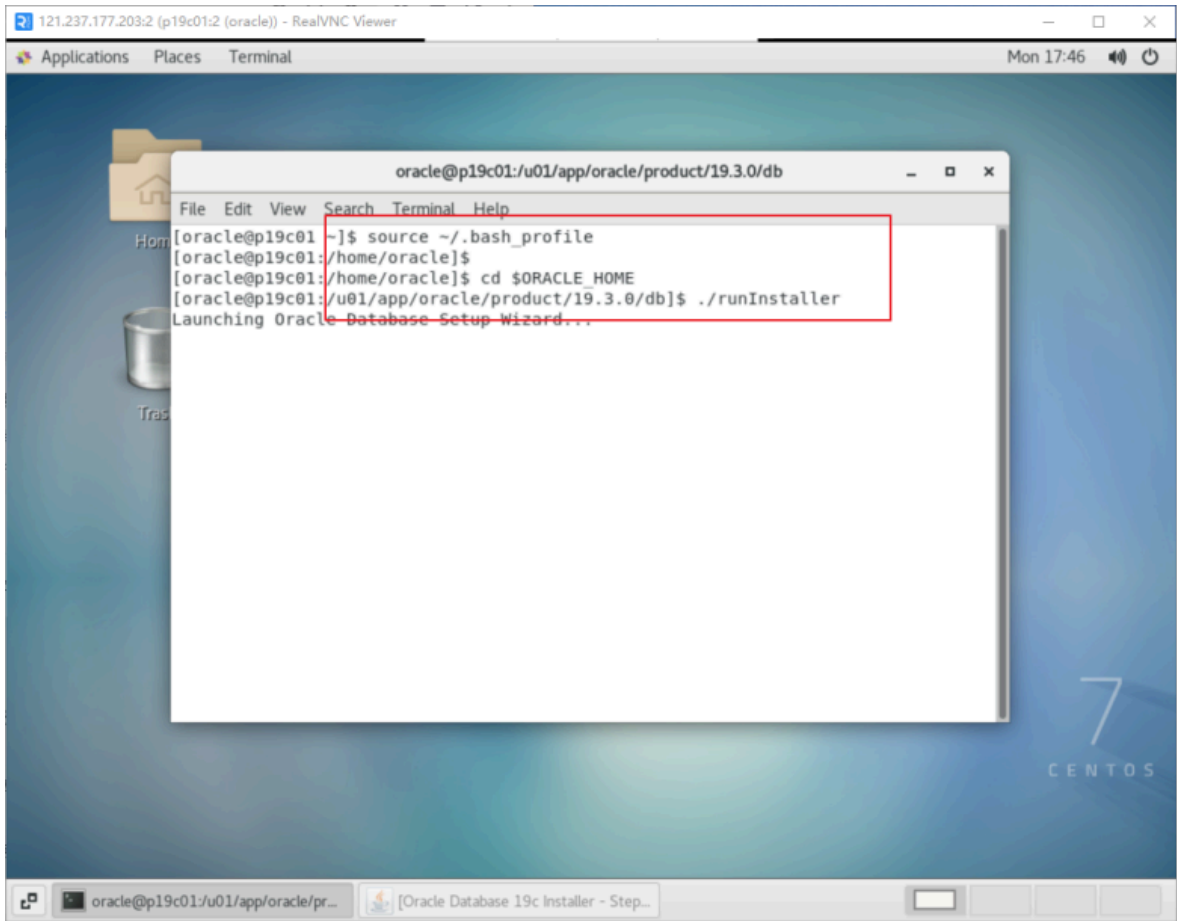
```
grid 20009 0.0 0.3 305292 54360 pts/0 S1 15:56 0:04 /bin/Xvnc :1 -auth  
/home/grid/.Xauthority -desktop p19c01:1 (grid) -fp catalogue:/etc/X11/fontpath.d -  
geometry 1024x768 -httpd /usr/share/vnc/classes -pn -rfbauth /home/grid/.vnc/passwd -  
rfbport 5901 -rfbwait 30000
```

```
grid 20022 0.0 0.0 113288 1200 pts/0 S 15:56 0:00 /bin/sh /home/grid/.vnc/xstartup
```

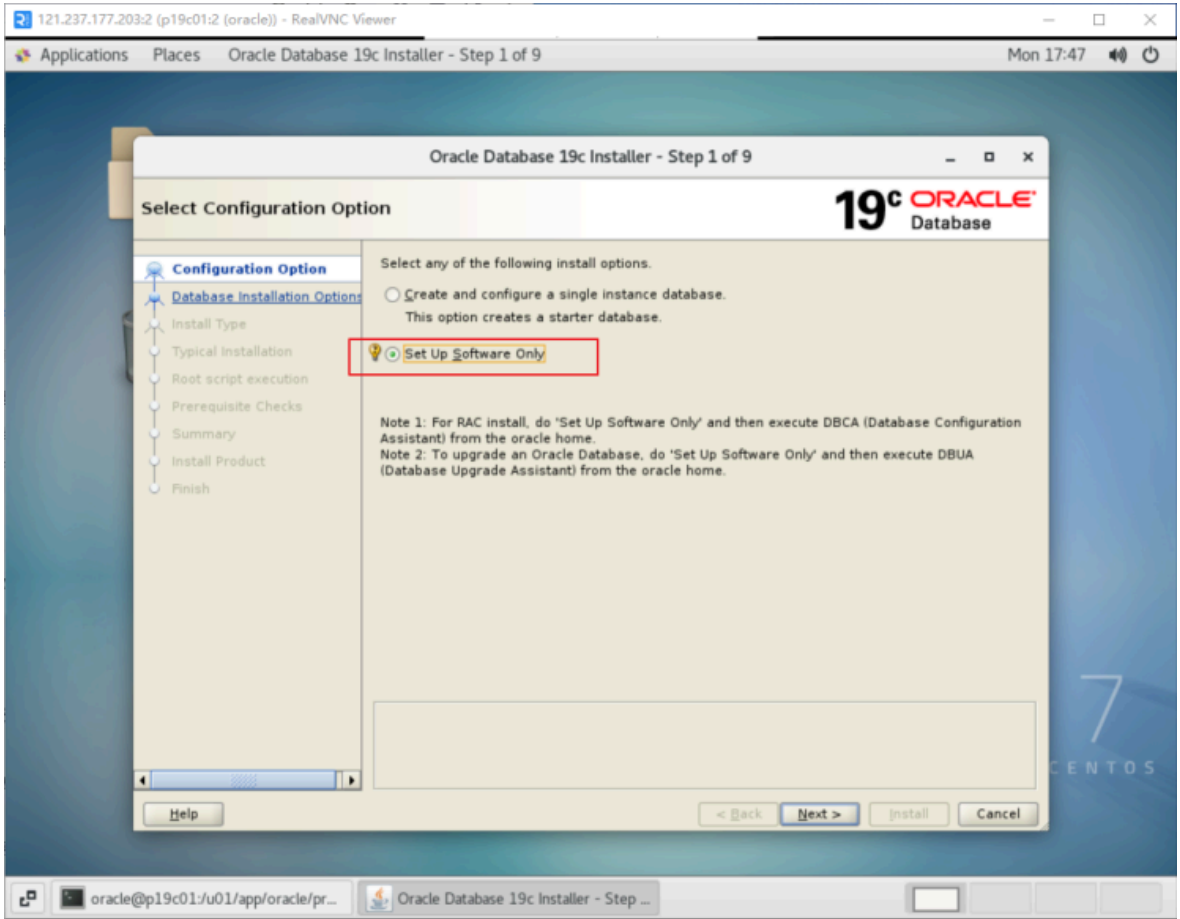
```
[oracle@p19c01:/home/oracle]$
```

在vnc客户端输入弹性IP加上端口2，接入图形界面。

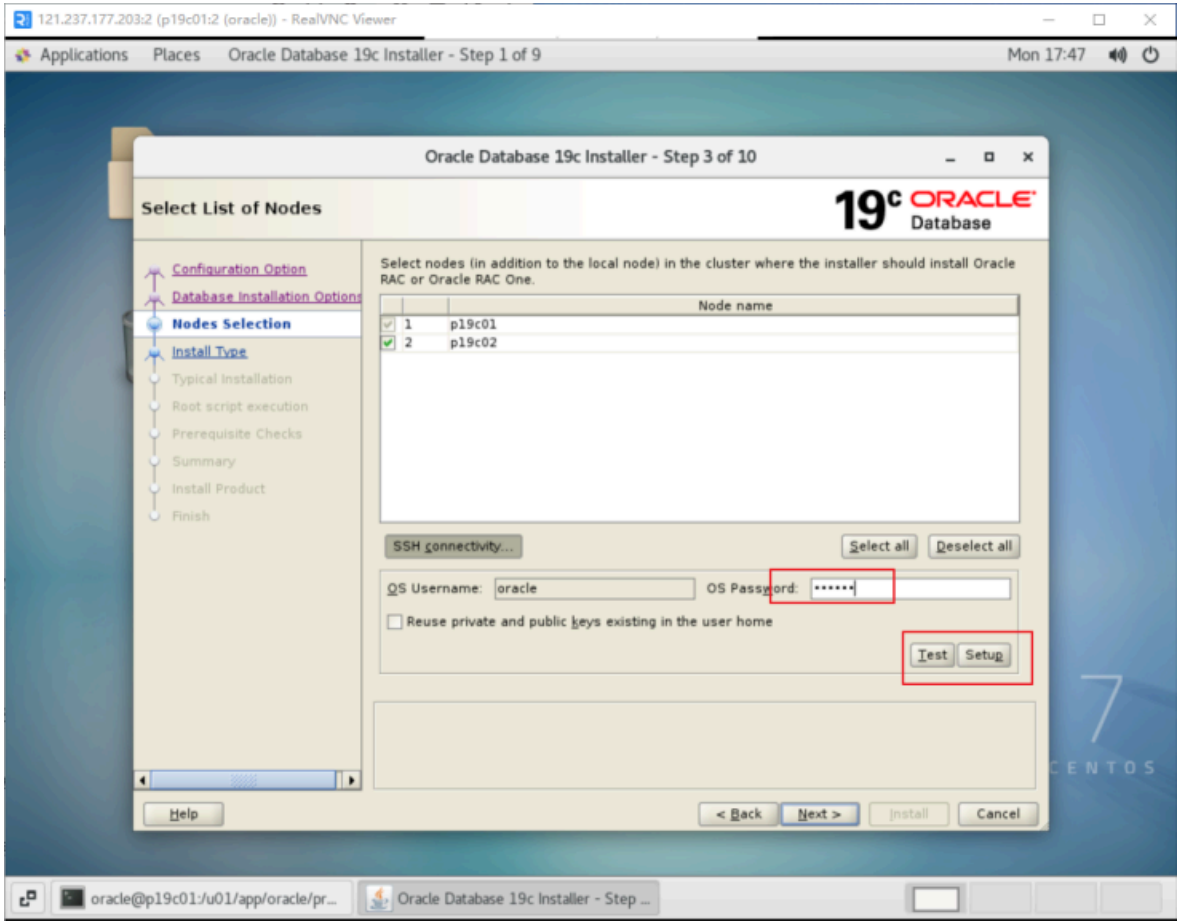
最佳实践



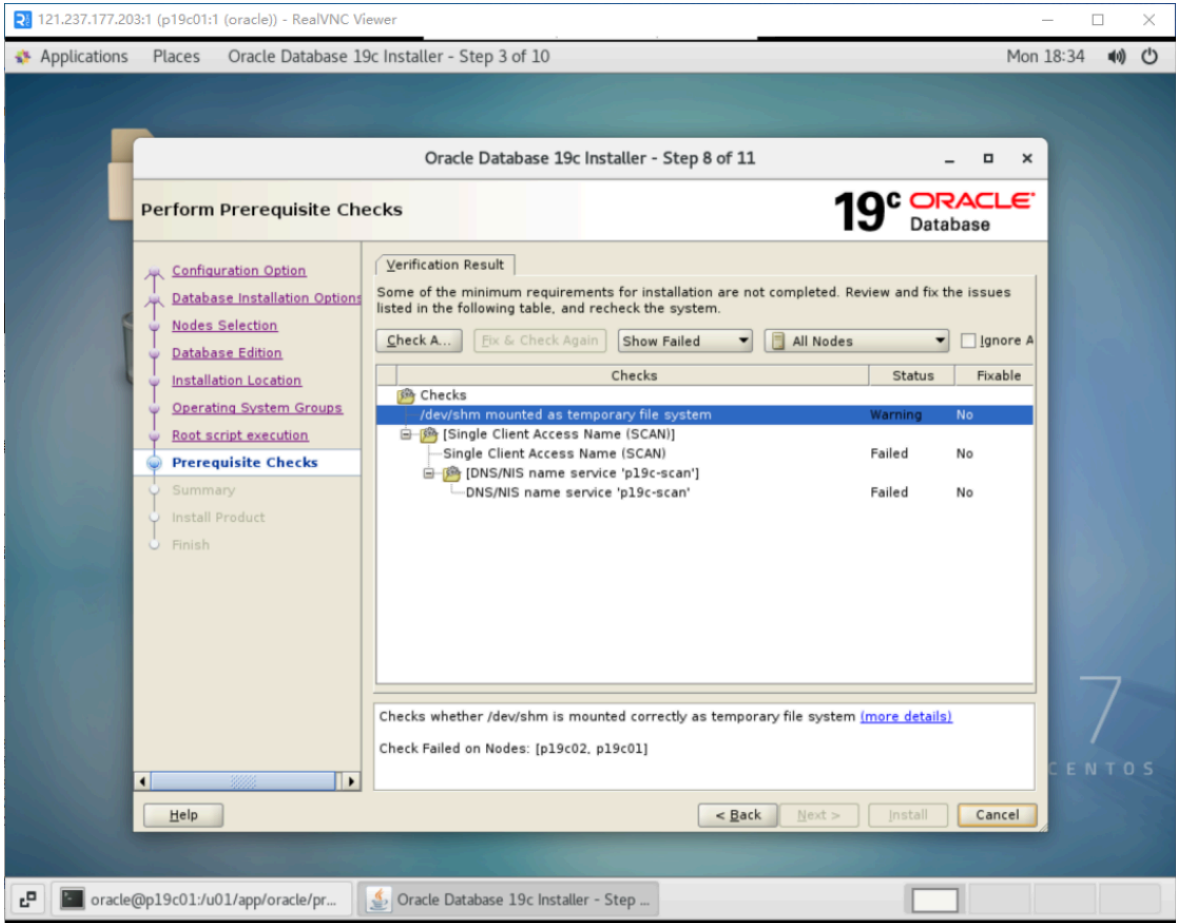
最佳实践

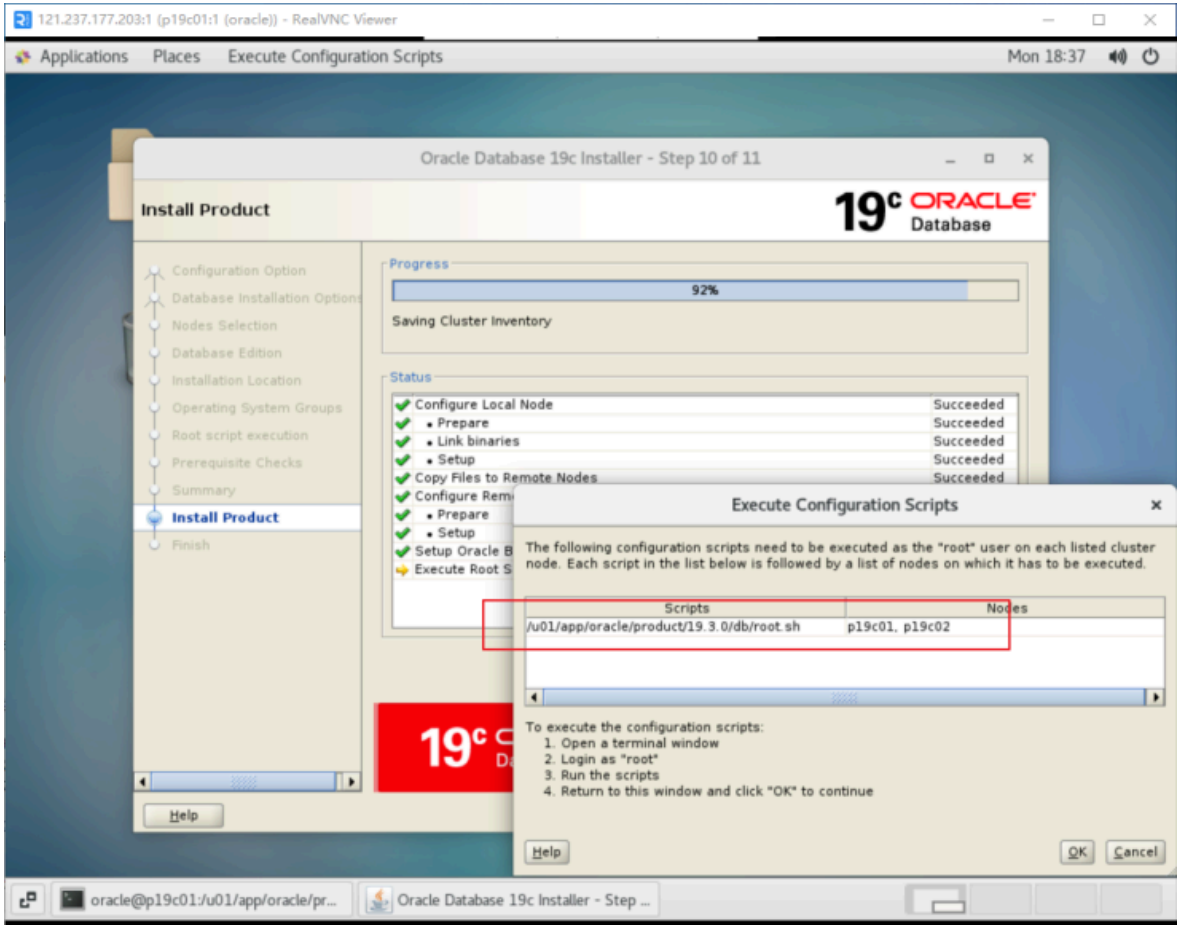


最佳实践



最佳实践



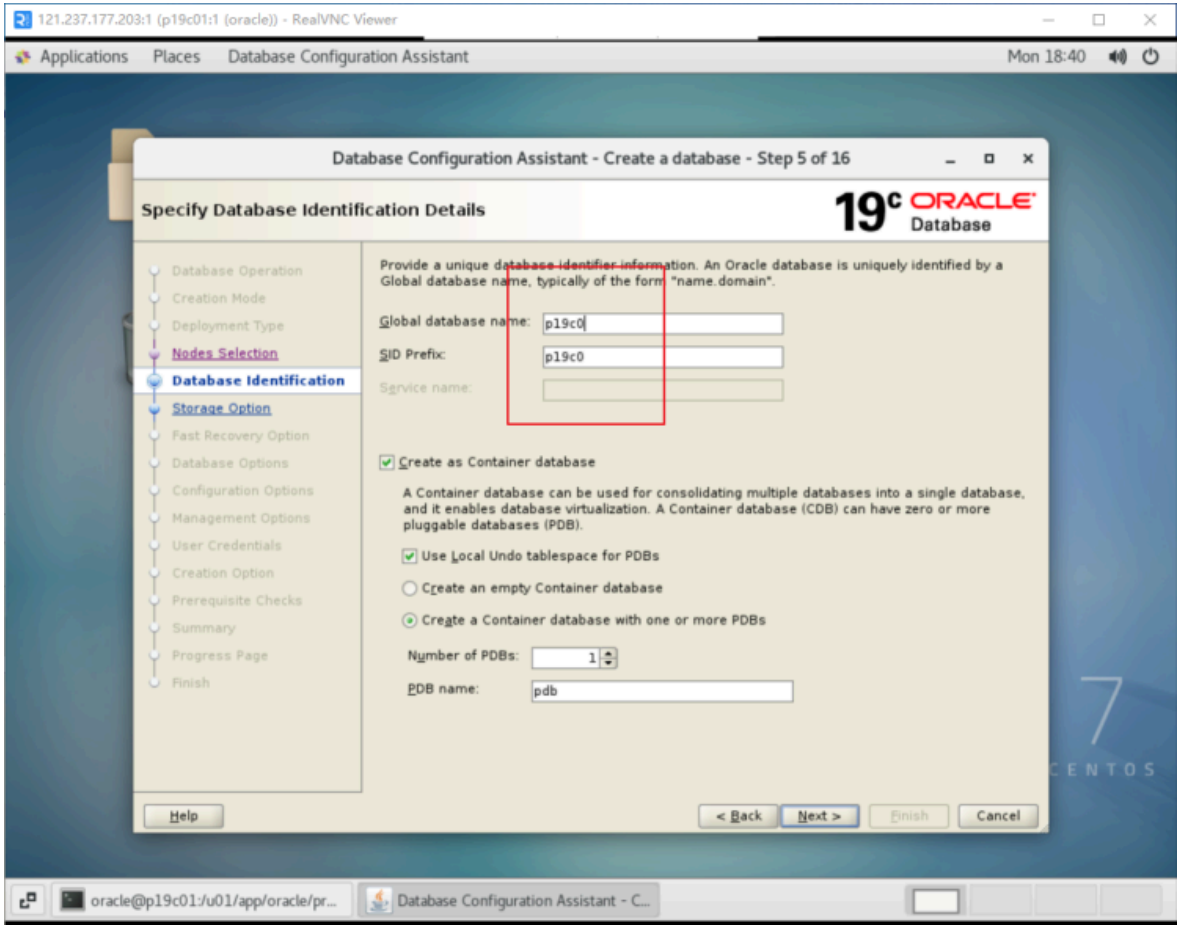


分别在两个节点上执行配置脚本，完成数据库的安装。

创建数据库实例

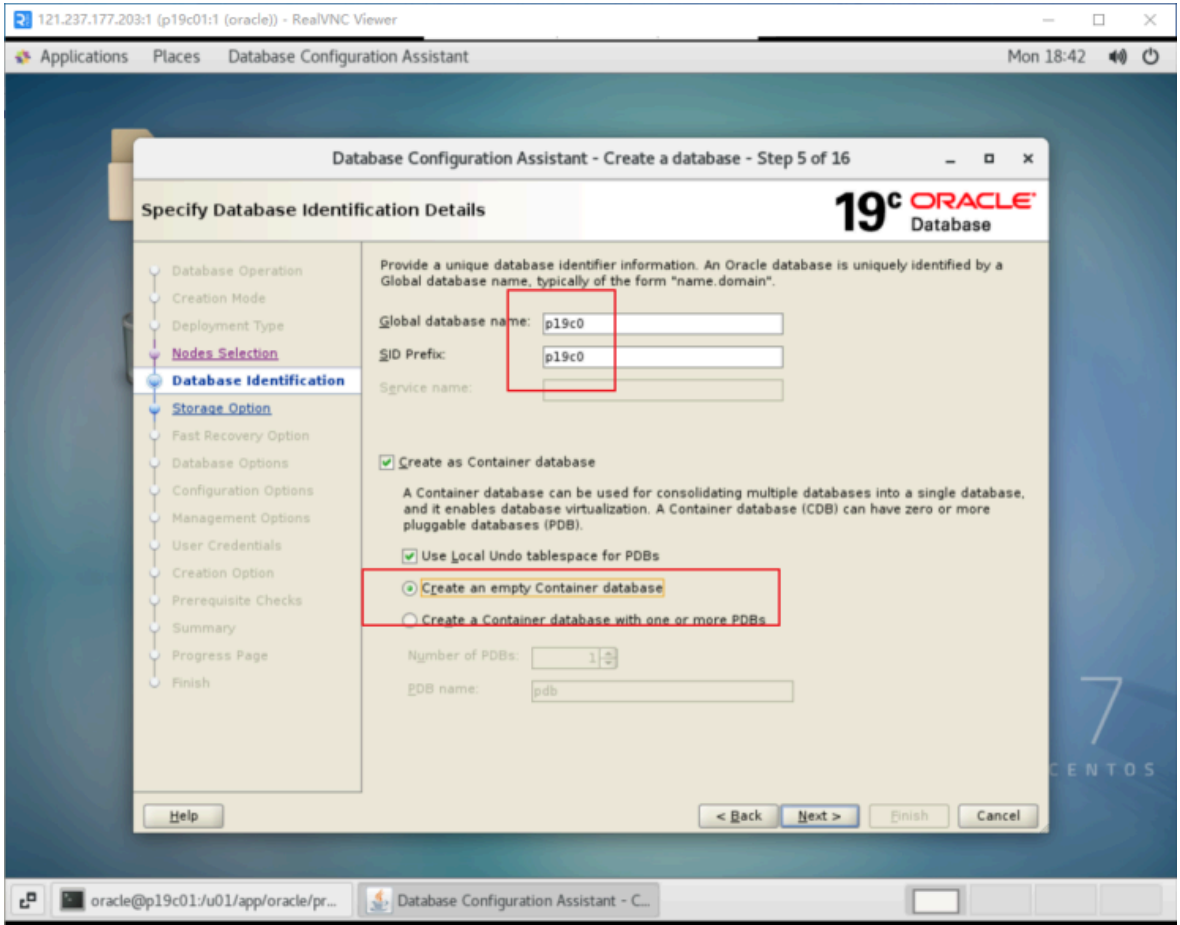
执行dbca命令弹出创建数据库界面：

最佳实践



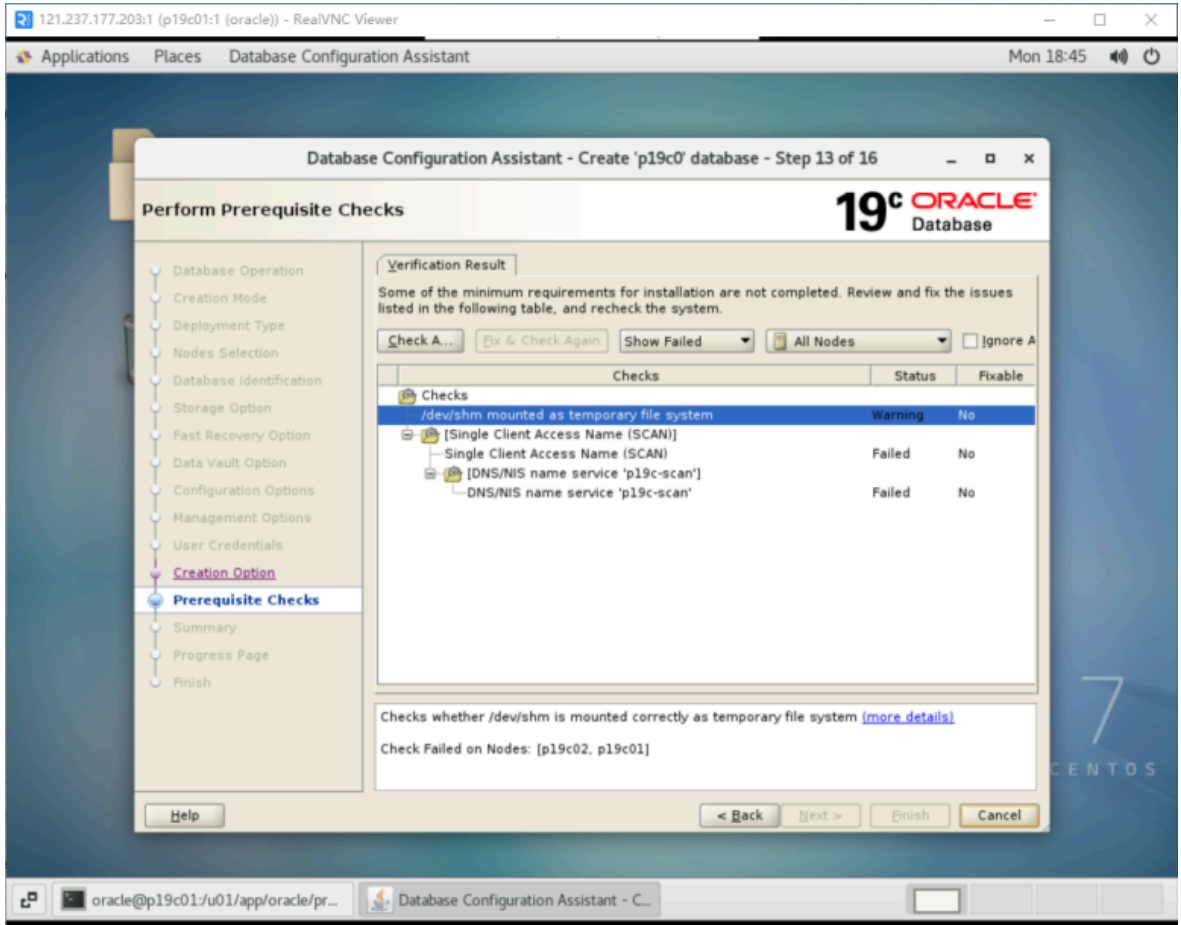
填写实例名称:

最佳实践



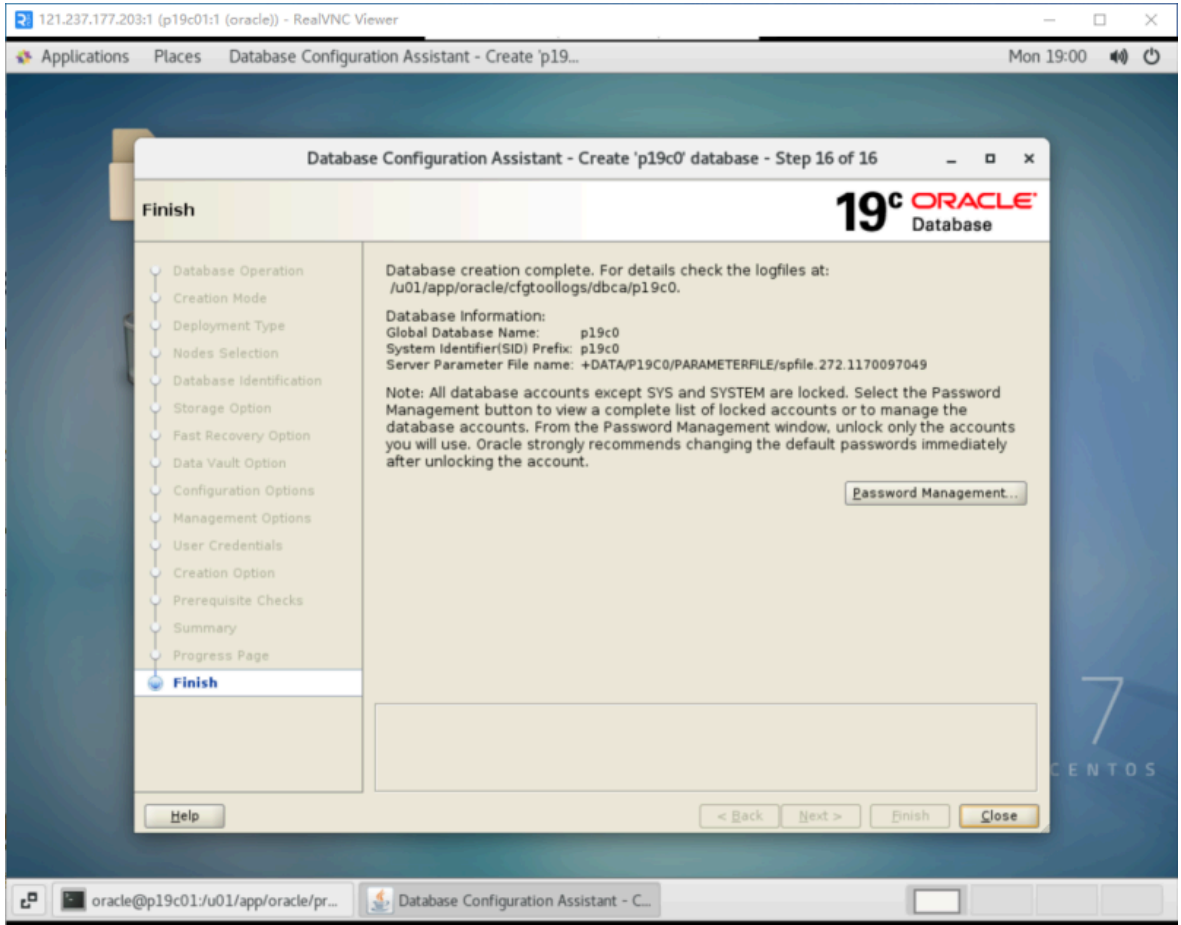
安装预检查，忽略：

最佳实践



完成数据库的创建:

最佳实践



```

1 liusu-huadong1-rc1
Every 2.0s: crsctl stat res -t
Mon May 27 19:00:40 2024
-----
Name          Target State   Server          State details
-----
Local Resources
-----
ora.LISTENER.lsnr
  ONLINE ONLINE p19c01 STABLE
  ONLINE ONLINE p19c02 STABLE
ora.chad      ONLINE ONLINE p19c01 STABLE
              ONLINE ONLINE p19c02 STABLE
ora.net1.network
  ONLINE ONLINE p19c01 STABLE
  ONLINE ONLINE p19c02 STABLE
ora.ons       ONLINE ONLINE p19c01 STABLE
              ONLINE ONLINE p19c02 STABLE
-----
Cluster Resources
-----
ora.ASHNET1LSNR_ASM.lsnr(ora.asmgroup)
  1 ONLINE ONLINE p19c01 STABLE
  2 ONLINE ONLINE p19c02 STABLE
  3 ONLINE OFFLINE STABLE
ora.DATA.dg(ora.asmgroup)
  1 ONLINE ONLINE p19c01 STABLE
  2 ONLINE ONLINE p19c02 STABLE
  3 OFFLINE OFFLINE STABLE
ora.LISTENER_SCAN1.lsnr
  1 ONLINE ONLINE p19c01 STABLE
ora.LISTENER_SCAN2.lsnr
  1 ONLINE ONLINE p19c02 STABLE
ora.LISTENER_SCAN3.lsnr
  1 ONLINE ONLINE p19c02 STABLE
ora.OCR.dg(ora.asmgroup)
  1 ONLINE ONLINE p19c01 STABLE
  2 ONLINE ONLINE p19c02 STABLE
  3 OFFLINE OFFLINE STABLE
ora.asm(ora.asmgroup)
  1 ONLINE ONLINE p19c01 Started,STABLE
  2 ONLINE ONLINE p19c02 Started,STABLE
  3 OFFLINE OFFLINE STABLE
ora.asmnet1.asmnetwork(ora.asmgroup)
  1 ONLINE ONLINE p19c01 STABLE
  2 ONLINE ONLINE p19c02 STABLE
  3 OFFLINE OFFLINE STABLE
ora.cvu       ONLINE ONLINE p19c02 STABLE
ora.p19c0.db
  1 ONLINE ONLINE p19c01 Open,HOME=/u01/app/oracle/product/19.3.0/db,STABLE
  2 ONLINE ONLINE p19c02 Open,HOME=/u01/app/oracle/product/19.3.0/db,STABLE
ora.p19c01.vip
  1 ONLINE ONLINE p19c01 STABLE
ora.p19c02.vip
  1 ONLINE ONLINE p19c02 STABLE
ora.qosmserver
  1 ONLINE ONLINE p19c02 STABLE
ora.scan1.vip
  1 ONLINE ONLINE p19c01 STABLE
ora.scan2.vip
  1 ONLINE ONLINE p19c02 STABLE
ora.scan3.vip
  1 ONLINE ONLINE p19c02 STABLE
-----

```

云主机迁移相关实践

跨账号同资源池迁移云主机

操作场景

本文通过采用镜像迁移方式实现跨账号同资源池迁移云主机，以实现跨账号同资源池使用镜像快速创建相同配置的弹性云主机。

方案介绍

镜像迁移方式实现跨账号同资源池迁移云主机的方案为：账号A将资源池A的云主机做成私有镜像，再共享给账号B；账号B接受账号A的共享镜像后，使用该镜像创建新的云主机。


说明

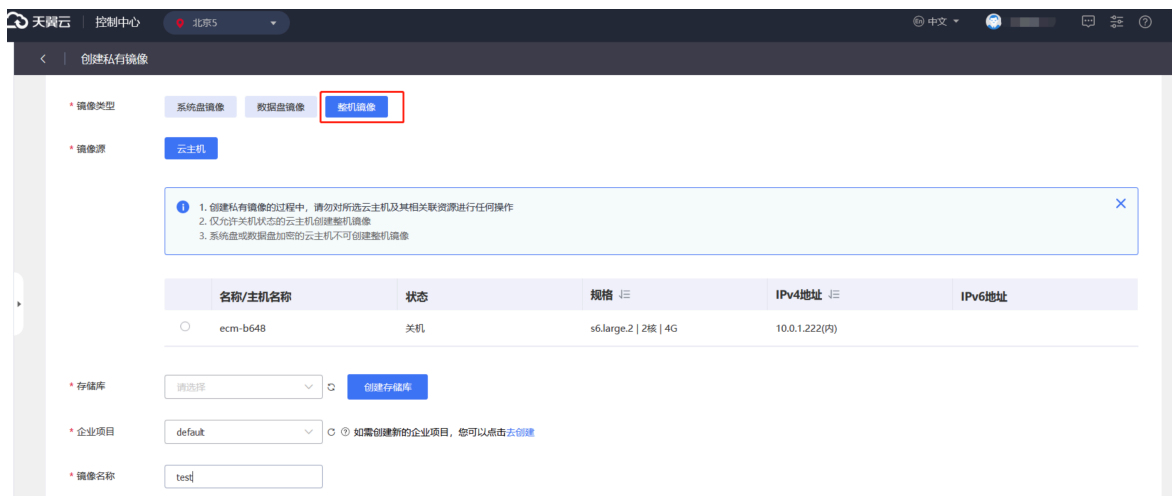
镜像为资源池级别，共享成功的镜像在该资源池所有可用区均可使用。

解决方案

以下以“北京-北京5”资源池的云主机迁移到另一个账号为实例进行详细介绍。

步骤1：创建私有镜像

1. 登陆控制中心。
2. 选择资源池  “北京-北京5”。
3. 选择“产品服务列表>计算>弹性云主机”，进入云主机列表页面。
4. 选择需要迁移的云主机，单击右侧操作列的“更多>制作镜像”，进入创建私有镜像页面。



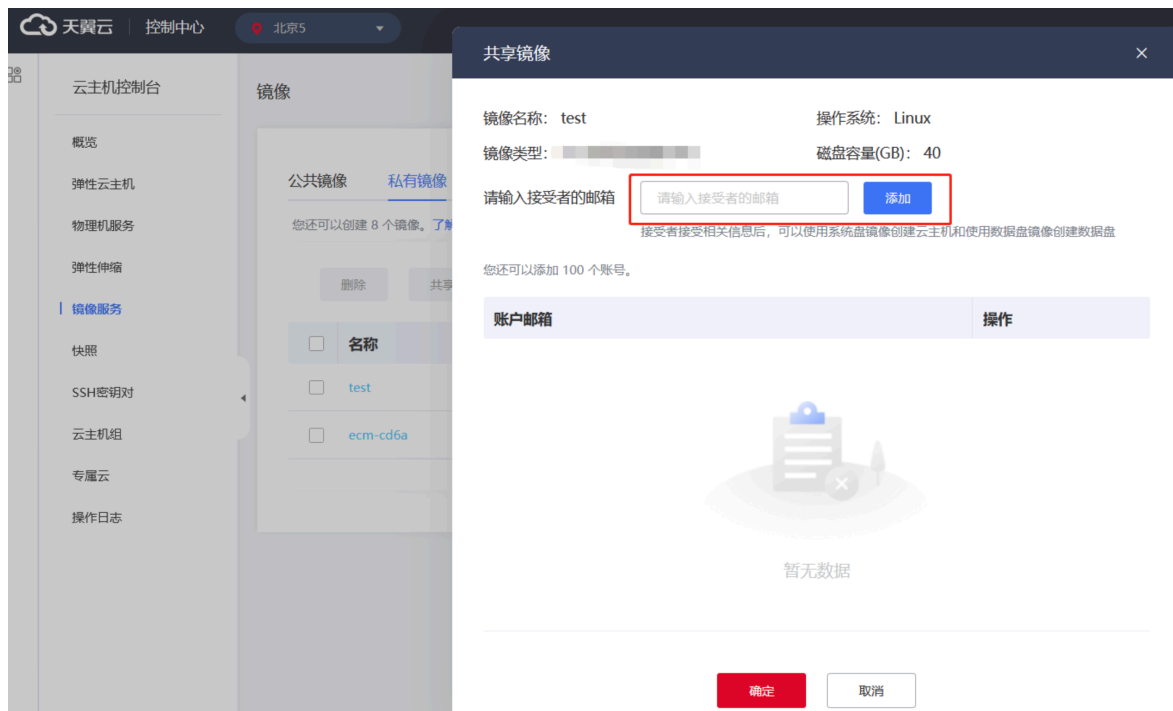
5. 填写页面参数，这里镜像类型选择“整机镜像”，镜像源选择“云主机”，选择要迁移的云主机，选择存储库和填写镜像名称。单击“下一步”。
6. 确认无误后，阅读并勾选协议，单击“确定下单”。
7. 返回私有镜像列表，在列表中查看新创建的私有镜像。

步骤2：共享私有镜像

1. 选择需要迁移的云主机，单击右侧操作列的“更多>共享”，进入共享镜像页面。
2. 填写共享镜像页面信息。

最佳实践

3. 选择需要共享的镜像和需要共享的账户邮箱，点击确认。



步骤3：接受共享私人镜像

1. 登陆需要将云主机迁移至的账号。
2. 登陆控制中心。
3. 选择资源池 北京-北京5。
4. 选择“产品服务列表>镜像服务”，进入镜像服务页面，单击“共享镜像”页签。
5. 选择需要迁移的共享镜像申请，单击接受共享，成功接受共享私人镜像。

步骤4：使用共享镜像创建云主机

1. 在共享镜像界面右侧操作栏，单击“申请云主机”，进入云主机创建界面。

<input type="checkbox"/>	名称	状态	操作系统	镜像类型	磁盘容量(GB)	创建时间	企业项目	操作
<input type="checkbox"/>	test	✔ 正常	Linux	ECS系统盘镜像(X86)	40	2023-09-03 10:58:...	default	申请云主机 更多
<input type="checkbox"/>	ecm-cd6a	✔ 正常	Linux	ECS系统盘镜像(X86)	40	2023-08-21 16:57:...	default	申请云主机 更多

2. 填写信息，完成弹性云主机的创建，可参考“[创建弹性云主机](#)”。

弹性云主机之间迁移最佳实践

介绍说明

当需要把不同天翼云账号下的主机资源整合到一个天翼云账号下，或者将同账户不同区域的CT-ECS资源整合到同一区域下时，可以采用主机迁移服务实现CT-ECS实例间的快速迁移。主机迁移服务支持同一区域或者不同区域的源端到目的端的迁移，下面为您介绍天翼云CT-ECS实例间的实现逻辑和操作步骤。

实现逻辑

CMS基础实现逻辑是在目标端部署PE轻量系统，在CMS控制台处进行任务配置与下发，通过协同CMS-Agent与CMS-PE实现迁移；

仅需通过确定PE轻量系统开通的账户与区域，即可限定迁移目标端的账户与区域。需要注意迁移过程中需要保证账户余额充足与目标区域已加载迁移专用镜像，未加载镜像区域不支持CMS迁移服务。

操作步骤

跨账号CT-ECS迁移（账户A迁移至B账户）

1. 在A账户下的ECS云主机下，使用B账户的云迁移服务AK/SK安装CMS-Agent。具体步骤请参见[在源端安装Agent](#)。



2. A账户主机即会被上报至B账户迁移平台，接下来可以进行迁移目标机设置。具体步骤请参见[创建目标机](#)，[检测目标机](#)和[绑定目标机](#)。



3. 绑定成功后可以配置并启动迁移任务。具体步骤请参见[迁移任务配置与开始全量迁移](#)。



4. 迁移任务配置完成后，单击“开始迁移”正常执行后续迁移操作，即可实现ECS云主机跨账户迁移。

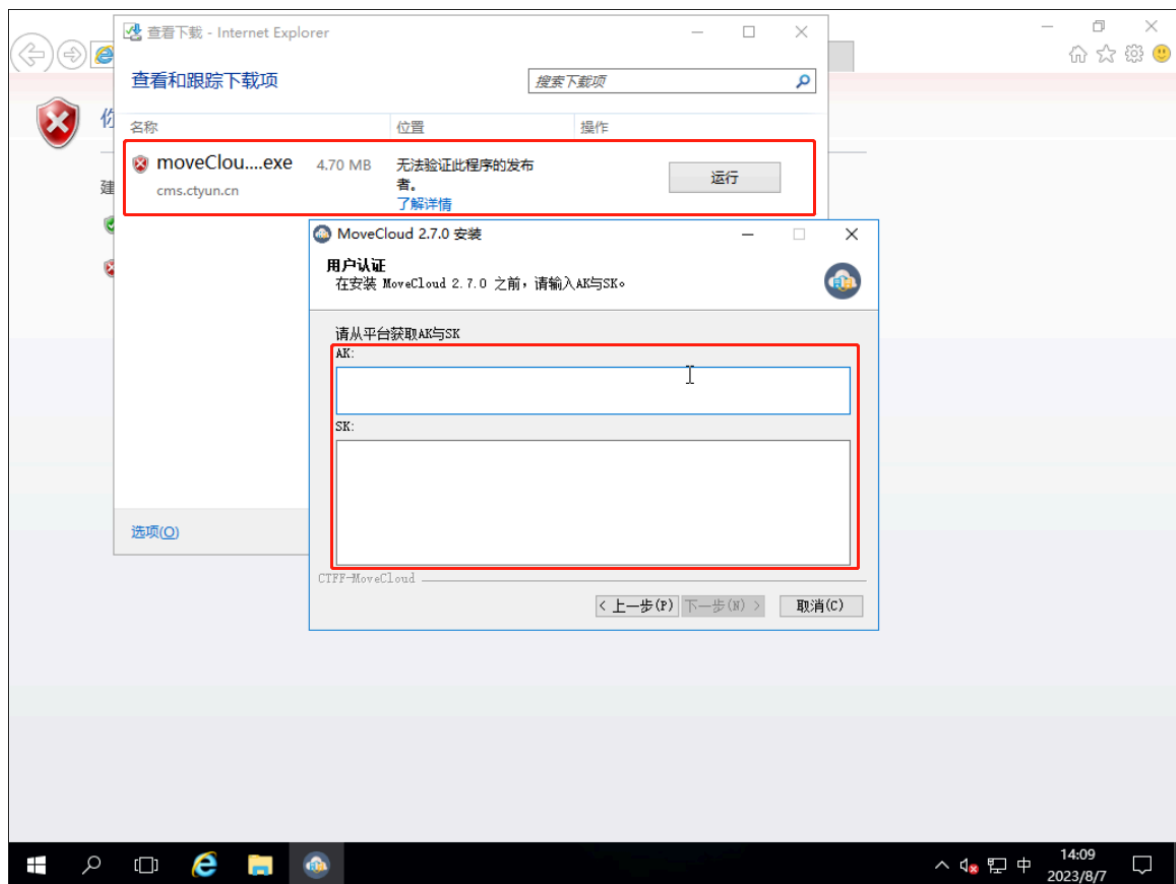


同账户下CT-ECS区域间迁移（C区域迁移至D区域）

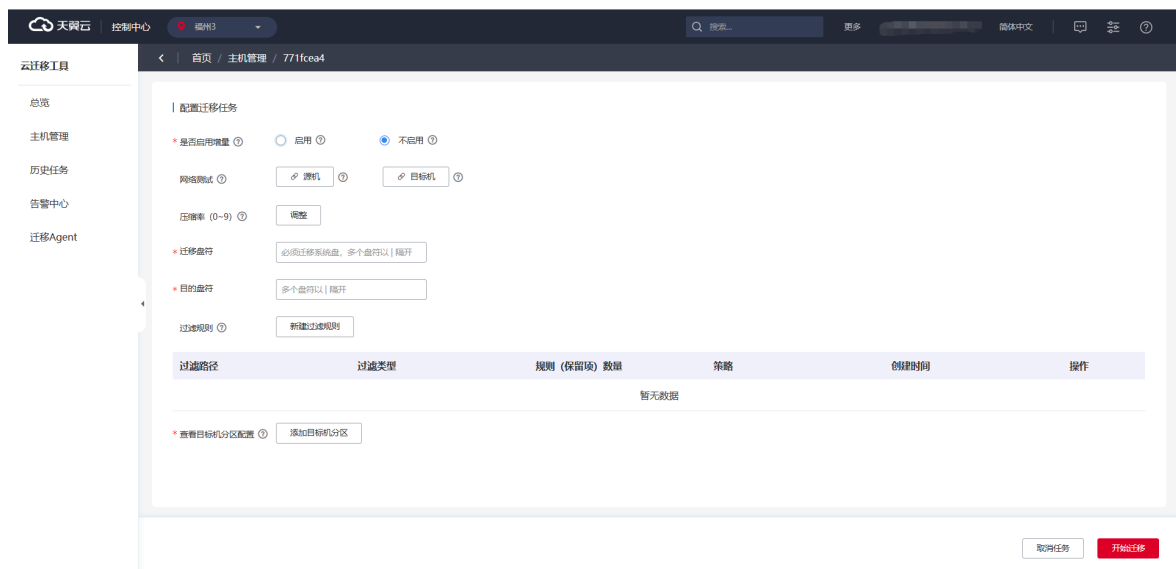
操作前提：D区域需支持云迁移服务，并且在D区域使用云迁移服务。

1. 在C区域云主机安装CMS-Agent，将迁移源上报；具体步骤请参见[在源端安装迁移Agent](#)。

最佳实践

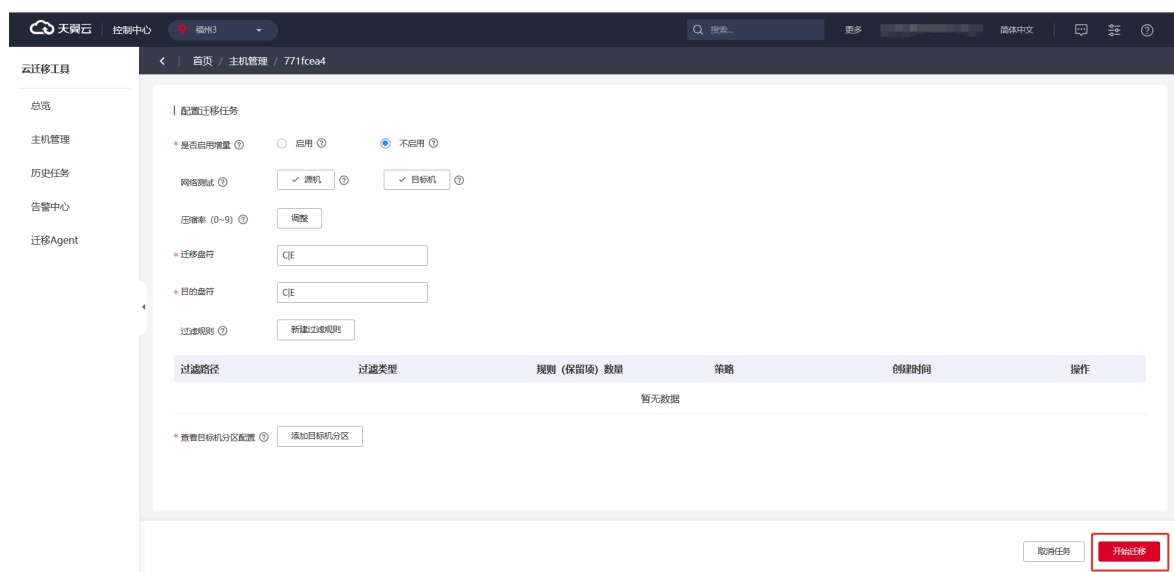


2. 在D区域创建目标机，具体步骤请参见[创建目标机](#)，[检测目标机](#)和[绑定目标机](#)。
3. 绑定成功后可以配置并启动迁移任务。具体步骤请参见[迁移任务配置与开始全量迁移](#)。



最佳实践

4. 迁移任务配置完成后，单击“开始迁移”正常执行后续迁移操作，即可实现同账户下CT-ECS云主机跨区域迁移。



OpenClaw 部署最佳实践

OpenClaw 安全风险提示

OpenClaw 是运行于系统级环境的通用 AI Agent，支持文件读写、代码执行、多步任务编排，可通过聊天工具接收指令执行对应操作。因其具备较高系统操作权限，强烈建议您仅在隔离、可控的云端环境中部署运行。

天翼云提供的 OpenClaw 软件环境来源于第三方或开源社区，仅供您参考与合规使用。您需自行评估使用相关的全部风险，因部署、配置、使用该镜像及相关软件产生的任何直接或间接损失、安全与合规责任，天翼云不承担相关责任。请您确保所有使用行为符合国家法律法规、监管要求及对应开源许可协议。

请您结合自身业务需求与安全要求，合理配置系统与权限策略，保障运行环境安全可控。为降低潜在安全风险，请您重点关注以下使用注意事项：

1. 及时更新版本

定期升级 OpenClaw 及相关组件，避免因已知漏洞带来安全风险。

2. 加强数据备份

建议建立定期备份机制，防止因系统故障或误操作导致数据丢失。

3. 控制网络暴露面

非必要情况下，不建议将 OpenClaw 服务端口直接暴露至互联网，可通过安全组、内网访问或代理方式进行访问控制。

4. 遵循最小权限原则

仅启用必要的工具和功能，避免授予 OpenClaw 过高的系统权限或无限制的自主执行能力。

5. 使用可信来源的 Skills 插件

建议优先使用官方或经过安全验证的 Skills，避免使用来源不明的插件，以降低安全风险。

最佳实践

6. 部署主机安全防护措施

建议安装主机安全防护软件。天翼云为云主机提供免费的主机安全卫士，建议在创建实例时启用相关安全服务。

7. 妥善保护访问凭证

对API Key、Token 等敏感凭证进行安全存储与加密管理，避免泄露。

8. 开启日志与审计机制

建议启用操作日志与行为审计功能，以确保关键操作可记录、可追溯。

使用云主机一键部署 OpenClaw（快速版）

OpenClaw 是一个开源的、运行于系统级环境的通用 AI 助手框架。支持文件读写、代码执行与多步任务编排，可通过聊天工具接收指令并完成实际操作。其核心特征是“可执行性”，适用于自动化办公与任务代理，通常需在隔离、可控的云端环境中运行。

说明

OpenClaw 作为开源项目，建议在使用前全面评估其安全性与稳定性，仔细阅读《OpenClaw 安全风险提示》并且严格按照对应的开源许可协议规范使用，确保系统和数据安全。

本文将为您介绍使用天翼云弹性云主机部署个人AI助手 OpenClaw 的最佳实践。

注意

本文使用最新版本镜像【OpenClaw-v2026.3.13-Ubuntu24.04】部署，建议您使用该镜像参考流程。低于【OpenClaw-v2026.3.13】版本的镜像，请参考实践[使用云主机部署OpenClaw（自定义配置版）](#)进行操作。

步骤1：购买云主机

方式1：通过公有云生态专区部署OpenClaw

说明

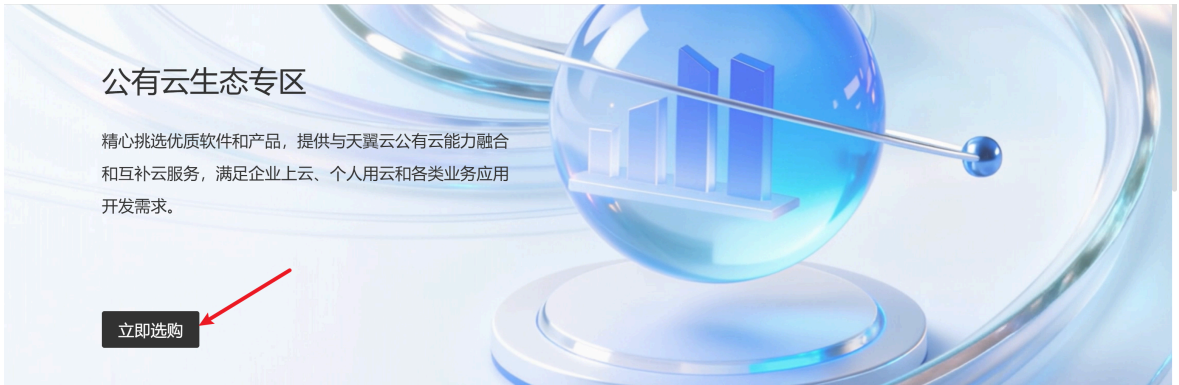
资源池 西南1、华南2、长沙42、武汉41、芜湖2、杭州7、华东1、武汉4、拉萨3、上海36、上海7 支持使用方式1部署。

1. 登录天翼云官网，进入[公有云生态专区](#)。



最佳实践

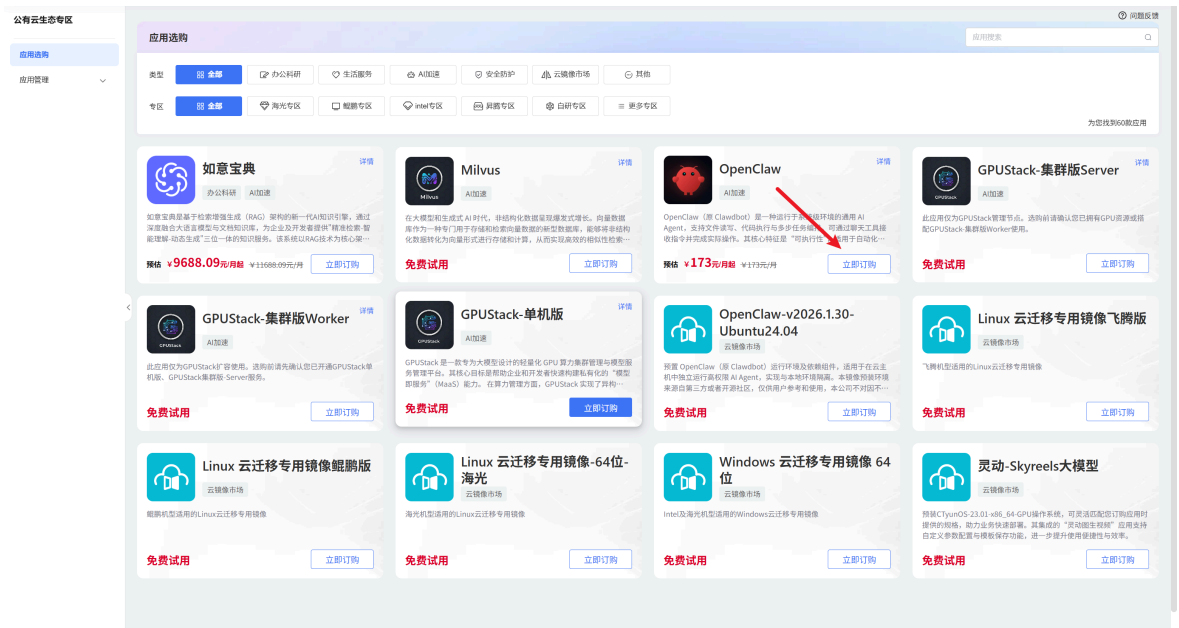
2. 选择“OpenClaw”应用，点击“立即选购”按钮。



热门应用推荐

 <p>OpenClaw (原 Clawbot) 是一种运行于系统级环境的通用 AI Agent，支持文件读写、代码执行与多步任务编排，可通过聊天工具接收指令并完成实际操作。其核...</p>	 <p>如意宝典是基于检索增强生成 (RAG) 架构的新一代 AI 知识引擎，通过深度融合大语言模型与文档知识库，为企业及开发者提供“精准检索-智能理解-动态生成”三位...</p>	 <p>在大模型和生成式 AI 时代，非结构化数据呈现爆发式增长。向量数据库作为一种专门用于存储和检索向量数据...</p>
---	--	---

3. 在应用选购列表中，选择OpenClaw，点击“立即订购”。



最佳实践

4. 根据指引，完成简单配置后，点击“立即下单”。

The screenshot displays the configuration interface for the OpenClaw application. The main configuration area includes:

- 加建应用:** OpenClaw (OpenClaw个人AI助理), version V2026.1.30.
- 资源池:** 西南1.
- 规格套餐:** 基础型. Description: 适用于个人调研. Includes: OpenClaw服务 (vCPU: 2核, 内存: 4 GB, 系统盘: 40 GB SSD-generic, 公网弹性IP: 1个, 带宽: 5 Mbps).
- 应用计费方式:** 包年包月.
- 实例名称:** App-OpenClaw-7m736 (18 / 63).
- 应用别名:** OpenClaw-nz4i (13 / 30).
- 密码:** [Redacted].

At the bottom, the purchase duration is set to 1 month. The total price is displayed as 云资源价格 ¥173.00元 and 应用价格 免费 ¥0.00元. A red arrow points to the **确认下单** button.

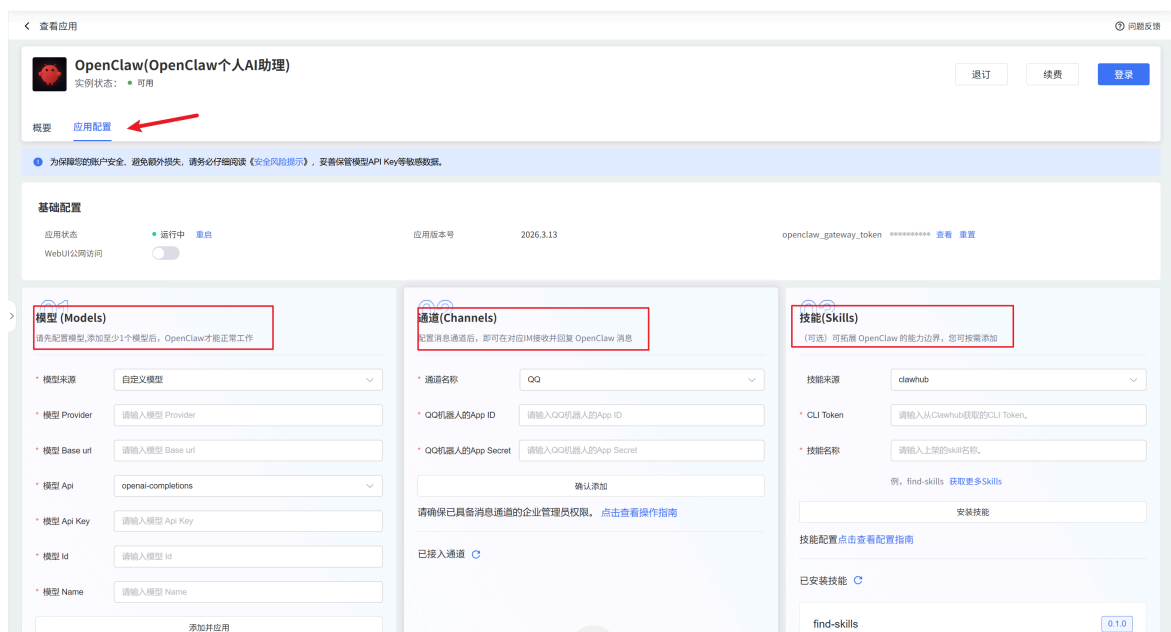
最佳实践

5. 完成支付后，返回生态专区“我的应用”页，点击查看“查看应用”按钮。

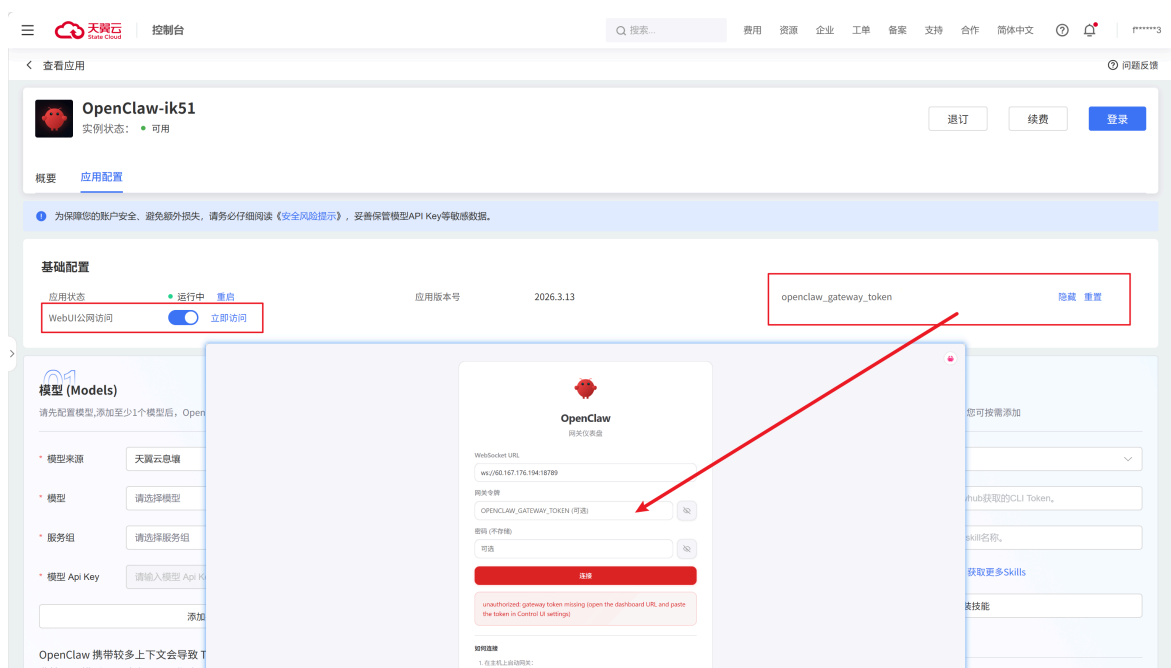


最佳实践

6. 进入应用详情页，点击“应用配置”页签，在页面中完成**模型配置**、**通道配置**、**技能配置**后，即可开始使用 OpenClaw。



7. 在确保模型、安全组已正确配置的前提下，可通过WebUI试用OpenClaw，开启WebUI公网访问开关，点击“立即访问”按钮，在打开的新页面中输入“openclaw_gateway_token”对应的凭证值，即可开始与OpenClaw 对话。



最佳实践

方式2：通过云镜像市场创建云主机部署OpenClaw

说明

以下资源池支持使用方式2部署：

西南1、华南2。

1. 登录天翼云官网，进入天翼云控制台-计算控制台-弹性云主机，点击“创建云主机”按钮。



最佳实践

2. 云主机配置。

CPU架构选择“X86计算”，规格-分类选择“经济型”，云主机规格推荐使用【经济型-e.large.2】的规格。

The screenshot shows the instance specification selection interface. Under the 'Economic' category, the 'e.large.2' instance is selected and highlighted with a red box. The table below lists the specifications for various instance types:

规格族	规格名称	vCPU (核)	内存 (GB)	最大带宽(Gbps) / 基准带宽(Gbps)	网卡多队列数	规格参考价格(元/月)
经济型e	e.large.1	2	2	2 / 0.2	1	45
经济型e	e.large.2	2	4	2 / 0.2	1	96
经济型e	e.large.4	2	8	2 / 0.2	1	144
经济型e	e.xlarge.2	4	8	3 / 0.4	1	192

注意

仅经济型（E）实例支持通过公有云生态专区应用进行应用配置。

镜像类型选择云镜像市场/应用镜像，并选取“OpenClaw-v2026.3.13-Ubuntu24.04”镜像。

The screenshot shows the 'Cloud Image Market' dialog box. The 'OpenClaw-v2026.3.13-Ubuntu24.04' image is selected and highlighted with a red box. Red arrows point to the 'OpenClaw-v2026.3.13-Ubuntu24.04' text, the '立即使用' button, and the '从云镜像市场选择' button. The dialog box also displays a warning about third-party application security and provides details about the selected image, including its system architecture (X86), provider (Tianyi Cloud Technology Co., Ltd.), and configuration (2GB RAM).

点击“下一步：网络配置”。

最佳实践

3. 网络配置。

点击“创建安全组”按钮，跳转至网络控制台-安全组页面。

安全组

选择安全组

创建安全组

点击“创建安全组”按钮，在弹窗中模板下拉菜单选择“通用Web服务器”。

创建安全组

×

* 名称：

sgs-f47a

* 模板：

通用Web服务器

* 虚拟私有云：



[创建VPC](#)

该VPC下已创建安全组2个，配额为100个

* 企业项目：

default



描述：

默认放通22、3389、80、443和ICMP协议。使用于需要

[查看模板规则](#)

>

取消

确定

点击“确定”，完成安全组创建。返回云主机订购页面，点击“选择安全组”按钮，在安全组列表中勾选刚才创建的安全组。

安全组

选择安全组

创建安全组



1.sgs-f47a / sg-i9klxxovfw

弹性IP选择“自动分配”，根据需要选择带宽大小。为确保OpenClaw顺畅运行，建议带宽选择5M以上。

* 弹性IP

不使用 自动分配 使用已有

自动为每台云主机分配独享带宽的弹性公网IP，创建弹性云主机过程中，请确保弹性公网IP配额充足。
该独享带宽弹性IP的付费方式与订购周期和云主机保持一致：包年/包月

IP版本

IPv4

* 带宽

5 Mbps

注意

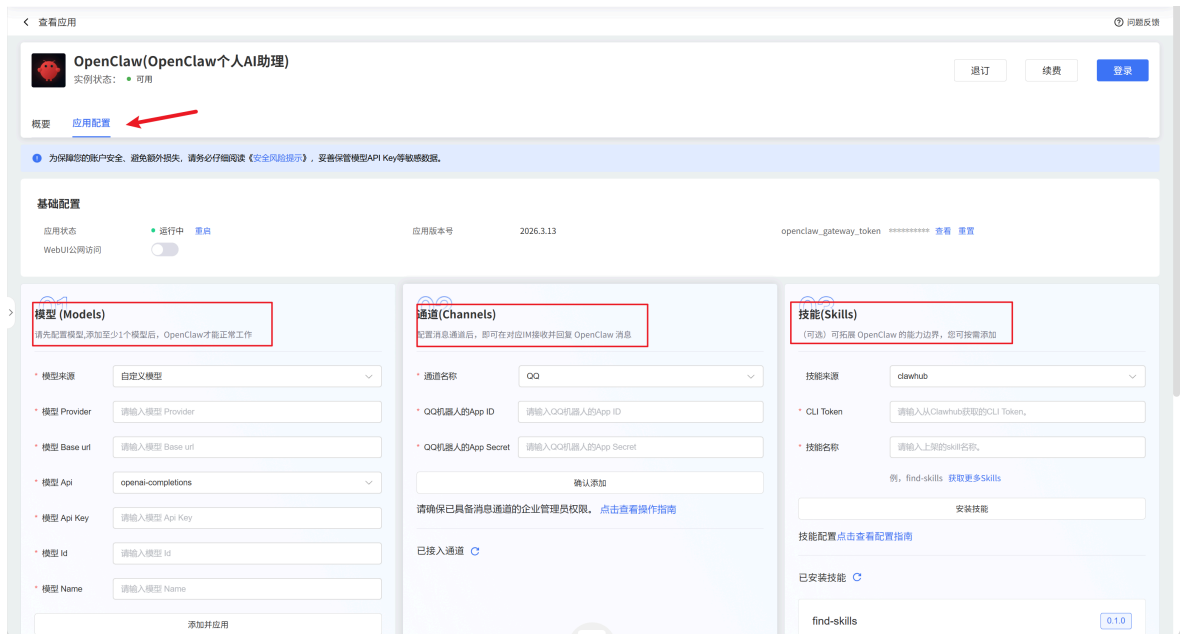
通过订购云主机部署OpenClaw时，请确保：

- 云主机已绑定弹性IP，否则OpenClaw应用将无法通过WebUI访问，也无法通过公有云生态专区应用页面进行模型、通道、技能等配置。
 - 云主机已绑定安全组，且入方向规则开放端口18789，否则无法在公有云生态专区进行应用配置。
- 根据提示完成高级配置，点击“下一步：确认配置”。确认配置无误后，点击“立即购买”。
 - 完成支付后，返回[云主机控制台](#)，确认云主机状态为“运行中”，进入云主机详情页，点击“应用管理”页签。

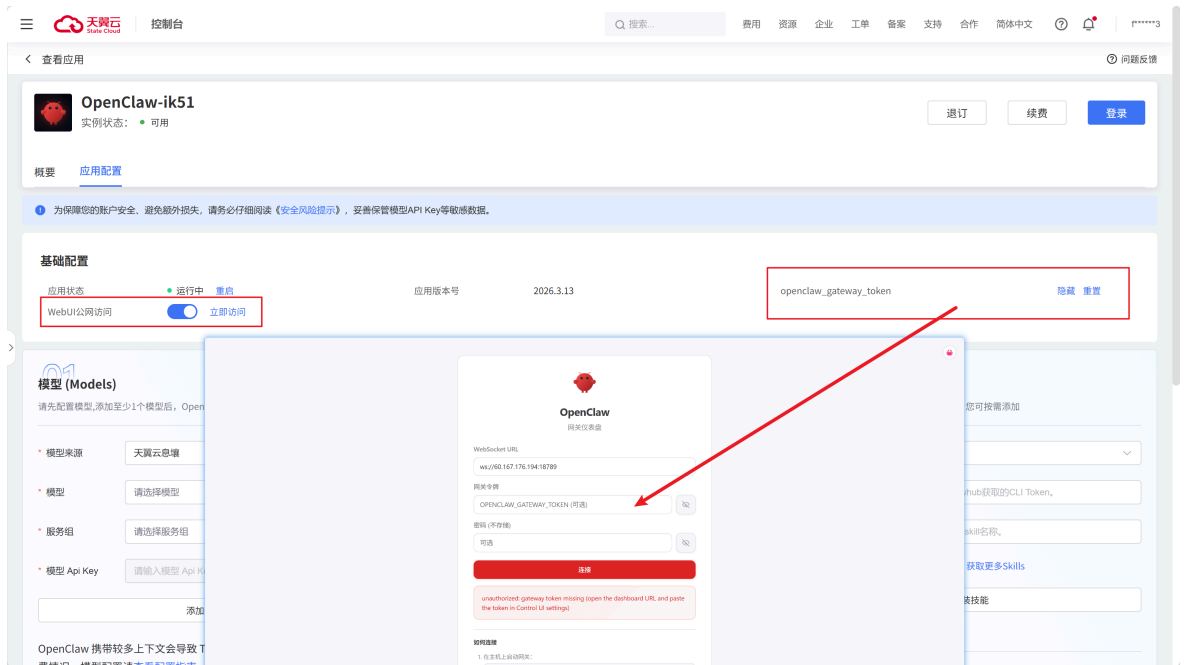


最佳实践

6. 进入应用详情页，点击“应用配置”页签，在页面中完成模型配置、通道配置、技能配置。



7. 在确保模型、安全组已正确配置的前提下，可通过WebUI试用OpenClaw，开启WebUI公网访问开关，点击“立即访问”按钮，在打开的新页面中输入“openclaw_gateway_token”对应的凭证值，即可开始与OpenClaw对话。



最佳实践

方式3：通过官网首页活动卡片部署OpenClaw

1. 登录天翼云官网，点击“立即体验”，进入OpenClaw活动专区。



2. 进入OpenClaw活动页面，点击“性能首选”。

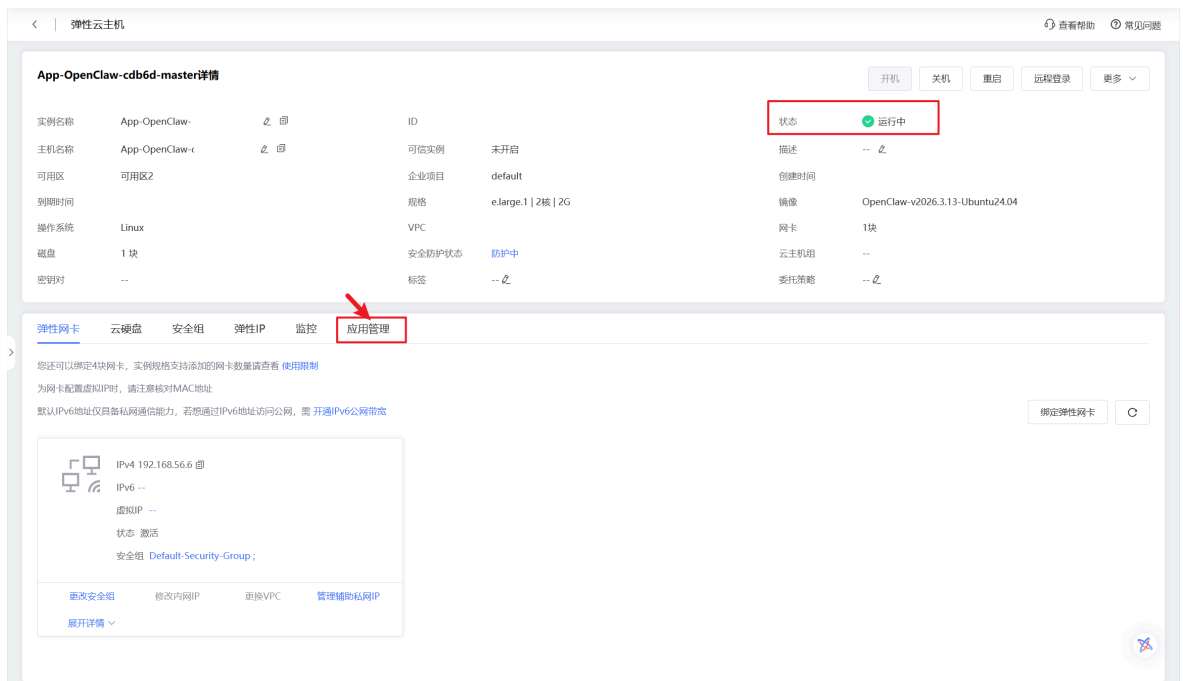


最佳实践

3. 根据您的业务需求，选择适用的OpenClaw云服务器套餐，点击活动卡片下方“立即订购”。



4. 完成支付后，返回云主机控制台，进入活动订购的云主机详情页，点击“应用管理”页签。



说明

云主机状态需为“运行中”。

最佳实践

5. 进入应用详情页，点击“应用配置”页签，在页面中完成 [模型配置](#)、[通道配置](#)、[技能配置](#) 后，即可开始使用OpenClaw。

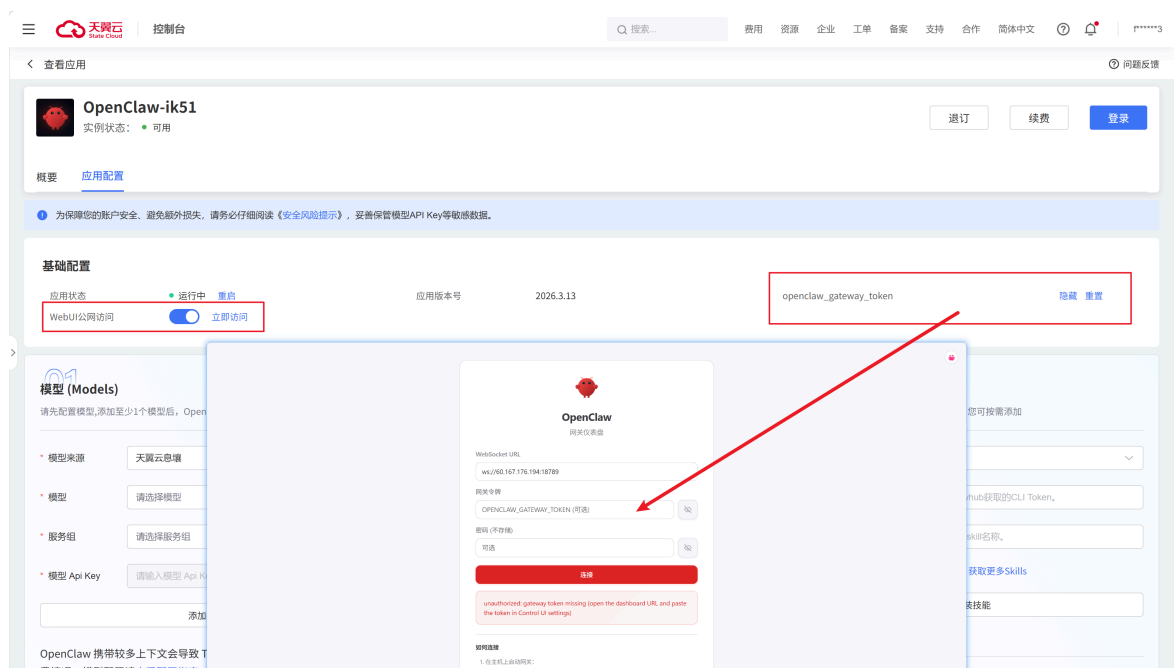
The screenshot displays the configuration interface for the OpenClaw-web-app. At the top, the application name is "OpenClaw-web-app" with a status of "可用" (Available). There are buttons for "退订" (Cancel), "续费" (Renew), and "登录" (Login). Below this is a navigation bar with "概要" (Overview) and "应用配置" (Application Configuration) tabs. A security warning is present: "为保障您的账户安全、避免额外损失，请务必仔细阅读《安全风险提示》，妥善保管模型API Key等敏感数据。"

The "应用配置" section is divided into three main panels, each with a red border:

- 基础配置 (Basic Configuration):** Shows application status (离线/重启), version (2026.3.13), and gateway token (openclaw_gateway_token).
- 模型 (Models):** A form for adding models. Fields include: 模型来源 (自定义模型), 模型 Provider, 模型 Base url, 模型 Api (openai-completions), 模型 Api Key, 模型 Id, and 模型 Name. A "添加并应用" button is at the bottom.
- 通道 (Channels):** A form for adding channels. Fields include: 通道名称 (QQ), QQ机器人的App ID, and QQ机器人的App Secret. A "确认添加" button is present. A note states: "请确保已具备消息通道的企业管理员权限。点击查看操作指南" (Ensure you have enterprise administrator permissions for the message channel. Click to view the operation guide). Below, it shows "已接入通道" (Connected Channels) with a list containing "QQ".
- 技能 (Skills):** A form for adding skills. Fields include: 技能来源 (ClawHub), CLI Token, and 技能名称. A note says: "(可选) 可拓展 OpenClaw 的能力边界，您可按需添加" (Optional: Can expand the capabilities of OpenClaw, you can add as needed). Below, it shows "已安装技能" (Installed Skills) with a list containing "find-skills" (version 0.1.0).

最佳实践

6. 在确保模型、安全组已正确配置的前提下，可通过WebUI试用OpenClaw，开启WebUI公网访问开关，点击“立即访问”按钮，在打开的新页面中输入“openclaw_gateway_token”对应的凭证值，即可开始与OpenClaw对话。



说明

1. 为保障您的数据安全，推荐您购买[云主机备份服务](#)，开启自动备份保护。
2. 云主机创建成功后，您可通过重置密码操作设置新密码，修改密码操作请参考[在控制台重置密码](#)。

使用云主机部署OpenClaw（自定义配置版）

使用云主机部署 OpenClaw

OpenClaw（原 Clawbot/Moltbot）是一种运行于系统级环境的通用 AI Agent，支持文件读写、代码执行与多步任务编排，可通过聊天工具接收指令并完成实际操作。其核心特征是“可执行性”，适用于自动化办公与任务代理，通常需在隔离、可控的云端环境中运行。

说明

OpenClaw 作为开源项目，建议在使用前全面评估其安全性与稳定性，并且严格按照对应的开源许可协议规范使用，确保系统和数据安全。

本文将为您介绍使用天翼云弹性云主机部署个人AI助手 OpenClaw 的最佳实践。

最佳实践

注意

本文、[使用云主机打造飞书私人助理](#) 及 [使用云主机打造钉钉私人助理](#) 最佳实践中，均使用天翼云云镜像市场（应用镜像）中的 OpenClaw 镜像进行部署，建议您使用该镜像参考流程。

步骤1：购买云主机

方式1：通过公有云生态专区部署OpenClaw

说明

资源池 西南1 支持使用方式1部署。

1. 登录天翼云官网，进入[公有云生态专区](#)。



2. 选择“OpenClaw”应用，点击“立即选购”按钮。



热门应用推荐

 OpenClaw AI加速	 如意宝典 办公科研	 Milvus AI加速
OpenClaw (原 Clawdbot) 是一种运行于系统级环境的通用 AI Agent，支持文件读写、代码执行与多步任务编排，可通过聊天工具接收指令并完成实际操作。其核...	如意宝典是基于检索增强生成 (RAG) 架构的新一代 AI 知识引擎，通过深度融合大语言模型与文档知识库，为企业及开发者提供“精准检索·智能理解·动态生成”三位...	在大模型和生成式 AI 时代，非结构化数据呈现爆发式增长。向量数据库作为一种专门用于存储和检索向量数据的新颖数据库，能够将非结构化数据转化为向量形式...

最佳实践

3. 在应用选购列表中，选择OpenClaw，点击“立即订购”。

公有云生态专区

应用选购

应用列表

应用选购

如意宝典

Milvus

OpenClaw

GPUStack-集群版Server

GPUStack-集群版Worker

GPUStack-单机版

OpenClaw-v2026.1.30-Ubuntu24.04

Linux 云迁移专用镜像 飞腾版

Linux 云迁移专用镜像 鲲鹏版

Linux 云迁移专用镜像-64位-海光

Windows 云迁移专用镜像 64位

灵动-Skyreels大模型

4. 根据指引，完成简单配置后，点击“立即下单”。

应用选购

OpenClaw

OpenClaw个人AI助理

V2026.1.30

资源池

西南1

规格套餐

基础型

适用于个人测试

此套餐包含

OpenClaw服务

vCPU 2核

内存 4 GB

系统盘 40 GB SSD-generic

公网弹性IP 1个

带宽 5 Mbps

应用计费方式

包年包月

实例名称

App-OpenClaw-7m736

应用别名

OpenClaw-nz4i

密码

购买时长 1个月

云资源价格 ¥173.00元

应用价格 免费

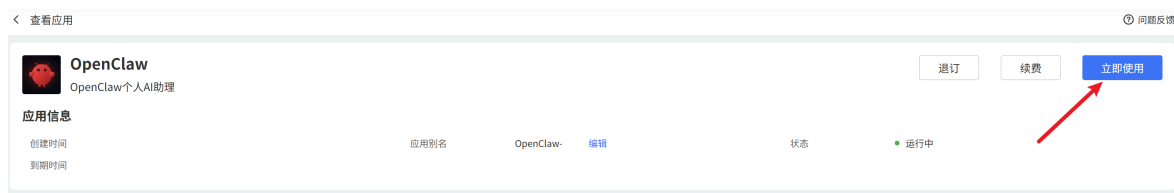
确认下单

最佳实践

5. 完成支付后，返回生态专区“我的应用”页，点击查看“查看应用”按钮。



6. 进入应用详情页，点击“立即使用”按钮登录。



方式2：通过云镜像市场创建云主机部署OpenClaw

说明

以下资源池支持使用方式2部署：

华北2、杭州7、南京5、佛山3、华东1、西安7、西南2-贵州、郑州5、上海36。

最佳实践

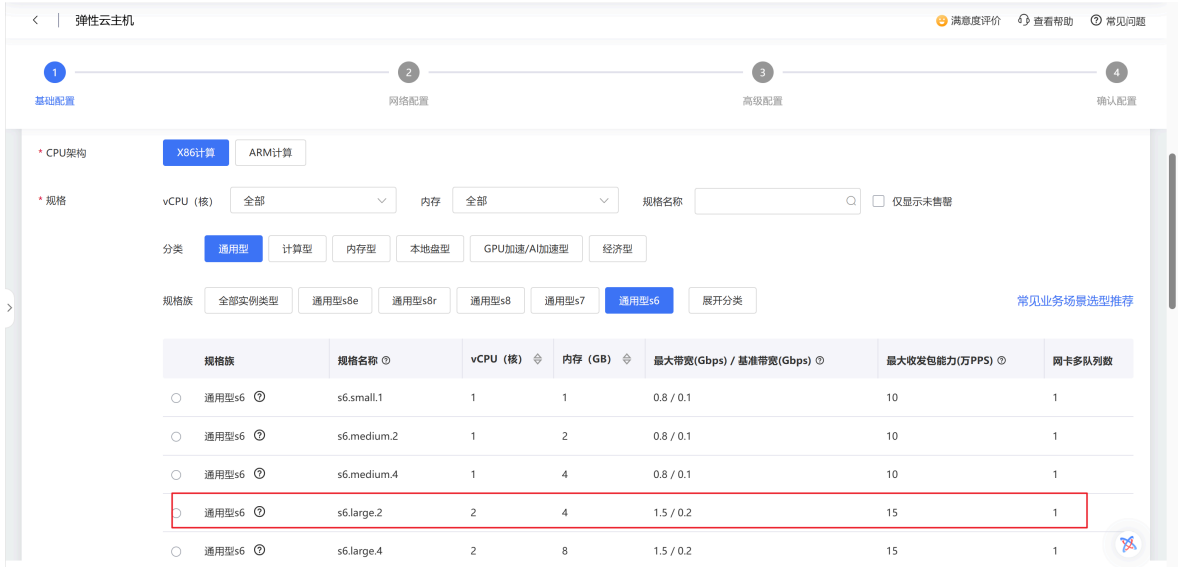
1. 登录天翼云官网，进入天翼云控制台-计算控制台-弹性云主机，点击“创建云主机”按钮。



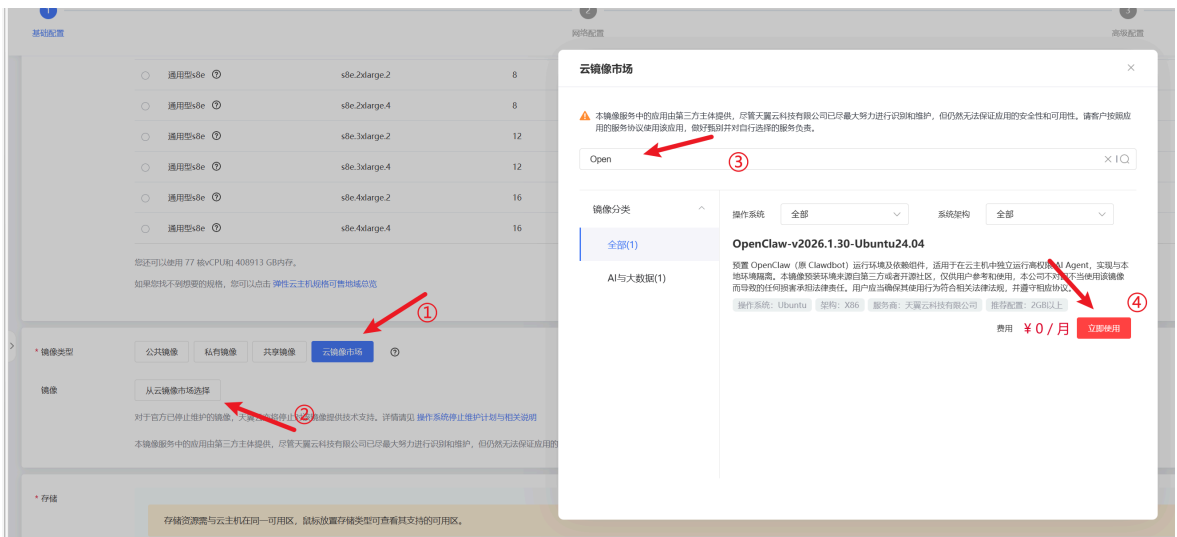
最佳实践

2. 云主机配置。

CPU架构选择“X86计算”，规格-分类选择“通用型”，云主机规格推荐使用【通用型-s6.large.2】或其他【2核4G】的规格。



镜像类型选择云镜像市场/应用镜像，并选取“OpenClaw-v2026.1.30-Ubuntu24.04”镜像。



点击“下一步：网络配置”。

最佳实践

3. 网络配置。

点击“创建安全组”按钮，跳转至网络控制台-安全组页面。

安全组

选择安全组

创建安全组

点击“创建安全组”按钮，在弹窗中模板下拉菜单选择“通用Web服务器”。

创建安全组

×

* 名称：

sgs-f47a

* 模板：

通用Web服务器

* 虚拟私有云：



[创建VPC](#)

该VPC下已创建安全组2个，配额为100个

* 企业项目：

default



描述：

默认放通22、3389、80、443和ICMP协议。使用于需要

[查看模板规则](#)

>

取消

确定

点击“确定”，完成安全组创建。返回云主机订购页面，点击“选择安全组”按钮，在安全组列表中勾选刚才创建的安全组。

最佳实践

安全组

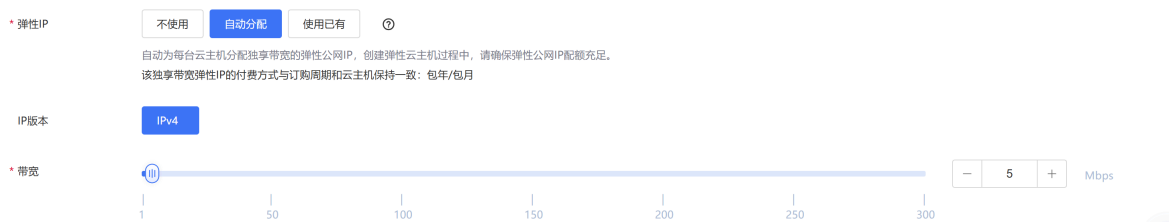
选择安全组

创建安全组



1.sgs-f47a / sg-i9klxxovfw

弹性IP选择“自动分配”，根据需要选择带宽大小。为确保OpenClaw顺畅运行，建议带宽选择5M以上。



4. 根据提示完成高级配置，点击“下一步：确认配置”。确认配置无误后，点击“立即购买”。
5. 完成支付后，返回云主机控制台，确认云主机状态为“运行中”，点击“重置密码”按钮设置云主机登录密码。



6. 密码重置成功后，点击远程登录按钮，在弹出的管理终端页面输入用户名/密码，登录云主机。

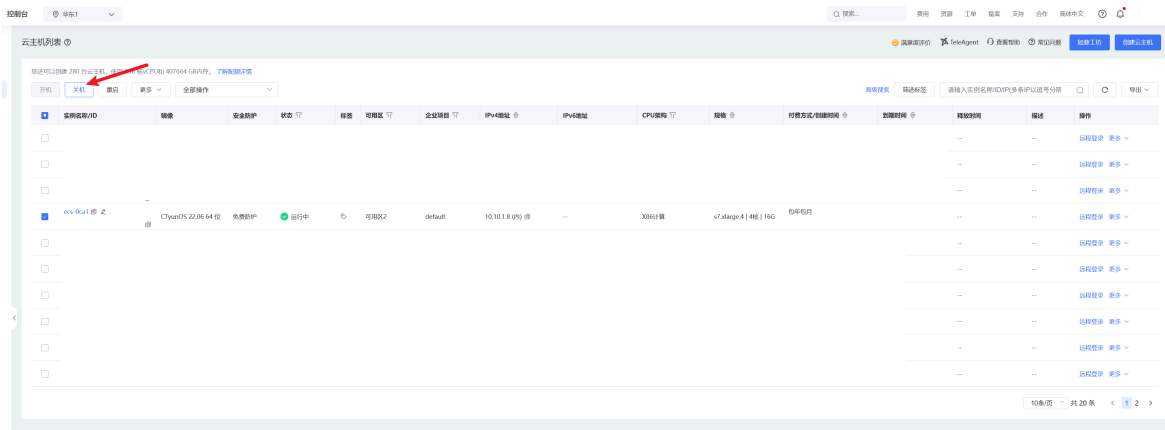


方式3：使用已有云主机部署OpenClaw

若您希望在已购买的云主机上部署OpenClaw，可通过云主机控制台云主机一键重装能力部署，具体操作如下：

最佳实践

1. 登录天翼云控制台，在云主机列表中，选中希望部署的云主机，执行关机操作。



2. 关机成功后，点击更多，在下拉菜单中选择一键重装。



最佳实践

3. 在弹窗中，镜像类型选择“云镜像市场”，重装选择“OpenClaw-v2026.1.30-Ubuntu24.04”镜像，并重新输入密码及确认密码，点击“确定”按钮，触发重装操作。

一键重装

×

- 1、重装后IP地址不变，但系统盘会清空，请预先做好云硬盘备份。
- 2、如果云主机存在云主机快照或系统盘快照，请先将快照删除，否则无法重装云主机。
- 3、重装后云主机备份将无法用于恢复数据。
- 4、重装后系统盘所在的云硬盘备份策略无法继续使用。请在重装前将系统盘从所在备份策略中移出，重装后重新加入；若未提前移出，请在重装后及时新建云盘备份策略。
- 5、Linux系统下使用了LVM的用户，LVM如果加入了数据盘，请先从LVM卸载掉数据盘，再一键重装，否则会导致主机重装后无法使用。
- 6、多次批量一键重装操作建议间隔120s以上，否则可能造成重装失败。
- 7、使用加密系统盘重装大约需要10~25分钟，容量越大加密耗时越久。
- 8、重装操作系统可能会使云备份、云容灾等关联业务产生影响，请操作前判断是否需要清理备份计划、容灾复制资源等操作。
- 9、涉及到收费镜像时，不支持批量重装操作。
- 10、如果重装的主机已挂载XSSD类型的系统盘或数据盘，仅支持选择“NVMe驱动”为“已安装”的镜像。部分增强型云主机（搭载了智能网卡的云主机），不受此条限制。
- 11、若已使用服务器安全卫士（原生版）产品，选择非公共镜像重装后无法继续使用，请在控制台中重新安装Agent。

* 镜像类型

公共镜像

私有镜像

共享镜像

云镜像市场

?

重装选择 OpenClaw-v2026.1.30-Ubuntu24.04(40GB) [重新选择](#)

对于官方已停止维护的镜像，天翼云亦将停止对该镜像提供技术支持。详情请见 [操作系统停止维护计划与相关说明](#)

本镜像服务中的应用由第三方主体提供，尽管天翼云科技有限公司已尽最大努力进行识别和维护，但仍然无法保证应用的安全性和可用性。请客户按照应用的服务协议使用该应用，做好甄别并对自行选择的服务负责。

* 主机名称

ecs-0ca1

?

* 密码

请输入密码

🔒

* 确认密码

请再次输入密码

🔒

云监控 开启详细监控

在云服务器创建成功后，3-5分钟内将完成详细监控Agent安装，即开启云服务器CPU，内存，网络，磁盘，进程等指标详细监控。若不开启，则无任何监控数据。若误操作删除Agent，也会导致监控数据无法获取。

* 用户数据：

* 重装后开机：

重装操作系统后，将为您自动开机，若您是按需计费，开机后您的云主机将开始计费。

取消

确定

最佳实践

4. 重装成功后，确认云主机状态为“运行中”，可继续执行后续OpenClaw部署流程。



提示：推荐使用云主机备份，对云主机内所有云硬盘数据同时提供备份保护服务，并支持利用备份数据恢复云主机数据，详见[整机备份恢复](#)。

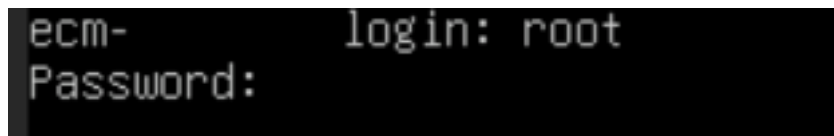
步骤2：在云主机上配置大模型

推荐使用[息壤模型推理服务](#)，本文以息壤模型推理服务为例，介绍配置步骤。

1. 登录天翼云控制台，根据您的需求选择适合的模型，本文以DeepSeek模型为例，详细开通步骤可参考[DeepSeek模型调用-快捷版](#)。
2. 登录目标云主机。登录天翼云官网，进入计算控制台，点击远程登录按钮。



本文以Linux系统操作为例，在管理终端页面中，输入用户名（默认为root）及密码，登录云主机。



最佳实践

3. 在终端管理页面中执行以下3条命令:

```
openclaw config set models.providers.[供应商名称] '{"baseUrl":"https://[大模型服务的IP或者域名]/v1","apiKey":"[api key]","api":"openai-completions","models": [{"id": "[模型 id]","name": "[模型名称]","reasoning": false,"input": ["text"],"contextWindow": 65536,"maxTokens": 65536}]'
```

```
openclaw config set agents.defaults.model.primary "[供应商名称]/[模型 id]"
```

```
openclaw config set agents.defaults.models.[供应商名称]/[模型 id] {}
```

命令中, 需要替换的参数描述如下:

参数	填写说明	示例
[供应商名称]	模型供应商名称, 可自定义命名。	xirang
[大模型服务的IP或者域名]	模型服务调用地址。 说明 即息壤模型服务生成的 baseUrl。	https://wishub-x6.ctyun.cn
[api key]	模型服务API Key。 说明 即息壤模型服务生成的app key。	123456789
[模型ID]	调用模型的ID。 说明 息壤模型服务详情页可查询模型ID。	DeepSeek-V3.2-Standard
[模型名称]	调用模型的名称。 说明 息壤模型详情页可查询模型名称。	DeepSeek-V3.2-Standard
contextWindows	上下文窗口。 由模型决定, 可根据息壤详情页中标注的模型上下文最大长度计算。如: 模型上下文长度为128k, 那么contextWindow就是128*1024=131072。	131072

最佳实践

参数	填写说明	示例
maxToken	最大Token数。 由模型决定，可根据息壤详情页中标注的模型上下文最大长度计算。maxToken理论上不超过上下文最大长度。	32768

完整的命令示例如下（此处仅为示例，请替换参数后执行）：

```
openclaw config set models.providers.XiRang '{"baseUrl":"https://wishub-x6.ctyun.cn/v1","apiKey":"123456789","api":"openai-completions","models":[{"id":"DeepSeek-V3.2-Standard","name":"DeepSeek-V3.2-Standard","reasoning":false,"input":["text"],"contextWindow":131072,"maxTokens":32768}]}'
```

```
openclaw config set agents.defaults.model.primary "XiRang/DeepSeek-V3.2-Standard"
```

```
openclaw config set agents.defaults.models.[XiRang/DeepSeek-V3.2-Standard] {}
```

步骤3：配置聊天软件

聊天软件的配置将决定与OpenClaw进行交互的界面，目前我们已经支持飞书、钉钉的接入，具体教程参见：[使用云主机打造飞书私人助理](#)、[使用云主机打造钉钉私人助理](#)。

使用云主机打造OpenClaw飞书私人助理

通过配置聊天软件与OpenClaw进行交互，本文将为您介绍介绍打造飞书个人助理的最佳实践。

注意

OpenClaw 作为开源项目，建议在使用前全面评估其安全性与稳定性，并且严格按照对应的开源许可协议规范使用，确保系统和数据安全。

前置依赖

1. 完成OpenClaw（原Clawdbot/Moltbot）服务器的部署。
2. 完成大模型配置。

说明

以上流程，可参考[使用云主机一键部署OpenClaw](#)。

步骤1：飞书机器人创建（个人用户可使用）

1. 进入飞书[应用中心](#)，选择创建企业自建应用。



最佳实践

2. 填写信息，点击创建

创建企业自建应用



该应用仅可在 内部使用，应用发布需经过企业管理员审核，请仔细阅读 [应用审核规则](#)。

应用名称*

私人AI助手

6/32

应用描述*

基于Clawdbot的AI助手

15/120

应用图标*

JPEG/PNG/SVG/BMP 格式，2 MB 以内，大于 240*240 px，无圆角

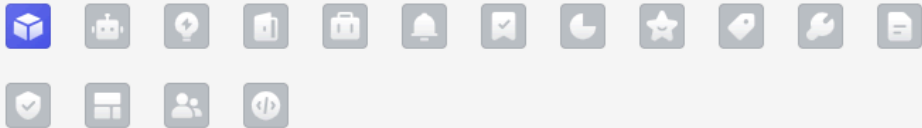


自定义上传

选择背景色



选择图标

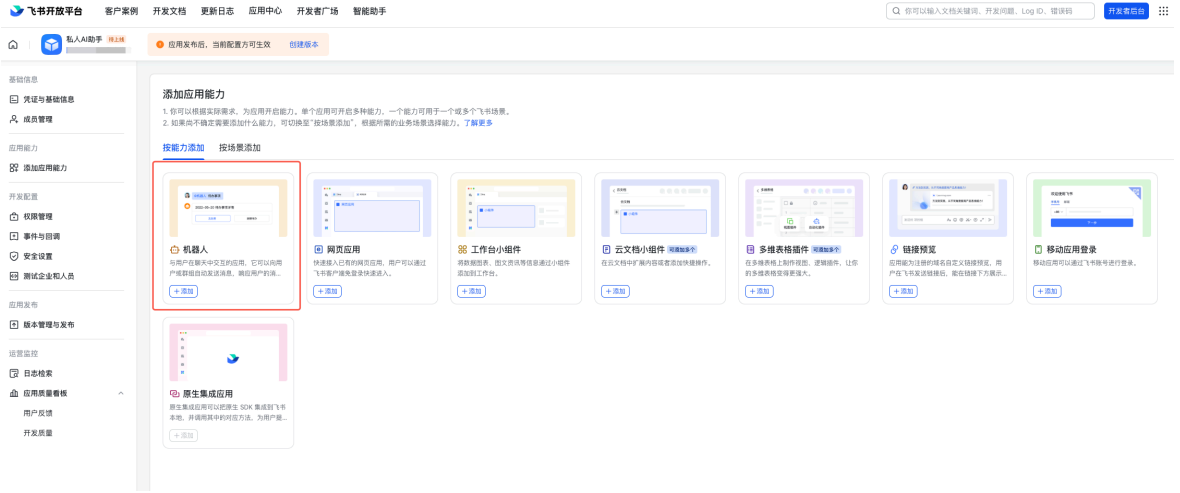


取消

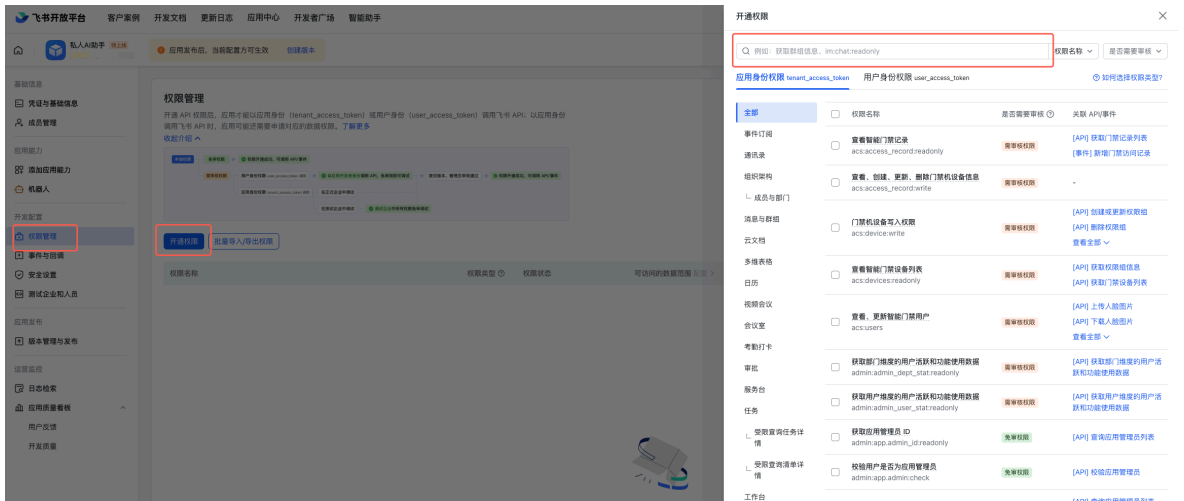
创建

最佳实践

3. 选择机器人应用，点击添加机器人。



4. 为应用添加权限。点击权限管理-开通权限，在输入框中输入目标权限，必需权限和可选权限配置见下表。



必需权限	范围	说明
contact:user:base:readonly	用户信息	获取用户基本信息
contact:contact:base:readonly	通讯录	获取通讯录基本信息
im:message	消息	发送和接收消息
im:message.p2p_msg:readonly	私聊	读取发给机器人的私聊消息
im:message.grouop_at_msg:readonly	群聊	接收群内 @机器人 的消息
im:message:send_as_bot	发送	以机器人身份发送消息

最佳实践

必需权限	范围	说明
im:resource	媒体	上传和下载图片/文件

可选权限	范围	说明
im:message.group_msg	群聊	读取所有群消息（敏感）
im:message:readonly	读取	获取历史消息
im:message:update	编辑	更新/编辑已发送消息
im:message:recall	撤回	撤回已发送消息
im:message.reactions:read	表情	查看消息表情回复

权限添加完后状态如下：

权限管理

开通 API 权限后，应用才能以应用身份 (smart_access_token) 或用户身份 (user_access_token) 调用飞书 API；以应用身份调用飞书 API 时，应用可能还需要申请对应的数据权限。了解更多

收起介绍

权限名称 权限范围 权限状态 可访问的数据范围 操作

获取用户基本信息 contact:user:basic:readonly	群 应用身份	已开通	通讯录 与应用的可用范围一致	相关 API/事件 关闭
获取与发送群聊、群组成员消息 im:message	群 应用身份	已开通	-	相关 API/事件 关闭
接收群聊中机器人消息事件 im:message.group_at_msg:readonly	群 应用身份	已开通	-	相关 API/事件 关闭
接收用户上传机器人的群聊消息 im:message:group_msg:readonly	群 应用身份	已开通	-	相关 API/事件 关闭
以应用的身份发送消息 im:message:send_at_bot	群 应用身份	已开通	-	相关 API/事件 关闭
获取与上传图片文件资源 im:resource	群 应用身份	已开通	-	相关 API/事件 关闭

5. 获取并保存飞书的APP ID 和 APP Secret 用于“步骤3-在云主机上配置飞书插件”，获取入口如下：

应用凭证

App ID

App Secret

综合信息

应用图标

JPEG/PNG/SVG/BMP 格式，2 MB 以内，大于 240*240 px，无边角

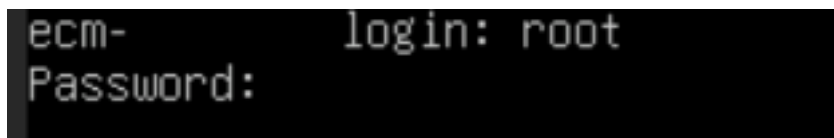
最佳实践

步骤2: 在云主机上配置飞书插件信息

1. 登录目标云主机。登录天翼云官网，进入计算控制台，点击远程登录按钮。



本文以Linux系统操作为例，在管理终端页面中，输入用户名（默认为root）及密码，登录云主机。



2. 在云主机管理终端页面中执行下方3条命令，完成飞书的相关信息配置：

```
openclaw config set channels.feishu.appId "[飞书的App ID]"
```

```
openclaw config set channels.feishu.appSecret "[飞书的App Secret]"
```

```
openclaw config set channels.feishu.enabled true
```

注意

若您选择安装的镜像名称中的版本信息为v2026.3.13或以上版本的，需额外进行配置 pairing 认证或执行以下命令关闭 pairing 模式：

```
openclaw config set channels.feishu.dmPolicy open
```

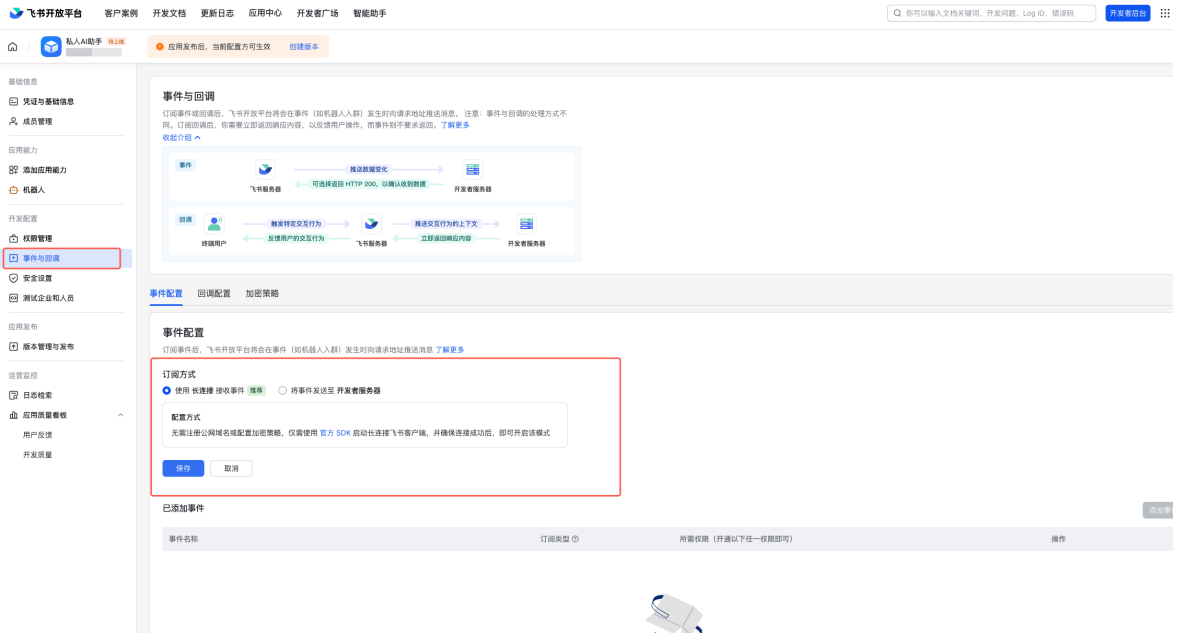
3. 重启 OpenClaw。在云主机中执行以下命令：

```
openclaw gateway restart
```

最佳实践

步骤3: 添加事件订阅

1. 选择订阅方式为长连接



添加事件，具体添加项见下表：

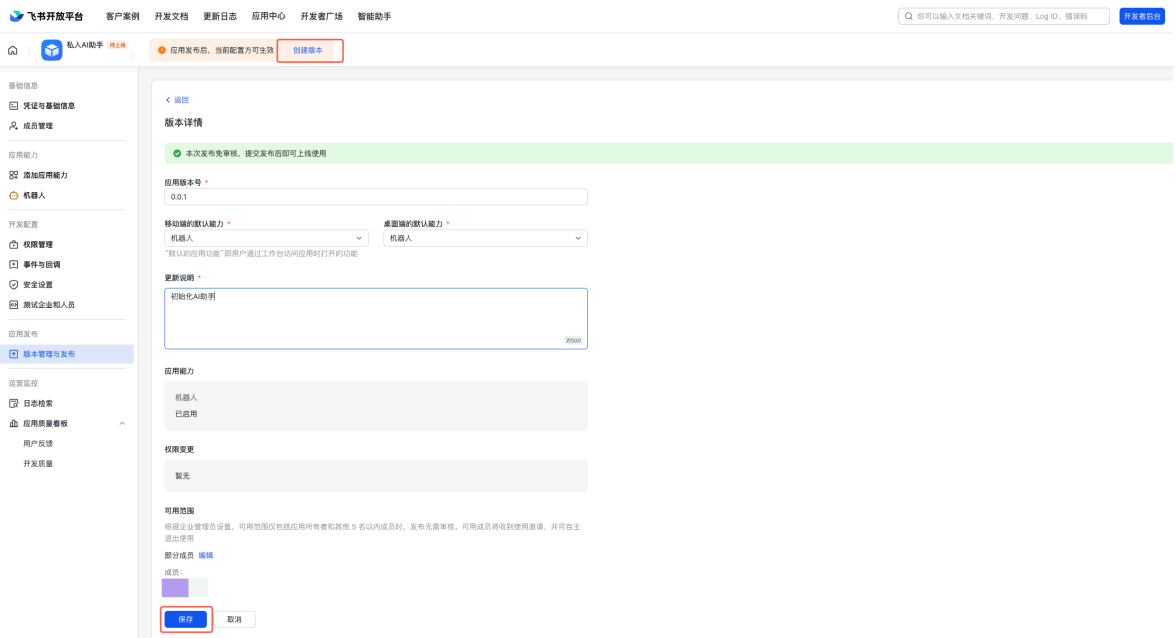


事件	说明
im.message.receive_v1	接收消息（必需）
im.message.message_read_v1	消息已读回执

最佳实践

事件	说明
im.chat.member.bot.added_v1	机器人进群
im.chat.member.bot.deleted_v1	机器人被移出群

配置完成后，进行发版

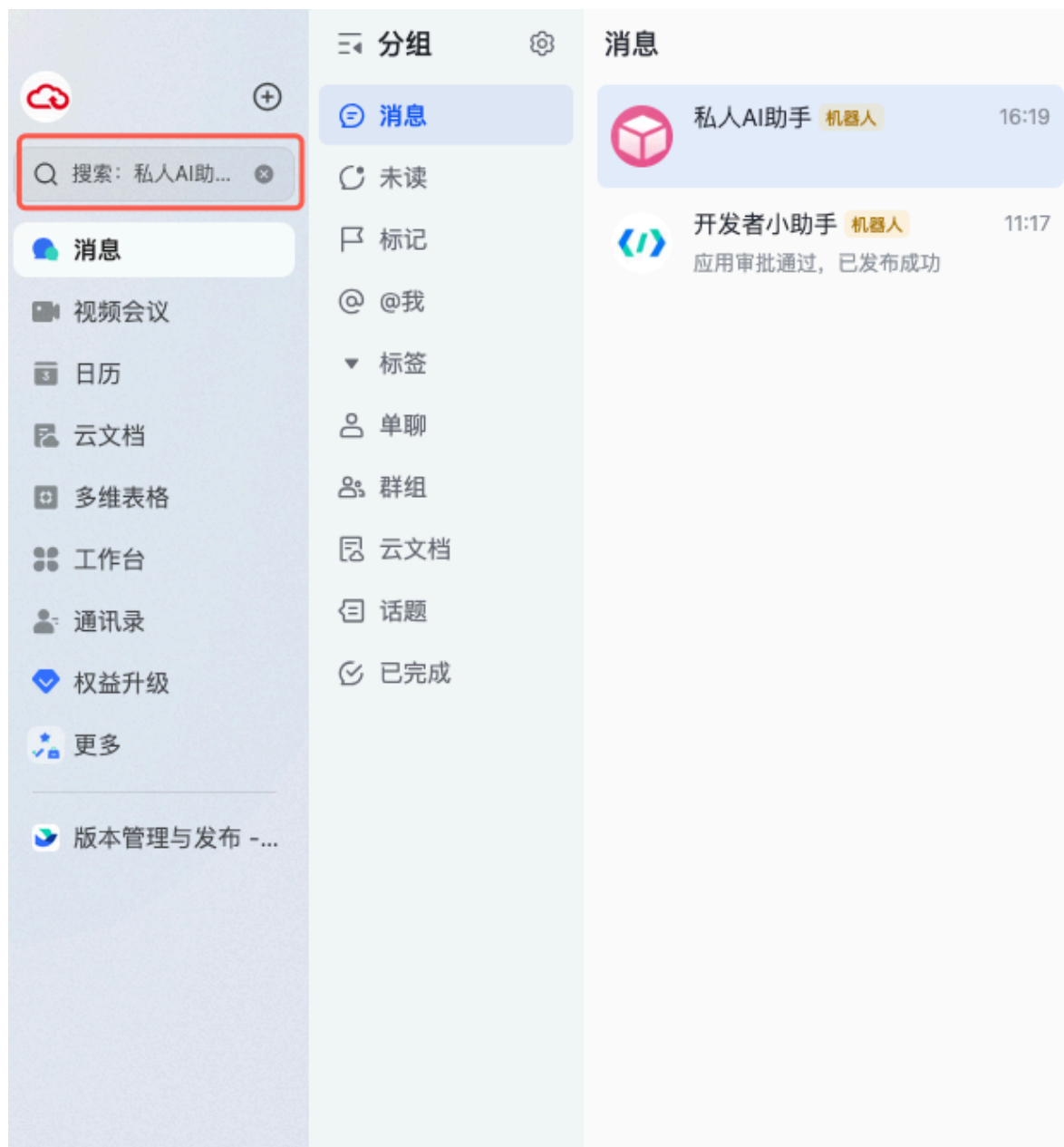


注意

事件订阅必须在OpenClaw添加飞书配置并启动后，才可以创建，否则无法创建成功。

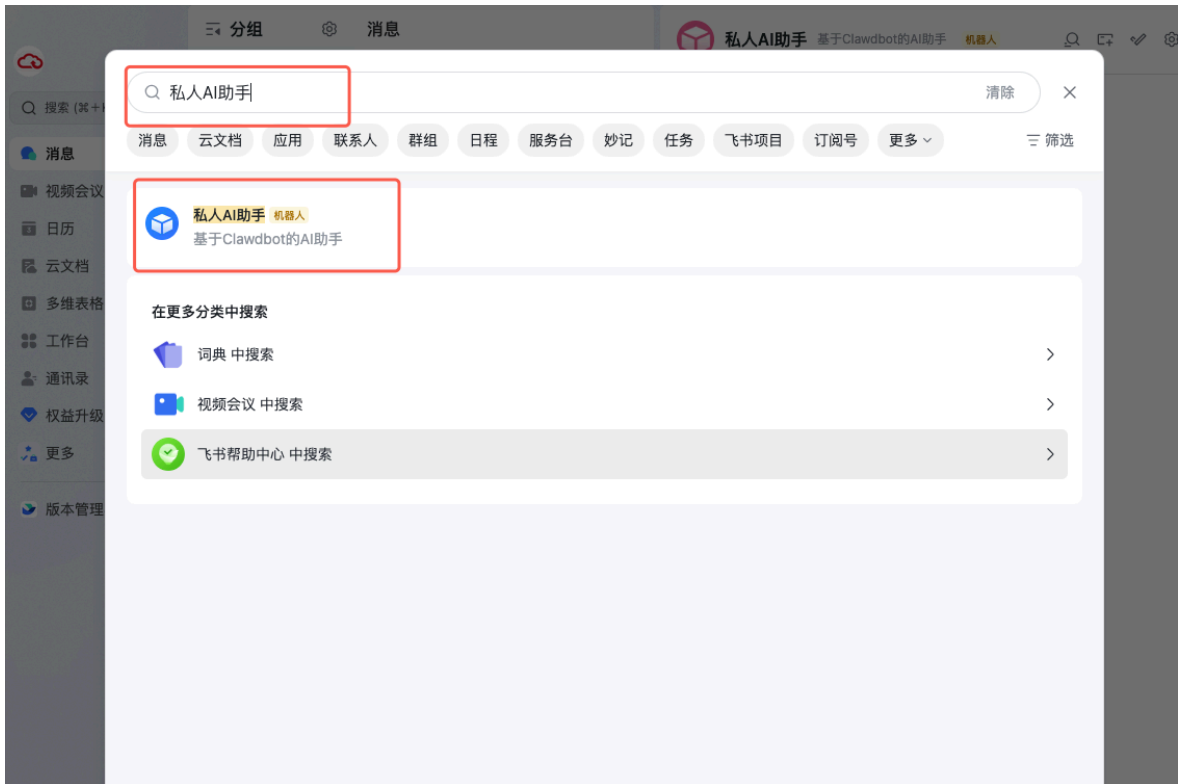
最佳实践

2. 飞书机器人已创建成功，在飞书搜索栏中搜索机器人应用名称。




例如本文“步骤1”命名的“私人AI助手”，点击应用即可开启对话。

最佳实践




现在可以在飞书中和机器人对话，开启全新AI助手体验。

最佳实践


 **私人AI助手** 基于Clawdbot的AI助手 机器人 🔍 📄 ✍️ ⚙️

11:58

 延钦

你好，请介绍一下你自己~

💬 1 条回复

 | 回复 你好，请介绍一下你自己~

你好！我是 Clawdbot，一个 AI 助手。

****我能做什么：****

- 📁 文件管理、代码操作
- 🔍 网络搜索和信息查询
- 💬 消息收发（支持飞书、Discord、Slack 等）
- 🧠 记忆和学习（会记住我们聊过的重要事情）
- 🛠️ 帮你处理各种任务

****我的风格：****

简洁、有帮助、有点个性。不喜欢废话，有问题直接解决。

我刚上线，还没给自己起名字呢 👁️ 你可以叫我任何你喜欢的名字！或者告诉我你怎么称呼，我以后就这么叫你。

有什么我可以帮你的吗？

发送给 私人AI助手 Aa 😊 @ ✂️ + ↗️ ➤ | ▾

使用云主机打造OpenClaw钉钉私人助理

通过配置聊天软件与OpenClaw进行交互，本文将为您介绍打造钉钉个人助理的最佳实践。

注意

OpenClaw 作为开源项目，建议在使用前全面评估其安全性与稳定性，并且严格按照对应的开源许可协议规范使用，确保系统和数据安全。

前置依赖

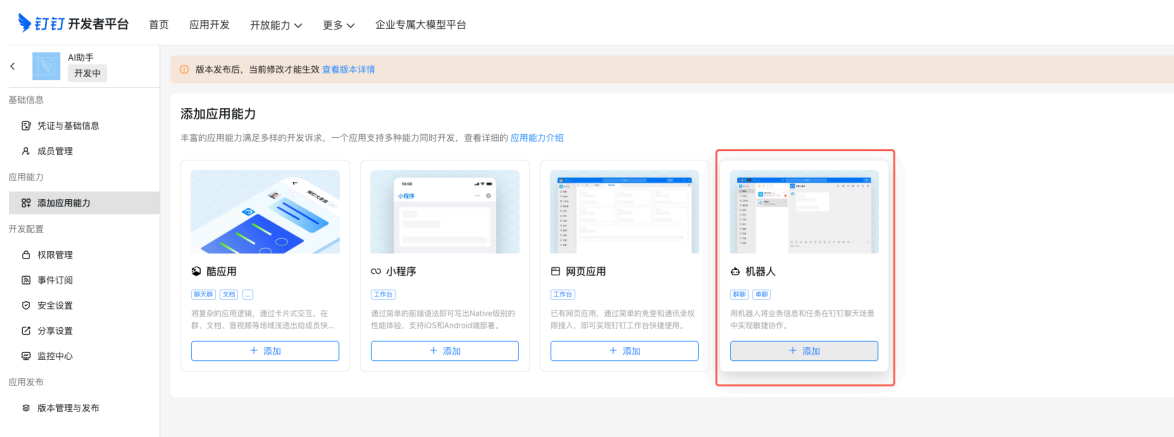
1. 完成OpenClaw（原Clawdbot/Moltbot）服务器的部署。
2. 完成大模型配置。

说明

以上流程，可参考[使用云主机一键部署OpenClaw](#)。

步骤1：钉钉机器人创建（需创建企业）

1. 进入[钉钉开发者平台](#)。
2. 进入“添加应用能力”菜单，选择添加机器人应用。



最佳实践

3. 配置机器人，选择stream模式。

钉钉 开发者平台 首页 应用开发 开放能力 更多 企业专属大模型平台

AI助手 开发中

基础信息

- 凭证与基础信息
- 成员管理

应用能力

- 添加应用能力
- 机器人**



开发配置

- 权限管理
- 事件订阅
- 安全设置
- 分享设置
- 监控中心

应用发布

- 版本管理与发布

版本发布后，当前修改才能生效 [查看版本详情](#)

- * 机器人名称 [使用应用名称](#)
AI助手
- * 机器人图标 [使用应用图标](#)

可上传JPG/PNG格式、240*240px以上、1:1、2MB 以内的无圆角图标
- * 机器人简介
机器人简介
请简要描述机器人信息，最多10个字符，该描述将在群内添加机器人时向用户进行展示
- * 机器人描述
机器人标题描述
请描述机器人功能，最多200字符，该描述将在机器人名片信息和群内添加机器人时向用户进行展示
- * 机器人消息预览图

选择图标
图片格式必须为：png、jpeg、jpg，不超过2M，可修改。

[英文](#) [日文](#) [繁体](#) [越南](#) [香港](#) [泰语](#) [印尼语](#) [马来西亚语](#) [葡萄牙语](#) [西班牙语](#) [韩语](#) [法语](#) [俄语](#) [土耳其语](#)

机器人名称
机器人名称最大长度为20个字符，该名称将在机器人名片信息和群内添加机器人时向用户进行展示

机器人简介
请简要描述机器人信息，最多10个字符，该描述将在群内添加机器人时向用户进行展示

机器人描述
请描述机器人功能，最多200字符，该描述将在机器人名片信息和群内添加机器人时向用户进行展示

消息接收模式：[\(推荐使用Stream模式，无需注册公网回调地址，详情参考\[文档链接\]\(#\)\)](#)

Stream模式

发布

最佳实践

4. 获取并保存钉钉的Client ID 和 Client Secret 用于“步骤3-在云主机上配置钉钉插件”，获取入口如下：

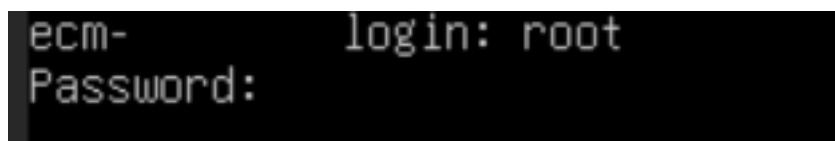


步骤2：在云主机上配置钉钉插件信息

1. 登录目标云主机。登录天翼云官网，进入计算控制台，点击远程登录按钮。



本文以Linux系统操作为例，在管理终端页面中，输入用户名（默认为root）及密码，登录云主机。



最佳实践

2. 在云主机管理终端页面中执行下方6条命令，完成钉钉的相关信息配置：

```
openclaw config set gateway.auth.token "[密码]"
```

```
openclaw config set channels.dingtalk-connector.gatewayToken "[密码]"
```

```
openclaw config set gateway.http.endpoints.chatCompletions.enabled true
```

```
openclaw config set channels.dingtalk-connector.clientId "[钉钉的Client ID]"
```

```
openclaw config set channels.dingtalk-connector.clientSecret "[钉钉的Client Secret]"
```

```
openclaw config set channels.dingtalk-connector.enabled true
```

说明

命令中的密码可自定义，两次密码输入需保持一致。

3. 重启 OpenClaw。在云主机中执行以下命令：

```
openclaw gateway restart
```

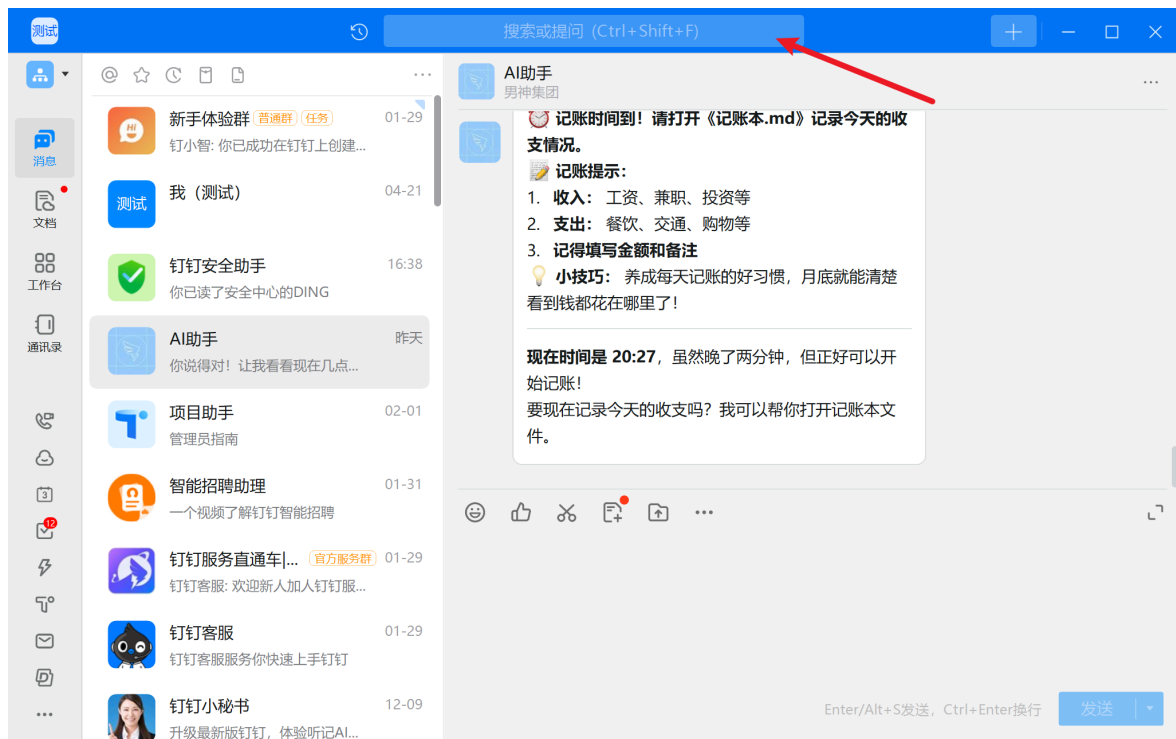
步骤3：发布应用

1. 回到[钉钉开发者平台](#)，发布版本。

The screenshot displays the '钉钉开发者平台' (DingTalk Developer Platform) interface. The top navigation bar includes '钉钉开发者平台', '首页', '应用开发', '开放能力', '更多', and '企业专属大模型平台'. The left sidebar lists various management options, with '版本管理与发布' (Version Management and Release) highlighted in a red box. The main content area shows the '版本详情' (Version Details) page for an application named 'AI助手' (AI Assistant) in '开发中' (In Development) status. A warning message at the top states: '版本发布后，当前修改才能生效 查看版本详情' (After version release, current modifications will take effect. View version details). The page includes a '返回' (Return) link, a '版本详情' (Version Details) header, and a note: '应用版本可设置应用版本号、应用能力和应用可用范围，所有变更需要保存并发布后才会正式生效，请谨慎操作' (Application version can be set with application version number, application capabilities, and application availability range. All changes need to be saved and released to take effect. Please operate with caution). The '应用版本号' (Application Version Number) field is set to '1.0.0'. The '版本描述' (Version Description) field is set to '初始化' (Initialization). The '待发布内容' (Content to be released) section shows '应用信息' (Application Information) as '已启用，当前版本无数据更新' (Enabled, no data updates for current version) and '机器人' (Robot) as 'AI助手'. The '应用可用范围' (Application Availability Range) is set to '全部员工' (All employees). A '保存' (Save) button is located at the bottom.

最佳实践

2. 发布后，返回钉钉聊天窗口，在顶部搜索栏中搜索应用名称。



例如本文“步骤1”命名的“AI助手”，点击应用，即可开启对话。



部署Palworld幻兽帕鲁服务器最佳实践

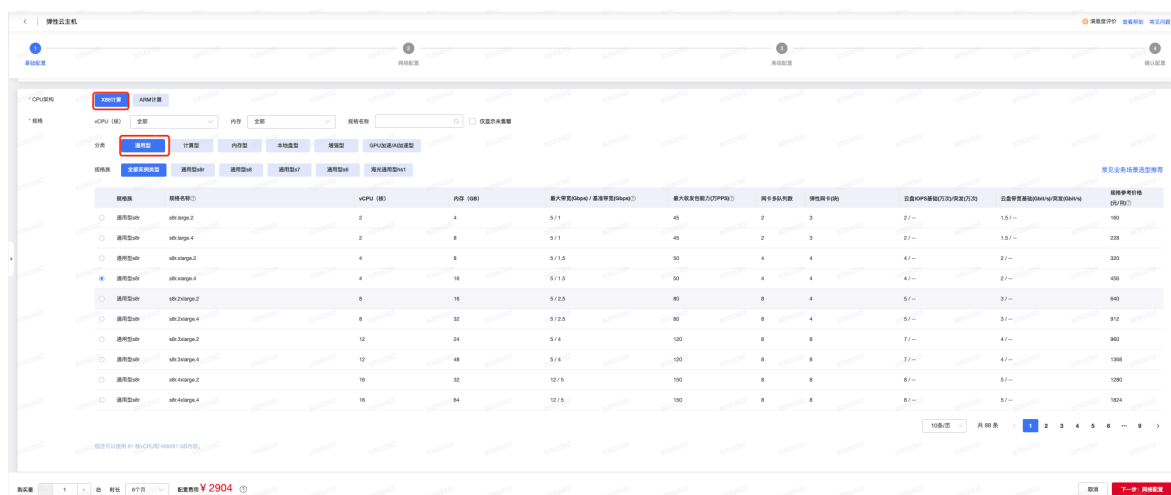
部署Palworld幻兽帕鲁服务器最佳实践（Ubuntu）

1. 登录云主机控制台，选择创建云主机的资源池，点击“创建云主机”按钮。



2. 基础配置。

CPU架构选择“X86计算”，规格-分类选择“通用型”。为确保游戏顺畅运行，建议您选择4C16G及以上的规格。



镜像类型选择“应用镜像”，镜像下拉菜单中选择“ubuntu” - “幻兽帕鲁（Palworld）-Ubuntu”。



点击“下一步：网络配置”。

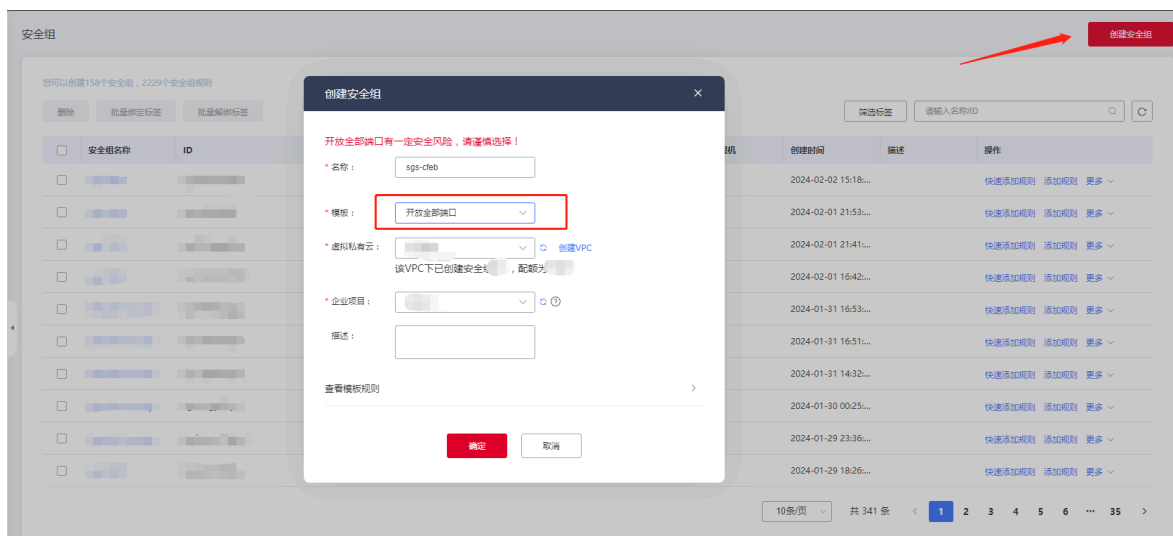
最佳实践

3. 网络配置。

点击“创建安全组”按钮，跳转至网络控制台-安全组页面。

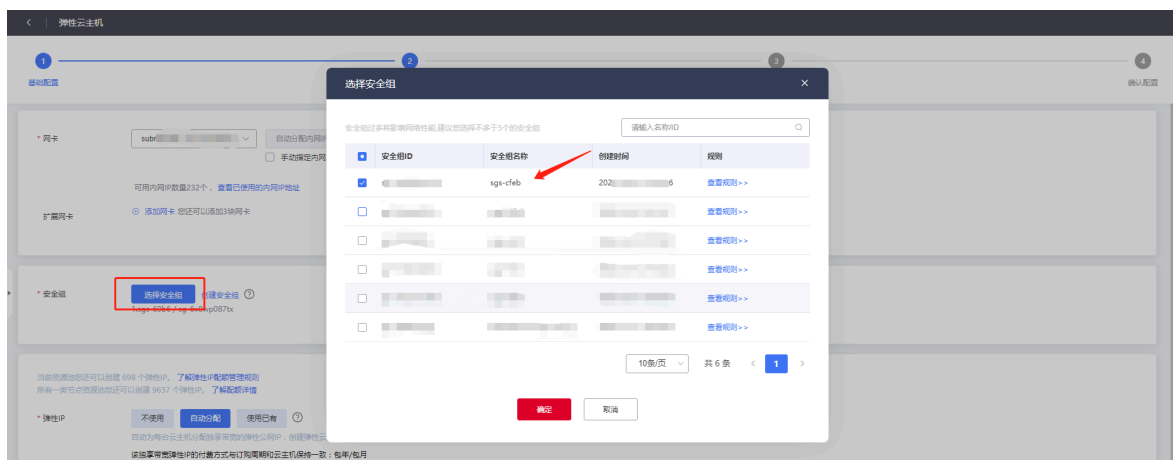


点击“创建安全组”按钮，在弹窗中等待模板下拉菜单选择“开放全部端口”。



点击“确定”，完成安全组创建。

返回云主机订购页面，点击“选择安全组”按钮，在安全组列表中勾选刚才创建的安全组。



点击“确定”，完成安全组选择。

最佳实践

弹性IP选择“自动分配”，根据需要选择带宽大小。为确保游戏顺畅运行，建议带宽选择10M以上。



点击“下一步：高级配置”。

4. 根据提示完成高级配置，点击“下一步：确认配置”。

确认配置无误后，点击“立即购买”。

5. 完成支付后，返回云主机控制台，确认云主机状态为“运行中”，则可以进行后续操作。



6. 添加安全组规则

在云主机列表中点击云主机名称，进入云主机详情页。点击安全组页签，点击“添加规则”按钮，新增IP放行。



在添加规则弹窗中：

- 1) 协议选择“UDP”。
- 2) 端口范围填写“8211”。

端口也可自定义进行修改，修改后可提高服务器的安全性：

登录云主机后，首先使用密码登录root用户，执行以下命令进入编辑配置界面。

```
vim /etc/rc.local
```

按i进入编辑状态，将第二行改为：

```
su - steam -c "cd  
/home/steam/.steam/SteamApps/common/PalServer;steamcmd +login anonymous +app_update 2394010 validate +  
./PalServer.sh -port=xxx &"
```

其中xxx为1-65535中的任意希望启用的端口，修改完成后按esc退出编辑，再输入:wq进行保存并退出。

- 3) 地址处将“1.1.1.1”更改为自己的IP地址。

添加规则 ✕

* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略： 允许 拒绝

* 优先级： 1

* 协议
TCP、UDP报文分片后，分片不带有端口信息，需要将端口范围指定为1-65535，不进行端口过滤。

* 端口范围

* 源地址： IP 安全组 前缀列表

描述

点击“确定”，完成安全组规则修改。

最佳实践

7. 进入游戏。

先在云主机控制台-云主机列表中找到云主机的公网IP地址。

实例/主机名称	镜像	安全防护	状态	标签	可用区	企业项目	IPv4地址	IPv6地址	CPU架构	规格	付费方式/创建时间	操作
			运行中		可用区1				X86计算	s7.small.1 1核 1G	包年包月 2024-02-01 17:00	远程登录 更多
			运行中		可用区1				X86计算	s7.xlarge.4 4核 16G	按量付费 2024-02-01 16:22	远程登录 更多

启动游戏，在游戏界面选择“加入多人游戏（专用服务器）”。在地址输入框输入“公网IP:8211”，点击“联系”即可进入帕鲁世界。



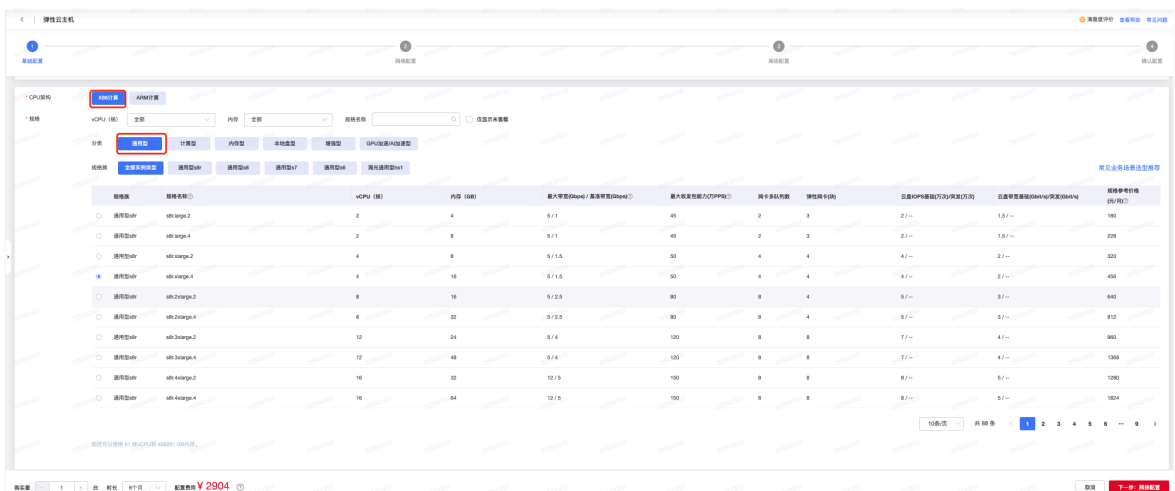
最佳实践



部署Palworld幻兽帕鲁服务器最佳实践（Windows）

1. 登录云主机控制台，选择创建云主机的资源池，点击“创建云主机”按钮。
2. 基础配置。

CPU架构选择“X86计算”，规格-分类选择“通用型”。为确保游戏顺畅运行，建议您选择4C16G及以上的规格。



镜像类型选择“应用镜像”，镜像下拉菜单中选择“windows” - “幻兽帕鲁（Palworld）”。

点击“下一步：网络配置”。

3. 网络配置。

点击“创建安全组”按钮，跳转至网络控制台-安全组页面。



点击“创建安全组”按钮，在弹窗中等待模板下拉菜单选择“开放全部端口”。



点击“确定”，完成安全组创建。

返回云主机订购页面，点击“选择安全组”按钮，在安全组列表中勾选刚才创建的安全组。



点击“确定”，完成安全组选择。

弹性IP选择“自动分配”，根据需要选择带宽大小。为确保游戏顺畅运行，建议带宽选择10M以上。



点击“下一步：高级配置”。

4. 根据提示完成高级配置，点击“下一步：确认配置”。

确认配置无误后，点击“立即购买”。

5. 完成支付后，返回云主机控制台，确认云主机状态为“运行中”，则可以后续操作。



6. 添加安全组规则。

在云主机列表中点击云主机名称，进入云主机详情页。点击安全组页签，点击“添加规则”按钮，新增IP放行。



在添加规则弹窗中：

- 1) 协议选择“UDP”。
- 2) 端口范围填写“8211”。

端口也可自定义进行修改，修改后可提高服务器的安全性：Windows镜像云主机登录后，将C:\steamcmd路径下start.bat文件中修改最后一行，改为：

```
start "" "C:\PalServer\PalServer.exe" -port=xxx
```

其中xxx为1-65535中的任意希望启用的端口，并重新启动服务器即可生效。

- 3) 地址处将“1.1.1.1”更改为自己的IP地址。

添加规则 ✕

* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略: 允许 拒绝

* 优先级: ?

* 协议 ?

TCP、UDP报文分片后，分片不带有端口信息，需要将端口范围指定为1-65535，不进行端口过滤。

* 端口范围

* 源地址: IP 安全组 前缀列表

?

描述

点击“确定”，完成安全组规则修改。

7. 进入游戏。

先在云主机控制台-云主机列表中找到云主机的公网IP地址。



启动游戏，在游戏界面选择“加入多人游戏（专用服务器）”。在地址输入框输入“公网IP:8211”，点击“联系”即可进入帕鲁世界。



Palworld幻兽帕鲁世界参数修改最佳实践（Ubuntu）

操作场景

本文以Ubuntu操作系统为例，为您介绍通过Palworld专有镜像一键部署Palworld服务器后对游戏世界参数进行修改的具体操作。

操作步骤

1. 登录root用户。

在云主机控制台点击云主机列表中“远程登录”按钮，登录云主机。

输入root后回车，再输入创建时使用的密码，即可进入root用户。

2. 执行命令 `su - steam`。
3. 执行命令 `cd /home/steam/.steam/SteamApps/common/PalServer`，进入该目录下。
4. 执行命令 `cp DefaultPalWorldSettings.ini Pal/Saved/Config/LinuxServer/PalWorldSettings.ini`。
5. 使用vim编辑器对配置进行编辑。

```
vim Pal/Saved/Config/LinuxServer/PalWorldSettings.ini
```

进入后按i进入编辑状态，修改完成后按一下Esc退出编辑，再输入:wq进行保存并退出。

6. 输入 `reboot` 重启服务器，配置即可生效。

参数对应表

Difficulty	英文	中文
DayTimeSpeedRate	Day time speed	白天时间流逝速度
NightTimeSpeedRate	Night time speed	夜晚时间流逝速度
ExpRate	EXP rate	经验获取倍率
PalCaptureRate	Pal capture rate	帕鲁捕获率
PalSpawnNumRate	Pal appearance rate	帕鲁刷新倍率
PalDamageRateAttack	Damage from pals multipiler	帕鲁攻击力倍率
PalDamageRateDefense	Damage to pals multipiler	帕鲁防御力倍率
PlayerDamageRateAttack	Damage from player multipiler	玩家攻击力倍率
PlayerDamageRateDefense	Damage to player multipiler	玩家防御力倍率
PlayerStomachDecreaceRate	Player hunger depletion rate	玩家饥饿消耗率
PlayerStaminaDecreaceRate	Player stamina reduction rate	玩家体力减少率
PlayerAutoHPRegeneRate	Player auto HP regeneration rate	玩家自动HP回复率
PlayerAutoHpRegeneRateInSleep	Player sleep HP regeneration rate	玩家睡眠HP回复率
PalStomachDecreaceRate	Pal hunger depletion rate	帕鲁饥饿消耗率
PalStaminaDecreaceRate	Pal stamina reduction rate	帕鲁体力减少率
PalAutoHPRegeneRate	Pal auto HP regeneration rate	帕鲁自动HP回复率

最佳实践

Difficulty	英文	中文
PalAutoHpRegeneRateInSleep	Pal sleep health regeneration rate (in Palbox)	帕鲁睡眠HP回复率（帕鲁终端中）
BuildObjectDamageRate	Damage to structure multipiler	建筑物血量
BuildObjectDeteriorationDamageRate	Structure determination rate	建筑物风化速率
CollectionDropRate	Getherable items multipiler	采集物掉落率
CollectionObjectHpRate	Getherable objects HP multipiler	采集物回复率
CollectionObjectRespawnSpeedRate	Getherable objects respawn interval	采集物刷新率
EnemyDropItemRate	Dropped Items Multipiler	击败敌人物品掉落率
DeathPenalty	Death penalty None : No lost, Item : Lost item without equipment, ItemAndEquipment : Lost item and equipment, All : Lost All item, equipment, pal(in inventory)	死亡惩罚None：没有丢失；Item：丢失物品，不丢失帕鲁；ItemAndEquipment：丢失的物品和装备；All：丢失所有物品、装备、帕鲁（背包中）
GuildPlayerMaxNum	Max player of Guild	公会最大玩家数
PalEggDefaultHatchingTime	Time(h) to incubate massive egg	孵化巨大蛋的时间(h)
BaseCampWorkerMaxNum	Max worker in camp	每个据点工作帕鲁数量上限
WorkSpeedRate	Rate of working speed	工作效率
ServerPlayerMaxNum	Maximum number of people who can join the server	服务器最多可加入人数
ServerName	Server name	服务器名称
ServerDescription	Server description	服务器描述
AdminPassword	AdminPassword	管理员密码
ServerPassword	Set the server password.	设置服务器密码(仅作为社群服务器时有效)
PublicPort	Public port number	公共端口号
PublicIP	Public IP	公共IP
RCONEnabled	Enable RCON	启用RCON
RCONPort	Port number for RCON	RCON的端口号

Palworld幻兽帕鲁世界参数修改最佳实践（Windows）

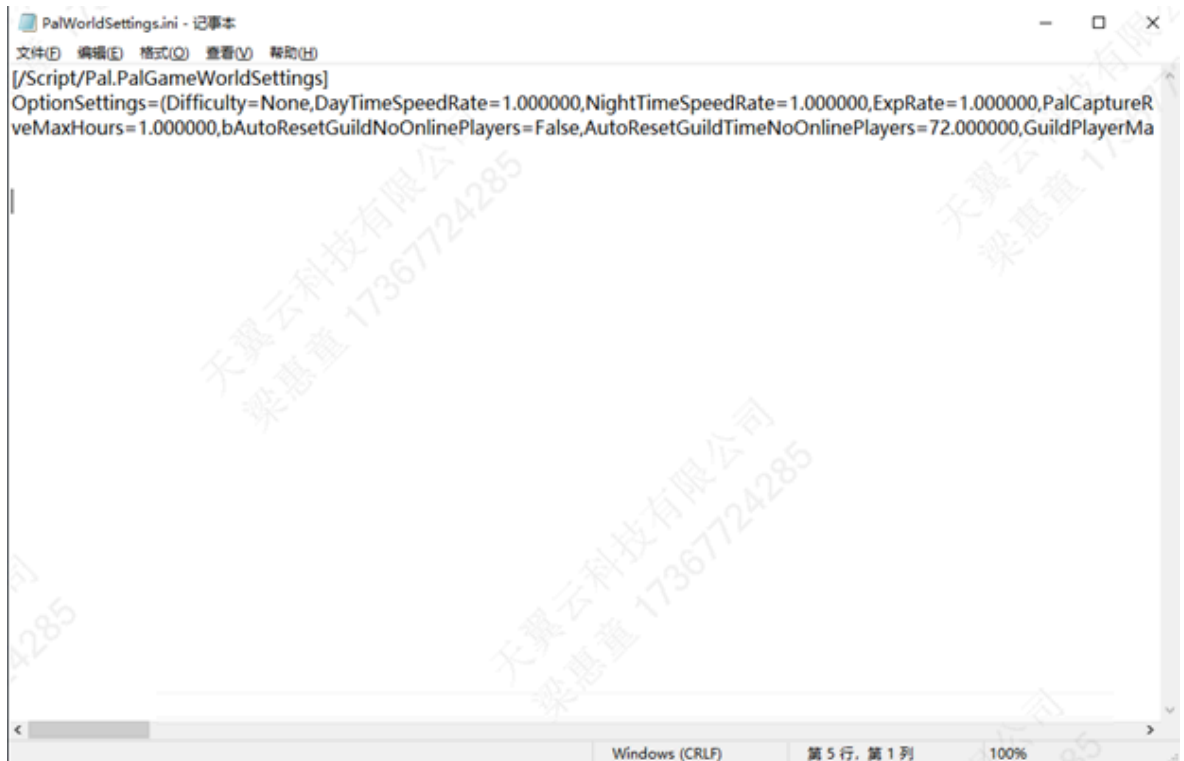
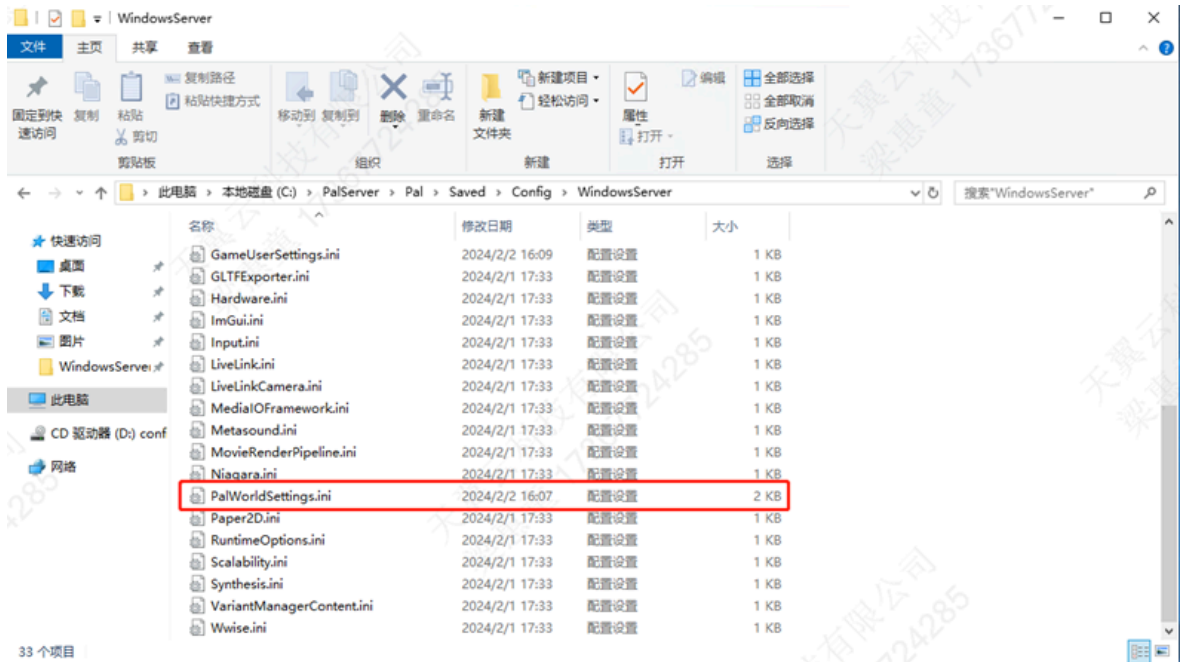
操作场景

本文以Windows操作系统为例，为您介绍通过Palworld专有镜像一键部署Palworld服务器后对游戏世界参数进行修改的具体操作。

最佳实践

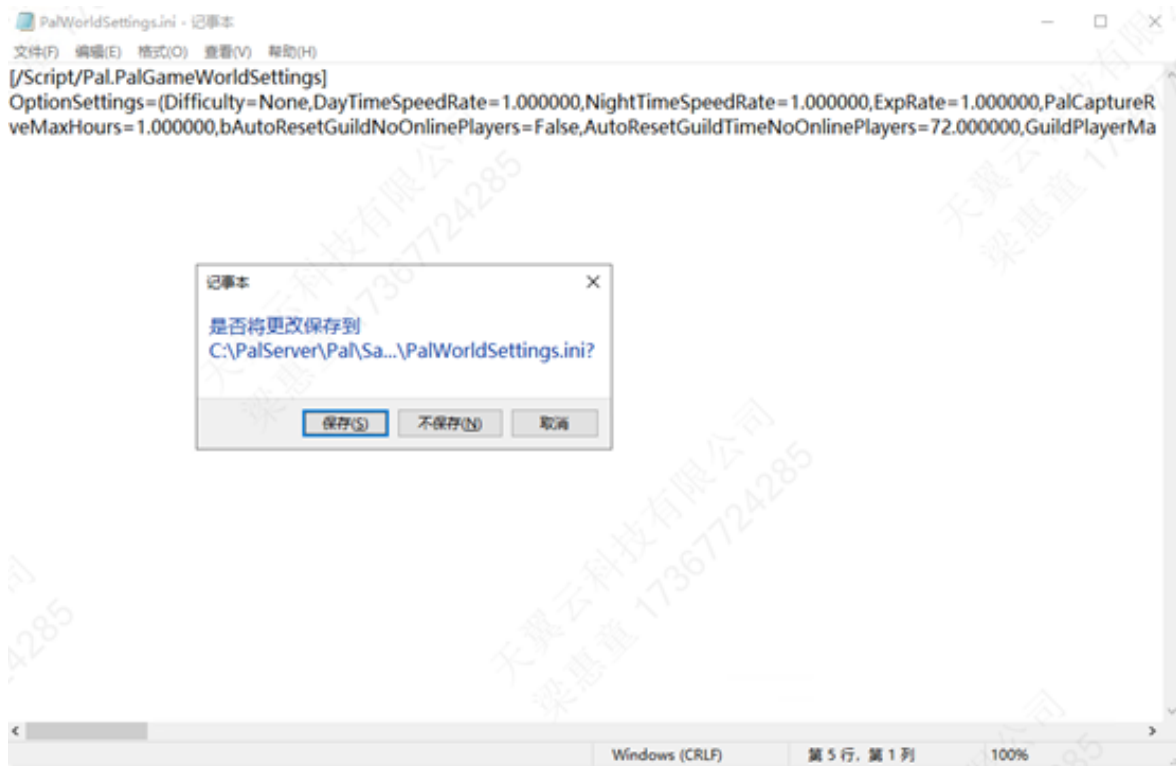
操作步骤

1. 进入C:\PalServer\Pal\Saved\Config\WindowsServer 路径。
2. 使用记事本打开PalWorldSettings.ini文件。



最佳实践

3. 根据参数对应表修改自己喜欢的世界参数，修改完成后点击“保存”。



4. 重启云主机，配置的参数即可生效。

参数对照表

Difficulty	英文	中文
DayTimeSpeedRate	Day time speed	白天时间流逝速度
NightTimeSpeedRate	Night time speed	夜晚时间流逝速度
ExpRate	EXP rate	经验获取倍率
PalCaptureRate	Pal capture rate	帕鲁捕获率
PalSpawnNumRate	Pal appearance rate	帕鲁刷新倍率
PalDamageRateAttack	Damage from pals multipiler	帕鲁攻击力倍率
PalDamageRateDefense	Damage to pals multipiler	帕鲁防御力倍率
PlayerDamageRateAttack	Damage from player multipiler	玩家攻击力倍率
PlayerDamageRateDefense	Damage to player multipiler	玩家防御力倍率
PlayerStomachDecreaceRate	Player hunger depletion rate	玩家饥饿消耗率
PlayerStaminaDecreaceRate	Player stamina reduction rate	玩家体力减少率
PlayerAutoHPRegeneRate	Player auto HP regeneration rate	玩家自动HP回复率
PlayerAutoHpRegeneRateInSleep	Player sleep HP regeneration rate	玩家睡眠HP回复率

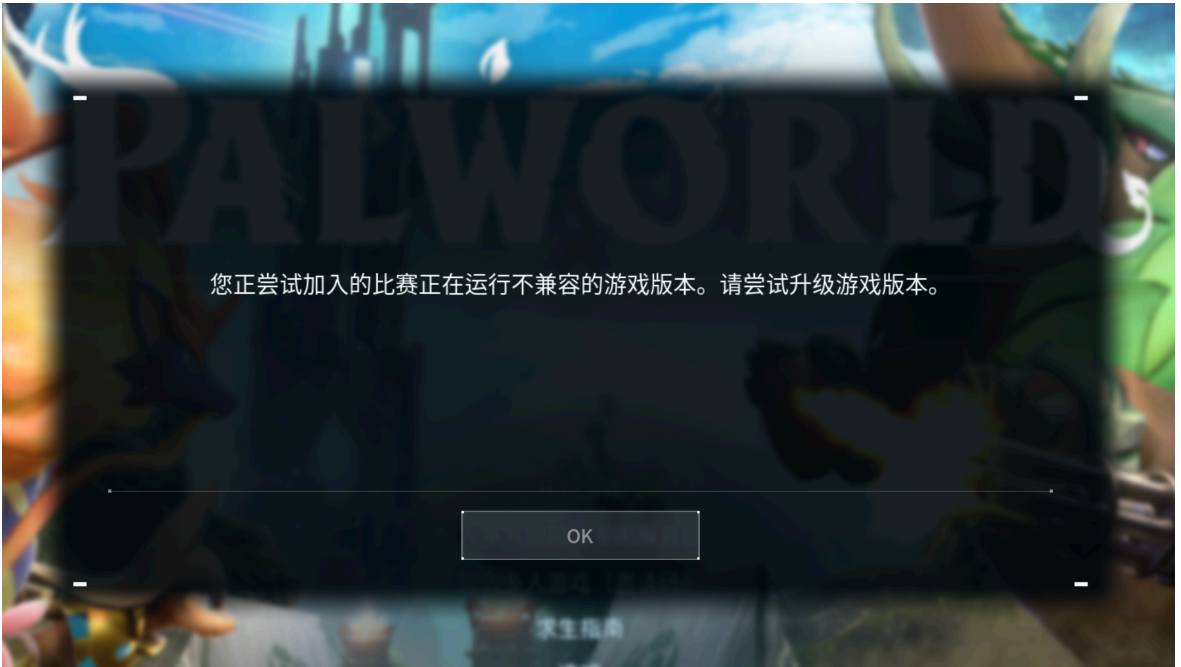
最佳实践

Difficulty	英文	中文
PalStomachDecreaseRate	Pal hunger depletion rate	帕鲁饥饿消耗率
PalStaminaDecreaseRate	Pal stamina reduction rate	帕鲁体力减少率
PalAutoHPRegeneRate	Pal auto HP regeneration rate	帕鲁自动HP回复率
PalAutoHpRegeneRateInSleep	Pal sleep health regeneration rate (in Palbox)	帕鲁睡眠HP回复率（帕鲁终端中）
BuildObjectDamageRate	Damage to structure multipiler	建筑物血量
BuildObjectDeteriorationDamageRate	Structure determination rate	建筑物风化速率
CollectionDropRate	Getherable items multipiler	采集物掉落率
CollectionObjectHpRate	Getherable objects HP multipiler	采集物回复率
CollectionObjectRespawnSpeedRate	Getherable objects respawn interval	采集物刷新率
EnemyDropItemRate	Dropped Items Multipiler	击败敌人物品掉落率
DeathPenalty	Death penalty None : No lost, Item : Lost item without equipment, ItemAndEquipment : Lost item and equipment, All : Lost All item, equipment, pal(in inventory)	死亡惩罚None：没有丢失；Item：丢失物品，不丢失帕鲁；ItemAndEquipment：丢失的物品和装备；All：丢失所有物品、装备、帕鲁（背包中）
GuildPlayerMaxNum	Max player of Guild	公会最大玩家数
PalEggDefaultHatchingTime	Time(h) to incubate massive egg	孵化巨大蛋的时间(h)
BaseCampWorkerMaxNum	Max worker in camp	每个据点工作帕鲁数量上限
WorkSpeedRate	Rate of working speed	工作效率
ServerPlayerMaxNum	Maximum number of people who can join the server	服务器最多可加入人数
ServerName	Server name	服务器名称
ServerDescription	Server description	服务器描述
AdminPassword	AdminPassword	管理员密码
ServerPassword	Set the server password.	设置服务器密码(仅作为社群服务器时有效)
PublicPort	Public port number	公共端口号
PublicIP	Public IP	公共IP
RCONEnabled	Enable RCON	启用RCON
RCONPort	Port number for RCON	RCON的端口号

Palworld幻兽帕鲁更新游戏最佳实践

如果您进入游戏时遇到如图报错，请参考本文对游戏进行更新。

最佳实践



Ubuntu

启动云主机时可以触发后台的自动检查游戏版本，如需更新游戏，在云主机控制台点击“重启”即可。



Windows

自动更新

启动云主机时可以触发后台的自动检查游戏版本，如需更新游戏，在云主机控制台点击“重启”即可。



手动更新

1. 在本地Windows电脑搜索远程桌面连接，并输入云主机的公网IP地址。



2. 使用设置的密码进行登录，用户为Administrator，密码输入设置的云主机密码并连接。
3. 打开任务管理器，找到Pal进程，右键点击并选择结束任务。

已有天翼云Ubuntu云主机，搭建幻兽帕鲁服务器

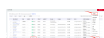
操作前提

- 如果您已购买的云主机位于【青岛20、西南1】资源池（其余资源池镜像正在加载中），且满足以下条件，请参考本文部署幻兽帕鲁服务器。
 - 操作系统：Ubuntu 20.04
 - 配置：4核16G及以上
 - 已开通公网IP
 - 云主机未到期

操作步骤

1. 登录云主机控制台，前往您已购买的云主机所在的资源池。确认云主机状态为“已关机”，则可以进行重装操作系统操作。

点击“更多”按钮，点击下拉菜单中“一键重装”按钮。



最佳实践

2. 在重装操作系统弹窗中，镜像类型选择“应用镜像”，并在重装选择下拉菜单中选择“Ubuntu”、“幻兽帕鲁（Palworld）”。



设置并确认密码，点击“确定”按钮，完成重装操作系统操作。

3. 返回云主机控制台，选中已完成重装操作系统的云主机，点击“开机”按钮。确认云主机状态为“运行中”，则可以进行后续操作。

4. 添加安全组规则。

在云主机列表中点击云主机名称，进入云主机详情页。点击安全组页签，点击“添加规则”按钮，新增IP放行。



在添加规则弹窗中：

- 1) 协议选择“UDP”。
- 2) 端口范围填写“8211”。

端口也可自定义进行修改，修改后可提高服务器的安全性：

登录云主机后，首先使用密码登录root用户，执行以下命令进入编辑配置界面。

```
vim /etc/rc.local
```

按i进入编辑状态，将第二行改为：

```
su - steam -c "cd  
/home/steam/.steam/SteamApps/common/PalServer;steamcmd +login anonymous +app_update 2394010 validate +  
./PalServer.sh -port=xxx &"
```

其中xxx为1-65535中的任意希望启用的端口，修改完成后按esc退出编辑，再输入:wq进行保存并退出。

- 3) 地址处将“1.1.1.1”更改为自己的IP地址。

添加规则 ✕

* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略: 允许 拒绝

* 优先级: ?

* 协议 ?

TCP、UDP报文分片后，分片不带有端口信息，需要将端口范围指定为1-65535，不进行端口过滤。

* 端口范围

* 源地址: IP 安全组 前缀列表

?

描述

点击“确定”，完成安全组规则修改。

5. 登录游戏。

在云主机控制台找到云主机的公网IP地址。



启动游戏，在游戏界面选择“加入多人游戏（专用服务器）”。在地址输入框输入该地址，点击“联系”即可进入帕鲁世界（“:8211”不用进行修改）。



已有天翼云Windows云主机，搭建幻兽帕鲁服务器

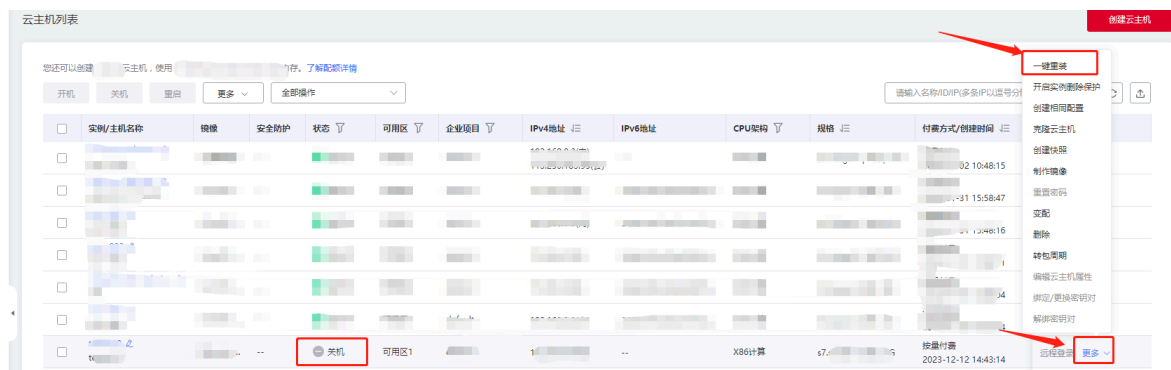
操作前提

- 如果您已购买的云主机位于【青岛20、西南1】资源池（其余资源池镜像正在加载中），且满足以下条件，请参考本文部署幻兽帕鲁服务器。
 - 操作系统：Windows
 - 配置：4核16G及以上
 - 已开通公网IP
 - 云主机未到期

操作步骤

1. 登录云主机控制台，前往您已购买的云主机所在的资源池。确认云主机状态为“已关机”，则可以进行重装操作系统操作。

点击“更多”按钮，点击下拉菜单中“一键重装”按钮。



最佳实践

2. 在重装操作系统弹窗中，镜像类型选择“应用镜像”，并在重装选择下拉菜单中选择“Windows”、“幻兽帕鲁 (Palworld)”。

一键重装 ×

⚠️ 1、重装后IP地址不变，但系统盘会清空，请预先做好云硬盘备份。重装后云主机备份将无法用于恢复数据、云主机快照为错误状态，不支持申请云主机、恢复数据。
2、重装后系统盘所在的云硬盘备份策略无法继续使用。请在重装前将系统盘从所在备份策略中移出，重装后重新加入；若未提前移出，请在重装后及时新建云盘备份策略。
3、Linux系统下使用了LVM的用户，LVM如果加入了数据盘，请先从LVM卸载掉数据盘，再一键重装，否则会导致主机重装后无法使用。
4、多次批量一键重装操作建议间隔120s以上，否则可能造成重装失败。
5、使用加密系统盘重装大约需要10~25分钟，容量越大加密耗时越久。

* 镜像类型：

* 重装选择：

* 密码：

* 确认密码：

* 用户数据：

以下云主机将一键重装

实例名称	状态	IPv4地址	规格	镜像	系统盘容量	创建时间
ecm-2c53	关机	1...	s7.xlarge.4 4核...	3...	80GB	2024-02-04 11:...

设置并确认密码，点击“确定”按钮，完成重装操作系统操作。

3. 返回云主机控制台，选中已完成重装操作系统的云主机，点击“开机”按钮。确认云主机状态为“运行中”，则可以进行后续操作。

4. 添加安全组规则。

在云主机列表中点击云主机名称，进入云主机详情页。点击安全组页签，点击“添加规则”按钮，新增IP放行。



在添加规则弹窗中：

- 1) 协议选择“UDP”。
- 2) 端口范围填写“8211”。

端口也可自定义进行修改，修改后可提高服务器的安全性：Windows镜像云主机登录后，将C:\steamcmd路径下start.bat文件中修改最后一行，改为：

```
start "" "C:\PalServer\PalServer.exe" -port=xxx
```

其中xxx为1-65535中的任意希望启用的端口，并重新启动服务器即可生效。

- 3) 地址处将“1.1.1.1”更改为自己的IP地址。

添加规则 ✕

* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略: 允许 拒绝

* 优先级: ?

* 协议 ?

TCP、UDP报文分片后，分片不带有端口信息，需要将端口范围指定为1-65535，不进行端口过滤。

* 端口范围

* 源地址: IP 安全组 前缀列表

?

描述

点击“确定”，完成安全组规则修改。

最佳实践

5. 登录游戏。

在云主机控制台找到云主机的公网IP地址。



启动游戏，在游戏界面选择“加入多人游戏（专用服务器）”。在地址输入框输入该地址，点击“联系”即可进入帕鲁世界（“:8211”不用进行修改）。

A screenshot of the Palia game's "DEDICATED SERVER" interface. The interface is in Chinese and shows a list of servers with columns for "世界名称" (World Name), "Day", "人数" (Players), "Ping", and "Version". A search bar is at the top, and a sidebar on the left contains buttons for "官方服务器", "社群服务器", "最近访问过的服务器列表", "建立服务器的方法", and "服务器故障信息". At the bottom, there is an "输入密码" (Enter Password) field containing "127.0.0.1:8211" and a "联系" (Contact) button. A red arrow points from the "服务器故障信息" button to the password field.

世界名称	Day	人数	Ping	Version
Official EU Server 0024	2	31/32	298	v0.1.3.0
Official EU Server 0052	2	31/32	280	v0.1.3.0
Official China Server 0101	5	30/32	78	v0.1.3.0
Official China Server 0099	49	30/32	92	v0.1.3.0
Official NA Server 0080	3	29/32	233	v0.1.3.0
Official China Server 0150	84	29/32	44	v0.1.3.0
Official Asia Server 0074	393	28/32	98	v0.1.3.0
Official EU Server 0138	7	27/32	282	v0.1.3.0
Official NA Server 0190	3	23/32	211	v0.1.3.0
Official China Server 0146	17	23/32	113	v0.1.3.0
Official China Server 0082	12	22/32	65	v0.1.3.0
Official NA Server 0111	10	21/32	243	v0.1.3.0

大模型学习机最佳实践

一键部署与登录LLaMA 3/LLaMA 2大模型学习机

按需购买学习机

1. 通过[大模型学习机专属活动优惠页](#)一键购买开通大模型学习机。



一键部署Llama3大模型学习机

基于天翼云主机, 预装最新主流大模型LLaMA3和StableDiffusion及开源web框架, 可浏览器直接访问, 在网页上进行操作。

2. 点击“新老同享优惠”，选择您要购买的学习机规格，点击“立即订购”。

大模型学习机：面向新老用户，提供多种学习机规格

入门款8核32G云主机低至3折起，新老共享，限购1台，推荐购买5M及以上 [活动规则>>](#)

GPU学习机-入门款(LLaMA3)	GPU学习机-进阶款(LLaMA3)	CPU学习机-入门款(LLaMA2)	CPU学习机-进阶款(LLaMA3)
<p>适合希望进行初步模型微调学习的用户，支持低精度模型推理与训练，采用专业级T4显...</p> <p>PI2型GPU云主机 T4加速</p> <p>规格选择 8核32G ▾</p> <p>系统盘大小 (GB) 60</p> <p>带宽 (M) 1</p> <p>时长 1个月 ▾</p> <p>新老用户均可购买</p> <p>1,967.35 元</p> <p>立省1,609.65元</p> <p>立即订购</p>	<p>适合希望进行深入模型调研学习的用户，支持各类主流大模型推理与训练，采用专业级A...</p> <p>PI7型GPU云主机 A10加速</p> <p>规格选择 16核64G ▾</p> <p>系统盘大小 (GB) 60</p> <p>带宽 (M) 1</p> <p>时长 1个月 ▾</p> <p>新老用户均可购买</p> <p>4,509.43 元</p> <p>立即订购</p>	<p>适合刚开始接触大模型，且对推理时延要求不高的用户，支持部分主流大模型推理，在保...</p> <p>S7通用型云主机 8C32G</p> <p>规格选择 8核32G ▾</p> <p>系统盘大小 (GB) 60</p> <p>带宽 (M) 1</p> <p>时长 1个月 ▾</p> <p>新老用户均可购买</p> <p>284.83 元</p> <p>立省664.61元</p> <p>立即订购</p>	<p>适合希望体验更多大模型，同时注重性价比的用户，提供性能更高的CPU和更大的内存...</p> <p>S7通用型云主机 16C64G</p> <p>规格选择 16核64G ▾</p> <p>系统盘大小 (GB) 60</p> <p>带宽 (M) 1</p> <p>时长 1个月 ▾</p> <p>新老用户均可购买</p> <p>551.06 元</p> <p>立省1,285.82元</p> <p>立即订购</p>

最佳实践

3. 您可以根据您的实际需求，调整学习机的系统盘大小、带宽大小以及购买时长。

GPU学习机-入门款(llama3)

×

区域	重庆2-可用区
规格选择	8核32G
镜像	LLaMA3-StableDiffusion-Wi
系统盘类型	通用型SSD
系统盘大小 (GB)	60
数据盘类型	通用型SSD
数据盘大小 (GB)	0
带宽 (M)	1
时长	1个月
台数	1

5.49折

1,967.35 元

立省1,609.65元

立即订购

4. 再次点击“立即订购”，支付成功后返回云主机控制台，选择订购云主机所在的资源池。云主机状态变更为“运行中”后即为开通成功，可以进行后续操作。

云主机控制台

云主机列表

您还可以创建 149 台云主机，使用 1184 核vCPU和 409536 GB内存。[了解配额详情](#)

开机 关机 重启 更多 全部操作 请输入名称/ID/IP(多条IP以逗号分隔)

<input type="checkbox"/>	实例/主机名称	镜像	安全防护	状态	企业项目	IPv4地址	操作
<input type="checkbox"/>	llama-2 ecm-424b	LLaMA3-..	防护中	运行中	default	IP地址 (公网)	远程登录 更多

最佳实践

登录大模型

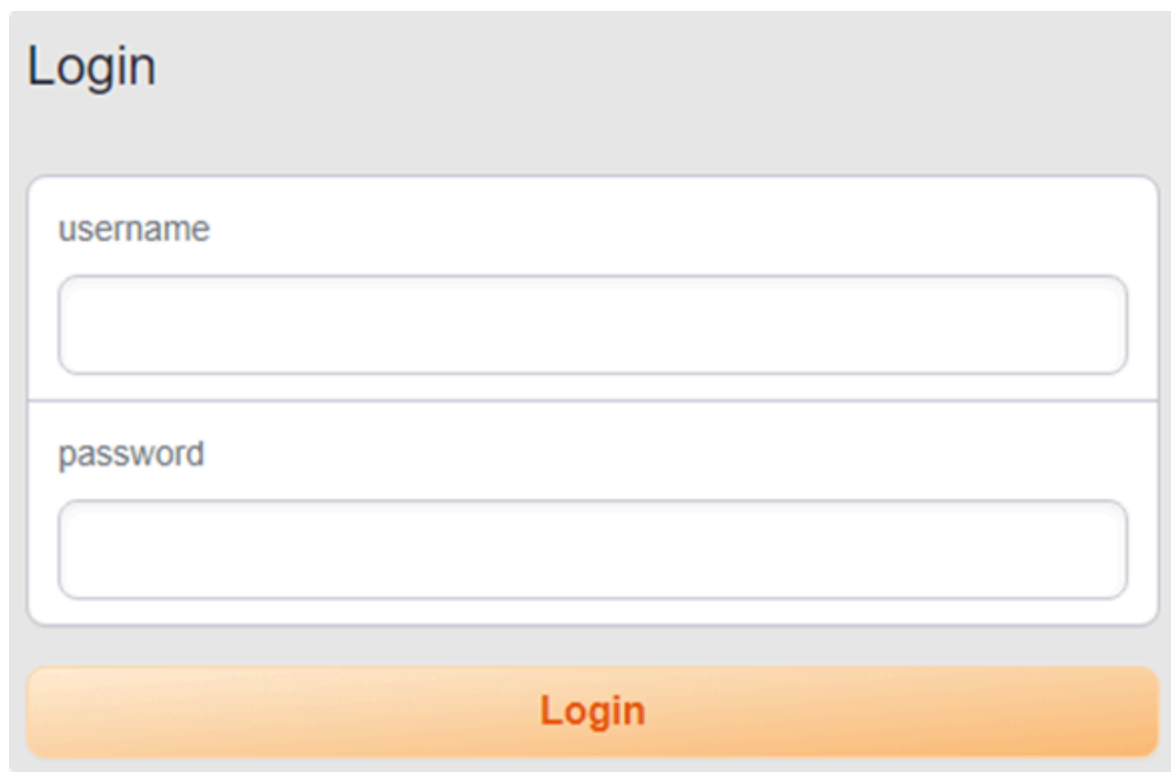
1. 登录前准备：在开通云主机列表的ipv4地址处，复制列表中的公共弹性IP。



最佳实践

2. 登录文本对话大模型（LLaMA2/LLaMA3）

将刚刚复制到的弹性IP替换到`http://弹性IP:7861/`中，并在浏览器中输入该地址跳转至登录页面。



The image shows a login interface with a light gray background. At the top left, the word "Login" is written in a dark blue font. Below this, there are two input fields: the first is labeled "username" and the second is labeled "password". Both fields are empty and have rounded corners. At the bottom of the form, there is a large orange button with the word "Login" written in white text.

您可以任选以下两个账号其一进行登录：

注意

建议您登录后尽快修改密码，修改密码方法在文档“[修改大模型学习机登录密码](#)”可查看。

账号一：

username: user

password: **\$I\$CFLvkJ69I**

账户二：

username: user2

password: C700hgW5fxkN

3. 登录AI图片生成大模型（StableDiffusion）

将刚刚复制到的弹性IP替换到<http://弹性IP:7860/> 中并在浏览器中输入跳转至登录页面。

您可以任选以下两个账号其一进行登录：

注意

建议您登录后尽快修改密码，修改密码方法在文档“[修改大模型学习机登录密码](#)”可查看。

账户一：

username: user

password: **\$I\$CFLvkJ69I**

账户二：

username: user2

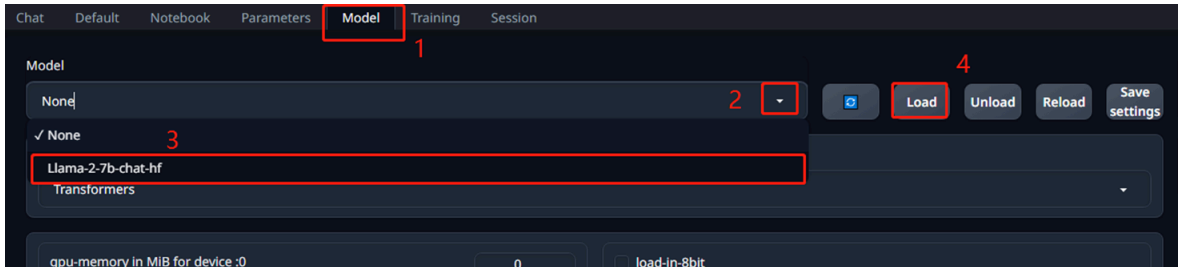
password: C700hgW5fxkN

最佳实践

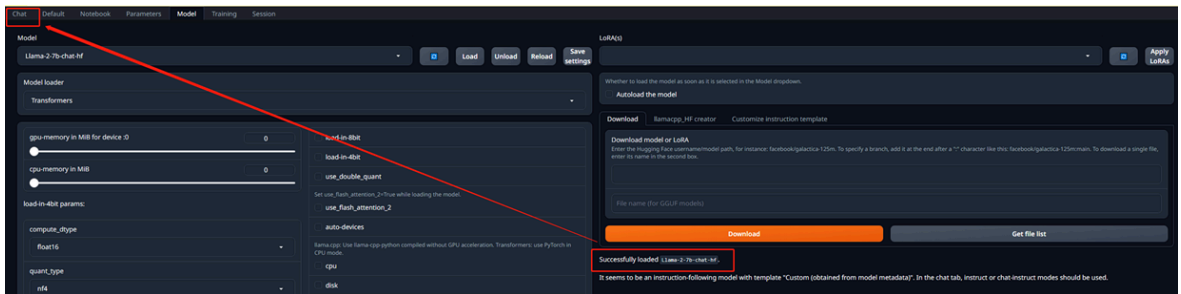
使用大模型进行推理计算

1. 文本对话大模型（LLaMA2/LLaMA3）

首先需要在Model标签页点击LLaMA2/LLaMA3大模型，然后点击Load按钮进行加载。



稍等一会，提示“Successfully loaded”，即可点击左上角的“chat”按钮，体验使用文本对话大模型（LLaMA2/LLaMA3）啦！

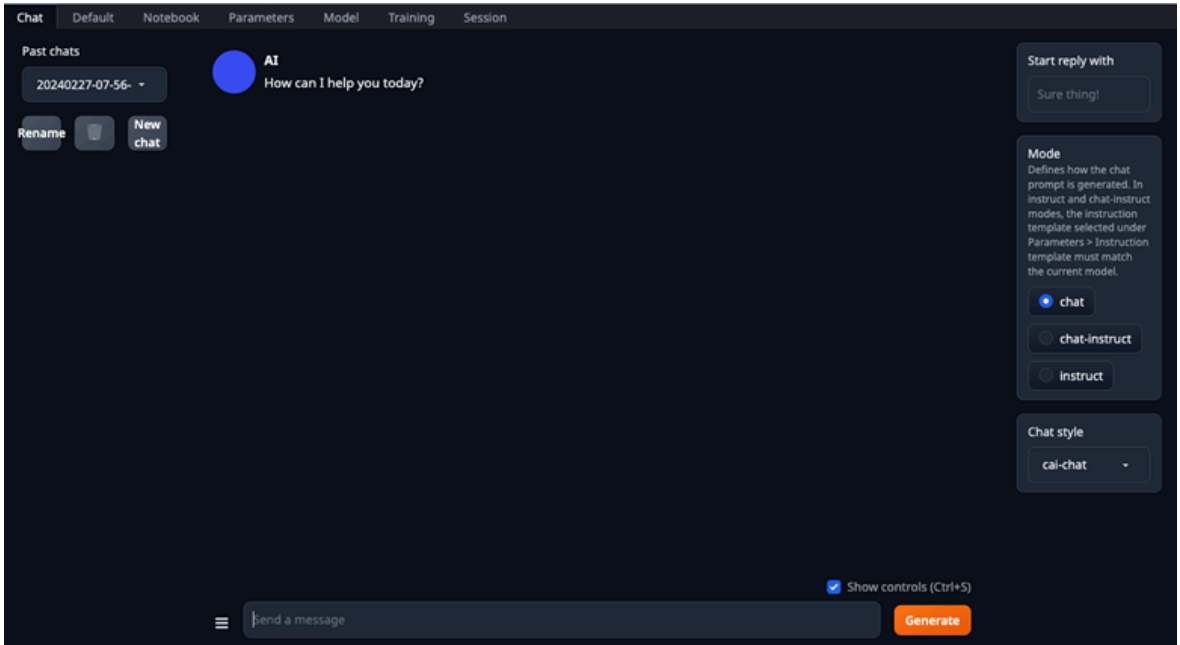


注意

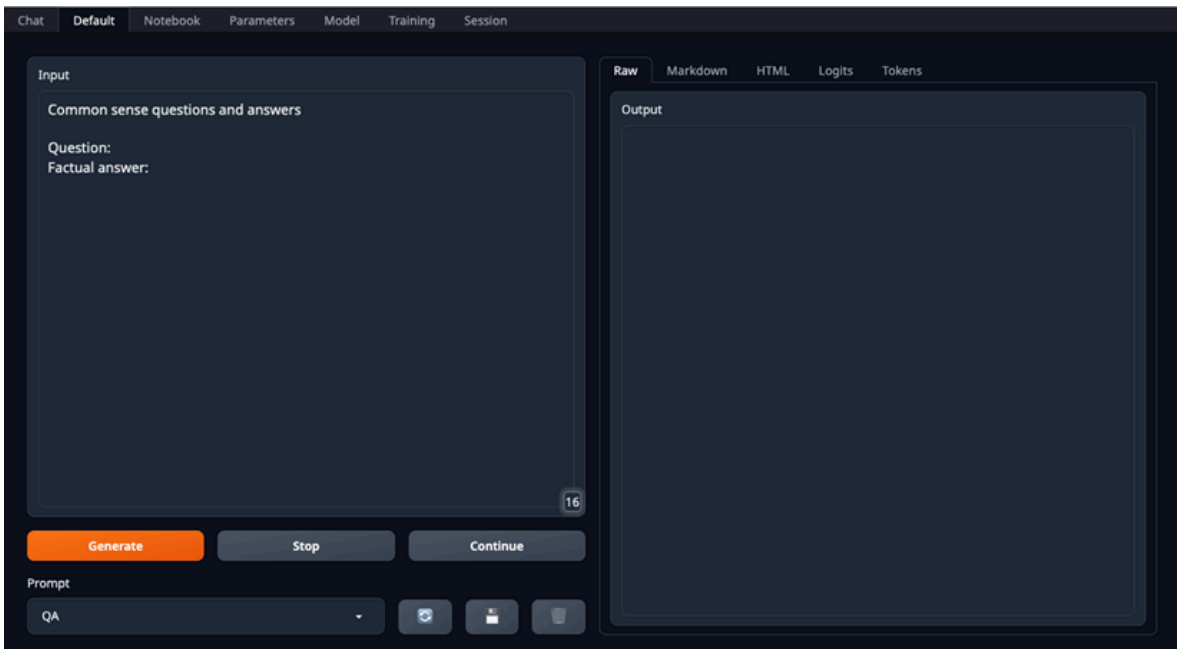
在加载Llama 2或者LLaMA3模型时，Pi2和s7.2xlarge.4规格使用默认配置会因为显存/内存不足加载失败，需要通过勾选load-in-8bit选项降低模型精度或关闭StableDiffusion服务后再加载Llama 2/LLaMA3模型。参考文档：[大模型学习机服务启停](#)。

使用文本多轮对话页面：

最佳实践



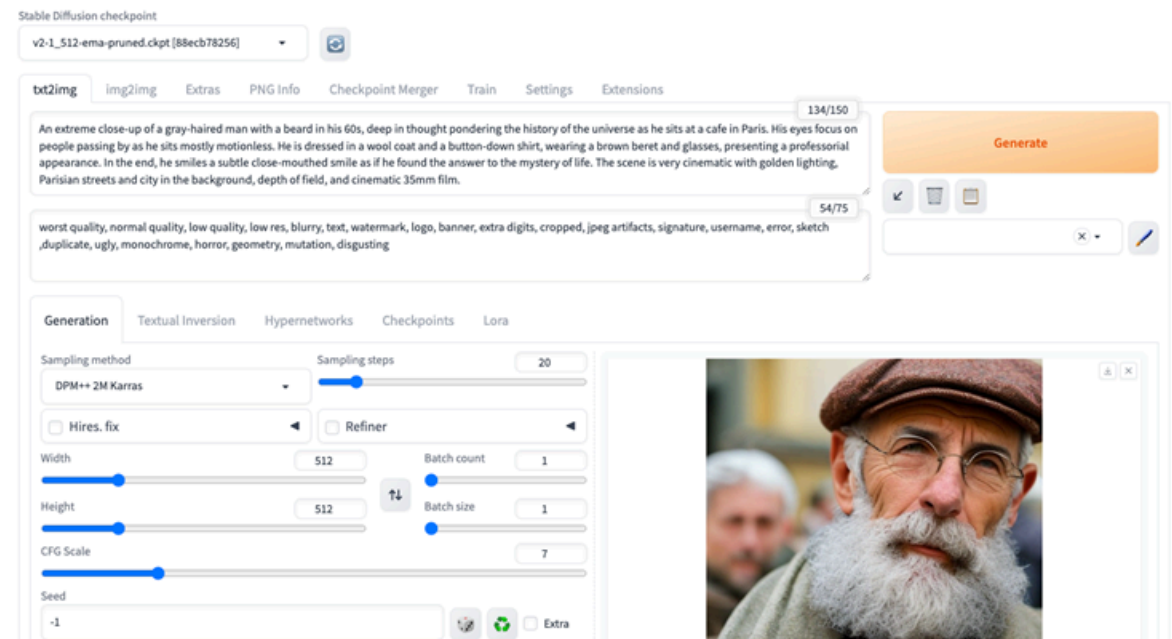
文章生成页面:



最佳实践

2. AI图片生成大模型（Stable Diffusion）：

AI图片生成大模型登录后可以直接使用，以下为图片生成页面。



部署与登录LLaMA 3/LLaMA 2大模型学习机

您可以通过本文的指导，使用镜像部署大模型学习机。

购买云主机

1. 登录[云主机控制台](#)，选择创建云主机资源池，点击“创建云主机”按钮。



最佳实践

2. 您可在以下资源池进行不同规格的开通:

注意

在加载Llama 2或者LLaMA3模型时, Pi2和s7.2xlarge.4规格使用默认配置会因为显存/内存不足加载失败, 需要通过勾选load-in-8bit选项降低模型精度或关闭StableDiffusion服务后再加载Llama 2/LLaMA3模型. 参考文档:[大模型学习机服务启停](#)。

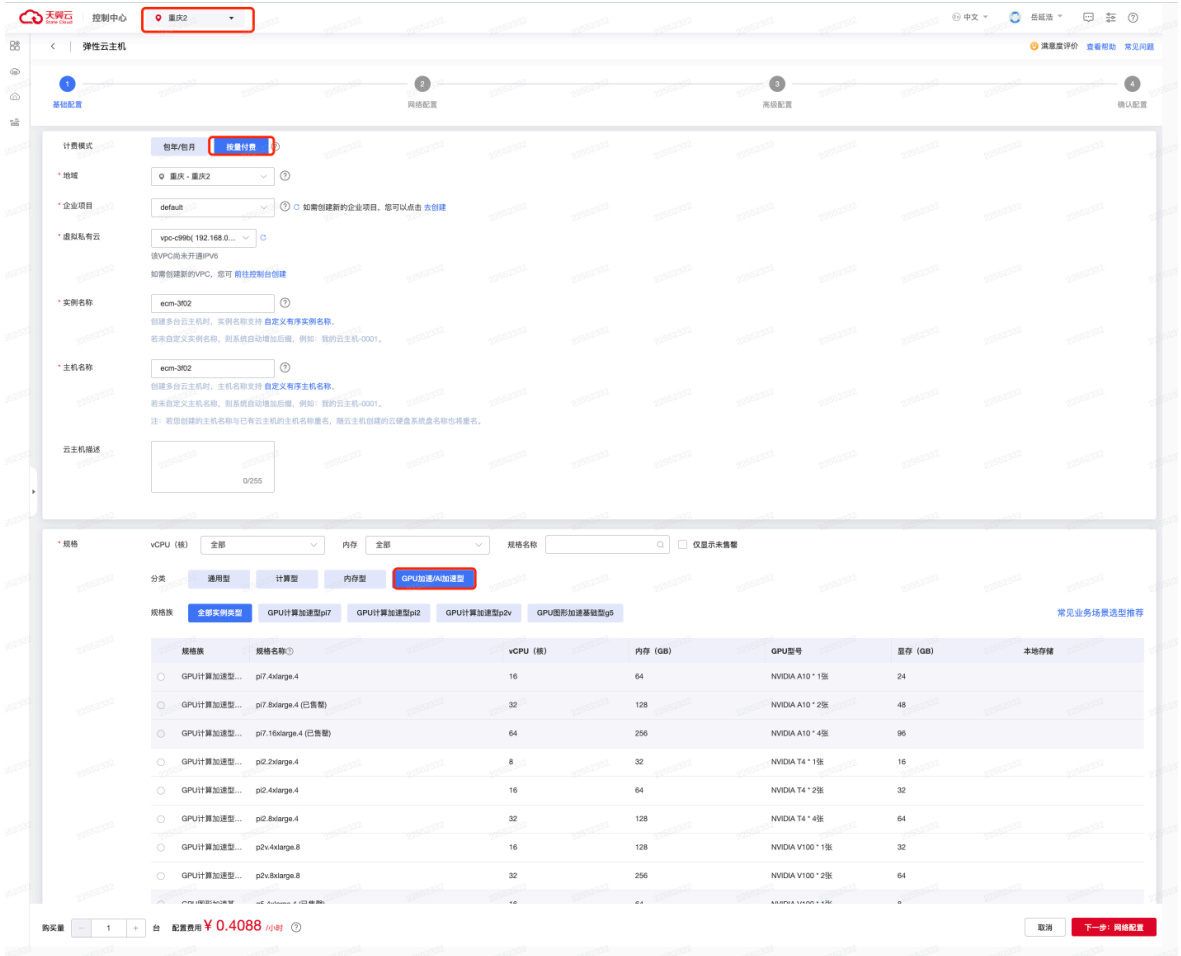
云服务器规格族	云服务器规格	支持大模型镜像资源池	镜像
GPU计算加速型	P17	重庆-重庆2、宁夏-中卫5、辽宁-辽阳1	LLaMA2-StableDffusion-WebUI-GPU LLaMA3-StableDffusion-WebUI-GPU
GPU计算加速型	P12	重庆-重庆2、上海-上海7	LLaMA2-StableDffusion-WebUI-GPU LLaMA3-StableDffusion-WebUI-GPU
通用型	S7(64G以上规格)	西南1-az1/az2	LLaMA2-StableDffusion-WebUI-CPU LLaMA3-StableDffusion-WebUI-CPU

最佳实践

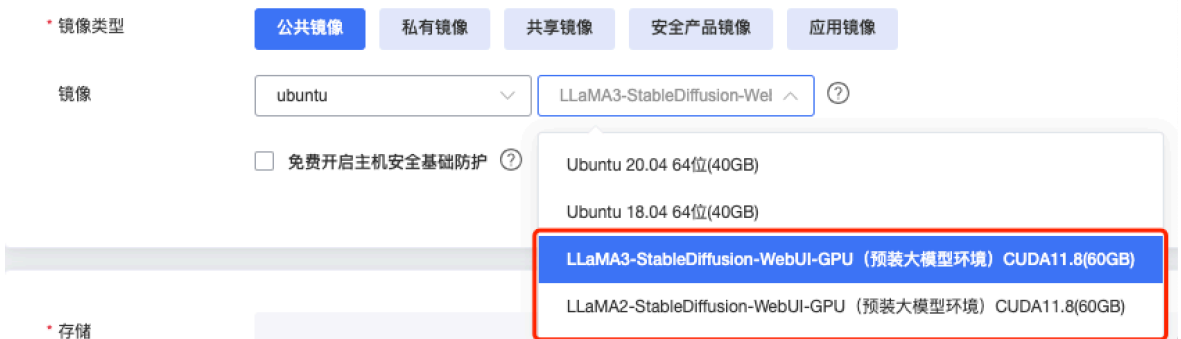
基础配置

1. 根据您的需要选择计费方式：包年/包月或按量付费。

规格分类处选择“GPU加速/AI加速型”，在下方的规格列表的pi2和pi7中选择需要的规格。



2. 选择镜像“ubuntu-LLaMA2-StableDiffusion-WebUI-GPU（预装大模型环境）（60GB）”或“ubuntu-LLaMA3-StableDiffusion-WebUI-GPU（预装大模型环境）（60GB）”，点击“下一步：网络配置”。



最佳实践

网络配置

1. 点击“创建安全组”按钮，跳转至网络控制台-安全组页面。
2. 点击“创建安全组”按钮，在弹窗中等待模板下拉菜单选择“开放全部端口”。
3. 点击“确定”，完成安全组创建。
4. 返回云主机订购页面，点击“选择安全组”按钮，在安全组列表中勾选刚才创建的安全组。
5. 点击“确定”，完成安全组选择。
6. 弹性IP选择“自动分配”，根据需要选择带宽大小。为确保大模型学习机顺畅运行，如有其他数据下载需求，建议10M以上，否则建议5M及以下。
7. 点击“下一步：高级配置”。
8. 根据提示完成高级配置，点击“下一步：确认配置”。确认配置无误后，点击“立即购买”。
9. 支付成功后返回云主机控制台，选择订购云主机所在的地域。云主机状态变更为“运行中”后即为开通成功，可以进行后续操作。

添加安全组规则

1. 点击云主机名称进入云主机详情页。

<input type="checkbox"/>	实例/主机名称	镜像	安全防护	状态	企业项目	IPv4地址	IPv6地址	规格	付费方式/创建时间	操作
<input type="checkbox"/>	ecm-c44a	LLaMA2...	未防护	运行中	default			pi7.4xlarge.4 16核 ... GPU: NVIDIA A10*1 ...		远程登录 更多

2. 选择云主机详情页下方的安全组页签。

The screenshot shows the '弹性云主机' (Elastic Cloud Server) details page. The '安全组' (Security Groups) tab is selected and highlighted with a red box. The page displays a table of security groups with columns for '安全组' (Security Group), '出口方向规则' (Egress Rules), and '入口方向规则' (Ingress Rules). The default security group is highlighted with a red box.

3. 点击默认安全组展开详情按钮。

The screenshot shows the expanded details for the default security group. The '安全组' (Security Group) tab is selected and highlighted with a red box. The page displays a table of security groups with columns for '安全组' (Security Group), '出口方向规则' (Egress Rules), and '入口方向规则' (Ingress Rules). The default security group is highlighted with a red box.

最佳实践

4. 点击“添加规则”，在端口范围处填写7860，点击确定。



5. 再次点击“添加规则”，在端口范围处填写7861，点击确定。

最佳实践

登录大模型

1. 登录前准备：选择云主机详情的“弹性IP”页签，复制列表中的弹性IP。

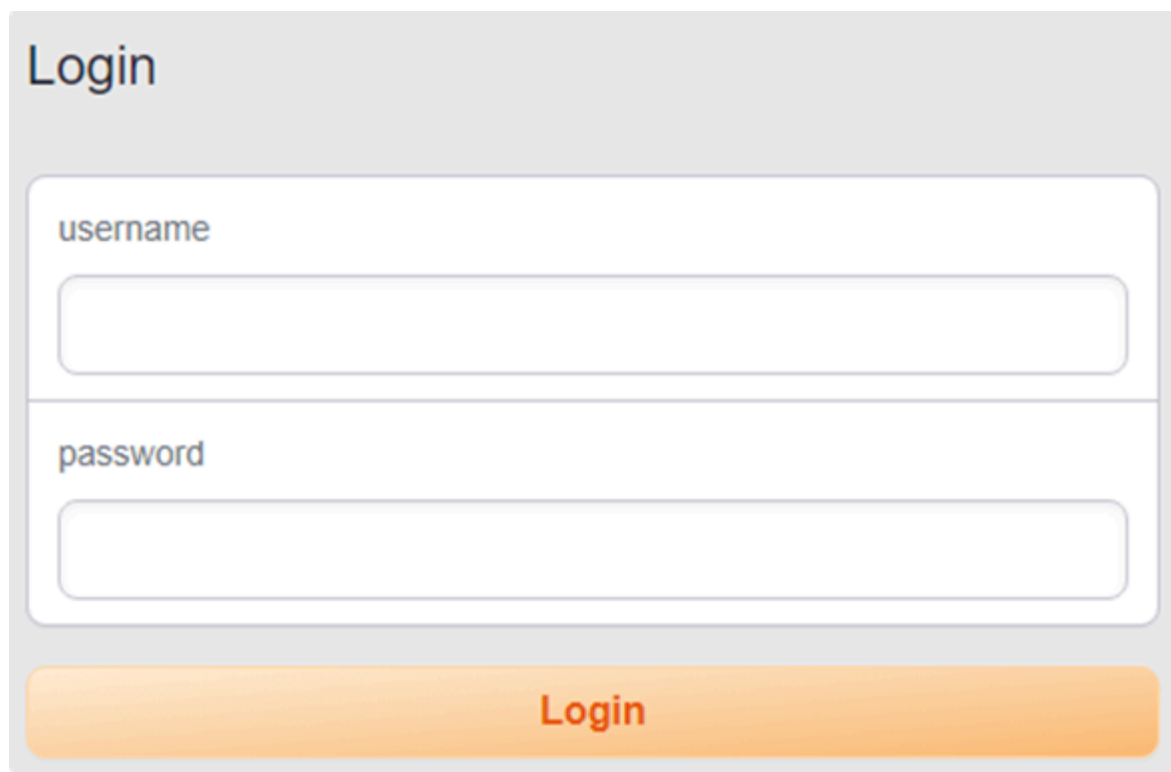
The screenshot displays the 'Elastic Cloud Server' (弹性云主机) details page. At the top, there are tabs for 'Elastic Network Card' (弹性网卡), 'Cloud Disk' (云硬盘), 'Security Group' (安全组), 'Elastic IP' (弹性IP), and 'Monitoring' (监控). The 'Elastic IP' tab is selected. Below the tabs, there is a text block: '您还可以绑定1个弹性IP（通过网卡及虚拟IP绑定），实例规格支持绑定的弹性IP数量请查看 [使用限制](#)。旧子网升级IPv6后，子网下的云主机需重启后才能支持IPv6，重启方法可参考 [重启实例](#)。' To the right of this text are buttons for 'Bind Elastic IP' (绑定弹性IP) and 'View Elastic IP' (查看弹性IP). Below the text is a table with the following columns: 'Elastic IP' (弹性IP), 'Bandwidth Size (Mbps)' (带宽大小 (Mbps)), 'Bound Internal IP' (已绑定内网IP), 'Creation Time' (创建时间), and 'Action' (操作). The table contains one row with a single Elastic IP address, a bandwidth of 5M, and a 'Release' (释放) button in the action column.

弹性IP	带宽大小 (Mbps)	已绑定内网IP	创建时间	操作
[Redacted]	5M	[Redacted]	[Redacted]	释放

最佳实践

2. 登录文本对话大模型（LLaMA2/LLaMA3）

将刚刚复制到的弹性IP替换到 `http://弹性IP:7861/` 中，并在浏览器中输入该地址跳转至登录页面。



The image shows a login form with a light gray background. At the top left, the word "Login" is written in a dark blue font. Below this, there are two input fields: the first is labeled "username" and the second is labeled "password". Both fields are empty and have a light gray border. At the bottom of the form, there is a large orange button with the word "Login" written in white text.

您可以任选以下两个账号其一进行登录：

注意

建议您登录后尽快修改密码，修改密码方法在文档“[修改大模型学习机登录密码](#)”可查看。

账号一：

- username: user
- password: **\$I\$CFLvkJ69I**

账户二：

- username: user2
- password: C700hgW5fxkN

3. 登录AI图片生成大模型（StableDiffusion）

将刚刚复制到的弹性IP替换到http://弹性IP:7860/ 中并在浏览器中输入跳转至登录页面。

您可以任选以下两个账号其一进行登录：

注意

建议您登录后尽快修改密码，修改密码方法在文档“[修改大模型学习机登录密码](#)”可查看。

账户一：

- username: user
- password: `ICFLvkJ69I`

账户二：

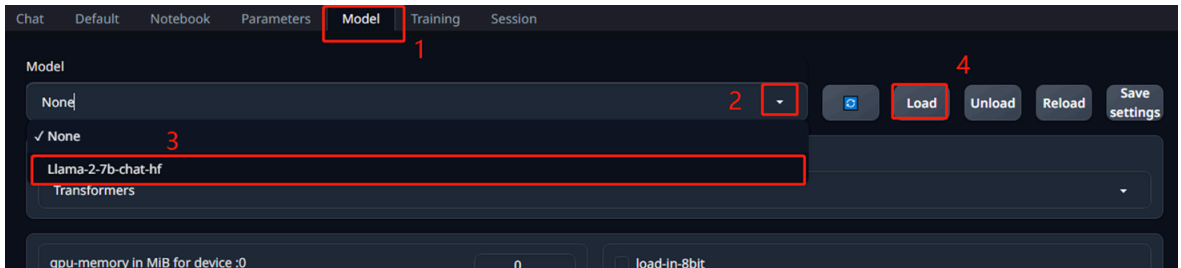
- username: user2
- password: `C700hgW5fxkN`

最佳实践

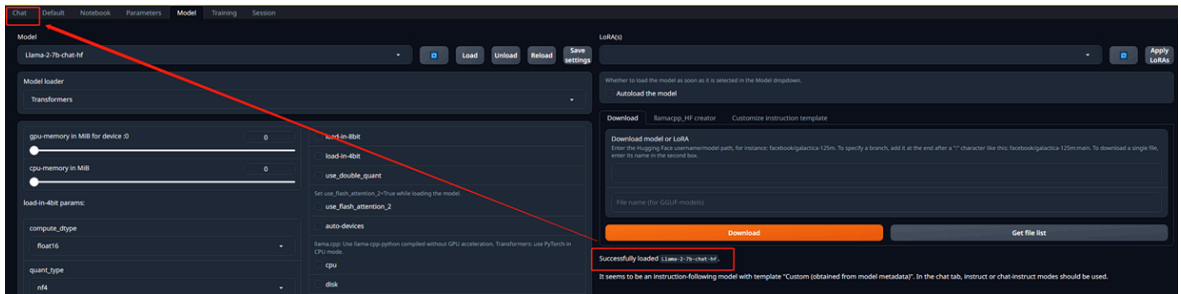
使用大模型进行推理计算

1. 文本对话大模型（LLaMA2/LLaMA3）

首先需要在Model标签页点击LLaMA2/LLaMA3大模型，然后点击Load按钮进行加载。



稍等一会，提示“Successfully loaded”，即可点击左上角的“chat”按钮，体验使用文本对话大模型（LLaMA2/LLaMA3）啦！

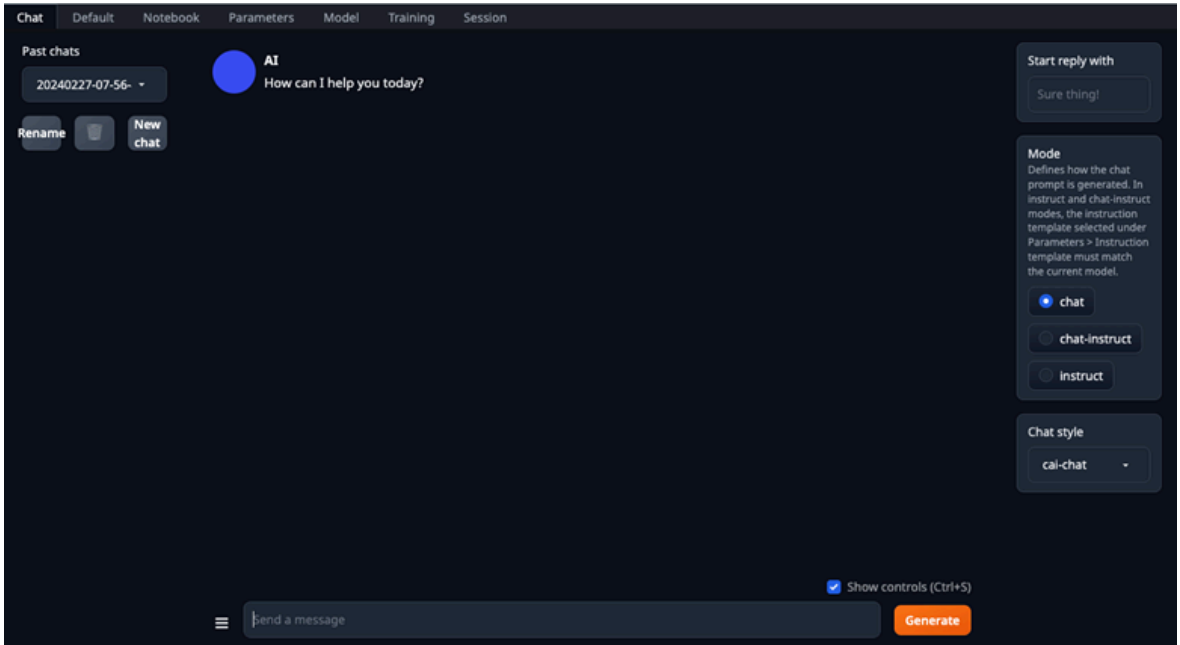


注意

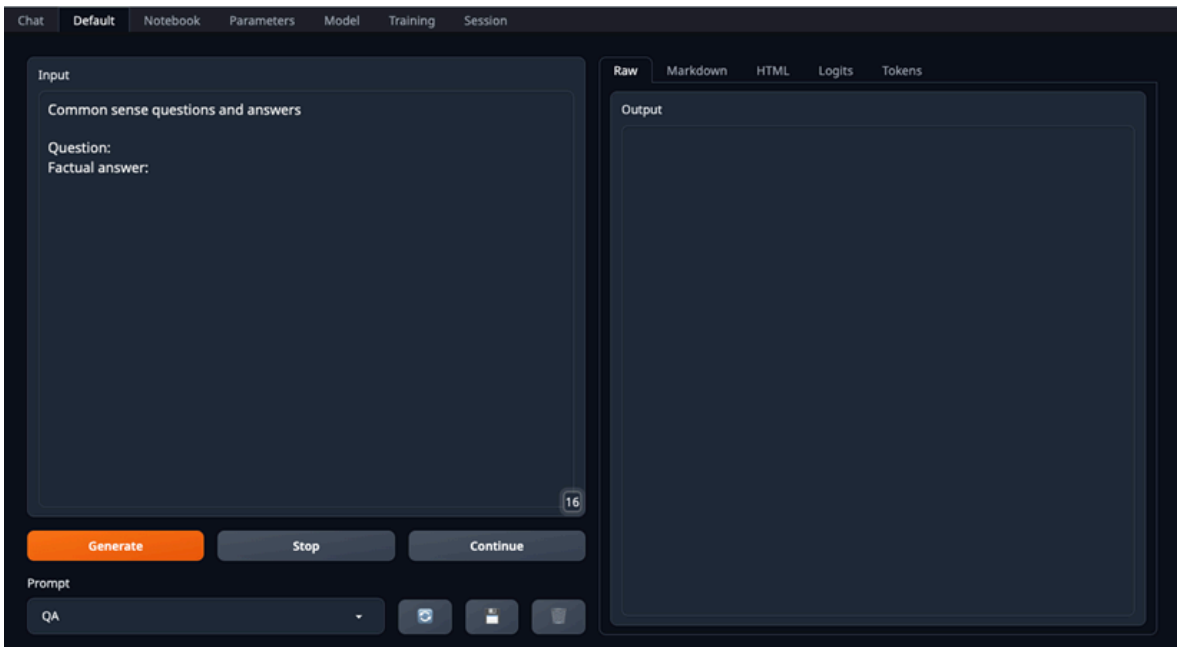
在加载Llama 2或者LLaMA3模型时，Pi2和s7.2xlarge.4规格使用默认配置会因为显存/内存不足加载失败，需要通过勾选load-in-8bit选项降低模型精度或关闭StableDiffusion服务后再加载Llama 2/LLaMA3模型。参考文档：[大模型学习机服务启停](#)。

使用文本多轮对话页面：

最佳实践



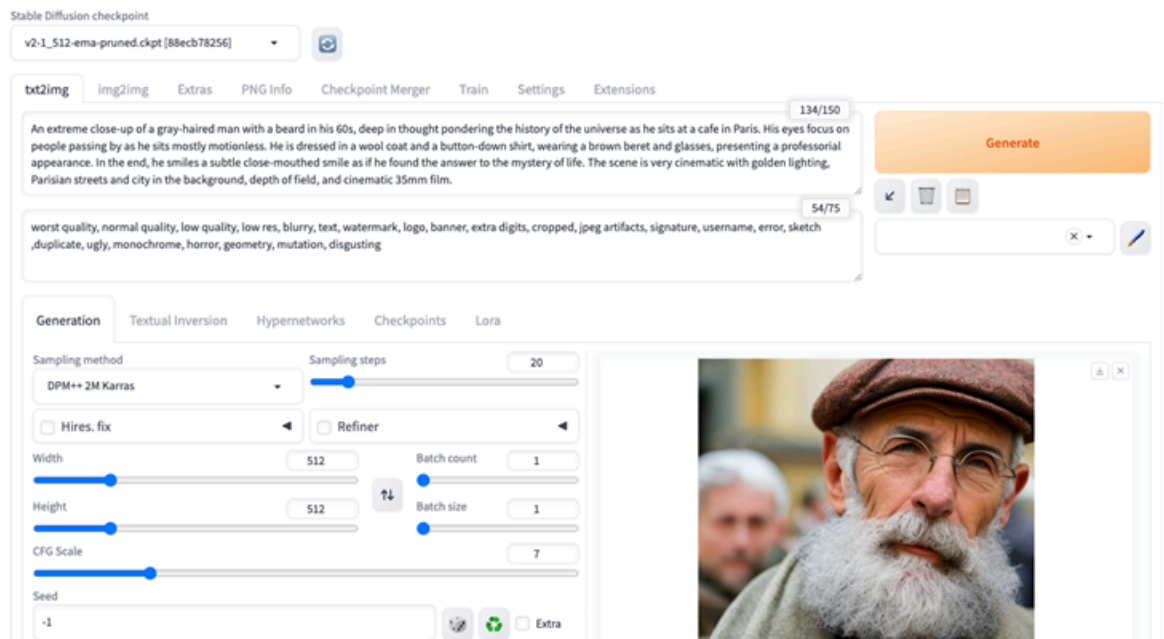
文章生成页面:



最佳实践

2. AI图片生成大模型（Stable Diffusion）：

AI图片生成大模型登录后可以直接使用，以下为图片生成页面。



已有天翼云云主机，部署与登录LLaMA 3/LLaMA 2大模型学习机

操作前提

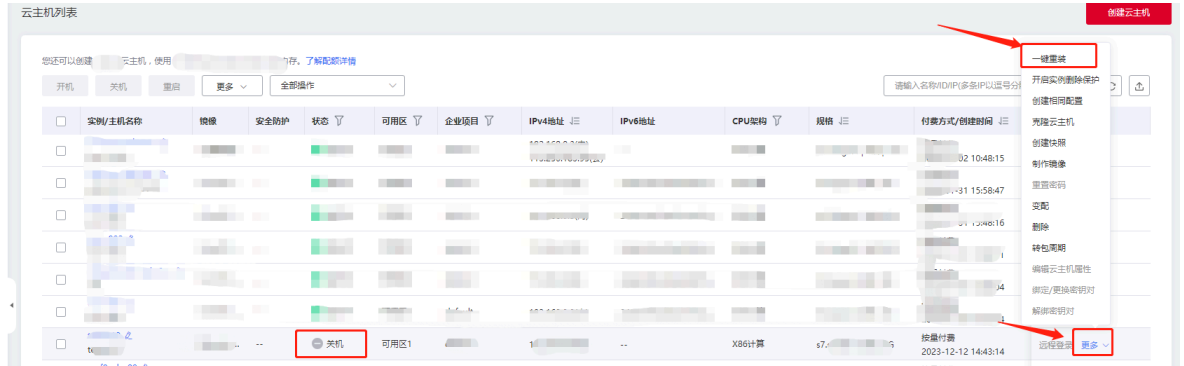
- 如果您尚未购买天翼云云主机，推荐您使用活动页上更简单快捷的一键部署方案。
- 如果您已购买的云主机位于【重庆-重庆2、宁夏-中卫5】资源池（其余资源池镜像正在加载中），且满足以下条件，请参考本文部署大模型学习机。
 - 配置：内存32G，显存16G以上
 - 已开通公网IP
 - 云主机未到期

最佳实践

操作步骤

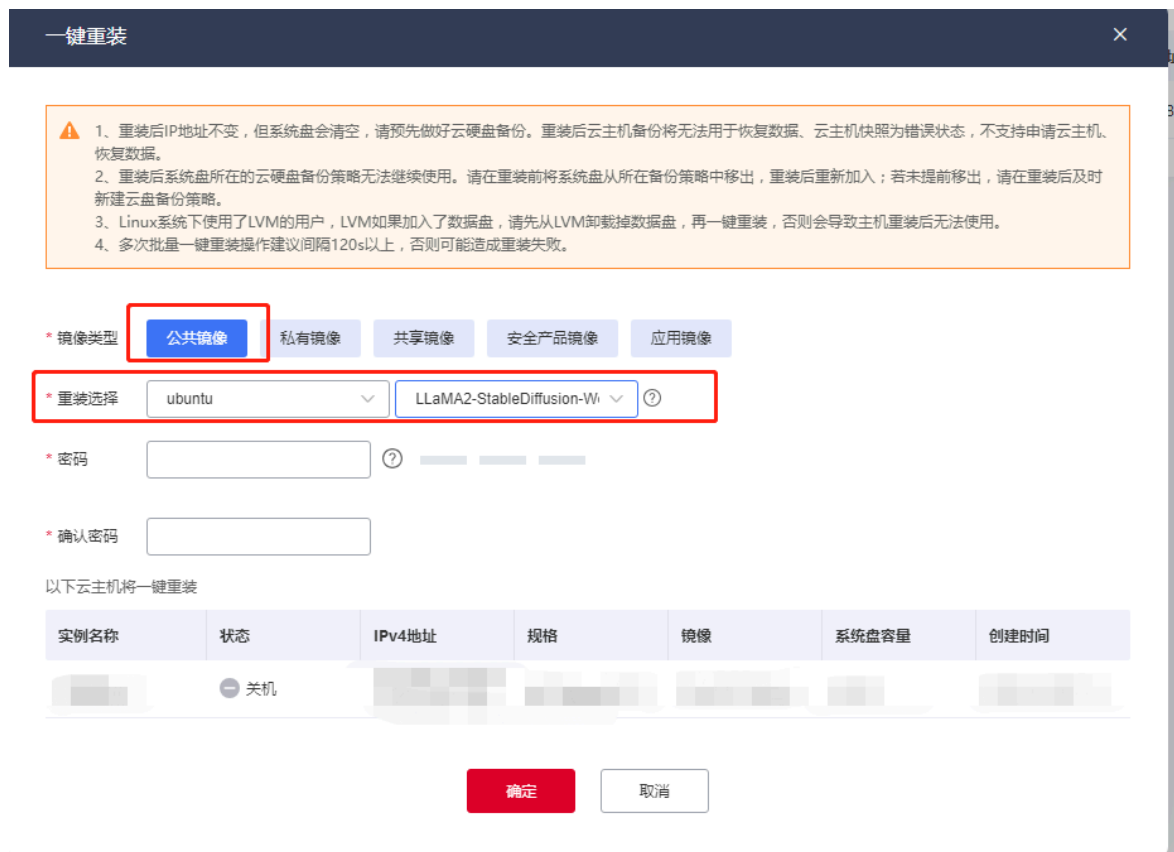
重装操作系统

1. 登录云主机控制台，前往您已购买的云主机所在的资源池。确认云主机状态为“已关机”，则可以开始重装操作系统操作。点击“更多”按钮，点击下拉菜单中“一键重装”按钮。



最佳实践

- 在重装操作系统弹窗中，镜像类型选择“公共镜像”，并在重装选择下拉菜单中选择“ubuntu-LLaMA2-StableDiffusion-WebUI-GPU（预装大模型环境）（60GB）”或ubuntu-LLaMA3-StableDiffusion-WebUI-GPU（预装大模型环境）（60GB）。



设置并确认密码，点击“确定”按钮，完成重装操作系统操作。

- 返回云主机控制台，选中已完成重装操作系统的云主机，点击“开机”按钮。确认云主机状态为“运行中”，则可以进行后续操作。

添加安全组规则

1. 点击云主机名称进入云主机详情页。

<input type="checkbox"/>	实例/主机名称	镜像	安全防护	状态	企业项目	IPv4地址	IPv6地址	规格	付费方式/创建时间	操作
<input type="checkbox"/>	ecm-c44a	LLaMA2...	未防护	运行中	default	[]	[]	pi7.4xlarge.4 16核 ... GPU: NVIDIA A10*1 ...	[]	远程登录 更多

最佳实践

2. 选择云主机详情页下方的安全组页签。

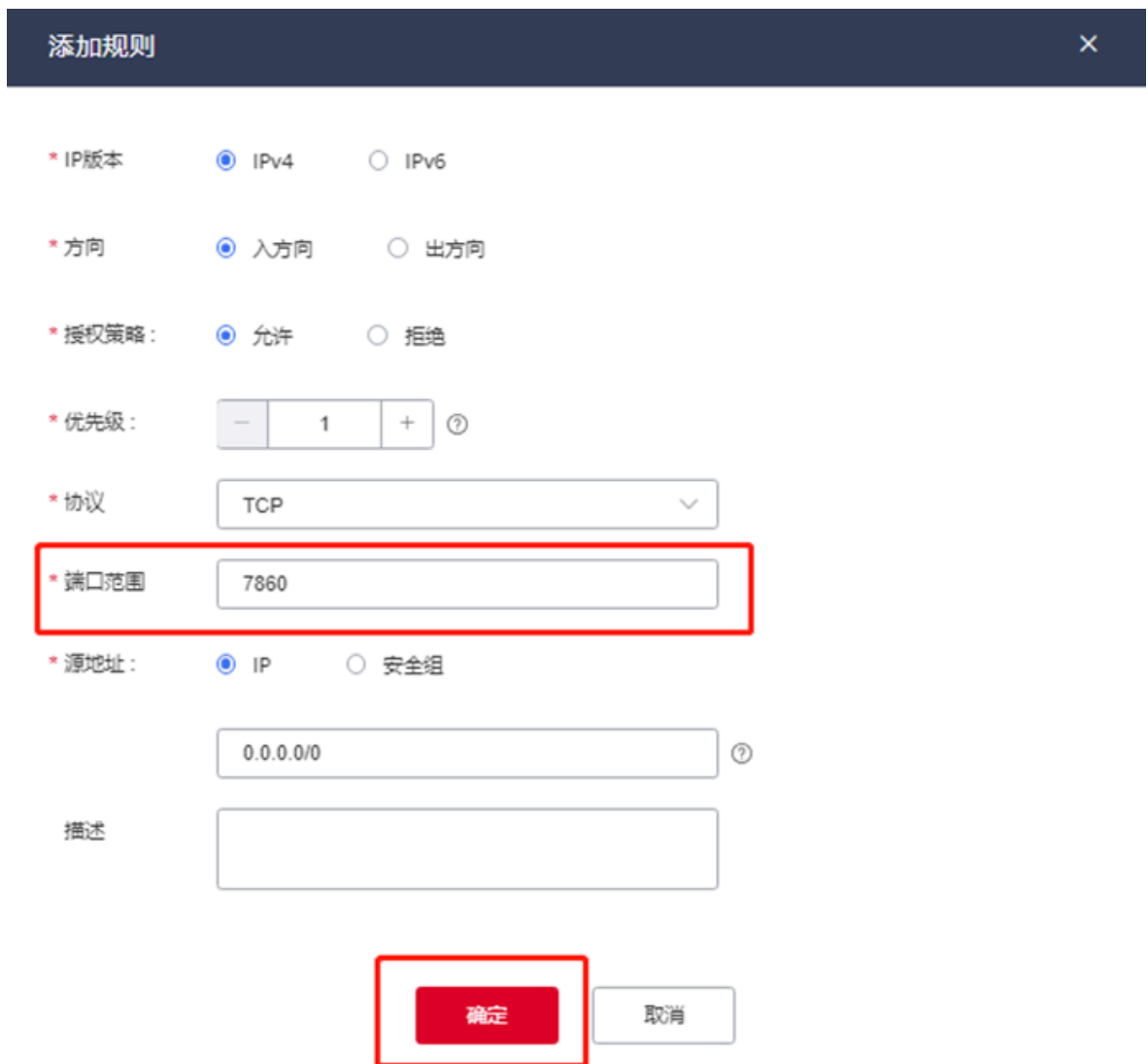


3. 点击默认安全组展开详情按钮。



最佳实践

4. 点击“添加规则”，在端口范围处填写7860，点击确定。



最佳实践

5. 再次点击“添加规则”，在端口范围处填写7861，点击确定。

添加规则 ×

* IP版本 IPv4 IPv6

* 方向 入方向 出方向

* 授权策略: 允许 拒绝

* 优先级: ?

* 协议 ▼

* 端口范围

* 源地址: IP 安全组

?

描述

最佳实践

登录大模型

1. 登录前准备：选择云主机详情的“弹性IP”页签，复制列表中的弹性IP。

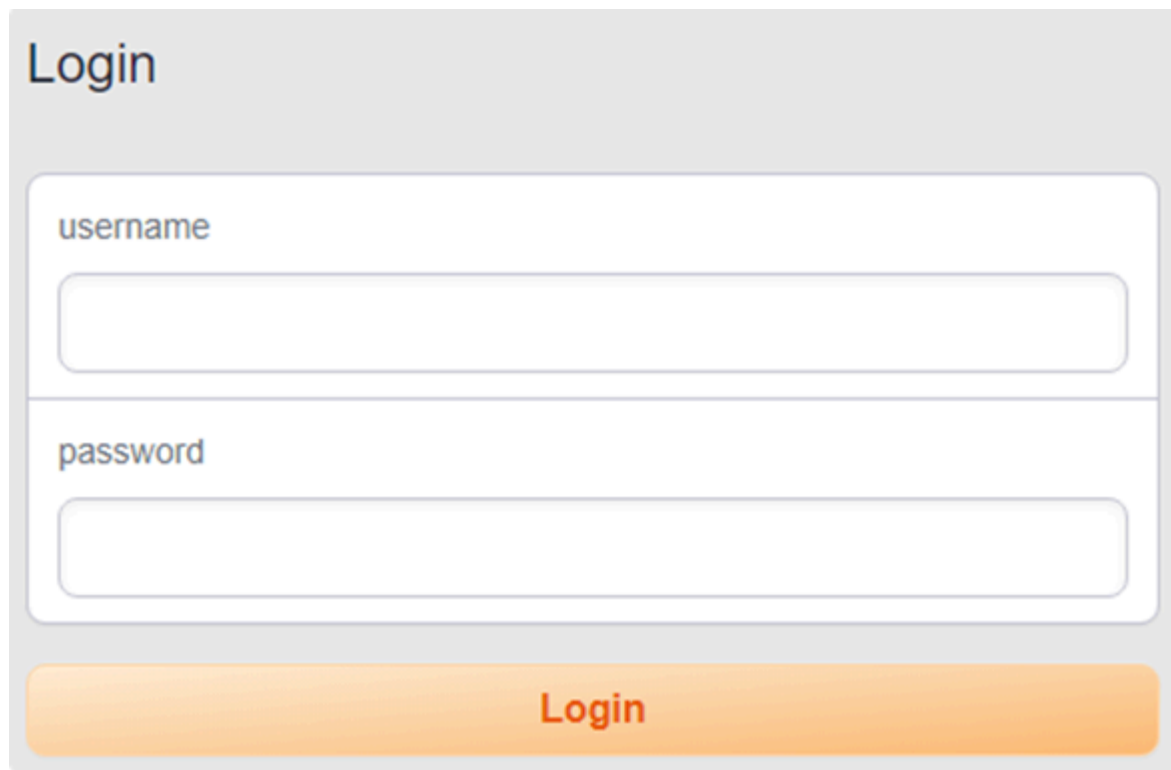
The screenshot displays the 'Elastic Cloud Host' (弹性云主机) management interface. The top section shows the instance 'ecm-e976' in a 'Running' (运行中) state. Key details include: ID, Untrusted Instance (可信实例) status (未开启), Description (描述), Creation Time (创建时间), Instance Type (规格) 4G, VPC, Instance Deletion Protection (实例删除保护) (已关闭), Password (密码), OS (操作系统), Disk (磁盘) 1 piece, Cloud Host Group (云主机组), and Delegation Strategy (委托策略). The 'Elastic IP' (弹性IP) tab is selected, showing a table with one entry: Elastic IP 14.2.231, Bandwidth (5M), and a 'Bind' (绑定) action button. A 'View Elastic IP' (查看弹性IP) button is also present.

弹性IP	带宽大小 (Mbps)	已绑定内网IP	创建时间	操作
14.2.231	5M			绑定

最佳实践

2. 登录文本对话大模型（LLaMA2/LLaMA3）

将刚刚复制到的弹性IP替换到 `http://弹性IP:7861/` 中，并在浏览器中输入该地址跳转至登录页面。



The image shows a login form with a light gray background. At the top left, the word "Login" is written in a dark blue font. Below it, there are two input fields: the first is labeled "username" and the second is labeled "password". Both fields are empty and have a light gray border. At the bottom of the form, there is a large orange button with the word "Login" written in white text.

您可以任选以下两个账号其一进行登录：

注意

建议您登录后尽快修改密码，修改密码方法在文档“[修改大模型学习机登录密码](#)”可查看。

账号一：

username: user

password: **\$I\$CFLvkJ69I**

账户二：

username: user2

password: C700hgW5fxkN

3. 登录AI图片生成大模型（StableDiffusion）

将刚刚复制到的弹性IP替换到<http://弹性IP:7860/> 中并在浏览器中输入跳转至登录页面。

您可以任选以下两个账号其一进行登录：

注意

建议您登录后尽快修改密码，修改密码方法在文档“[修改大模型学习机登录密码](#)”可查看。

账户一：

username: user

password: **\$I\$CFLvkJ69I**

账户二：

username: user2

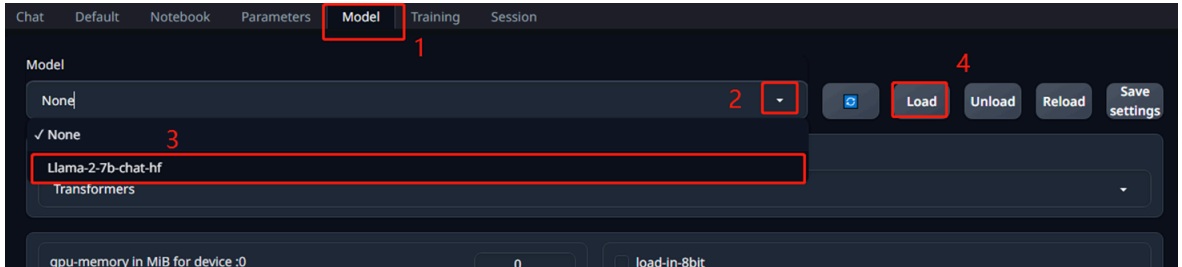
password: C700hgW5fxkN

最佳实践

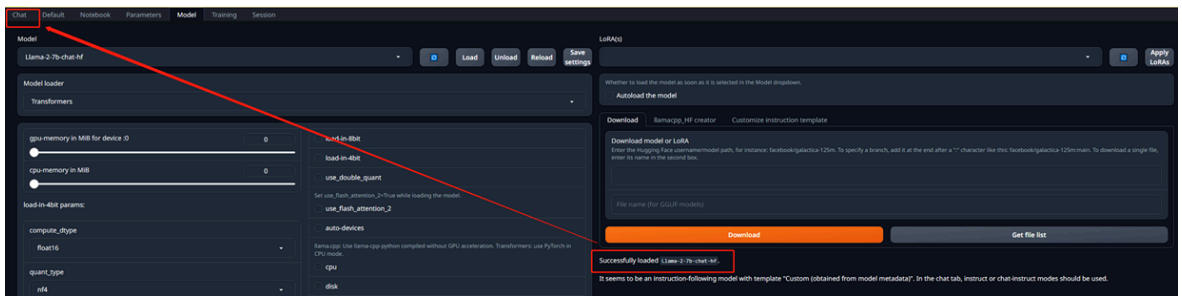
使用大模型进行推理计算

1. 文本对话大模型（LLaMA2/LLaMA3）

首先需要在Model标签页点击LLaMA2/LLaMA3大模型，然后点击Load按钮进行加载。



稍等一会，提示“Successfully loaded”，即可点击左上角的“chat”按钮，体验使用文本对话大模型（LLaMA2/LLaMA3）啦！

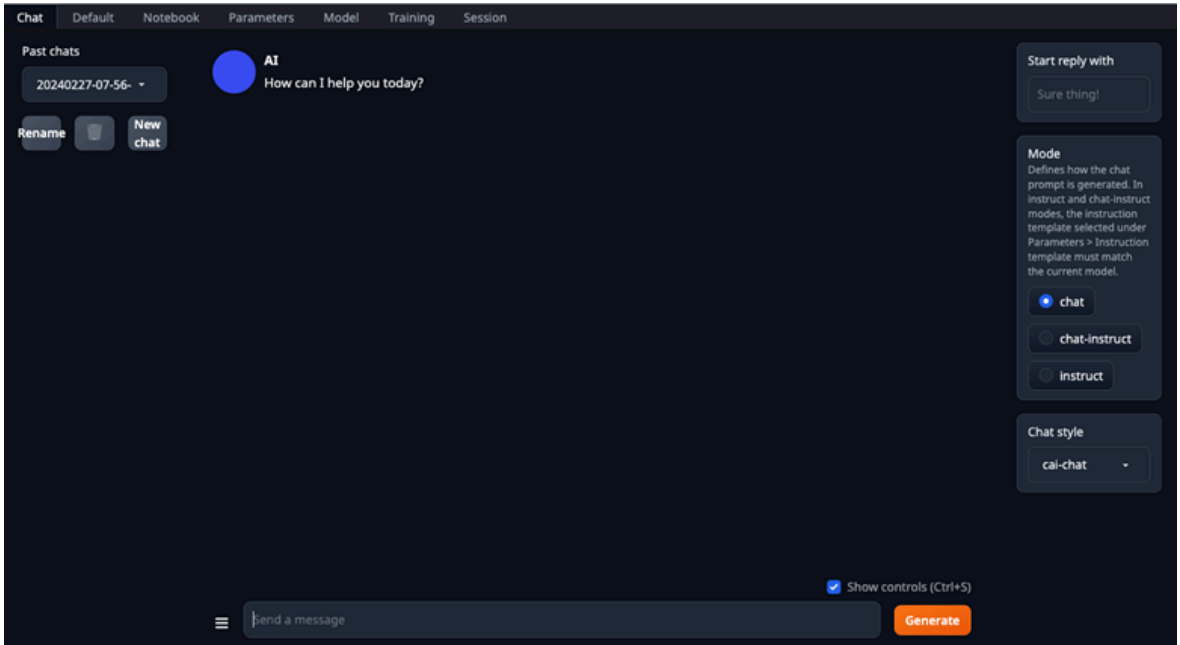


注意

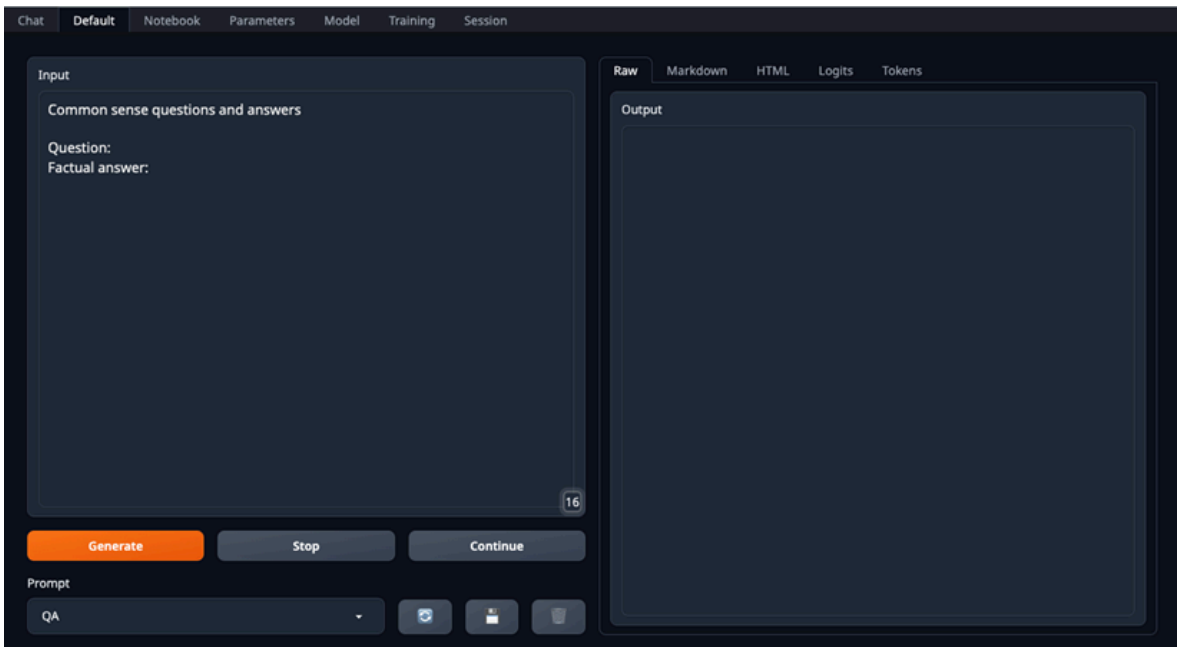
在加载时参数默认以fp16精度进行加载，Pi2和s7.2xlarge.4规格需要勾选load in 8 bit，否则会因显存/内存不足而加载失败。

使用文本多轮对话页面：

最佳实践



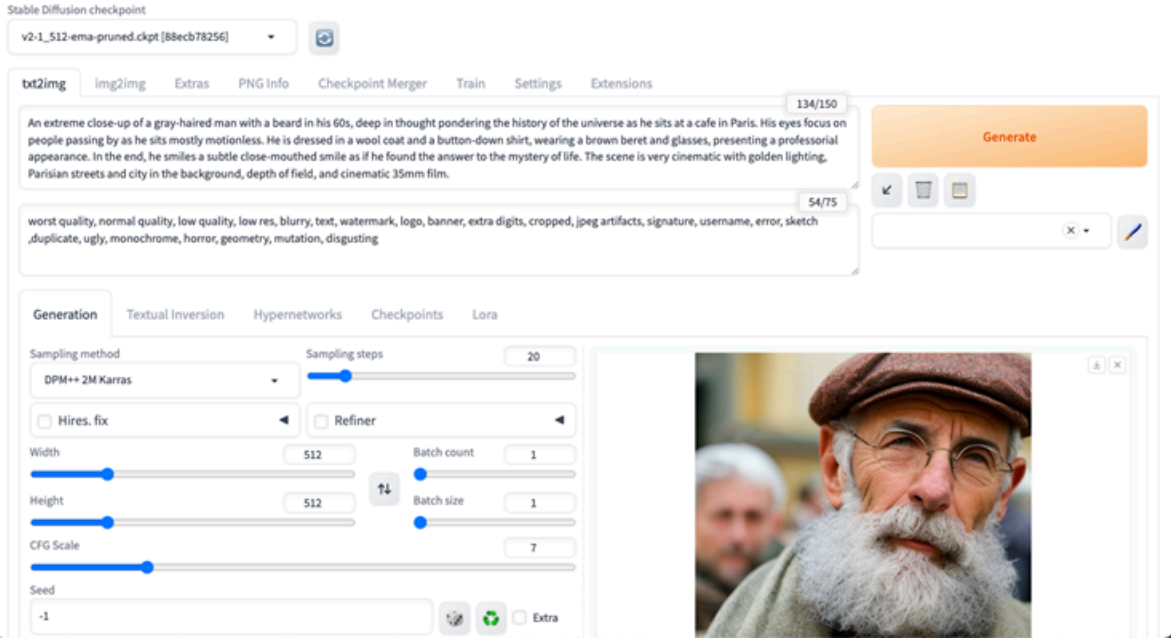
文章生成页面:



最佳实践

2. AI图片生成大模型（Stable Diffusion）：

AI图片生成大模型登录后可以直接使用，以下为图片生成页面。

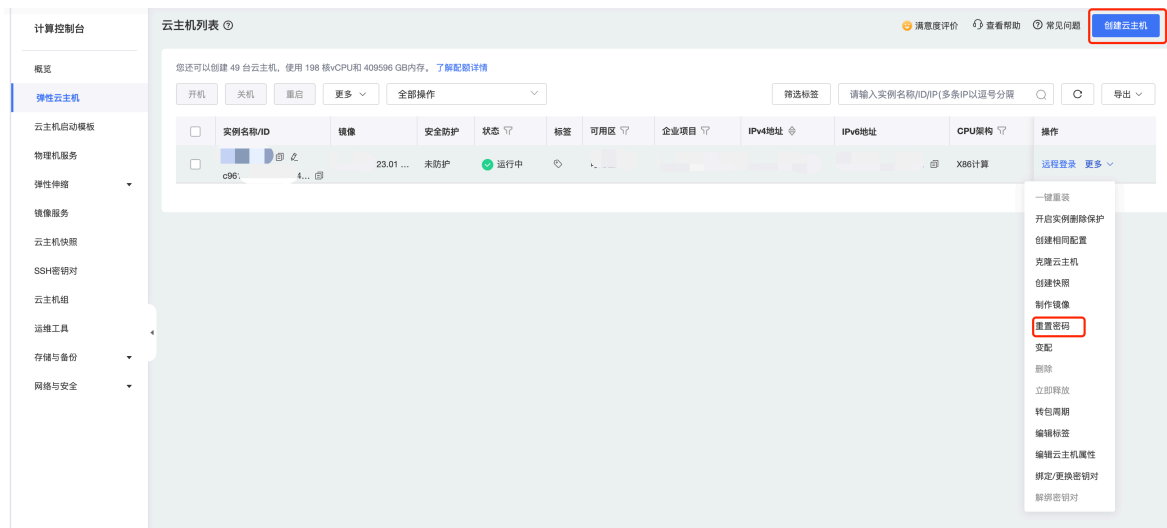


修改大模型学习机登录密码

本文向您介绍如何修改大模型学习机的登录密码，以下为修改大模型学习机的登录密码的文字教程。

最佳实践

1. 在云主机控制台，选择您购买云主机的地域，点击云主机的操作-更多-重置密码。



2. 点击“远程登录”按钮。



最佳实践

3. 进入虚机内部，输入 root 回车，在password处输入刚刚修改的密码（密码不可见，请输入完直接回车）。

```
IPv6 address for eth0:    240e:981:b000:1a::309:349

* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
  just raised the bar for easy, resilient and secure K8s cluster deployment.

  https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

154 updates can be applied immediately.
112 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

4 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

The list of available updates is more than a week old.
To check for new updates run: sudo apt update
New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Thu Mar  7 13:59:52 CST 2024 on tty1
root@ecm-4549:~# _
```

最佳实践

4. 点击右上角的“粘贴输入”，在输入框中粘贴以下指令，点击“确定发送”，点击虚拟机屏幕后回车发送指令。

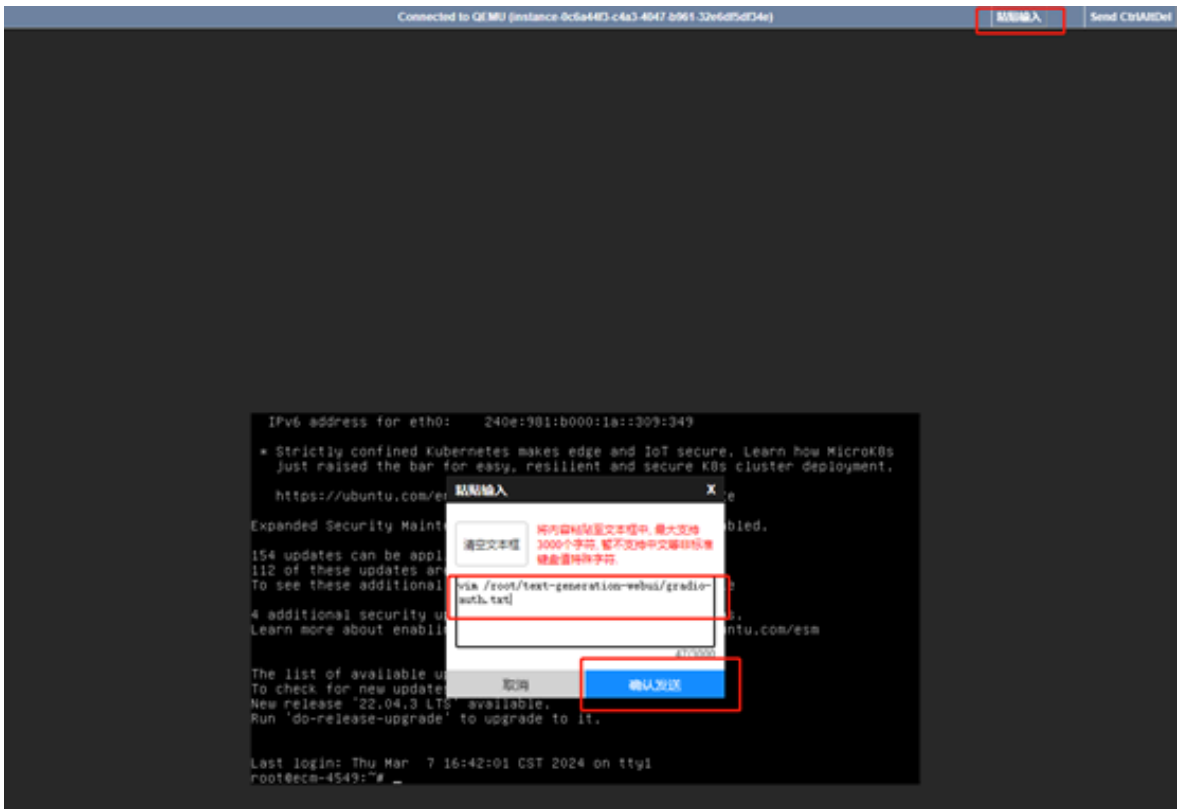
修改文本大模型密码指令：

```
vim /root/text-generation-webui/gradio-auth.txt
```

或

修改图像生成大模型密码指令：

```
vim /root/stable-diffusion-webui/gradio-auth.txt
```



进入修改对应大模型登录密码界面。

大模型学习机服务启停最佳实践

前提条件

学习机内通过docker容器的方式部署了text-generation-webui与stable-diffusion-webui两套框架的运行环境。可以通过以下命令进行查看:

```
docker ps -a
```

```
root@ecm-4549:~# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS        NAMES
dfc4c8ab835a   stable_diffusion:gpu  "/bin/bash"            16 minutes ago  Up 4 minutes  stable_diffusion
61de5996b808   llama:gpu      "/bin/bash"            16 minutes ago  Up 4 minutes  llama
```

网页服务的进程在容器内启动, 框架代码与模型放置在本机的/root目录下, 通过挂载的方式传入到容器中。

您无需进入容器执行操作, 只需要直接操作/root/text-generation-webui和/root/stable-diffusion-webui下的文件即可将改动同步到容器内部。

停止服务

直接停止 Docker 容器, 即可停止对应的服务, 可选择执行以下命令:

```
docker stop llama
```

```
docker stop stable_diffusion
```

```
root@ecm-4549:~# docker stop llama
llama
```

重启服务

通过执行预置的重启脚本启动容器与服务, 可选择执行以下命令:

```
sh /root/restart_llama_webui.sh
```

```
sh /root/restart_sd_webui.sh
```

```
root@ecm-4549:~# sh /root/restart_llama_webui.sh
container_status:exited
llama exited, start it
llama
```

也可以只重启容器, 不启动服务, 随后手动进入容器启动服务。

此方式可以用来查看框架的具体执行日志, 以便于追查问题等。

```
sh /root/docker_run_llama.sh
```

```
sh /root/docker_run_sd.sh
```

```
root@ecm-4549:~# sh docker_run_llama.sh
container_status:exited
llama exited, start it
llama
```


最佳实践

1. 基础模型加载

页面最上方左侧下拉菜单展开后会显示所有存放于云主机/root/stable-diffusion-webui/models/Stable-diffusion目录下的模型。学习机初始预装了原版v2-1_512-ema-pruned.ckpt,您也可随时下载其他版本的StableDiffusion放入目录下进行加载。v2-1版模型占用约3G显存。

```
root@ecm-4549:~/stable-diffusion-webui/models/Stable-diffusion# ll
total 5092656
drwxr-xr-x  2 root root    4096 Feb 27 13:30 ./
drwxr-xr-x 14 root root    4096 Feb 27 14:13 ../
-rw-r--r--  1 root root      0 Feb 27 11:45 'Put Stable Diffusion checkpoints here.txt'
-rw-r--r--  1 root root 5214865159 Dec  7 2022 v2-1_512-ema-pruned.ckpt
```



2. LoRA模型加载

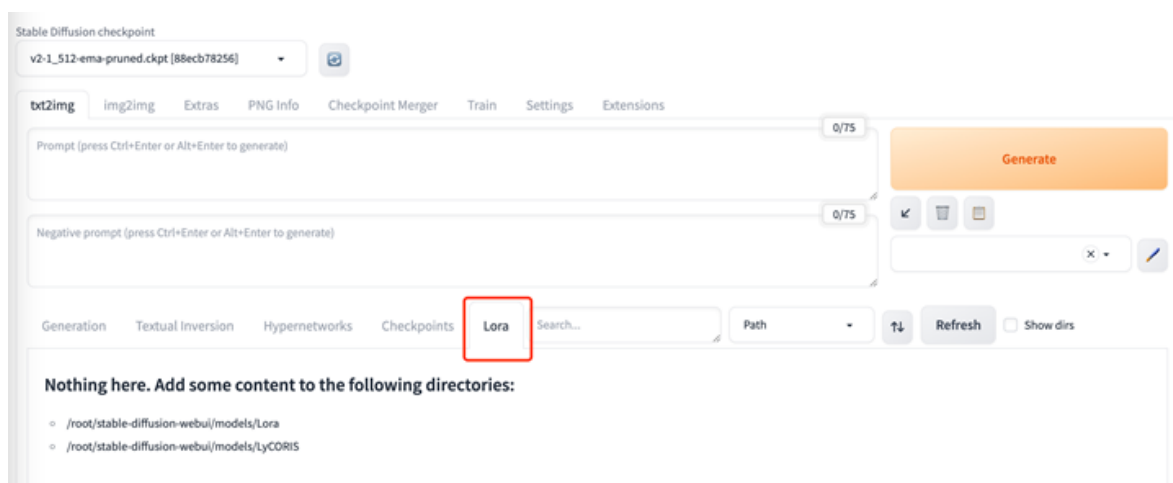
由于基础大模型参数量极大,训练需要的资源量极大,因此基于基础大模型进行微调的小模型应运而生。LoRA技术便是其中最主流的一种。

LoRA小模型无法独立使用,需要在加载完基础大模型后再进行加载,对基础大模型的能力进行扩展。

学习机内没有预装LoRA小模型,您可以在网上下载其他基于StableDiffusion的LoRA模型并放到学习机的/root/stable-diffusion-webui/models/Lora目录下。

StableDiffusion模型的LoRA模型以提示词(prompt)的形式进行使用,您可以在下载LoRA模型时进行具体使用方法查阅。

```
root@ecm-4549:~/text-generation-webui/loras# ll
total 8
drwxr-xr-x  2 root root 4096 Feb 27 11:45 ./
drwxr-xr-x 22 root root 4096 Feb 29 14:27 ../
-rw-r--r--  1 root root   0 Feb 27 11:45 place-your-loras-here.txt
```

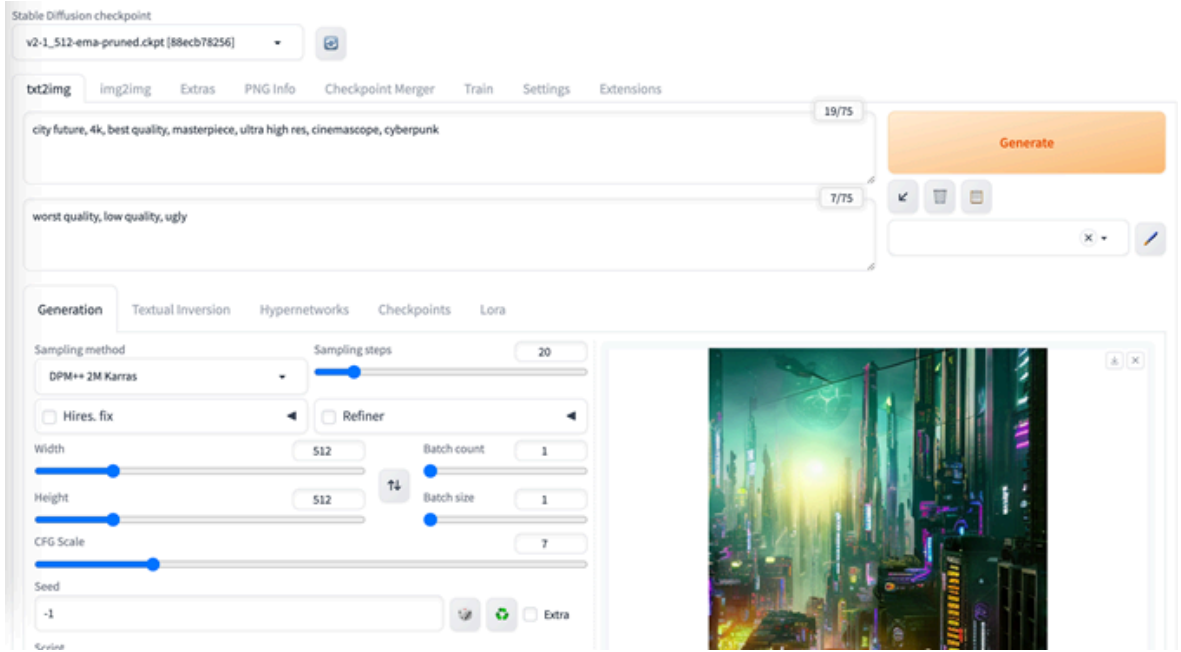


二、StableDiffusion模型使用

1. 文本生成图像模式

在txt2img标签页分别输入提示词(prompt)和负向提示词(prompt), 随后点击Generate按钮。

StableDiffusion的提示词写法要求较高, 这里无法做详细介绍. 您可搜索相关资料进行学习。

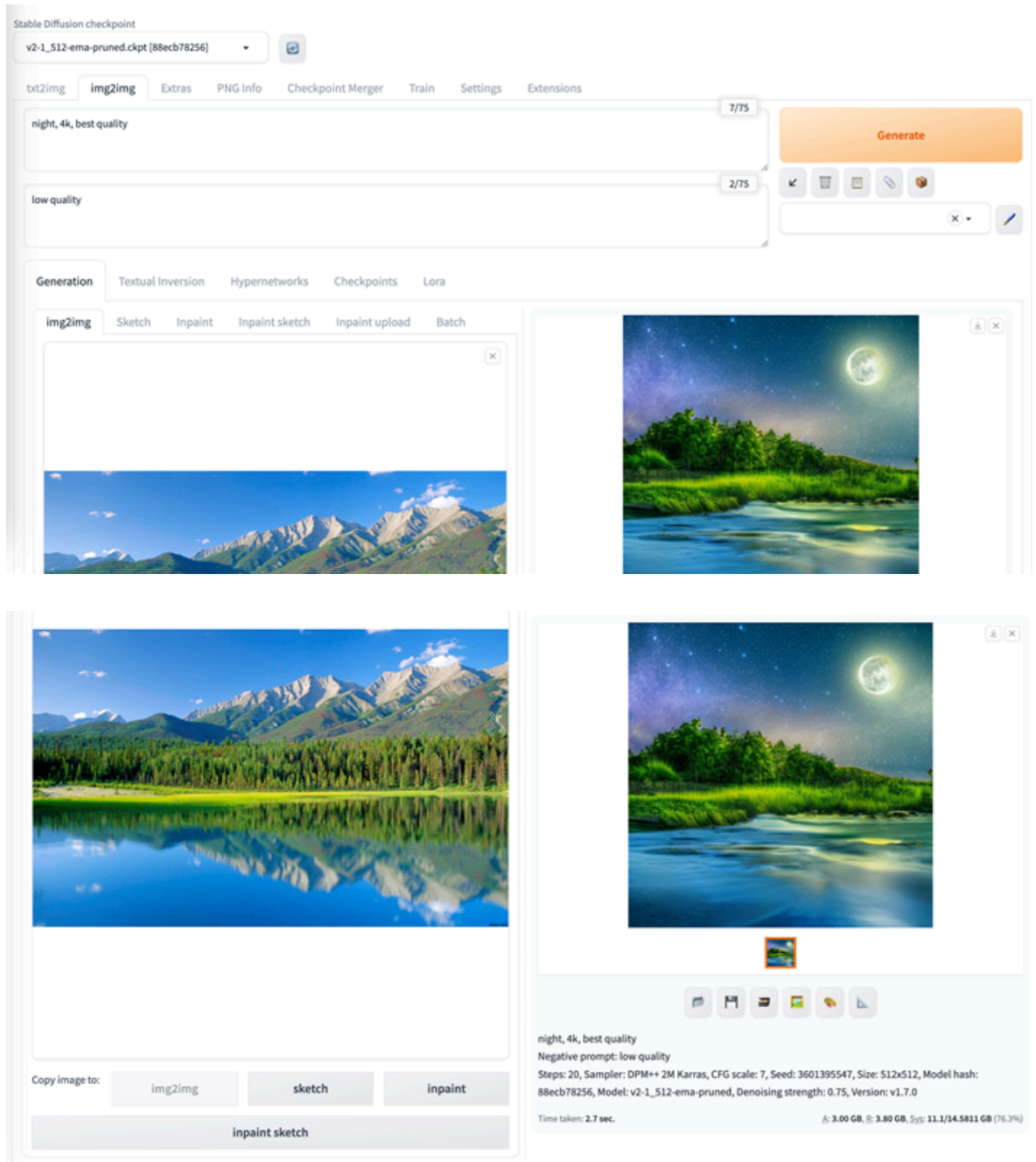


最佳实践

2. 图像生成图像模式

在img2img标签页分别输入提示词(prompt)和负向提示词(negative prompt), 同时在左下方选择上传一张图片, 最后点击Generate按钮。

StableDiffusion将给予给定图片和提示词, 创作一张新的图片。



3. 生成质量控制

StableDiffusion模型生成图片质量可以在页面下方Generation标签页进行调整,但需要您具有一定的diffusion模型(扩散模型)背景知识。这里仅对其中部分基础参数进行介绍:

- Sampling Steps: 采样步数. 数值越大图像越精细。
- Width & Height: 图像的宽度与高度。
- Batch count: 生成图像总批次。
- Batch size: 每一批次生成图像的个数。
- CFG Scale: 控制生成图像与提示词的关联度, 数值越大关联度越高, 但图片质量可能越差。
- Seed: 随机数种子. 控制模型生成结果的随机性, 如果输入-1则代表不固定随机数种子, 相同的提示词每次生成图片不同; 如果输入其他数字则代表固定随机数种子, 相同提示词每次生成图片相同。

大模型学习机文本生成最佳实践

前提条件

- 大模型学习机预装了Llama 2-7b-chat模型与stable-diffusion-base-v2.1模型, 以及配套的开源服务框架text-generation-webui与stable-diffusion-webui, 使您不需单独进行下载模型与配置环境。
- 云主机开通与如何登录网页页面参见[部署文档](#)。

以下将向您介绍如何进行Llama 2模型文本生成任务:

注意

框架默认角色名称为"You", 但主流大模型默认角色名称一般为"User", 可能会导致模型效果显著变差。

修改角色名称的方式有:

1. 在网页Parameters -> Chat -> User标签下修改Name字段为"User", 此方法刷新页面后就会失效。
2. 修改配置文件/root/text-generation-webui/modules/shared.py中的'name': 'User', 并重启llama服务, 此方法永久生效。

1. 大语言模型加载

说明

以下内容基于开源框架text-generation-webui。

1.1 基础模型加载

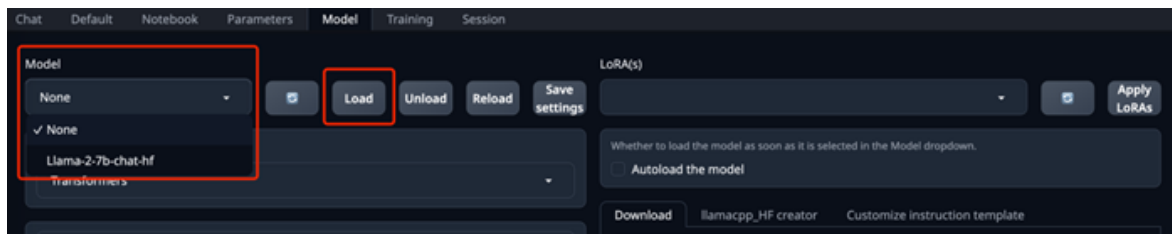
大语言模型体积较大, 需要占用较多的内存/显存资源, 因此在刚启动服务时并未进行加载, 需要手动选择模型进行加载。

在页面上切换到Model标签页, 左侧下拉菜单展开后会显示所有存放于云主机/root/text-generation-webui/models目录下的模型。学习机初始预装了Llama-2-7b-chat模型(huggingface格式), 您也可随时下载其他大模型并放入models目录下进行加载。

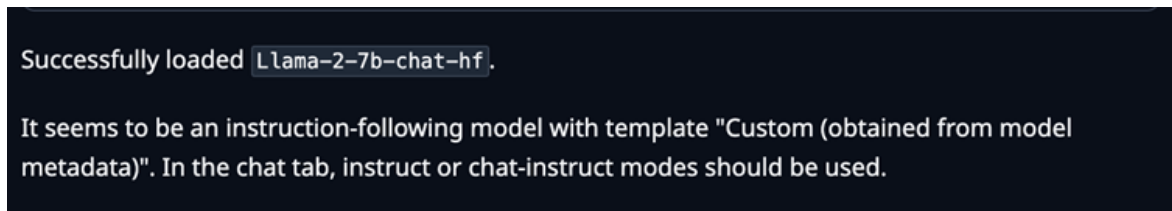
默认Model loader: Transformers能够支持huggingface的主流大模型, 在自行加载其他模型时您需要确认模型格式是否匹配。

最佳实践

```
root@ecm-4549:~/text-generation-webui/models# ll
total 20
drwxr-xr-x  3 root root 4096 Feb 27 14:24 ./
drwxr-xr-x 22 root root 4096 Feb 29 14:27 ../
drwxr-xr-x  2 root root 4096 Feb 23 15:24 Llama-2-7b-chat-hf/
-rw-r--r--  1 root root 5724 Feb 27 11:45 config.yaml
-rw-r--r--  1 root root   0 Feb 27 11:45 place-your-models-here.txt
```



右下角将出现模型加载样式，直至加载成功。耗时可能较长，期间不要退出页面。



注意

在加载预装的Llama-2-7b模型时，如果您使用的是GPU学习机，模型将默认以fp16半精度进行加载，消耗13.1G显存；如果您使用的是CPU学习机，模型将以fp32单精度进行加载，消耗26G内存。

因此，如果您的机型是内存为32G或显存为16G的型号，则会因为资源不足导致无法直接加载Llama-2-7b模型。

解决办法：

- (1) 参考[大模型学习机服务启停](#)文档说明，加载Llama-2-7b模型前先将图像生成服务关停。
- (2) GPU学习机能够支持对模型进行量化，降低模型精度的同时缩减模型大小。您可以在Model页下方找到load-in-8bit的开关并进行勾选，此时Llama-2-7b模型将消耗7G内存，可以和图像生成服务中的StableDiffusion模型共存。且8bit精度对大语言模型的生成来说影响不会特别大。

注意

由于huggingface.io网站访问受限，Model标签页的Download按钮无法直接下载模型。

最佳实践

1.2 LoRA模型加载

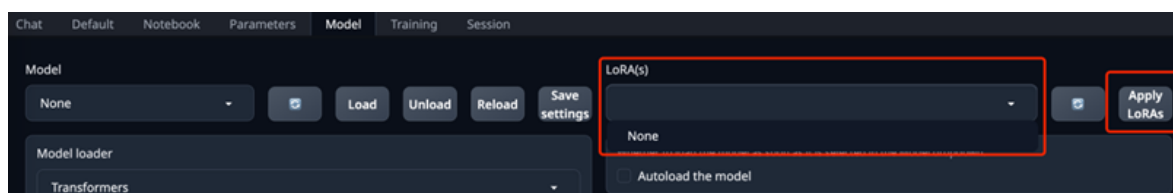
由于基础大模型参数量极多，训练需要的资源量极大，因此基于基础大模型进行微调的小模型应运而生。LoRA技术便是其中最主流的一种。

LoRA小模型无法独立使用，需要在加载完基础大模型后再进行加载，对基础大模型的能力进行扩展。

学习机内没有预装LoRA小模型，您可以在网上下载其他基于Llama-2-7b-chat的LoRA模型并放到学习机的/root/text-generation-webui/loras目录下。在Model标签页右侧选择LoRA模型并点击Apply LoRAs进行加载。

例如您可以从相关技术网站找到进行了中文微调的LoRA模型(也有直接在基础大模型上进行训练得到的新的基础大模型，注意区分)。

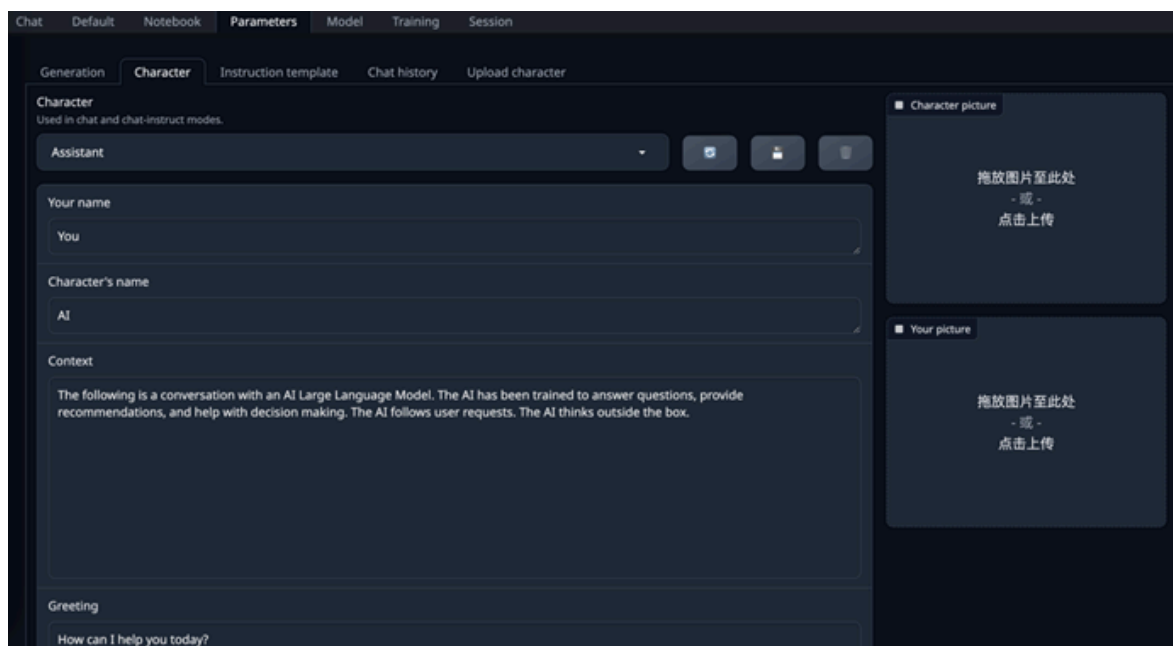
```
root@ecm-4549:~/text-generation-webui/loras# ll
total 8
drwxr-xr-x  2 root root 4096 Feb 27 11:45 ./
drwxr-xr-x 22 root root 4096 Feb 29 14:27 ../
-rw-r--r--  1 root root   0 Feb 27 11:45 place-your-loras-here.txt
```



2. 大语言模型使用

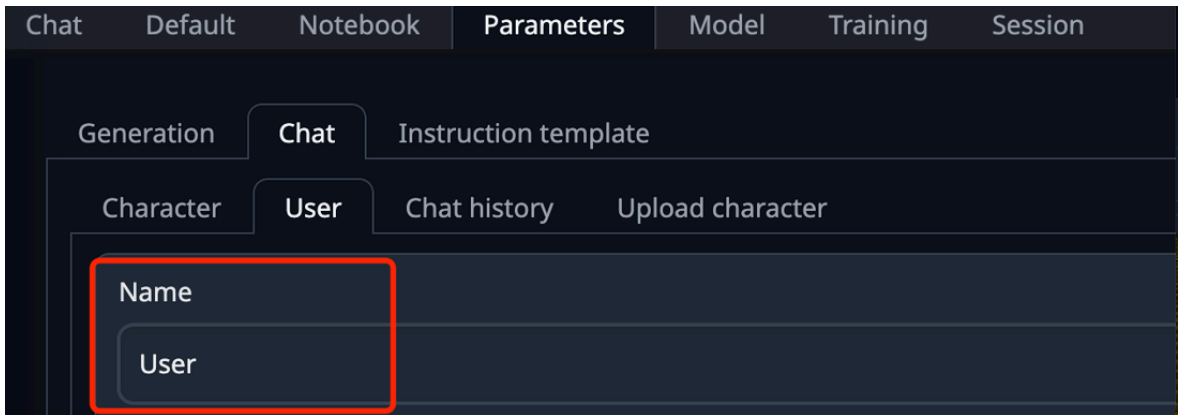
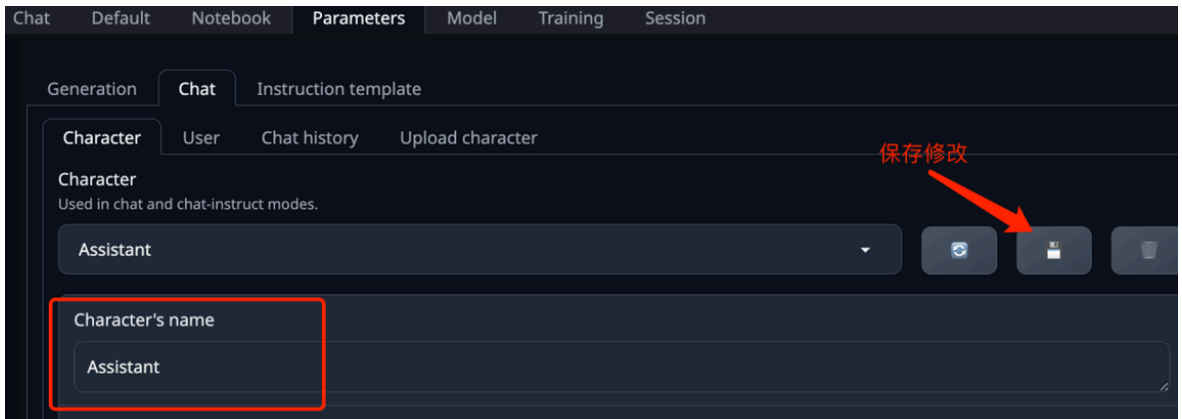
2.1 对话模式

大模型需要在一定的提示词(prompt)下进行生成。在对话模式下，框架预设了一些背景知识以控制生成方向。可在Parameters标签页的Character标签下进行查看和修改。



最佳实践

需要参考下图将Parameter页Chat标签下默认的名称修改为Character's Name: Assistant; User Name: User。否则在进行chat对话时部分模型可能出现错误。

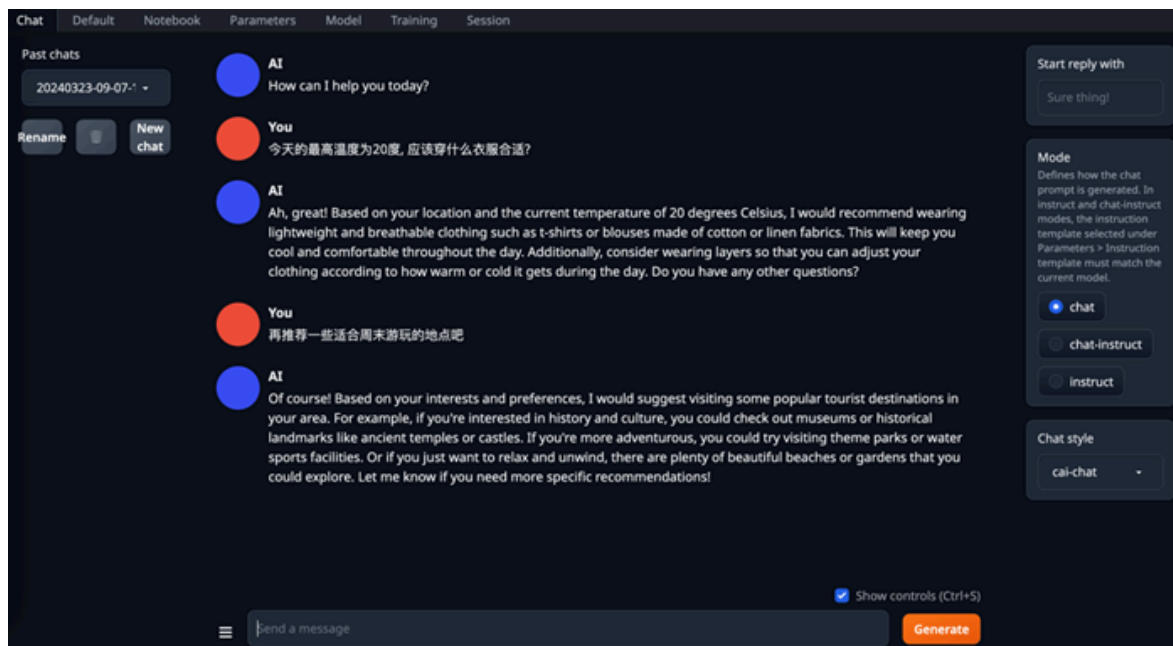


在Chat标签页我们可以和大模型展开对话。

注意

原版Llama-2-7b-chat模型对中文支持较弱，可以补充下载其他基于Llama-2-7b-chat的中文LoRA小模型，或换用其他中文的基础大模型，例如电信星辰（telechat）大模型。

最佳实践

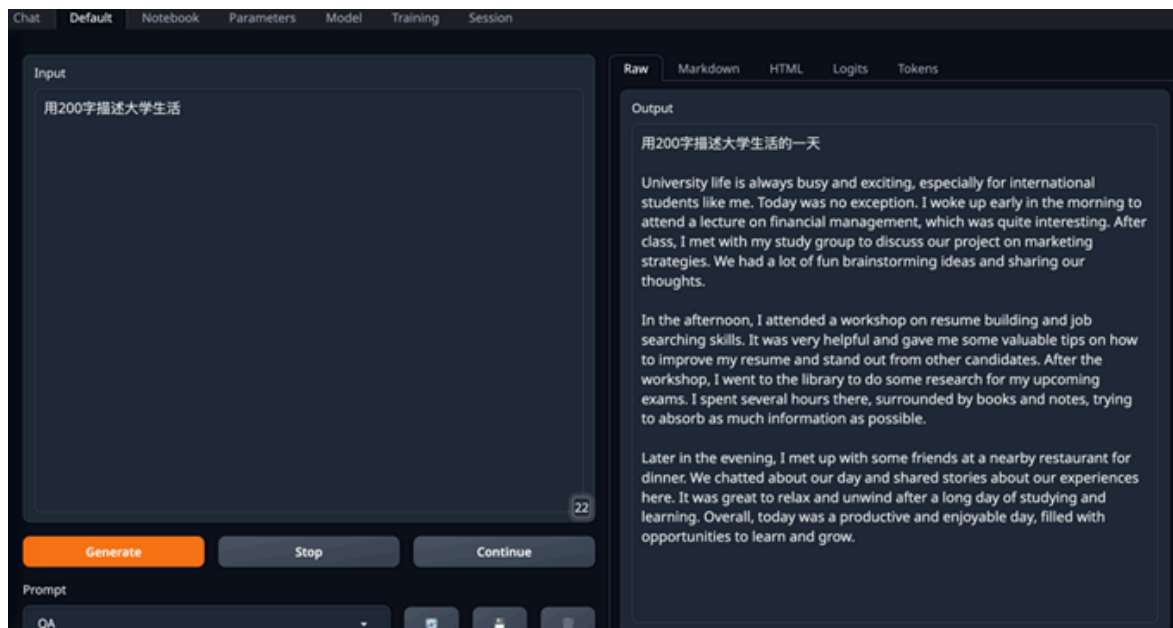


2.2 文本生成模式

在Default标签页我们可以利用大模型进行文本生成创作。

注意

原版Llama-2-7b-chat模型对中文支持较弱, 可以补充下载其他基于Llama-2-7b-chat的中文LoRA小模型, 或换用其他中文的基础大模型, 例如电信星辰 (telechat) 大模型。



2.3 生成方向控制

大语言模型生成方向可以在Parameters标签页的Generation标签下进行调整，但需要您具有一定的大语言模型背景知识。这里仅对其中部分基础参数进行介绍。

注意

大模型中token的概念：文本中最小的语义单元。例如在英文中一个简单单词(love)可能是一个token，一个复杂单词(transformer)可能是多个token的组合(trans + former)；在中文中一个字(爱)可能是一个token，多个字(爱情)也可能组成一个token。

- `max_new_tokens`: 生成token的最大数量。
- `temperature`: 在生成下一个token时，控制各个候选token被选择的概率的平滑程度。
`temperature`取值越大，每个候选token的概率越相似。
- `top_p`: 在生成下一个token时，按候选token的概率从大到小计算累计概率，达到p后丢弃后续的候选token，控制候选token的数量。
- `top_k`: 在生成下一个token时，按候选token的概率从大到小，只取前k个候选token，控制候选token的数量。
- `repetition_penalty`: 重复惩罚，控制生成结果中文字的重复次数。

大模型学习机文本生成模型微调最佳实践

说明

模型训练涉及到较多的专业知识，这里以Llama 2的LoRA微调为例给出一版示例。

1. 模型微调简介

由于基础大模型参数量极大，训练需要的资源量极大，因此基于基础大模型进行微调的小模型应运而生。LoRA技术便是其中最主流的一种。

LoRA小模型无法独立使用，需要在加载完基础大模型后再进行加载，对基础大模型的能力进行扩展。

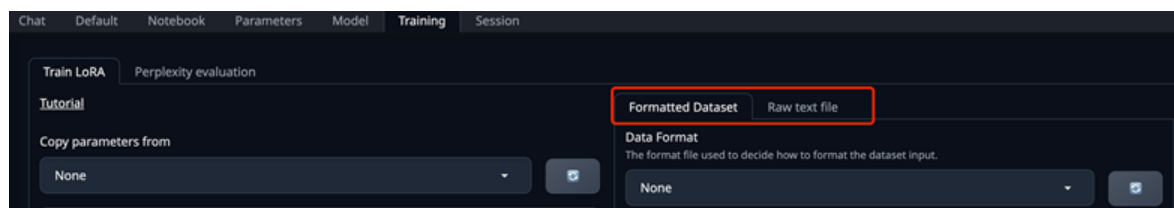
已有LoRA模型如何加载可参考[文本生成模型使用最佳实践](#)。

(1) 准备训练数据

支持多种格式，主要分为格式化数据与原始文本数据。

格式化数据以alpaca-chat-format为例，数据以json方式进行组织，每条数据分为instruction, input, output三个部分。

原始文本数据直接将文章或对话的原文作为输入，框架自动进行切分和训练。



下面以普通txt文本文件为例，介绍后续训练过程。

最佳实践

(2) 上传数据到云主机

windows系统使用命令提示符，macos/linux系统使用终端，执行scp命令将数据文件上传到云主机的/root/text-generation-webui/training/datasets目录下。

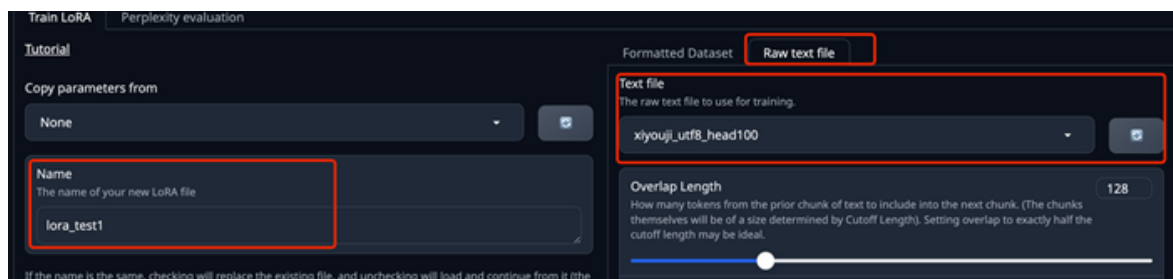
```
scp [本机文件路径] root@[ip]:/root/text-generation-webui/training/datasets
```

```
night@Night-Macbook Downloads % scp /Users/night/Downloads/xiyouji.txt root@219.152.96.190:/root/text-generation-webui/training/datasets/  
root@219.152.96.190's password:  
xiyouji.txt 100% 2725KB 7.0MB/s 00:00
```

(3) 训练参数配置

必填: Name, Text file选择刚才上传的文件。

其他参数的设定需要有足够的背景知识，这里不做详细介绍，您可自行学习模型训练时的基本参数。



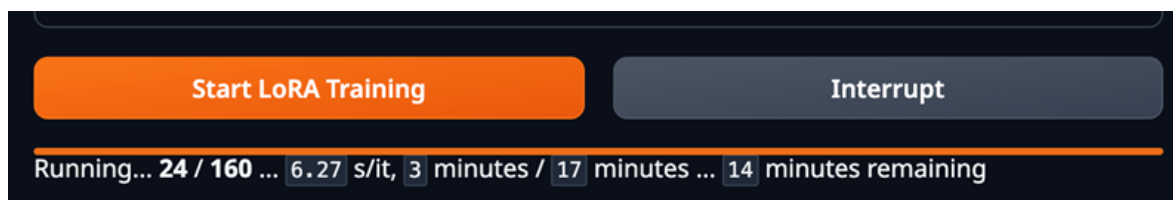
(4) 启动训练

在启动前保证Model页已经Load过基础模型。

注意

Llama 2-7b模型的训练默认fp16精度大约消耗18G显存，入门款T4显卡机型无法支持，需要在加载基础模型时选择load-in-8bit以降低内存占用。

点击页面右下方 Start LoRA Training按钮，下方将会出现提示文字，等待模型完成训练与保存。



注意

- 1.在此阶段您可能会遇到各类问题导致训练失败，建议参考[大模型学习机服务启停文档](#)中关于打印服务日志的内容。常见例如: 中文不是utf8编码，显存不足等。
- 2.由于huggingface.io网站访问受限，学习机在启动服务时需要检查是否在启动脚本run.sh里写入了export HF_HUB_OFFLINE=1，否则在训练启动和结束时都会自动访问huggingface网站导致超时失败。
- 3.完成训练后的基础模型已经被训练影响，需要重新reload一次基础模型以恢复初始状态。

最佳实践

```
root@ecm-4549:~/text-generation-webui# cat run.sh
#!/bin/bash

export PATH=$PATH:/root/python310/bin
export HF_HUB_OFFLINE=1 #can not access to huggingface.io

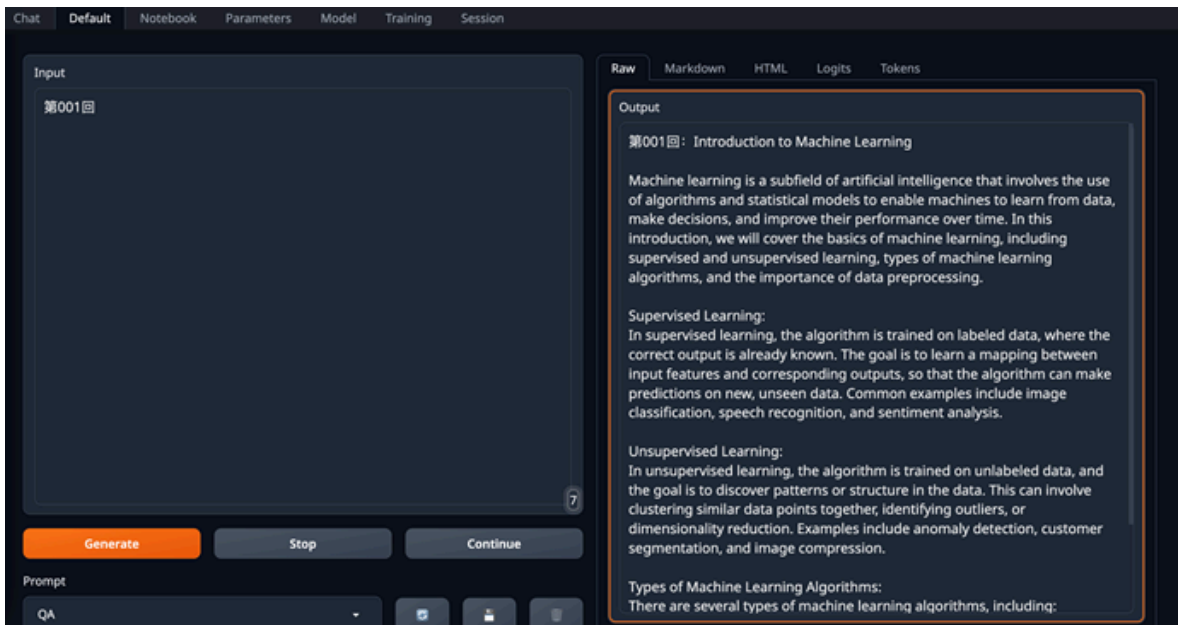
bash start_linux.sh \
  --listen \
  --listen-port 7861 \
  --gradio-auth-path /root/text-generation-webui/gradio-auth.txt
```

(5) 加载微调过的模型

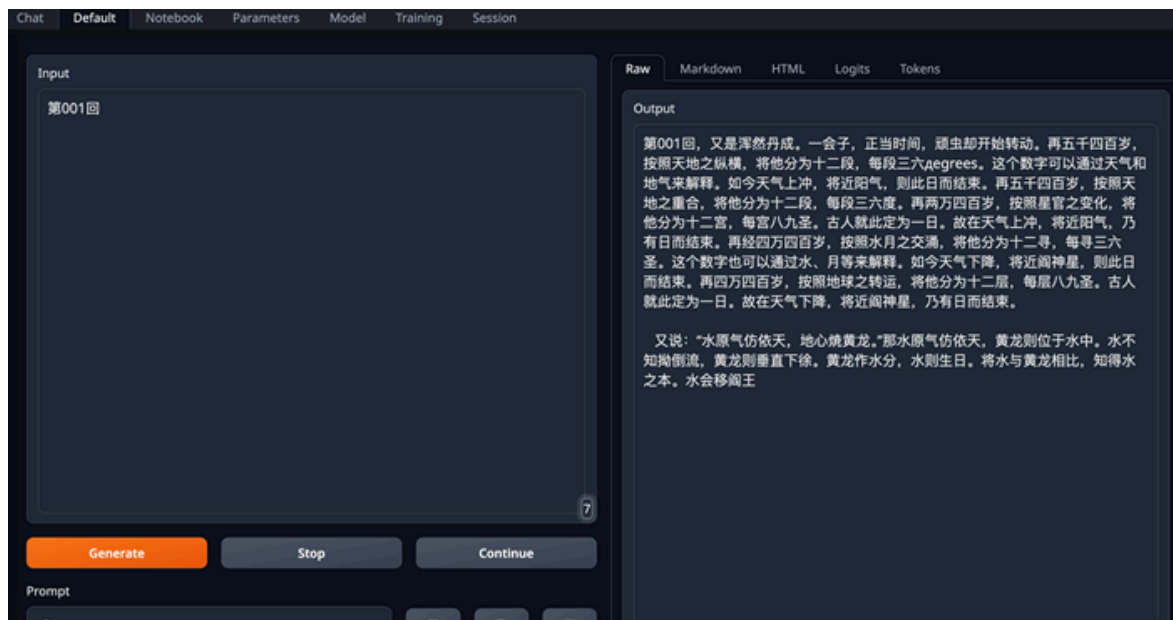
完成训练后，LoRA小模型会被保存到root/text-generation-webui/loras目录下。

回到Model页，右侧LoRA标签下选择刚才训练出的模型并点击Apply LoRAs按钮，在基础大模型之上加载LoRA小模型。

本次示例我们采用了西游记原文的前100行作为训练数据，在不加载LoRA小模型前，生成效果以英文为主。



加载我们训练的LoRA小模型后，生成内容风格贴合西游记原文。



使用大模型学习机微调自己的文本生成模型

前提条件

- 大模型学习机预装了Llama 2-7b-chat模型与stable-diffusion-base-v2.1模型，以及配套的开源服务框架text-generation-webui与stable-diffusion-webui，使您不需单独进行下载模型与配置环境。
- 云主机开通与如何登录网页页面参见[部署文档](#)。

1. 模型微调简介

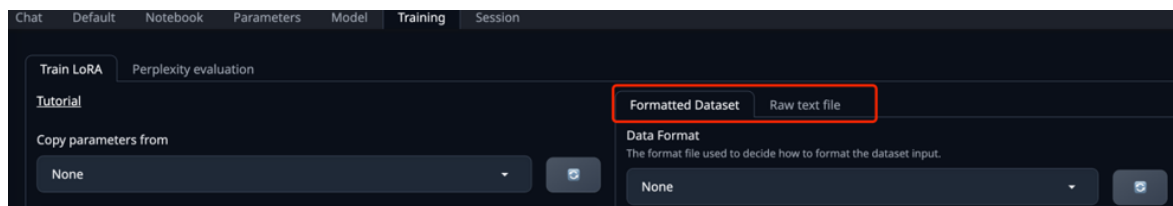
由于基础大模型参数量极大，训练需要的资源量极大，因此基于基础大模型进行微调的小模型应运而生。学习机内框架自带的LoRA技术便是其中最主流的一种。

LoRA小模型无法独立使用，需要在加载完基础大模型后再进行加载，对基础大模型的能力进行扩展。

已有LoRA模型，如何加载可参考[文本生成模型使用最佳实践](#)。

2. 准备训练数据

模型训练支持多种输入格式，主要分为格式化数据(Formatted Dataset)与原始文本数据(Raw text file)。



2.1 格式化数据(Formated Dataset)

格式化数据一般用于对话领域，以json形式进行组织，可以在/root/text-generation-webui/training/formats目录下找到各类格式的配置文件的。以alpaca-chatbot-format为例，

最佳实践

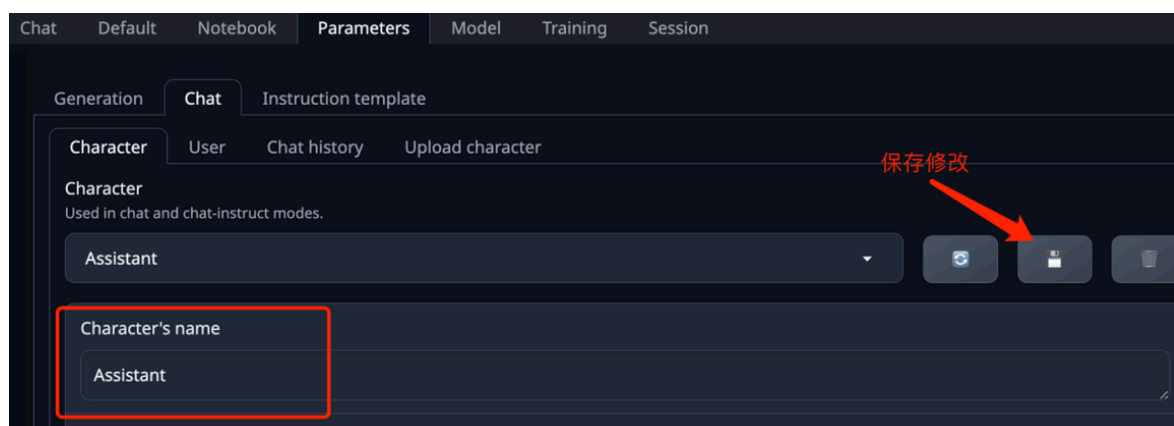
每条数据包括instruction, output或instruction, input, output三个部分。参考示例数据如下:

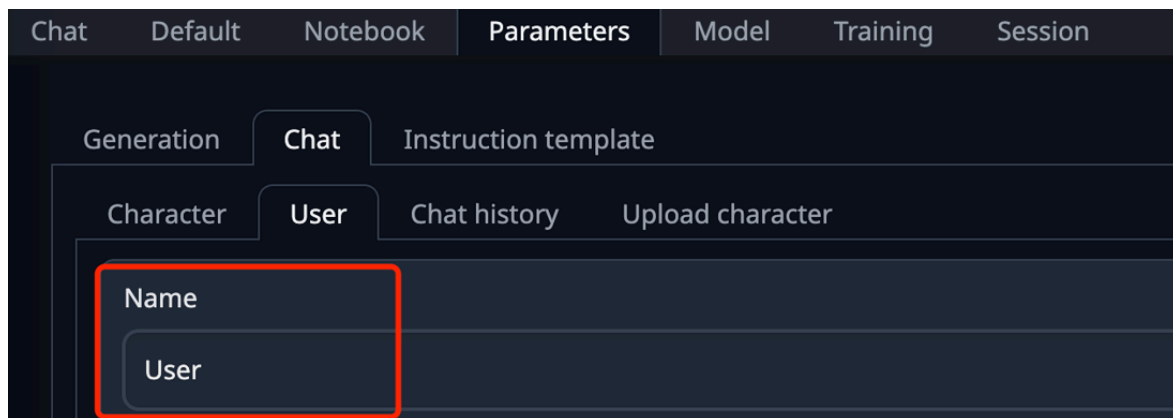
```
[
  {
    "instruction":
"我想基于天翼云云主机通过镜像部署大模型学习机，你能告诉我具体怎么操作吗？",
    "output":
"当然可以。首先，您需要登录云主机控制台，并选择创建云主机资源池，然后点击“创建云主机”按钮。"
  },
  {
    "instruction": "好的，我已经点击了“创建云主机”按钮，接下来应该怎么做？",
    "output":
"在当前可用选项中，您可以在重庆-重庆2或宁夏-中卫5购买GPU计算加速型云主机的Pi7规格；或者在重庆-重庆2购买GPU计算加速型云主机的Pi2规格。"
  }
]
```

您需要将私有数据组织成上述格式，并保存为[文件名].json文件。例如xuexiji_chatdata.json

参考/root/text-generation-webui/training/formats/alpaca-chatbot-format.json文件内容，格式化输入数据将被组织成 User: %instruction%\nAssistant: %output%输入模型，此处的 User和 Assistant务必与Parameter页面的Chat标签下的Character和User的Name保持一致。对话时的角色标签对大模型来说是一个很强的特征。

需要参考下图将Parameter 页Chat 标签下默认的名称 修改为Character' s Name: Assistant; User Name: User 。否则在进行chat 对话时小模型几乎完全无效。





2.2 原始文本数据(Raw text file)

原始文本数据直接将文章或对话的原文作为输入，没有格式转化的过程，框架自动进行切分和训练。

例如我们直接截取大模型学习机部署文档作为输入：

本文基于天翼云云主机，为您介绍通过镜像部署大模型学习机的具体操作。

购买云主机。

登录云主机控制台，选择创建云主机资源池，点击“创建云主机”按钮。

当前您可在重庆-重庆2、宁夏-中卫5购买GPU计算加速型云主机的Pi7规格；也可在重庆-重庆2购买GPU计算加速型云主机的Pi2规格。

您需要将私有数据以utf8编码格式写入到文件中，并保存为[文件名].txt文件。例如xuexiji_docdata.txt。

3. 数据上传到云主机

windows系统使用命令提示符， macos/linux系统使用终端，执行scp命令将数据文件上传到云主机的/root/text-generation-webui/training/datasets目录下。

```
scp [本机文件路径] root@[ip]:/root/text-generation-webui/training/datasets
```

```
night@Night-Macbook Downloads % scp /Users/night/Downloads/xiyouji.txt root@219.152.96.190:/root/text-generation-webui/training/datasets/
root@219.152.96.190's password:
xiyouji.txt 100% 2725KB 7.0MB/s 00:00
```

如果您擅长使用linux系统，也可采用例如ftp，lrzsz等其他方式将数据从本地上传到云主机内。

4. 启动训练

4.1 加载基础大模型

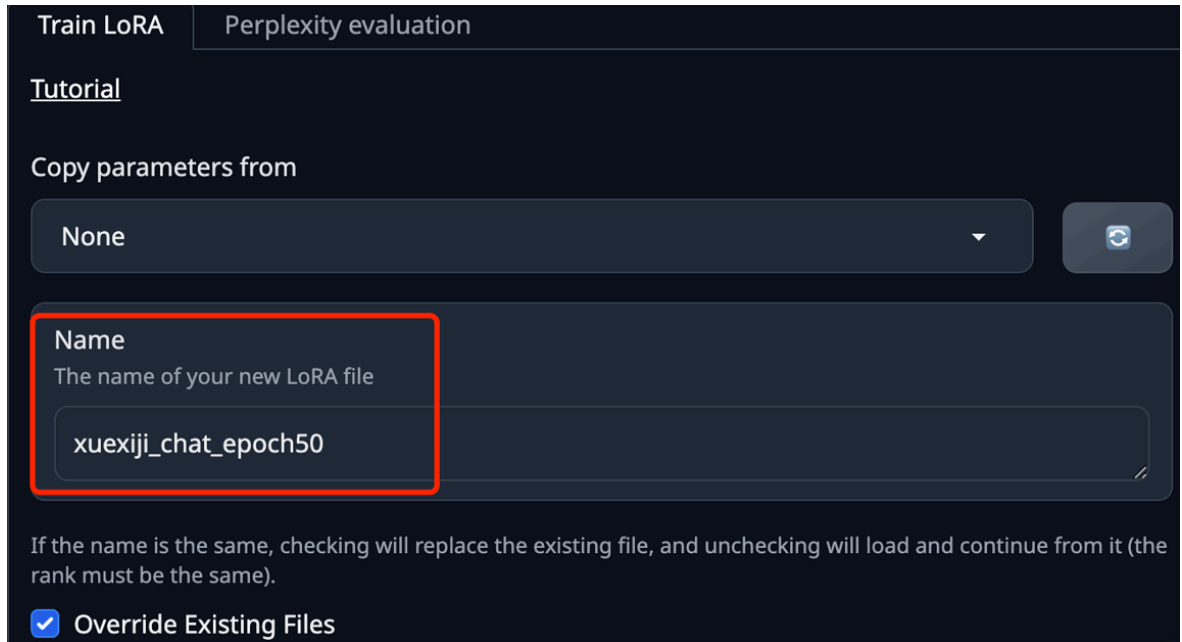
由于小模型需要在基础大模型的基础上才能使用，因此在训练前也要先保证在Model页完成了基础大模型的加载。

1. Llama 2-7b模型在训练时，默认的fp16精度大约要消耗18G显存，入门款T4显卡机型无法支持。需要在加载基础模型时选择load-in-8bit以降低内存占用。
2. 如果刚才进行过训练，则已加载进显存/内存的基础大模型也已经受到了训练的影响，需要先进行模型reload恢复初始状态。

4.2 训练参数配置

在Training页面进行参数配置。

- Name: 必填参数，输出模型将以此命名写入到/root/text-generation-webui/loras文件夹下。



The screenshot shows the 'Train LoRA' interface with two tabs: 'Train LoRA' and 'Perplexity evaluation'. Under the 'Train LoRA' tab, there is a 'Tutorial' section. Below it, the 'Copy parameters from' dropdown is set to 'None'. A text input field for 'Name' is highlighted with a red box, containing the text 'xuexiji_chat_epoch50'. Below the input field, there is a checkbox labeled 'Override Existing Files' which is checked. A note below the checkbox states: 'If the name is the same, checking will replace the existing file, and unchecking will load and continue from it (the rank must be the same).'

- LoRA rank: 小模型的维度，取值越大学习能力越强(但模型也越大)。可以直接采用默认值32，如果发现模型效果不佳，可以逐步增加到128进行尝试。
- LoRA Alpha: 推荐保持为LoRA rank的两倍。
- Batch Size: 每一批参与训练的数据条数。采用默认值128即可。如果您的训练数据条数过少，则也可修改为64或32。

最佳实践

LoRA Rank

128

Also called dimension count. Higher values = larger file, more content control. Smaller values = smaller file, less control. Use 4 or 8 for style, 128 or 256 to teach, 1024+ for fine-detail on big data. More VRAM is needed for higher ranks.

LoRA Alpha

256

This divided by the rank becomes the scaling of the LoRA. Higher means stronger. A good standard value is twice your Rank.

Batch Size

32

Global batch size. The two batch sizes together determine gradient accumulation (gradientAccum = batch / microBatch). Higher gradient accum values lead to better quality training.

Micro Batch Size

4

Per-device batch size (NOTE: multiple devices not yet implemented). Increasing this will increase VRAM usage.

- Epochs: 数据训练的重复次数。一般保持1-3即可。如果您的训练数据特别少(例如不足1000条), 则可以适当增大epoch以重复使用训练数据。例如本示例只用了30条训练数据, 可以使用epochs=50或100以加强训练效果(实际使用时不推荐过大的epoch, 示例仅为了保证能够训练出明显效果, 已达到过拟合的状态)。

Save every n steps

If above 0, a checkpoint of the LoRA will be saved every time this many steps pass.

0

Epochs

Number of times every entry in the dataset should be fed into training. So 1 means feed each item in once, 5 means feed it in five times, etc.

50

Learning Rate

In scientific notation. $3e-4$ is a good starting base point. $1e-2$ is extremely high, $1e-6$ is extremely low.

其他参数的设定需要有足够的背景知识, 这里不做详细介绍, 您可自行学习模型训练时的基本参数。

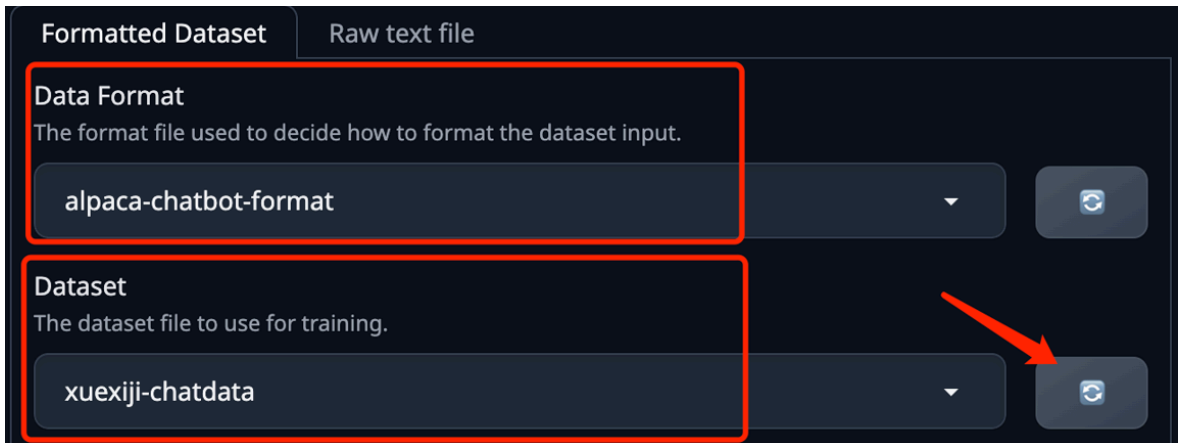
最佳实践

4.3 加载样本配置

在Training页面进行样本配置。您可以选择以下的其中一种数据进行训练。

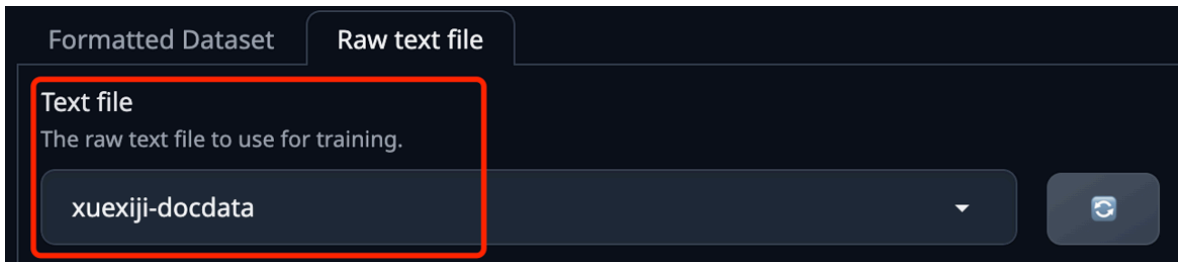
4.3.1 加载格式化数据(Formatted Dataset)

选择Formatted Dataset，设置Data Format和Dataset为刚才准备好的数据文件。如果无法查看到文件，尝试点击右侧的刷新按钮重新加载，以及检查文件名后缀是否为.json。



4.3.2 加载原始文本数据(Raw text file)

选择Raw text file，设置Text file为刚才准备好的数据文件。如果无法查看到文件，尝试点击右侧的刷新按钮重新加载，以及检查文件名后缀是否为.txt。



4.4 启动训练

在启动前保证Model页已经Load过基础模型。

点击页面右下方 Start LoRA Training按钮，下方将会出现提示文字，等待模型完成训练与保存。



说明

1. 在此阶段您可能会遇到各类问题导致训练失败，建议参考服务启停最佳实践文档[文档链接]中关于打印服务日志的内容。常见例如：中文不是utf8编码，显存不足等。

最佳实践

- 在打印的日志中可以查看到模型训练时的loss指标。一般loss下降到0.02以下即为基本训练达标。
- 由于huggingface.io网站访问受限，学习机在启动服务时需要检查是否在启动脚本run.sh里写入了export HF_HUB_OFFLINE=1，否则在训练启动和结束时都会自动访问huggingface网站导致超时失败。

```
root@ecm-4549:~/text-generation-webui# cat run.sh
#!/bin/bash

export PATH=$PATH:/root/python310/bin
export HF_HUB_OFFLINE=1 #can not access to huggingface.io

bash start_linux.sh \
  --listen \
  --listen-port 7861 \
  --gradio-auth-path /root/text-generation-webui/gradio-auth.txt
```

4.5 加载微调过的模型

完成训练后，LoRA小模型会被保存到/root/text-generation-webui/loras目录下。

注意训练完成后已加载进显存/内存的基础大模型也已经受到了训练的影响，需要先进行模型reload恢复初始状态，再进行其他任何操作。

回到Model页，右侧LoRA标签下选择刚才训练出的模型并点击Apply LoRAs按钮，在基础大模型之上加载LoRA小模型(如果看不到小模型，尝试点击Apply LoRAs按钮左侧的刷新按钮)。

4.6 使用模型示例

在不加载LoRA小模型前，生成效果以英文为主，且内容不贴合训练数据。

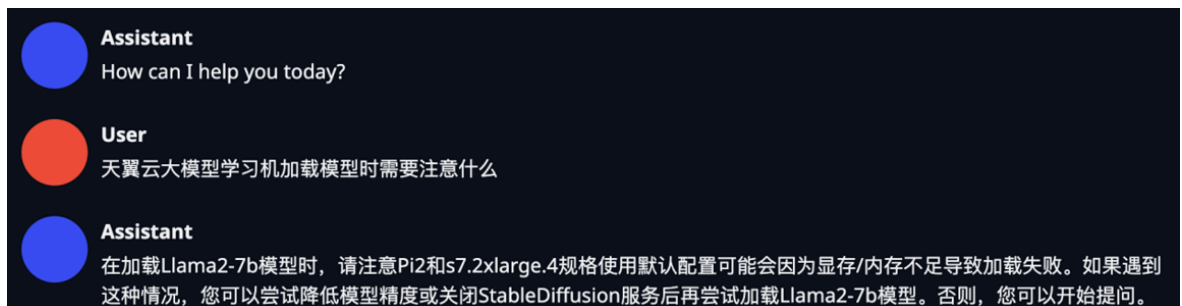
4.6.1 对话(chat)模式

参见本文2.1节的“注意”部分，先确保已将Parameter页Chat标签下默认的名称修改为Character's Name: Assistant; User Name: User。否则在进行chat对话时小模型几乎完全无效。

加载基于alpaca-chatbot-format格式数据训练的LoRA小模型后，回答内容贴合训练数据。

注意

大模型生成存在随机性，一定完全符合预期，您可参考”[大模型学习机文本生成最佳实践](#)”文档，调整参数降低模型随机性。如果始终无法符合预期，则可能是训练epoch不足，可以继续加大epoch测试。



说明

最佳实践

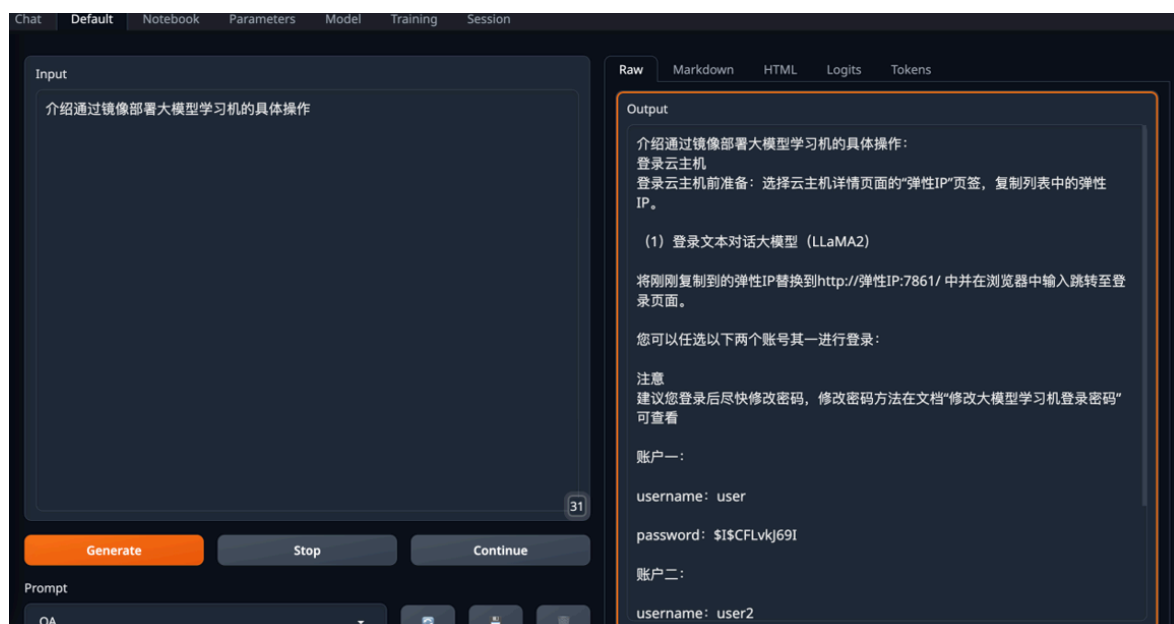
对话(chat)模式的本质是先将输入数据组织成[role属性] Assistant: [文本] User: [文本] Assistant: [文本]的格式, 然后再作为prompt输入模型。

4.6.2 默认(default)模式

加载基于raw text file数据训练的LoRA小模型后, 生成内容贴合训练数据。

注意

大模型生成存在随机性, 不一定完全符合预期, 您可参考”[大模型学习机文本生成最佳实践](#)”文档, 调整参数降低模型随机性。如果始终无法符合预期, 则可能是训练epoch不足, 可以继续加大epoch测试。



使用大模型学习机微调自己的图像生成模型

前提条件

- 大模型学习机预装了Llama 2-7b-chat模型与stable-diffusion-base-v2.1模型, 以及配套的开源服务框架text-generation-webui与stable-diffusion-webui, 使您不需单独进行下载模型与配置环境。
- 云主机开通与如何登录网页页面参见[部署文档](#)。

一、模型微调简介

由于基础大模型参数量极大, 训练需要的资源量极大, 因此基于基础大模型进行微调的技术应运而生。针对StableDiffusion进行微调的技术主流的有4种, 分别是:

1. Textual Inversion: 不改变原模型结构与参数, 只增加新的关键词并训练新的输入embedding。生成的新模型很小, 大约几十KB。由于没有影响原模型的结构与参数, 因此调整能力有限。
2. Hypernetwork: 在原模型的cross-attention网络结构中插入一层新网络, 与原模型结构串行执行。由于新网络会平均作用在整个模型上, 因此更适宜训练绘画风格。模型大小约几十MB。
3. LoRA: 在原模型的cross-attention网络结构中插入多个小网络结构, 与原模型结构并行执行。实践证明这种结构的效果要优于Hypernetwork, 因此LoRA是当前最主流的微调方案。模型大小约几十MB。

最佳实践

4. Dreambooth: 对整个原模型的参数进行微调，训练成本高，生成的模型相当于一个新的基础大模型。模型大小约几GB。

已有LoRA模型如何加载可参考[图像生成模型使用最佳实践](#)。

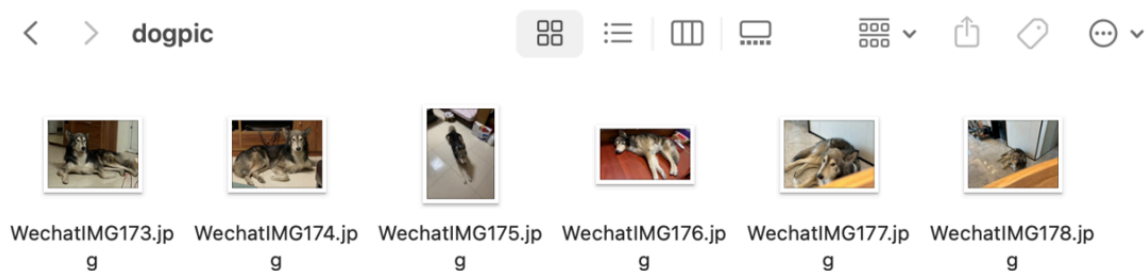
本文以Textual Inversion方式为例，介绍通过私有数据微调模型并进行图像生成的流程。如果您有微调新的LoRA模型的需求，本文也给出了相关插件的安装说明。

二、准备训练数据

1. 图片文件准备

准备一批用于训练的图片，可以是同一人物/物体的不同视角，也可以是同一种绘画风格。

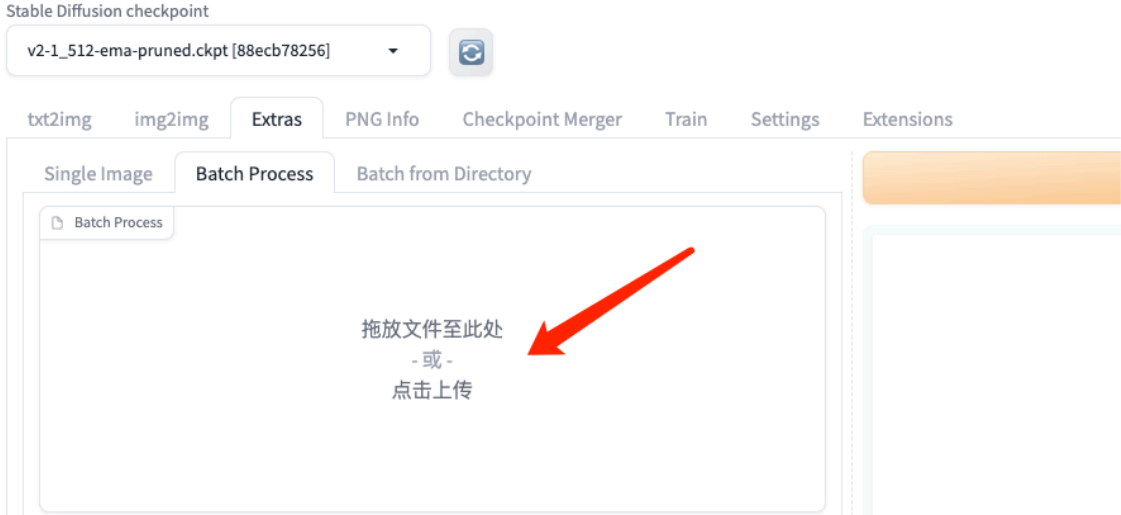
本示例中采用了同一只狗狗的不同视角图片。



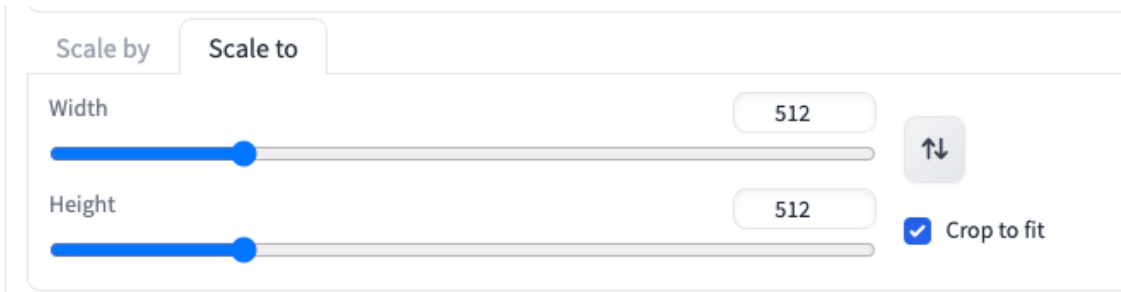
最佳实践

2. 图片文件上传与裁切

在Extras页面的Batch Process下点击文字，选择图片文件。



选择Scale to将图片裁切到512x512的大小。

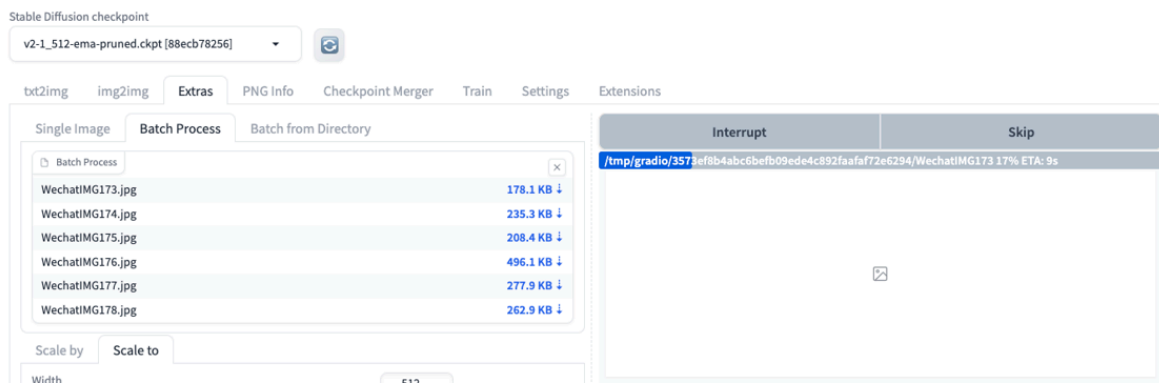


点击generate按钮，等待图片上传完毕。

注意

目前某些镜像中尚未预装deepbooru功能所需的模型。当您首次上传图片时，系统会访问github进行模型下载，该模型大小为614MB。依据您的网络带宽，这可能需要几分钟的时间。

最佳实践



上传完成的图片在云主机的 `/root/stable-diffusion-webui/output/extras-images` 目录下。

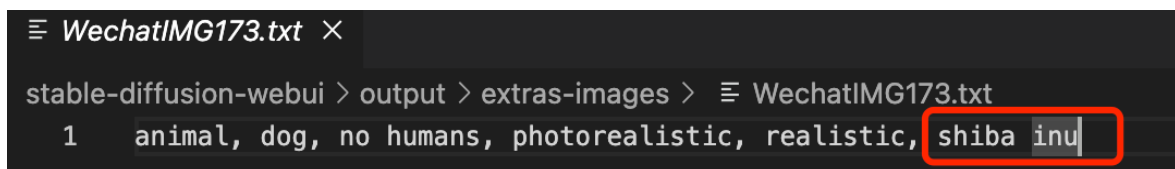
```
root@ecm-4549:~/stable-diffusion-webui/output/extras-images# ls -l
total 11600
drwxr-xr-x 2 root root 4096 Apr 7 10:44 ./
drwxr-xr-x 3 root root 4096 Apr 7 10:44 ../
-rw-r--r-- 1 root root 1494090 Apr 7 10:44 WechatIMG173.png
-rw-r--r-- 1 root root 60 Apr 7 10:44 WechatIMG173.txt
-rw-r--r-- 1 root root 1897223 Apr 7 10:44 WechatIMG174.png
-rw-r--r-- 1 root root 113 Apr 7 10:44 WechatIMG174.txt
-rw-r--r-- 1 root root 1830959 Apr 7 10:44 WechatIMG175.png
-rw-r--r-- 1 root root 68 Apr 7 10:44 WechatIMG175.txt
-rw-r--r-- 1 root root 2092483 Apr 7 10:44 WechatIMG176.png
-rw-r--r-- 1 root root 77 Apr 7 10:44 WechatIMG176.txt
-rw-r--r-- 1 root root 2303343 Apr 7 10:44 WechatIMG177.png
-rw-r--r-- 1 root root 92 Apr 7 10:44 WechatIMG177.txt
-rw-r--r-- 1 root root 2213769 Apr 7 10:44 WechatIMG178.png
```

建议您将文件通过 `mv` 命令移动到其他文件夹，方便区分不同的训练文件。例如：

```
cd /root/stable-diffusion-webui
mkdir -p train/picdata1
mv output/extras-images/* train/picdata1
```

您还可查看每张图片生成的 `txt` 文件，检查反推出来的提示词是否符合预期，并删除识别错误的提示词。

例如本示例中模式识别的提示词中有一个多余的“shiba inu”，需要将其删除。



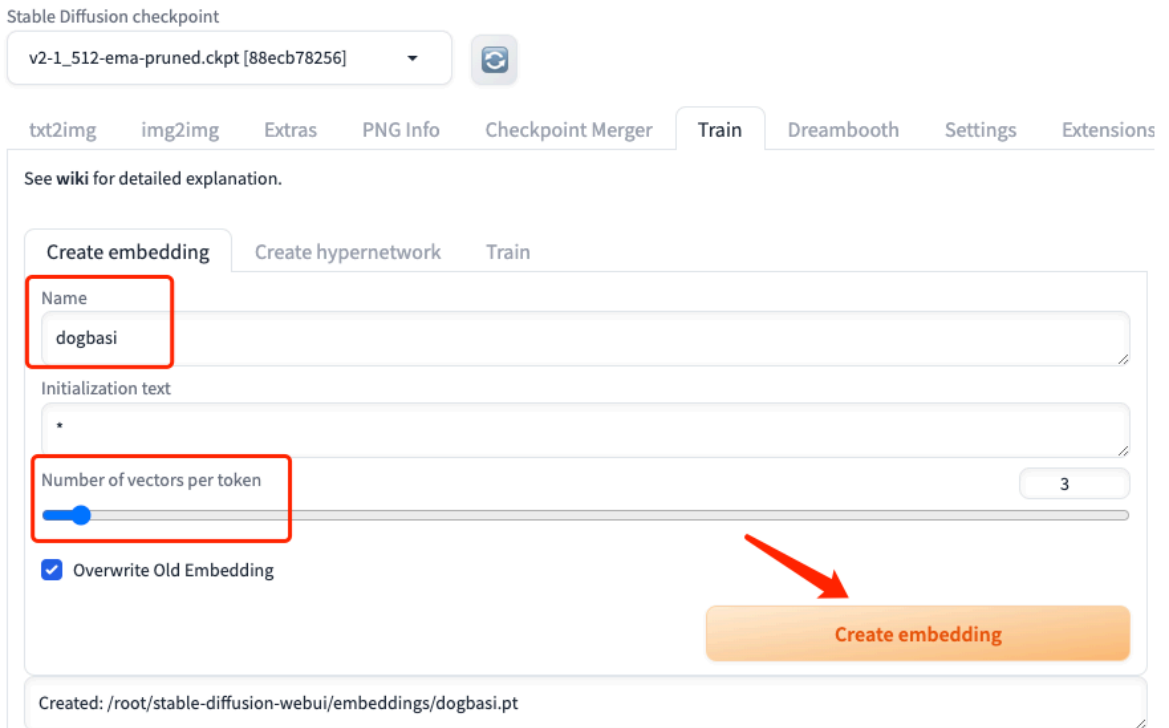
三、启动训练

stable-diffusion-webui框架在启动时默认已经加载好了基础大模型，无需做额外操作。

1. 生成初始模型文件

首先需要在Train页面的Create embedding标签下，创建一个空的初始模型文件。

- **Name:** 小模型的名称，也是未来使用小模型时需要用到的专有提示词名称。例如我们为训练图片中的狗狗取名字叫“八四”，因此模型的名字可以叫“dogbasi”，未来使用模型时也需要用“dogbasi”触发。注意这里的名称必须是特有的，避免和模型已有的关键词重复。
- **Number of vectors per token:** 小模型参数的维度，数值越大模型拟合能力越强，但也可能导致拟合过强。推荐设置在3-7左右。如果是学习绘画风格，一般设置尽量小一些；如果是学习具体物体，可以设置大一些。



最佳实践

2. 调整训练配置

在Train页面的Train标签页，修改如下设置：

- **Embedding:** 选择3.1步骤中创建过的初始embedding模型。
- **Dataset directory:** 填写2.2步骤中上传与调整过的训练图片目录。
- **Prompt template:** 如果期望训练绘画风格，选择style_filewords.txt；如果期望训练具体物体，选择subject_filewords.txt。
- **Max Steps:** 训练的总轮次，一般在图片数量不大的情况下(10张图左右)，推荐设置为1000及以上，以保证训练质量。

Stable Diffusion checkpoint
v2-1_512-ema-pruned.ckpt [88ecb78256]

txt2img img2img Extras PNG Info Checkpoint Merger **Train** Dreambooth Settings Extensions

See [wiki](#) for detailed explanation.

Create embedding Create hypernetwork **Train**

Train an embedding or Hypernetwork; you must specify a directory with a set of 1:1 ratio images [[wiki](#)]

Embedding
dogbasi

Hypernetwork

Dataset directory
/root/stable-diffusion-webui/output/extras-images

Prompt template
subject_filewords.txt

Max steps
1000

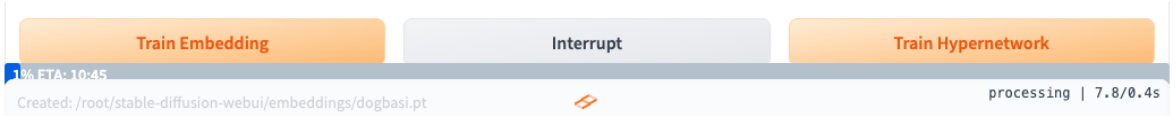
Save an image to log directory every N steps, 0 to disable
500

Save a copy of embedding to log directory every N steps, 0 to disable
500

最佳实践

3. 启动训练

点击页面底部Train Embedding按钮开始训练，等待进度条结束。

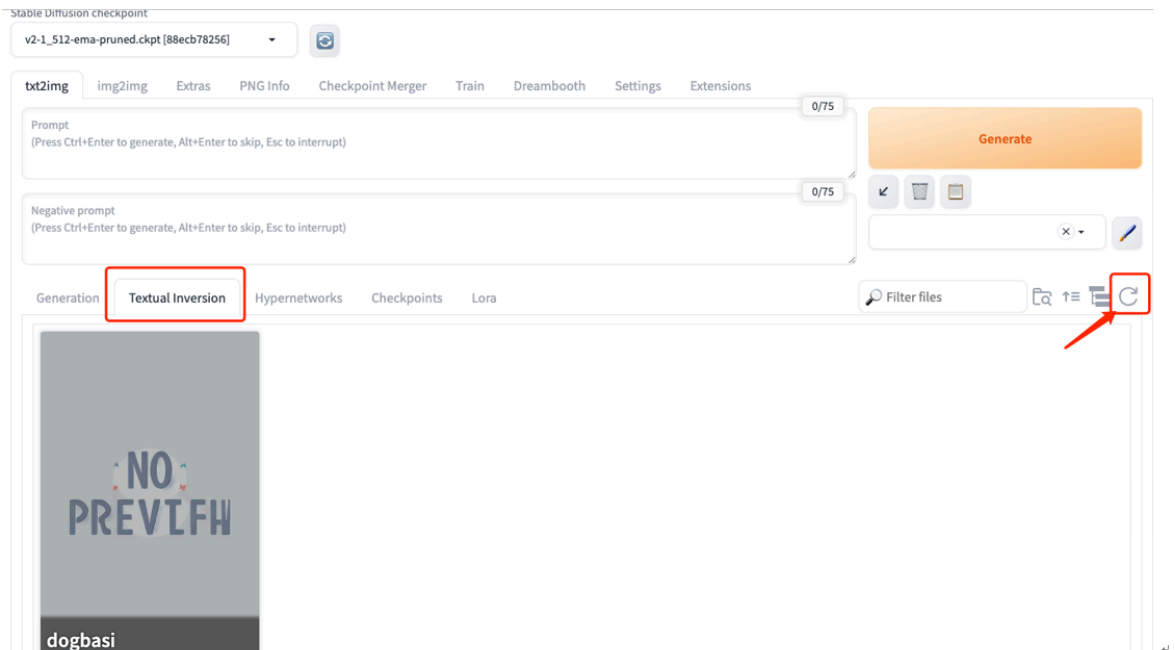


注意

在batch size默认设置为1的情况下，训练需要消耗6G显存。

4. 加载微调过的模型

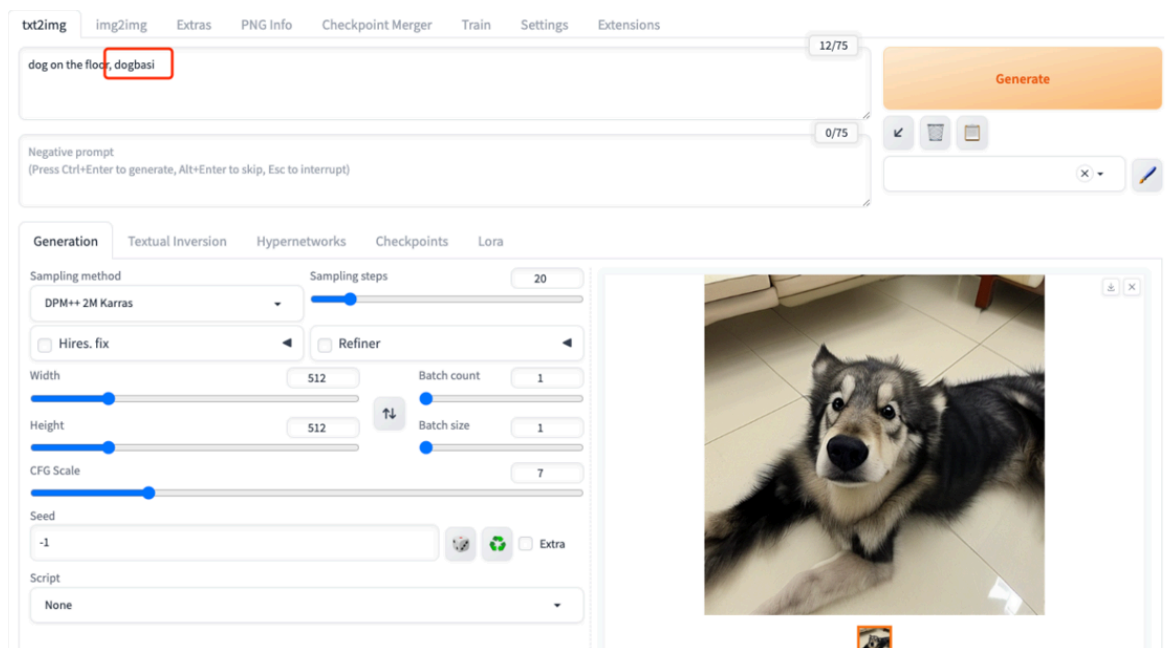
在txt2img页面的Textual inversion标签下，点击右侧的刷新按钮，如果看到新训的小模型即代表成功。



最佳实践

5. 使用模型示例

在提示词prompt中附上微调过的小模型名称，即可生成类似训练数据的新图片。如果发现图片质量不佳，可以尝试加大训练时的Max Steps，或者加大Create embedding时的Number of vectors per token，或者准备更多高质量的训练图片（数据最关键）。



四、微调新LoRA模型插件安装

stable-diffusion-webui本体没有微调LoRA模型的功能，需要安装Dreambooth插件。下面将以此为例介绍如何在本框架中安装插件。

stable-diffusion-webui本身提供了自动下载和安装插件的功能（见extensions页），但由于github和pip库访问受限，自动下载无法成功。因此本文采用了手动安装的方式。

手动安装需要您通过ssh或远程登录等方式登录到云主机系统内，您可以参考[修改大模型学习机登录密码](#)的相关内容。

同时，由于涉及到服务运行环境的修改，因此所有操作都将在docker容器内执行。进入容器的方式参考：

检查容器状态

```
docker ps -a
```

进入容器

```
docker exec -it stable_diffusion bash
```

注意

执行完进入容器的命令后，命令行界面并不会有明显变化，您可以通过执行命令docker ps是否能成功，来判断当前在容器内还是云主机内。

最佳实践

1. 配置github地址（可选）

如果您在执行`git clone`时始终因为网络问题无法成功，则可参考本节内容。

a. 获取可用的ip地址



注意

1. IP地址并不会一直固定，未来如无法访问github，您需要重新确认最新的IP地址。
2. 可以使用代理服务器彻底解决问题，但注意如果设置了`http_proxy`系统变量，需要在使用完毕后取消设置，或者在`/root/stable-diffusion-webui/run.sh`启动脚本中增加一项`--share \`，否则webui服务会报错无法访问localhost。

b. 修改hosts文件

请确保当前在docker容器内。

通过vim编辑器修改`/etc/hosts`文件，添加如下内容：

```
# 从ipaddress查到的真实ip地址  
140.82.114.3 github.com
```

完成配置后，访问github相关链接成功概率提升（但不保证一定成功）。

2. 下载插件文件

请确保当前在docker容器内。

进入到`/root/stable-diffusion-webui/extensions`目录下，并下载dreambooth插件。

```
cd /root/stable-diffusion-webui/extensions  
git clone https://github.com/d8ahazard/sd_dreambooth_extension.git
```

遇到连接失败可以多重试几次，如果始终失败，则参考4.1节的内容手动配置github地址。

3. 安装插件需要的依赖库

请确保当前在docker容器内。

`stable-diffusion-webui`采用了python venv的方式进行环境隔离，首先需要进入到虚拟环境。

```
cd /root/stable-diffusion-webui
```

最佳实践

```
source venv/bin/activate
```

执行成功后，命令行界面开头将出现(venv)标识。

```
root@ecm-4549:~/stable-diffusion-webui/extensions# cd /root/stable-diffusion-webui/  
root@ecm-4549:~/stable-diffusion-webui# source venv/bin/activate  
(venv) root@ecm-4549:~/stable-diffusion-webui#
```

然后在插件文件夹内安装依赖库。由于pip库访问受限问题，这里采用了清华的镜像源。

```
cd /root/stable-diffusion-webui/extensions/sd_dreambooth_extension  
pip install -r requirements.txt -i https://pypi.tuna.tsinghua.edu.cn/simple
```

安装成功后将出现如下提示：

```
Installing collected packages: dadaptation, gitpython, bitsandbytes  
Attempting uninstall: gitpython  
Found existing installation: GitPython 3.1.32  
Uninstalling GitPython-3.1.32:  
Successfully uninstalled GitPython-3.1.32  
Successfully installed bitsandbytes-0.43.0 dadaptation-3.2 gitpython-3.1.43  
  
[notice] A new release of pip available: 22.2.1 -> 24.0  
[notice] To update, run: pip install --upgrade pip
```

4. 重启服务以应用插件

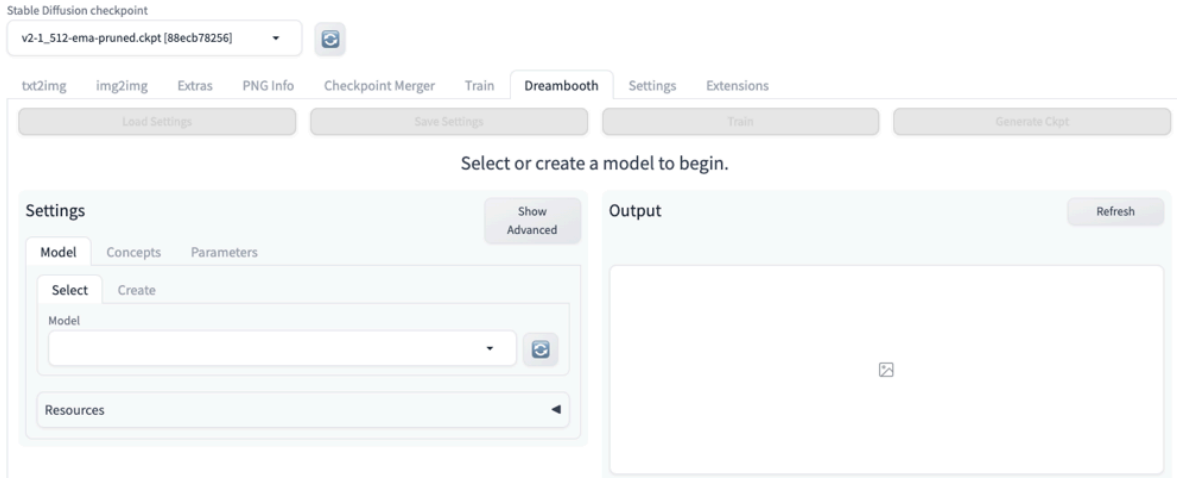
如果您当前在docker容器环境中，可直接执行exit命令退出到云主机内。

参考[大模型学习机服务启停最佳实践](#)文档，重启stable-diffusion-webui服务：

```
sh /root/restart_sd_webui.sh
```

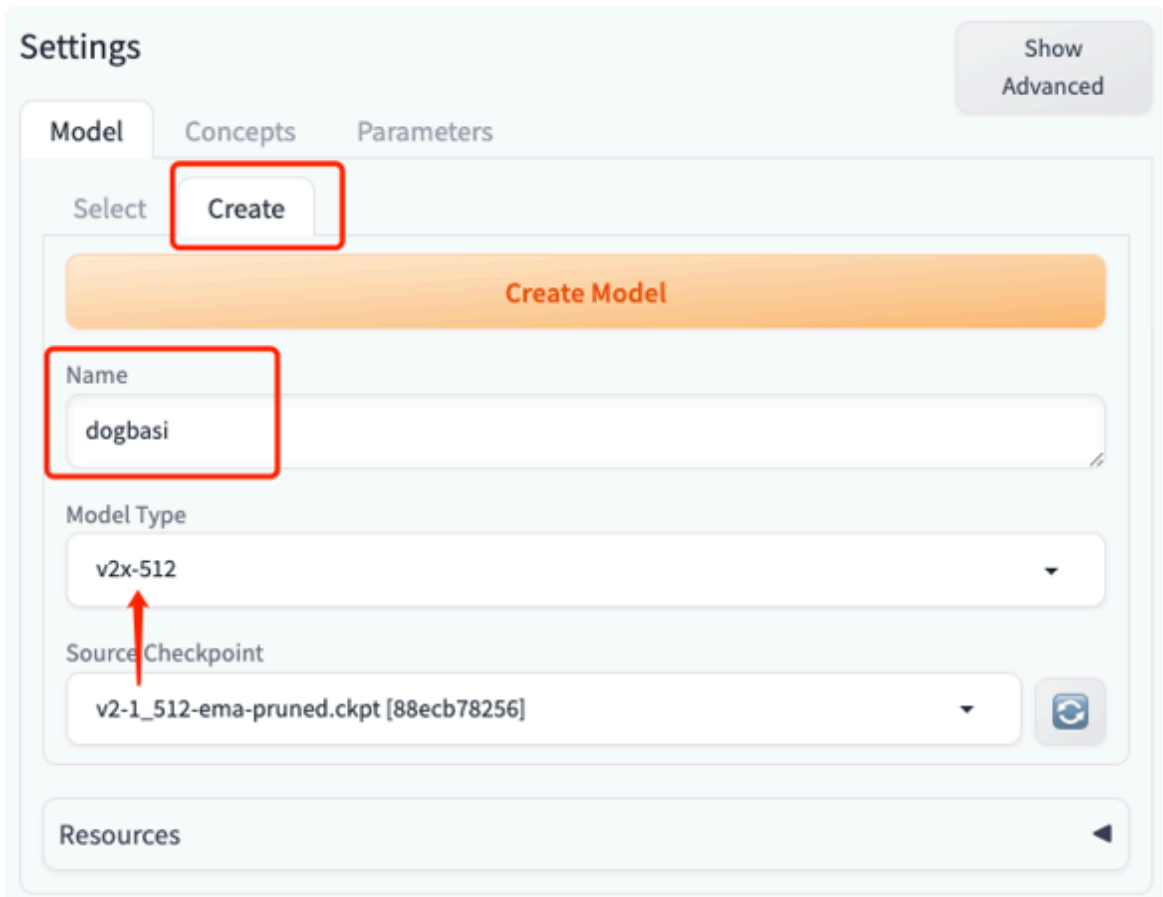
5. 确认插件安装完毕

重新登录页面后，应多出Dreambooth页面：



Dreambooth插件的使用方式和上文训练Textual Inversion是一致的。

a. 在Create页面创建一个空模型。内容参考下图。



- b. 首次创建时，插件会从github额外下载一些辅助模型，大小约几百MB。建议您提前自行下载并放到/root/stable-diffusion-webui/models/GFPGAN目录下，避免等待太长时间。

```
cd /root/stable-diffusion-webui/models/GFPGAN
```

```
wget
```

```
https://github.com/xinntao/face_xlib/releases/download/v0.1.0/detection_Resnet50_Final.pth
```

```
wget https://github.com/xinntao/face_xlib/releases/download/v0.2.2/parsing_parsenet.pth
```

- c. 插件还需要从huggingface网站下载StableDiffusion模型的部分内容，因此需要删除/root/stable-diffusion-webui/run.sh脚本中的内容：

```
# 删除这一行可恢复对huggingface的连接
```

```
export HF_HUB_OFFLINE=1 #can not access to
```

最佳实践

huggingface.io

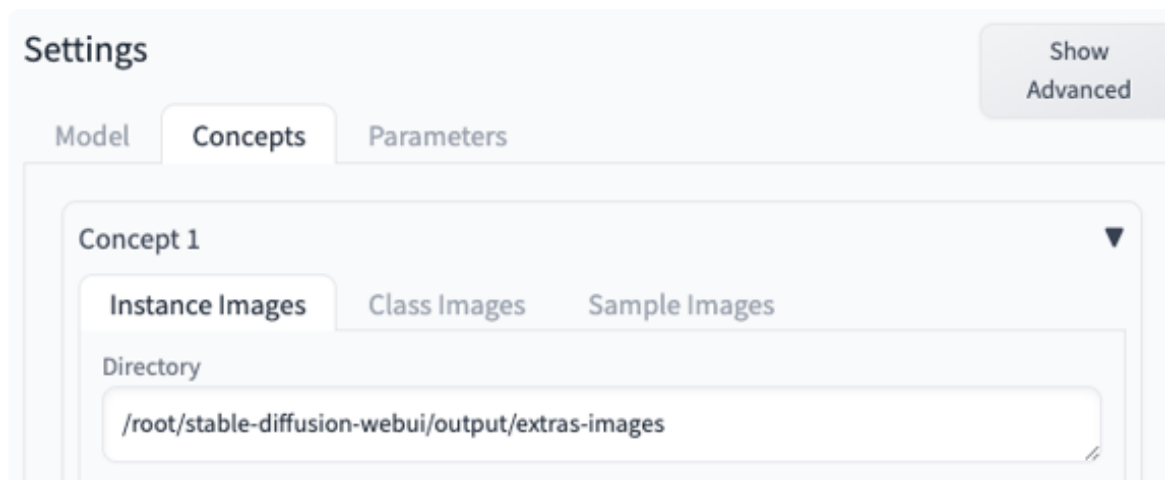
参考[配置github地址](#)对github真实ip的配置，也需要对huggingface.co进行相关配置，否则无法访问。

下载模型数据需要占据大量磁盘空间，建议额外购买50G-100G大小的数据盘，并挂载到/root/stable-diffusion-webui/models/dreambooth目录下（也可先将当前model文件夹改名，然后直接将数据盘挂载为新的model文件夹，最后将老数据mv进去）。

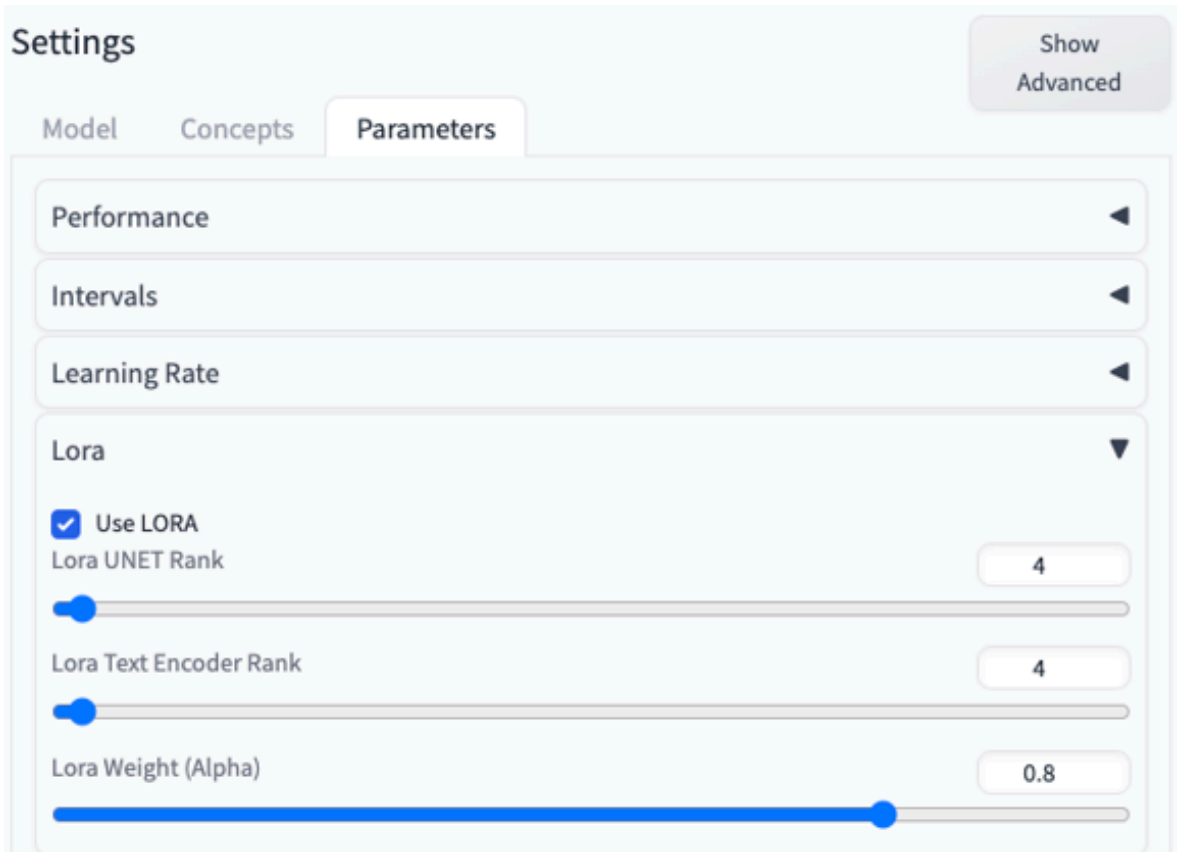
说明

即使配置了真实ip，在访问huggingface.co时也依然会失败，建议您使用代理服务器进行连接。配置代理服务器和http_proxy环境变量后，需要在/root/stable-diffusion-webui/run.sh脚本中增加`--share`，否则webui服务会报错无法访问localhost。

- d. 在Concepts页面设置样本路径。



- e. 在Parameters页开启LoRA训练。



其他训练相关的参数您可自行查看和调整（例如训练轮次，学习率等），本文不再一一赘述。

AIGC实践

在天翼云使用Ollama运行DeepSeek的最佳实践-7B等版本

什么是DeepSeek

DeepSeek 是一个基于 Transformer 架构的大型语言模型（LLM），由深度求索（DeepSeek）公司开发。它能够处理自然语言理解、生成、翻译、问答等多种任务。在目前大模型主流榜单中，DeepSeek-V3 在开源模型中位列榜首，与世界上最先进的闭源模型不分伯仲。在对话模型典型任务方面的评测效果如下：

最佳实践

Benchmark (Metric)	DeepSeek V3	DeepSeek V2.5	Qwen2.5	Llama3.1	Claude-3.5	GPT-4o
		0905	72B-Inst	405B-Inst	Sonnet-1022	0513
Architecture	MoE	MoE	Dense	Dense	-	-
# Activated Params	37B	21B	72B	405B	-	-
# Total Params	671B	236B	72B	405B	-	-
English						
MMLU (EM)	88.5	80.6	85.3	88.6	88.3	87.2
MMLU-Redux (EM)	89.1	80.3	85.6	86.2	88.9	88.0
MMLU-Pro (EM)	75.9	66.2	71.6	73.3	78.0	72.6
DROP (3-shot F1)	91.6	87.8	76.7	88.7	88.3	83.7
IF-Eval (Prompt Strict)	86.1	80.6	84.1	86.0	86.5	84.3
GPQA-Diamond (Pass@1)	59.1	41.3	49.0	51.1	65.0	49.9
SimpleQA (Correct)	24.9	10.2	9.1	17.1	28.4	38.2
FRAMES (Acc.)	73.3	65.4	69.8	70.0	72.5	80.5
LongBench v2 (Acc.)	48.7	35.4	39.4	36.1	41.0	48.1
Code						
HumanEval-Mul (Pass@1)	82.6	77.4	77.3	77.2	81.7	80.5
LiveCodeBench (Pass@1-COT)	40.5	29.2	31.1	28.4	36.3	33.4
LiveCodeBench (Pass@1)	37.6	28.4	28.7	30.1	32.8	34.2
Codeforces (Percentile)	51.6	35.6	24.8	25.3	20.3	23.6
SWE Verified (Resolved)	42.0	22.6	23.8	24.5	50.8	38.8
Aider-Edit (Acc.)	79.7	71.6	65.4	63.9	84.2	72.9
Aider-Polyglot (Acc.)	49.6	18.2	7.6	5.8	45.3	16.0
Math						
AIME 2024 (Pass@1)	39.2	16.7	23.3	23.3	16.0	9.3
MATH-500 (EM)	90.2	74.7	80.0	73.8	78.3	74.6
CNMO 2024 (Pass@1)	43.2	10.8	15.9	6.8	13.1	10.8
Chinese						
CLUEWSC (EM)	90.9	90.4	91.4	84.7	85.4	87.9
C-Eval (EM)	86.5	79.5	86.1	61.5	76.7	76.0
C-SimpleQA (Correct)	64.1	54.1	48.4	50.4	51.3	59.3

当前 DeepSeek 模型有多个版本，参数量从 1.5B、70B到671B不等，适用于不同的应用场景和计算资源。参数量越多，资源消耗越多，使用者可以根据自己实际需求选择不同模型版本。

注意

- 为维护良好的网络环境和公共秩序，禁止任何用户或机构利用天翼云平台从事违法活动、传播不良信息或实施违反社会公序良俗的行为。若发生此类行为，天翼云将依法采取必要措施，停止相关服务功能，并依法追究责任。
- 模型由DeepSeek公司提供，模型的结果仅供参考，并可能因不同的环境、数据或操作条件而有所不同，我们不对模型的任何结果负责。

快速体验DeepSeek

Ollama 是一个专为本地运行大语言模型（LLMs）而设计的工具，支持多种模型格式，并提供了简单易用的命令行接口。这里，我们以DeepSeek-r1-7b模型为例，讲解如何在天翼云使用Ollama运行 DeepSeek模型，读者可以根据自己实际需求修改云主机模型和云主机配置。

最佳实践

步骤一：创建GPU云主机

1. 进入创建云主机页面

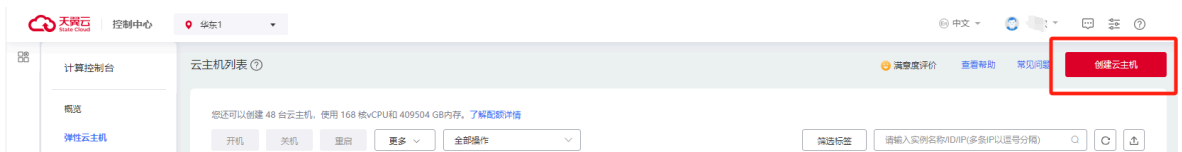
- a. 点击天翼云门户首页的“控制中心”，输入登录的用户名和密码，进入控制中心页面。



- b. 单击“产品服务列表>弹性云主机”，进入主机列表页。



- c. 单击“创建云主机”，进入弹性云主机创建页。



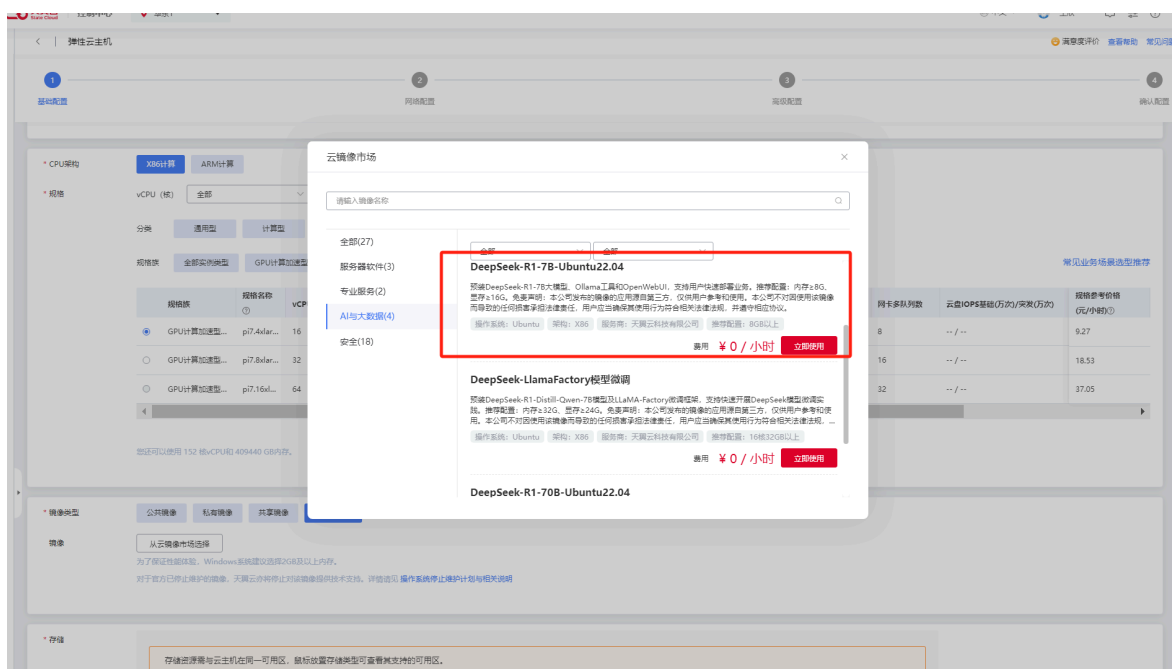
最佳实践

2. 进行基础配置

- 根据业务需求配置“计费模式”、“地域”、“企业项目”、“虚拟私有云”、“实例名称”、“主机名称”等。
- 选择规格。此处选择"CPU架构"为"X86"、"分类"为"GPU加速/AI加速型"、"规格族"为"GPU计算加速型 pi7"、"规格"为"pi7.4xlarge.4"。
- 选择镜像。“镜像类型”选择“镜像市场”，在云镜像市场中选择预置了DeepSeek R1模型的DeepSeek-R1-7B-Ubuntu22.04镜像。

注意

本镜像推荐配置：内存≥8G、显存≥16G。



注意

目前提供了预装多种量化精度、多种参数量的模型，如您有需求，也可在云镜像市场中进行选择。

- 设置云盘类型和大小。

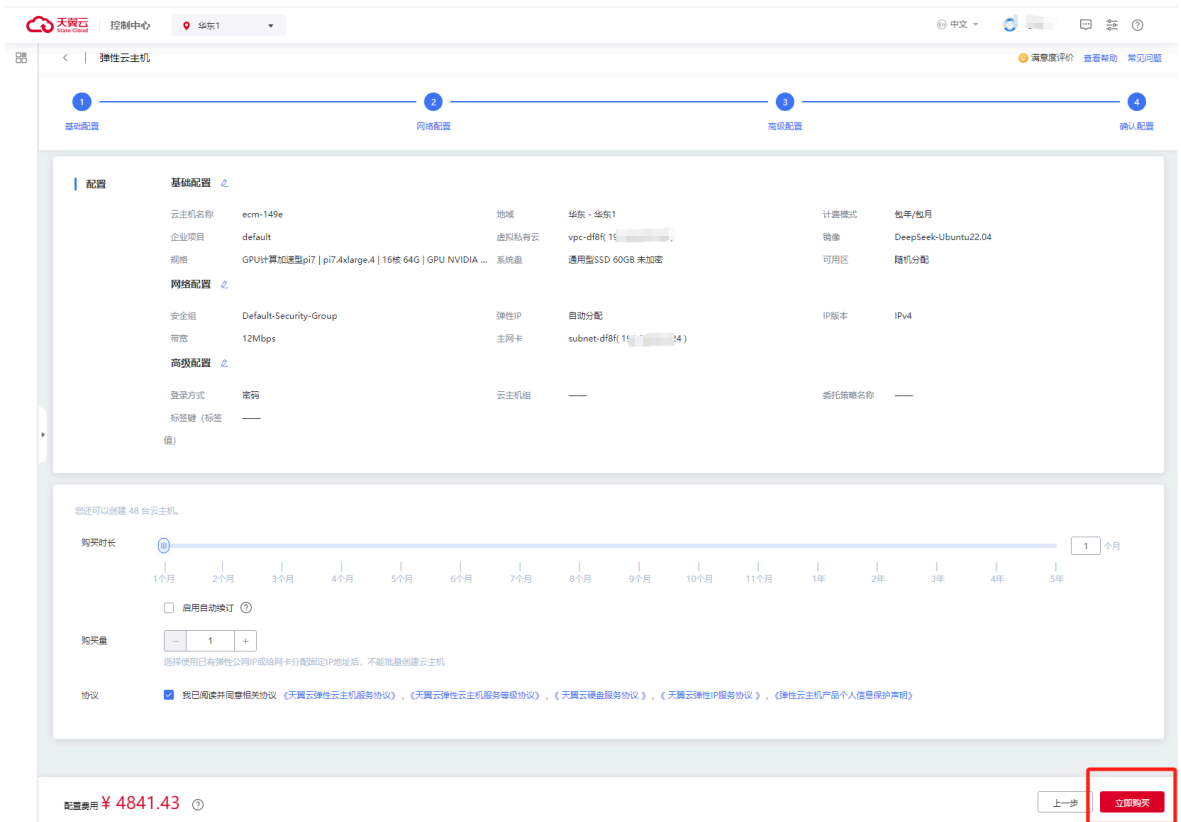
最佳实践



3. 网络及高级配置

设置网络，包括"网卡"、"安全组"，同时配备"弹性IP"用于下载和访问模型；设置高级配置，包括"登录方式"、"云主机组"、"用户数据"。

4. 确认并支付



最佳实践

步骤二：使用DeepSeek模型

1. 通过web界面进行模型交互

注意

镜像自带的 ollama 工具监听 127.0.0.1:11434、webui 监听 0.0.0.0:3000 端口，云主机默认不对外开放任何端口访问，请按需开放端口访问规则，避免数据泄露。

- 放行云主机安全组的 3000 端口。具体操作请参考[添加安全组规则-弹性云主机-用户指南-安全-安全组-配置安全组规则 - 天翼云](#)。

The screenshot shows a '添加规则' (Add Rule) dialog box with the following configuration:

- * IP版本: IPv4, IPv6
- * 方向: 入方向, 出方向
- * 授权策略: 允许, 拒绝
- * 优先级: [-] 1 [+] ⓘ
- * 协议: TCP (dropdown menu)
- * 端口范围: 3000
- * 源地址: IP, 安全组, 前缀列表
- 源地址输入框: 0.0.0.0/0 ⓘ
- 描述: (empty text box)

Buttons: 取消 (Cancel), 确定 (Confirm)

- 访问DeepSeek模型的可视化界面。登录地址为：`http://{公网ip地址}:3000`。

注意

云主机全自动安装DeepSeek模型和可视化界面，请等待云主机启动 5 分钟后，再访问登录界面。

首次登录页面如下：



c. 注册管理员账号。

01

开始使用 Open WebUI

① Open WebUI 不会与外部建立任何连接，您的数据会安全地存储在本地托管的服务器上。

名称

电子邮箱

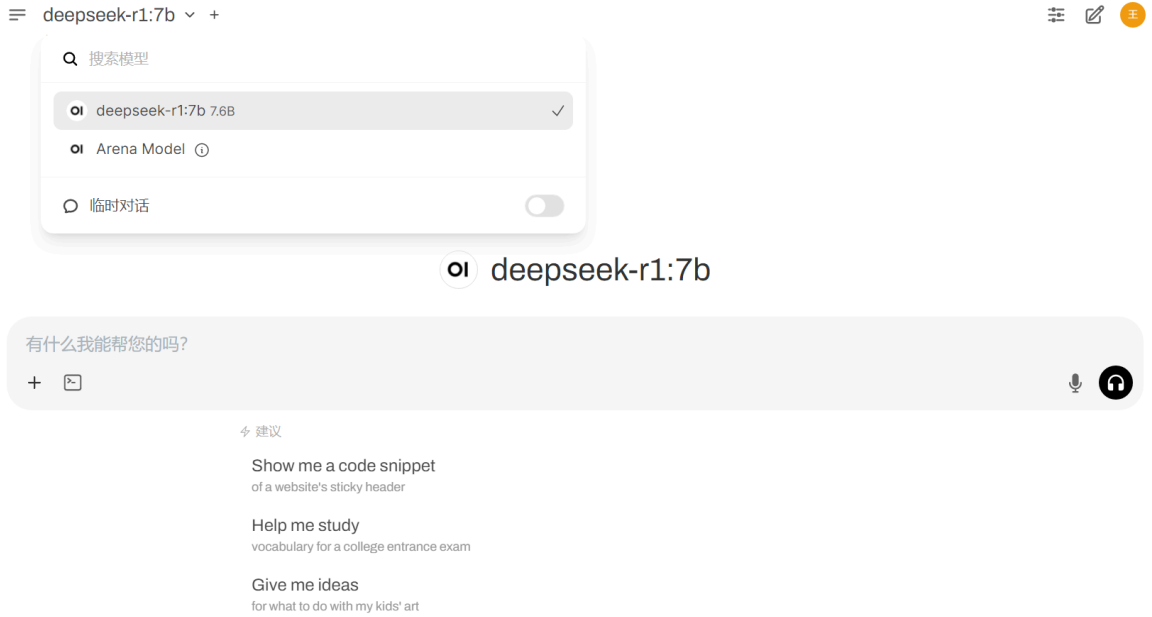
密码

创建管理员账号

d. 使用设置。

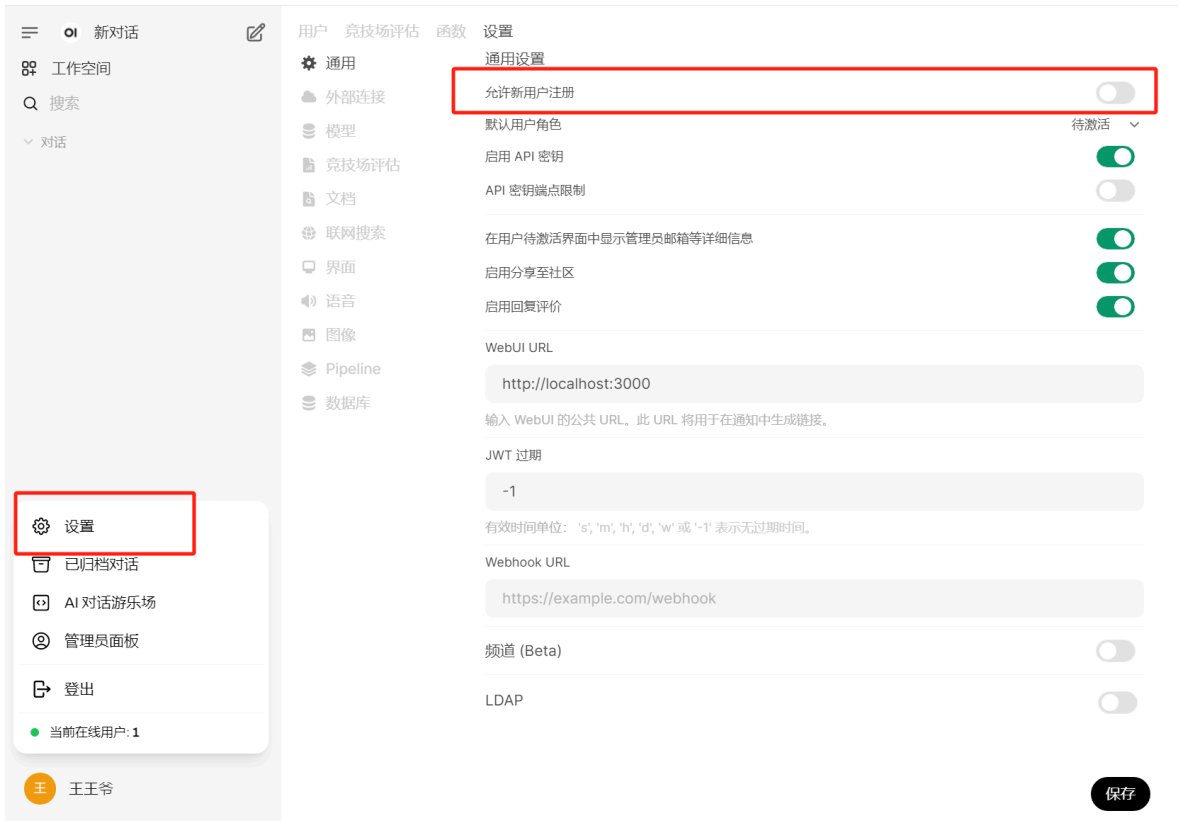
刷新进入首页，在模型下拉列表中，选择刚部署的DeepSeek:7b 模型。

最佳实践



点击左下角进入设置页面，如果您不想开放其他用户注册使用，则需要关闭“允许用户注册”功能。

最佳实践



如果您允许用户注册，还可以设置用户注册之后的行为，例如选择新用户注册后默认用户角色为“用户”/“待激活”等，需要管理员手动激活。

最佳实践

The screenshot shows a settings page with a sidebar on the left and a main content area on the right. The sidebar contains navigation options: 新对话 (New Conversation), 工作空间 (Workspace), 搜索 (Search), 对话 (Conversation), 用户 (User), 竞技场评估 (Arena Evaluation), 函数 (Function), and 设置 (Settings). The main content area is titled '设置' (Settings) and includes sections for '通用设置' (General Settings), 'WebUI URL', 'JWT 过期' (JWT Expiry), 'Webhook URL', '频道 (Beta)' (Channel Beta), and 'LDAP'. The '待激活' (Pending Activation) toggle switch is highlighted with a red box.

用户 竞技场评估 函数 设置

通用设置

- 允许新用户注册
- 默认用户角色
- 启用 API 密钥
- API 密钥端点限制
- 在用户待激活界面中显示管理员邮箱等详细信息
- 启用分享至社区
- 启用回复评价

WebUI URL

http://localhost:3000

输入 WebUI 的公共 URL。此 URL 将用于在通知中生成链接。

JWT 过期

-1

有效时间单位: 's', 'm', 'h', 'd', 'w' 或 '-1' 表示无过期时间。

Webhook URL

https://example.com/webhook

频道 (Beta)

LDAP

保存

设置模型可见性。多用户模式下，建议把模型设置为"Public"。

最佳实践

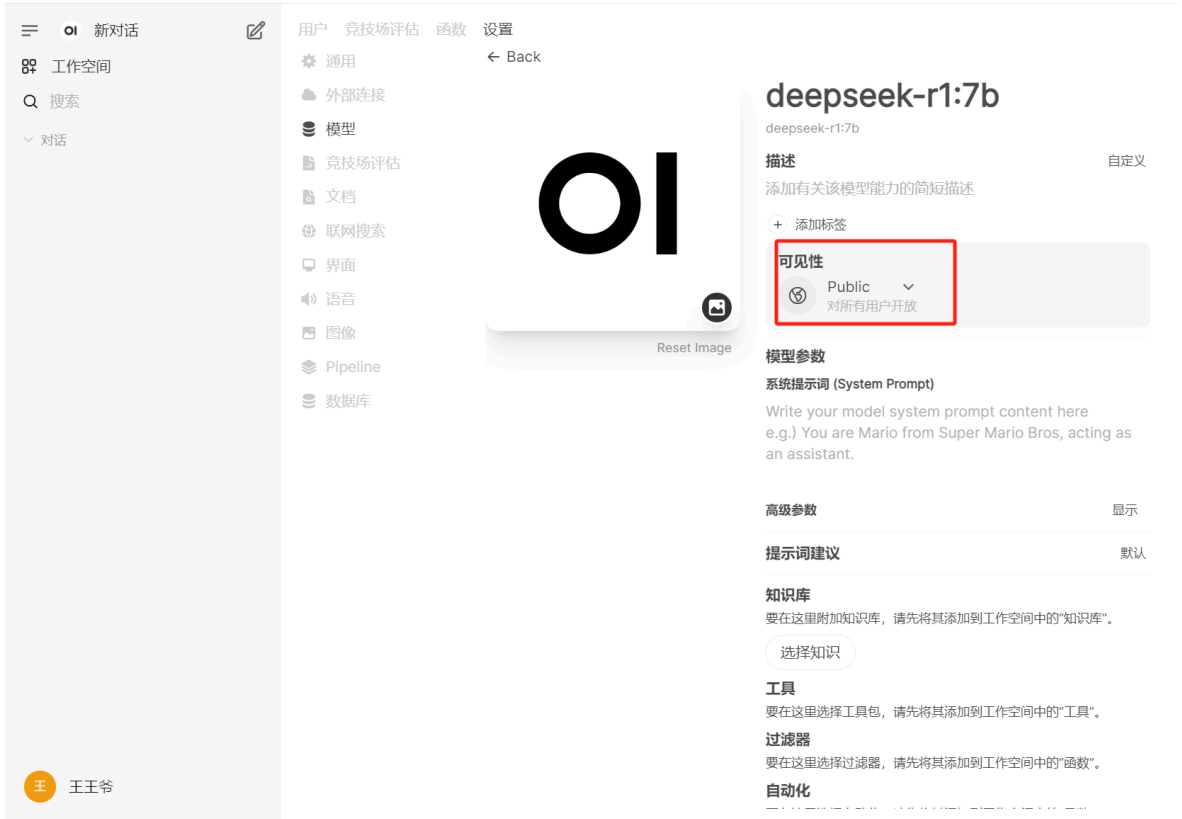
The screenshot displays a software interface with a sidebar on the left and a main content area on the right. The sidebar contains the following elements:

- 新对话 (New Conversation)
- 工作空间 (Workspace)
- 搜索 (Search)
- 对话 (Conversation)
- 王王爷 (User Profile)

The main content area is titled "模型 1" (Model 1) and includes the following components:

- 用户 竞技场评估 函数 设置 (User Arena Evaluation Function Settings)
- 通用 (General)
- 外部连接 (External Connections)
- 模型 (Model)
- 竞技场评估 (Arena Evaluation)
- 文档 (Documents)
- 联网搜索 (Web Search)
- 界面 (Interface)
- 语音 (Voice)
- 图像 (Image)
- Pipeline
- 数据库 (Database)

In the "模型" section, the model name "deepseek-r1:7b" is displayed. A red box highlights a pencil icon next to a green toggle switch, indicating that the model settings are being edited. Below this, there are two buttons: "导入预设" (Import Preset) and "导出预设" (Export Preset).



e. 使用DeepSeek模型进行模型推理。

最佳实践

☰ deepseek-r1:7b ▾ +
设为默认



🗨️ deepseek-r1:7b

有什么我能帮您的吗？



🔗 建议

Help me study

vocabulary for a college entrance exam

Grammar check

rewrite it for better readability

Give me ideas

for what to do with my kids' art

≡ deepseek-r1:7b v +

...  

您好，请帮我生成一段汉诺塔问题的Python脚本

OI deepseek-r1:7b

思考时间 几秒 v

当然可以！以下是汉诺塔问题的标准Python解决方案。这个脚本会演示如何将N个盘子从源柱移动到目标柱：

```
python 运行 保存 复制  
1 v def hanoi_tower(n, source, target, auxiliary):  
2 v     if n == 1:  
3         print(f"Move disk 1 from {source} to {target}")  
4 v     else:  
5         hanoi_tower(n-1, source, auxiliary, target)  
6         print(f"Move disk {n} from {source} to {target}")  
7         hanoi_tower(n-1, auxiliary, target, source)  
8  
9     # 用户可以根据需要调整的盘子数量  
10    n = 5  
11  
12    # 调用函数，将所有盘子从源柱移动到目标柱  
13    hanoi_tower(n, 'A', 'C', 'B')  
14  
15    # 输出结果会显示所有移动操作
```

输出解释：



输入消息

+ 



自定义部署DeepSeek

步骤一：规格选型

不同版本的模型参数量不同，参数量越多，资源消耗越多。不同规模的企业、客户有不同诉求，客户可根据自己的实际需求，选择所需镜像和模型版本。参数较少的DeepSeek模型可以在CPU上正常运行、如1.5B、7B模型，更大的模型需要GPU才能获得良好的体验。天翼云提供了不同模型的硬件配置建议，如下：

最佳实践

模型	推荐规格					
	通用云主机	T4	V100	V100S	A10	A100
DeepSeek-R1-1.5B	c7.2xlarge.2 (8C16G)	pi2.2xlarge.2 (显存: 1*16G)	2v.2xlarge.4 (显存: 1*32G)	2vs.2xlarge.4 (显存: 1*32G)	pi7.4xlarge.4 (显存: 1*24G)	p8a.6xlarge.4 (显存: 1*40G)
DeepSeek-R1-7B	c7.4xlarge.2 (16C32G)	pi2.4xlarge.2 (显存: 2*16G)	2v.2xlarge.4 (显存: 1*32G)	2vs.2xlarge.4 (显存: 1*32G)	pi7.4xlarge.4 (显存: 1*24G)	p8a.6xlarge.4 (显存: 1*40G)
DeepSeek-R1-8B	-	pi2.4xlarge.2 (显存: 2*16G)	2v.2xlarge.4 (显存: 1*32G)	2vs.2xlarge.4 (显存: 1*32G)	pi7.4xlarge.4 (显存: 1*24G)	p8a.6xlarge.4 (显存: 1*40G)
DeepSeek-R1-14B	-	pi2.8xlarge.2 (显存: 4*16G)	2v.4xlarge.4 (显存: 2*32G)	2vs.4xlarge.4 (显存: 2*32G)	pi7.8xlarge.4 (显存: 2*24G)	p8a.12xlarge.4 (显存: 2*40G)
DeepSeek-R1-32B	-	-	p2v.8xlarge.4 (显存: 4*32G)	2vs.8xlarge.4 (显存: 4*32G)	pi7.16xlarge.4 (显存: 4*24G)	p8a.24xlarge.4 (显存: 4*40G)
DeepSeek-R1-70B	-	-	-	-	-	p8a.24xlarge.4 (显存: 4*40G)

对应的GPU云主机规格请参见[NVIDIA GPU云主机-GPU云主机-产品简介-产品规格 - 天翼云](#)。

步骤二：云主机开通

[创建弹性云主机-弹性云主机-快速入门 - 天翼云](#)

[创建配备GPU驱动的GPU云主机 \(Linux\) -GPU云主机-用户指南-创建GPU云主机 - 天翼云](#)

步骤三：手动部署DeepSeek

1. 安装ollama

a. 下载安装脚本

使用非天翼云DeepSeek镜像时，需要手动安装ollama，这里以 amd64 为例，将如下脚本保存到本地，命名为 install_ollama.sh。

```
#!/bin/sh
# This script installs Ollama on Linux.
# It detects the current operating system architecture and installs the appropriate version of Ollama.

set -eu

red="$( (/usr/bin/tput bold || :) /usr/bin/tput setaf 1 || :) 2>&-)"
plain="$( (/usr/bin/tput sgr0 || :) 2>&-)"

status() { echo ">>> $" >&2; }
error() { echo "${red}ERROR:${plain} $" ; exit 1; }
warning() { echo "${red}WARNING:${plain} $" ; }
```

```
TEMP_DIR=$(mktemp -d)
cleanup() { rm -rf $TEMP_DIR; }
trap cleanup EXIT

available() { command -v $1 >/dev/null; }
require() {
    local MISSING=""
    for TOOL in $*; do
        if ! available $TOOL; then
            MISSING="$MISSING $TOOL"
        fi
    done

    echo $MISSING
}

["$(uname -s)" = "Linux"] || error 'This script is intended to run on Linux only.'

ARCH=$(uname -m)
case "$ARCH" in
    x86_64) ARCH="amd64" ;;
    aarch64|arm64) ARCH="arm64" ;;
    *) error "Unsupported architecture: $ARCH" ;;
esac

IS_WSL2=false

KERN=$(uname -r)
case "$KERN" in
    *icrosoft*WSL2 | *icrosoft*ws12) IS_WSL2=true;;
    *icrosoft) error "Microsoft WSL1 is not currently supported. Please use WSL2 with 'wsl --set-version <distro> 2" ;;
    *) ;;
esac

VER_PARAM="{OLLAMA_VERSION:+?version=$OLLAMA_VERSION}"

SUDO=
if [ "$(id -u)" -ne 0 ]; then
    # Running as root, no need for sudo
    if ! available sudo; then
        error "This script requires superuser permissions. Please re-run as root."
    fi

    SUDO="sudo"
fi
```

最佳实践

```
NEEDS=$(require curl awk grep sed tee xargs)
if [ -n "$NEEDS" ]; then
    status "ERROR: The following tools are required but missing:"
    for NEED in $NEEDS; do
        echo " - $NEED"
    done
    exit 1
fi

for BINDIR in /usr/local/bin /usr/bin /bin; do
    echo $PATH | grep -q $BINDIR && break || continue
done
OLLAMA_INSTALL_DIR=$(dirname ${BINDIR})

if [ -d "$OLLAMA_INSTALL_DIR/lib/ollama" ]; then
    status "Cleaning up old version at $OLLAMA_INSTALL_DIR/lib/ollama"
    $SUDO rm -rf "$OLLAMA_INSTALL_DIR/lib/ollama"
fi
status "Installing ollama to $OLLAMA_INSTALL_DIR"
$SUDO install -o0 -g0 -m755 -d $BINDIR
$SUDO install -o0 -g0 -m755 -d "$OLLAMA_INSTALL_DIR"
status "Downloading Linux ${ARCH} bundle"
curl --fail --show-error --location --progress-bar \
    "https://mirrors.ctyun.cn/ollama/v0.5.7/ollama-linux-${ARCH}.tgz${VER_PARAM}" | \
    $SUDO tar -xzf - -C "$OLLAMA_INSTALL_DIR"
if [ "$OLLAMA_INSTALL_DIR/bin/ollama" != "$BINDIR/ollama" ]; then
    status "Making ollama accessible in the PATH in $BINDIR"
    $SUDO ln -sf "$OLLAMA_INSTALL_DIR/ollama" "$BINDIR/ollama"
fi

# Check for NVIDIA JetPack systems with additional downloads
if [ -f /etc/nv_tegra_release ]; then
    if grep R36 /etc/nv_tegra_release > /dev/null ; then
        status "Downloading JetPack 6 components"
        curl --fail --show-error --location --progress-bar \
            "https://mirrors.ctyun.cn/ollama/v0.5.7/ollama-linux-${ARCH}-jetpack6.tgz${VER_PARAM}" | \
            $SUDO tar -xzf - -C "$OLLAMA_INSTALL_DIR"
    elif grep R35 /etc/nv_tegra_release > /dev/null ; then
        status "Downloading JetPack 5 components"
        curl --fail --show-error --location --progress-bar \
            "https://mirrors.ctyun.cn/ollama/v0.5.7/ollama-linux-${ARCH}-jetpack5.tgz${VER_PARAM}" | \
            $SUDO tar -xzf - -C "$OLLAMA_INSTALL_DIR"
    else
```

最佳实践

```
warning "Unsupported JetPack version detected. GPU may not be supported"
fi
fi

install_success() {
    status 'The Ollama API is now available at 127.0.0.1:11434.'
    status 'Install complete. Run "ollama" from the command line.'
}
trap install_success EXIT

# Everything from this point onwards is optional.

configure_systemd() {
    if ! id ollama >/dev/null 2>&1; then
        status "Creating ollama user..."
        $SUDO useradd -r -s /bin/false -U -m -d /usr/share/ollama ollama
    fi
    if getent group render >/dev/null 2>&1; then
        status "Adding ollama user to render group..."
        $SUDO usermod -a -G render ollama
    fi
    if getent group video >/dev/null 2>&1; then
        status "Adding ollama user to video group..."
        $SUDO usermod -a -G video ollama
    fi

    status "Adding current user to ollama group..."
    $SUDO usermod -a -G ollama $(whoami)

    status "Creating ollama systemd service..."
    cat <<EOF | $SUDO tee /etc/systemd/system/ollama.service >/dev/null
[Unit]
Description=Ollama Service
After=network-online.target

[Service]
ExecStart=$BINDIR/ollama serve
User=ollama
Group=ollama
Restart=always
RestartSec=3
Environment="PATH=$PATH"

[Install]
WantedBy=default.target
EOF
    SYSTEMCTL_RUNNING="$ (systemctl is-system-running || true)"
```

```
case $SYSTEMCTL_RUNNING in
  running|degraded)
    status "Enabling and starting ollama service..."
    $SUDO systemctl daemon-reload
    $SUDO systemctl enable ollama

    start_service() { $SUDO systemctl restart ollama; }
    trap start_service EXIT
    ;;
  *)
    warning "systemd is not running"
    if [ "$IS_WSL2" = true ]; then
      warning "see https://learn.microsoft.com/en-us/windows/wsl/systemd#how-to-enable-systemd-to-enable-it"
    fi
    ;;
esac
}

if available systemctl; then
  configure_systemd
fi

# WSL2 only supports GPUs via nvidia passthrough
# so check for nvidia-smi to determine if GPU is available
if [ "$IS_WSL2" = true ]; then
  if available nvidia-smi && [ -n "$(nvidia-smi | grep -o "CUDA Version: [0-9]*\.[0-9]*")" ]; then
    status "Nvidia GPU detected."
  fi
  install_success
  exit 0
fi

# Don't attempt to install drivers on Jetson systems
if [ -f /etc/nv_tegra_release ]; then
  status "NVIDIA JetPack ready."
  install_success
  exit 0
fi

# Install GPU dependencies on Linux
if ! available lspci && ! available lshw; then
  warning
  "Unable to detect NVIDIA/AMD GPU. Install lspci or lshw to automatically detect and install GPU dependencies."
  exit 0
fi
```

最佳实践

```
check_gpu() {
    # Look for devices based on vendor ID for NVIDIA and AMD
    case $1 in
        lspci)
            case $2 in
                nvidia) available lspci && lspci -d '10de:' | grep -q 'NVIDIA' || return 1 ;;
                amdgpu) available lspci && lspci -d '1002:' | grep -q 'AMD' || return 1 ;;
            esac ;;
        lshw)
            case $2 in
                nvidia) available lshw && $SUDO lshw -c display -numeric -disable network | grep -q
'vendor: *\[10DEV\]' || return 1 ;;
                amdgpu) available lshw && $SUDO lshw -c display -numeric -disable network | grep -q
'vendor: *\[1002\]' || return 1 ;;
            esac ;;
        nvidia-smi) available nvidia-smi || return 1 ;;
    esac
}

if check_gpu nvidia-smi; then
    status "NVIDIA GPU installed."
    exit 0
fi

if ! check_gpu lspci nvidia && ! check_gpu lshw nvidia && ! check_gpu lspci amdgpu
&& ! check_gpu lshw amdgpu; then
    install_success
    warning "No NVIDIA/AMD GPU detected. Ollama will run in CPU-only mode."
    exit 0
fi

if check_gpu lspci amdgpu || check_gpu lshw amdgpu; then
    status "Downloading Linux ROCm ${ARCH} bundle"
    curl --fail --show-error --location --progress-bar \
        "https://mirrors.ctyun.cn/ollama/v0.5.7/ollama-linux-${ARCH}-rocm.tgz${VER_PARAM}" | \
        $SUDO tar -xzf - -C "$OLLAMA_INSTALL_DIR"

    install_success
    status "AMD GPU ready."
    exit 0
fi

CUDA_REPO_ERR_MSG="NVIDIA GPU detected, but your OS and Architecture are not supported by NVIDIA.
Please install the CUDA driver manually https://docs.nvidia.com/cuda/cuda-installation-
guide-linux/"
# ref: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#rhel-7-centos-7
```

最佳实践

```
# ref: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#rhel-8-rocky-8
# ref: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#rhel-9-rocky-9
# ref: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#fedora
install_cuda_driver_yum() {
    status 'Installing NVIDIA repository...'

    case $PACKAGE_MANAGER in
        yum)
            $SUDO $PACKAGE_MANAGER -y install yum-utils
            if curl -I --silent --fail --location "https://developer.download.nvidia.
com/compute/cuda/repos/$1$2/$(uname -m | sed -e 's/aarch64/sbsa')/cuda-$1$2.repo" >/dev/null
; then
                $SUDO $PACKAGE_MANAGER config-manager --add-repo https://developer.download.
nvidia.com/compute/cuda/repos/$1$2/$(uname -m | sed -e 's/aarch64/sbsa')/cuda-$1$2.repo
            else
                error $CUDA_REPO_ERR_MSG
            fi
            ;;
        dnf)
            if curl -I --silent --fail --location "https://developer.download.nvidia.
com/compute/cuda/repos/$1$2/$(uname -m | sed -e 's/aarch64/sbsa')/cuda-$1$2.repo" >/dev/null
; then
                $SUDO $PACKAGE_MANAGER config-manager --add-repo https://developer.download.nvidia.
com/compute/cuda/repos/$1$2/$(uname -m | sed -e 's/aarch64/sbsa')/cuda-$1$2.repo
            else
                error $CUDA_REPO_ERR_MSG
            fi
            ;;
    esac

    case $1 in
        rhel)
            status 'Installing EPEL repository...'
            # EPEL is required for third-party dependencies such as dkms and libvdpa
            $SUDO $PACKAGE_MANAGER -y install https://dl.fedoraproject.org/pub/epel/epel-release-
latest-$2.noarch.rpm || true
            ;;
    esac

    status 'Installing CUDA driver...'

    if [ "$1" = 'centos' ] || [ "$1$2" = 'rhel7' ]; then
        $SUDO $PACKAGE_MANAGER -y install nvidia-driver-latest-dkms
    fi

    $SUDO $PACKAGE_MANAGER -y install cuda-drivers
}
```

最佳实践

```
# ref: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#ubuntu
# ref: https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#debian
install_cuda_driver_apt() {
    status 'Installing NVIDIA repository...'
    if curl -I --silent --fail --location "https://developer.download.nvidia.
com/compute/cuda/repos/$1$2/${uname -m | sed -e 's/aarch64/sbsa'}/cuda-keyring_1.1-1_all.deb"
>/dev/null ; then
        curl -fsSL -o $TEMP_DIR/cuda-keyring.deb https://developer.download.nvidia.
com/compute/cuda/repos/$1$2/${uname -m | sed -e 's/aarch64/sbsa'}/cuda-keyring_1.1-1_all.deb
        else
            error $CUDA_REPO_ERR_MSG
        fi

    case $1 in
        debian)
            status 'Enabling contrib sources...'
            $SUDO sed 's/main/contrib' < /etc/apt/sources.list | $SUDO tee
/etc/apt/sources.list.d/contrib.list > /dev/null
            if [ -f "/etc/apt/sources.list.d/debian.sources" ]; then
                $SUDO sed 's/main/contrib' < /etc/apt/sources.list.d/debian.sources | $SUDO tee
/etc/apt/sources.list.d/contrib.sources > /dev/null
            fi
            ;;
        esac

    status 'Installing CUDA driver...'
    $SUDO dpkg -i $TEMP_DIR/cuda-keyring.deb
    $SUDO apt-get update

    [ -n "$SUDO" ] && SUDO_E="$SUDO -E" || SUDO_E=
    DEBIAN_FRONTEND=noninteractive $SUDO_E apt-get -y install cuda-drivers -q
}

if [ ! -f "/etc/os-release" ]; then
    error "Unknown distribution. Skipping CUDA installation."
fi

. /etc/os-release

OS_NAME=$ID
OS_VERSION=$VERSION_ID

PACKAGE_MANAGER=
for PACKAGE_MANAGER in dnf yum apt-get; do
    if available $PACKAGE_MANAGER; then
        break
    fi
done
```

最佳实践

```
fi
done

if [ -z "$PACKAGE_MANAGER" ]; then
    error "Unknown package manager. Skipping CUDA installation."
fi

if ! check_gpu nvidia-smi || [ -z "$(nvidia-smi | grep -o "CUDA Version: [0-9]*\.[0-9]*")" ]; then
    case $OS_NAME in
        centos|rhel) install_cuda_driver_yum 'rhel' $(echo $OS_VERSION | cut -d '.' -f 1) ;;
        rocky) install_cuda_driver_yum 'rhel' $(echo $OS_VERSION | cut -c1) ;;
        fedora) [ $OS_VERSION -lt '39' ] && install_cuda_driver_yum $OS_NAME $OS_VERSION
    || install_cuda_driver_yum $OS_NAME '39';
        amzn) install_cuda_driver_yum 'fedora' '37' ;;
        debian) install_cuda_driver_apt $OS_NAME $OS_VERSION ;;
        ubuntu) install_cuda_driver_apt $OS_NAME $(echo $OS_VERSION | sed 's/\./') ;;
        *) exit ;;
    esac
fi

if ! lsmod | grep -q nvidia || ! lsmod | grep -q nvidia_uvm; then
    KERNEL_RELEASE="$(uname -r)"
    case $OS_NAME in
        rocky) $SUDO $PACKAGE_MANAGER -y install kernel-devel kernel-headers ;;
        centos|rhel|amzn) $SUDO $PACKAGE_MANAGER -y install kernel-
devel-$KERNEL_RELEASE kernel-headers-$KERNEL_RELEASE ;;
        fedora) $SUDO $PACKAGE_MANAGER -y install kernel-devel-$KERNEL_RELEASE ;;
        debian|ubuntu) $SUDO apt-get -y install linux-headers-$KERNEL_RELEASE ;;
        *) exit ;;
    esac

    NVIDIA_CUDA_VERSION=$(($SUDO dkms status | awk -F: '/added/ { print $1 }')
    if [ -n "$NVIDIA_CUDA_VERSION" ]; then
        $SUDO dkms install $NVIDIA_CUDA_VERSION
    fi

    if lsmod | grep -q nouveau; then
        status 'Reboot to complete NVIDIA CUDA driver install.'
        exit 0
    fi

    $SUDO modprobe nvidia
    $SUDO modprobe nvidia_uvm
fi

# make sure the NVIDIA modules are loaded on boot with nvidia-persistenced
```

最佳实践

```
if available nvidia-persistenced; then
  $SUDO touch /etc/modules-load.d/nvidia.conf
  MODULES="nvidia nvidia-uvmm"
  for MODULE in $MODULES; do
    if ! grep -qxF "$MODULE" /etc/modules-load.d/nvidia.conf; then
      echo "$MODULE" | $SUDO tee -a /etc/modules-load.d/nvidia.conf > /dev/null
    fi
  done
fi
```

```
status "NVIDIA GPU ready."
install_success
```

b. 执行安装

```
bash install_ollama.sh
```

如下显示则为安装成功。

```
root@iZ2vcj7pha89cupf784r4nZ:~# bash install_ollama.sh
>>> Installing ollama to /usr/local
>>> Downloading Linux amd64 bundle
##### 100.0%
>>> Adding ollama user to render group...
>>> Adding ollama user to video group...
>>> Adding current user to ollama group...
>>> Creating ollama systemd service...
>>> Enabling and starting ollama service...
>>> The Ollama API is now available at 127.0.0.1:11434.
>>> Install complete. Run "ollama" from the command line.
WARNING: No NVIDIA/AMD GPU detected. Ollama will run in CPU-only mode.
root@iZ2vcj7pha89cupf784r4nZ:~# ollama ps
```

c. 测试ollama服务安装情况

```
ollama ps
```

```
root@iZ2vcj7pha89cupf784r4nZ:~# ollama ps
NAME          ID          SIZE          PROCESSOR          UNTIL
```

2. 下载模型

天翼云[镜像站](#)也为常见镜像提供了加速能力，包括1.5B\7B\14B\70B\671B等相关模型，可以通过如下手段体验其他模型，以14B模型为例：

```
mkdir deepseek-r1-14b
```

```
cd deepseek-r1-14b
```

```
wget https://mirrors.ctyun.cn/ollama/models/deepseek-r1-14b/deepseek-r1-14b.gguf
```

```
wget https://mirrors.ctyun.cn/ollama/models/deepseek-r1-14b/ModelFile
```

最佳实践

```
ollama create deepseek-r1:14b -f ModelFile
```

```
root@test-deepseek-1:~# ollama create deepseek-r1:14b ModelFile
Error: accepts 1 arg(s), received 2
root@test-deepseek-1:~# ollama create deepseek-r1:14b -f ModelFile
gathering model components
copying file sha256:6e9f90f02bb3b39b59e81916e8cfce9deb45aeaeb9a54a5be4414486b907dc1e 100%
parsing GGUF
using existing layer sha256:6e9f90f02bb3b39b59e81916e8cfce9deb45aeaeb9a54a5be4414486b907dc1e
using existing layer sha256:369ca498f347f710d068cbb38bf0b8692dd3fa30f30ca2ff755e211c94768150
using existing layer sha256:f4d24e9138dd4603380add165d2b0d970bef471fac194b436ebd50e6147c6588
writing manifest
success
root@test-deepseek-1:~# ollama list
NAME                ID                SIZE      MODIFIED
deepseek-r1:14b    67711fd7c370     9.0 GB    4 seconds ago
deepseek-r1:7b     0a8c26691023     4.7 GB    3 days ago
```

如果你想体验其他模型，也可以直接通过ollama下载。

```
ollama pull deepseek-r1:14b
```

3. 使用ollama运行模型

```
ollama run deepseek-r1:14b
```

4. 通过web界面进行交互

a. 安装open-webui

```
docker pull ghcr.io/open-webui/open-webui:main
```

b. 启动容器

```
docker run -d --net=host -e PORT=3000 -e
OLLAMA_BASE_URL=http://127.0.0.1:11434 -e
ENABLE_SIGNUP=true -e ENABLE_OPENAI_API=False -v open-webui:/app/backend/data --name
open-webui --restart always ghcr.io/open-webui/open-webui:main
```

后续请参考上文“快速体验DeepSeek——步骤二：使用deepseek模型”。

FAQ

盘不够了，我应该怎么办？

如果您在模型部署过程中发现云盘的容量不够，可以采取如下措施：

1. 根据[云硬盘扩容概述-云硬盘-用户指南-扩容云硬盘](#) - 天翼云对已有云盘进行扩容。
2. 新建一块数据盘并挂载，相关操作见[挂载云硬盘-云硬盘-快速入门](#) - 天翼云、[初始化数据盘-弹性云主机-快速入门](#) - 天翼云。

如何修改ollama模型的存储位置？

在linux环境下，ollama默认模型存储目录是 /usr/share/ollama/.ollama/models/，我们建议您使用云硬盘独立挂载数据盘，将模型存储到数据盘中。模型存储位置是由环境变量控制的，我们需要修改ollama的环境变量重启服务才能修改存储目录，我们以 /data/ollama/models 目录为例：

最佳实践

1. 打开 ollama.service 文件

```
vi /etc/systemd/system/ollama.service
```

注意

请确保 ollama 用户组中的 ollama 用户具备访问该目录的读写权限。

2. 新增相关环境变量

```
[Unit]
Description=Ollama Service
After=network-online.target

[Service]
ExecStart=/usr/local/bin/ollama serve
User=ollama
Group=ollama
Restart=always
RestartSec=3
Environment="PATH=/usr/local/cuda/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin"
Environment="OLLAMA_MODELS=/data/ollama/models"

[Install]
WantedBy=default.target
```

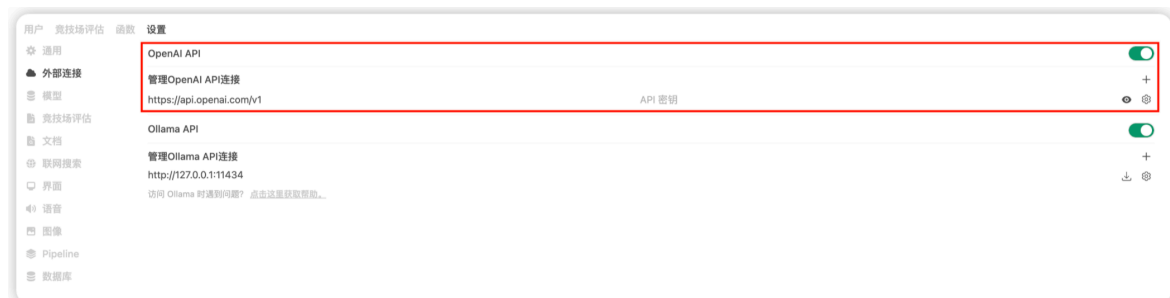
3. 重启服务

```
systemctl daemon-reload
systemctl restart ollama
```

使用Ollama+Open WebUI 部署的镜像，登录后页面白屏等待很长时间才能进入对话页面，该怎么办？

问题描述

1. 使用Ollama+openWebUI部署的DeepSeek公共镜像，默认会关闭OpenAI API访问，但是如果开启OpenAI API如下图所示，会导致公网访问WebUI界面加载、登录都很慢，甚至页面显示空白。



2. 当通过Open WebUI登录后，会去访问默认配置的OpenAI境外API链接，最终超时连接失败。

```
INFO [open_webui.routers.openai] get_all_models()
ERROR [open_webui.routers.openai] Connection error: Cannot connect to host api.openai.com:443 ssl:default [Multiple exceptions: [Errno 110] Connect call failed ('2a03:2880:f31:83:face:b00c:0:25de', 443, 0, 0), [Errno 110] Connect call failed ('157.248.16.59', 443)]
```

解决方法

使用Ollama部署的情况下，您可以在“管理员面板-设置-外部链接”中，关闭OpenAI API即可。如果您有需要使用的外部API连接，也可以按需配置API连接，并配置API密钥。

在天翼云使用vLLM运行DeepSeek的最佳实践-32B等版本

什么是DeepSeek

DeepSeek 是一个基于 Transformer 架构的大型语言模型（LLM），由深度求索（DeepSeek）公司开发。它能够处理自然语言理解、生成、翻译、问答等多种任务。在目前大模型主流榜单中，DeepSeek-V3 在开源模型中位列榜首，与世界上最先进的闭源模型不分伯仲。在对话模型典型任务方面的评测效果如下：

最佳实践

	DeepSeek V3	DeepSeek V2.5	Qwen2.5	Llama3.1	Claude-3.5	GPT-4o
Benchmark (Metric)		0905	72B-Inst	405B-Inst	Sonnet-1022	0513
Architecture	MoE	MoE	Dense	Dense	-	-
# Activated Params	37B	21B	72B	405B	-	-
# Total Params	671B	236B	72B	405B	-	-
English						
MMLU (EM)	88.5	80.6	85.3	88.6	88.3	87.2
MMLU-Redux (EM)	89.1	80.3	85.6	86.2	88.9	88.0
MMLU-Pro (EM)	75.9	66.2	71.6	73.3	78.0	72.6
DROP (3-shot F1)	91.6	87.8	76.7	88.7	88.3	83.7
IF-Eval (Prompt Strict)	86.1	80.6	84.1	86.0	86.5	84.3
GPQA-Diamond (Pass@1)	59.1	41.3	49.0	51.1	65.0	49.9
SimpleQA (Correct)	24.9	10.2	9.1	17.1	28.4	38.2
FRAMES (Acc.)	73.3	65.4	69.8	70.0	72.5	80.5
LongBench v2 (Acc.)	48.7	35.4	39.4	36.1	41.0	48.1
Code						
HumanEval-Mul (Pass@1)	82.6	77.4	77.3	77.2	81.7	80.5
LiveCodeBench (Pass@1-COT)	40.5	29.2	31.1	28.4	36.3	33.4
LiveCodeBench (Pass@1)	37.6	28.4	28.7	30.1	32.8	34.2
Codeforces (Percentile)	51.6	35.6	24.8	25.3	20.3	23.6
SWE Verified (Resolved)	42.0	22.6	23.8	24.5	50.8	38.8
Aider-Edit (Acc.)	79.7	71.6	65.4	63.9	84.2	72.9
Aider-Polyglot (Acc.)	49.6	18.2	7.6	5.8	45.3	16.0
Math						
AIME 2024 (Pass@1)	39.2	16.7	23.3	23.3	16.0	9.3
MATH-500 (EM)	90.2	74.7	80.0	73.8	78.3	74.6
CNMO 2024 (Pass@1)	43.2	10.8	15.9	6.8	13.1	10.8
Chinese						
CLUEWSC (EM)	90.9	90.4	91.4	84.7	85.4	87.9
C-Eval (EM)	86.5	79.5	86.1	61.5	76.7	76.0
C-SimpleQA (Correct)	64.1	54.1	48.4	50.4	51.3	59.3

当前 DeepSeek 模型有多个版本，参数量从 1.5B、70B 到 671B 不等，适用于不同的应用场景和计算资源。参数量越多，资源消耗越多，使用者可以根据自己实际需求选择不同模型版本。

注意

为维护良好的网络环境和公共秩序，天翼云平台特规定如下使用条款：禁止任何用户或机构利用平台进行违法活动、传播不良信息或违反社会公序良俗的行为。若发生此类行为，天翼云将依法采取必要措施，停止相关服务功能，并依法追究责任。

模型由DeepSeek公司提供。模型的结果仅供参考，并可能因不同的环境、数据或操作条件而有所不同，我们不对模型的任何结果负责。

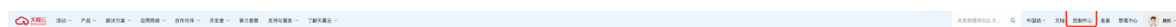
快速体验DeepSeek

vLLM 是一个专为本地运行大语言模型（LLMs）而设计的工具，并提供了简单易用的命令行接口。这里，我们以 Deepseek-R1-32B模型为例，讲解如何在天翼云使用vLLM运行 DeepSeek模型，读者可以根据自己实际需求修改云主机模型和云主机配置。

步骤一：创建GPU云主机

1. 进入创建云主机页面

点击天翼云门户首页的“控制中心”，输入登录的用户名和密码，进入控制中心页面。



单击“服务列表>弹性云主机”，进入主机列表页。



单击“创建云主机”，进入弹性云主机创建页。

最佳实践

2. 进行基础配置

基础配置：根据业务需求配置“计费模式”、“地域”、“企业项目”、“虚拟私有云”、“实例名称”、“主机名称”等。

选择规格：此处选择“CPU架构”为“X86”、“分类”为“GPU加速/AI加速型”、“规格族”为“GPU计算加速型pn8r”、“规格”为“pn8r.16xlarge.4”。

选择镜像：“镜像类型”选择“镜像市场”，在云镜像市场中选择预置了DeepSeek R1模型的DeepSeek-R1-32B-vLLM-Ubuntu22.04镜像。



设置系统盘：

• 存储



3. 网络及高级配置。

设置网络，包括“网卡”、“安全组”，同时配备“弹性IP”用于访问模型；设置高级配置，包括“登录方式”、“云主机组”、“用户数据”。

4. 确认并支付

步骤二：使用DeepSeek模型

通过云主机提供的Open WebUI直接体验大模型，具体请参考[Open WebUI快速入门](#)。

最佳实践

注意

镜像自带的ollama工具监听127.0.0.1:11434、webui 监听 0.0.0.0:3000 端口，云主机默认不对外开放任何端口访问，请按需开放端口访问规则，避免数据泄露。

自定义部署DeepSeek

步骤一：规格选型

不同版本的模型参数量不同，参数量越多，资源消耗越多。不同规模的企业、客户有不同诉求，客户可根据自己的实际需求，选择所需镜像和模型版本。参数较少的DeepSeek模型可以在CPU上正常运行、如1.5B、7B模型，更大的模型需要GPU才能获得良好的体验。天翼云提供了不同模型的硬件配置建议，如下：

模型	推荐规格					
	通用云主机	T4	V100	V100S	A10	A100
DeepSeek-R1-1.5B	c7.2xlarge.2	pi2.2xlarge.4	p2v.2xlarge.4	p2vs.2xlarge.4	g7.4xlarge.4	a8e.6xlarge.4
DeepSeek-R1-7B	c7.4xlarge.2	pi2.4xlarge.4	p2v.2xlarge.4	p2vs.2xlarge.4	g7.4xlarge.4	a8e.6xlarge.4
DeepSeek-R1-8B	-	pi2.4xlarge.4	p2v.2xlarge.4	p2vs.2xlarge.4	g7.4xlarge.4	a8e.6xlarge.4
DeepSeek-R1-14B	-	pi2.8xlarge.4	p2v.4xlarge.4	p2vs.4xlarge.4	g7.8xlarge.4	a8e.12xlarge.4
DeepSeek-R1-32B	-	-	p2v.8xlarge.4	p2vs.8xlarge.4	g7.16xlarge.4	a8e.24xlarge.4
DeepSeek-R1-70B	-	-	-	-	-	p8a.24xlarge.4

对应的GPU云主机规格请参见[NVIDIA GPU云主机-GPU云主机-产品简介-产品规格 - 天翼云](#)。

步骤二：云主机开通

[创建弹性云主机-弹性云主机-快速入门 - 天翼云](#)

[创建配备GPU驱动的GPU云主机（Linux）-GPU云主机-用户指南-创建GPU云主机 - 天翼云](#)

步骤三：手动部署DeepSeek

建议使用天翼云提供的DeepSeek镜像，减少安装过程中可能遇到的问题。如您必须通过vLLM部署模型，请参考如下步骤：

1. 安装依赖包

安装pip

```
curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
```

```
python get-pip.py
```

修改pip镜像源，以清华源为例

```
pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple
```

安装依赖包

```
pip install modelscope==1.22.3
```

```
pip install openai==1.61.0
```

```
pip install tqdm==4.67.1
```

```
pip install transformers==4.48.2
```

最佳实践

```
pip install vllm==0.7.1
pip install jinja2==3.1.0
```

2. 下载模型

为了方便，我们以较小的DeepSeek-R1-Distill-Qwen-1.5B为例（3.5G左右），将以下内容保存为model_download.py 文件，参数 cache_dir 为模型的下载路径，您可以按需修改，需确保存储空间足够存放模型。

```
from modelscope import snapshot_download
model_dir = snapshot_download('deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B', cache_dir='/root/model-path', revision='master')
```

运行该文件，启动模型下载。

```
python model_download.py
```

3. 运行模型

完成下载后，即可使用 vLLM 运行模型。

```
vllm serve /root/model-path/deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B --served-model-name DeepSeek-R1-Distill-Qwen-1.5B --max-model-len=2048 --api-key my-token --port=8000
```

注意

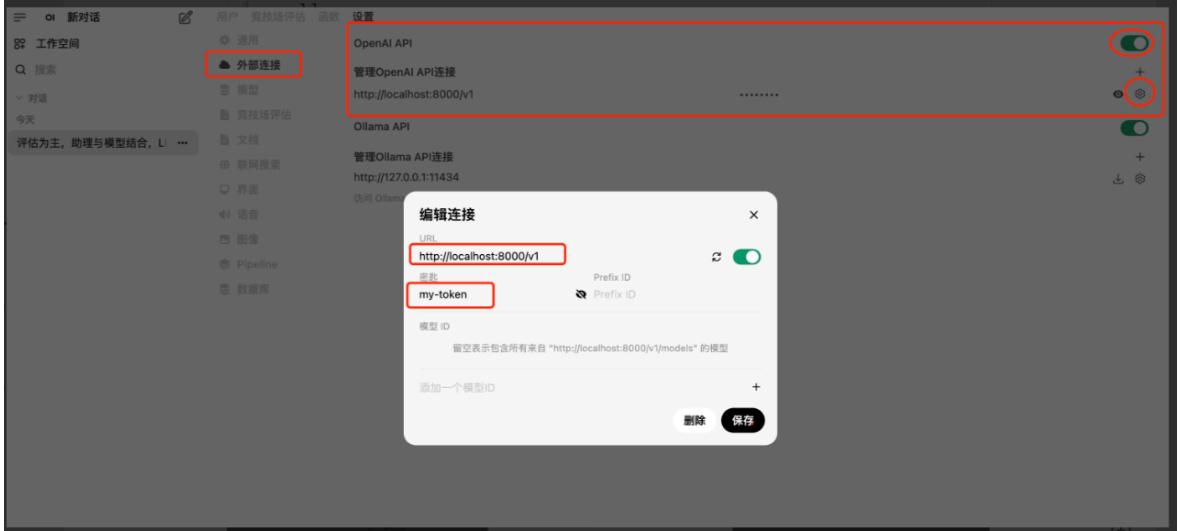
其中 --api-key 参数用于模型 api 调用时认证动作，可按需修改值。

最佳实践

4. 使用WebUI访问模型

基于我们已经安装、配置好的 OpenWebUI 页面，可以直接访问模型。

管理员设置中，配置 OpenAI API 访问地址，密钥就是上文中的 `--api-key`。



选择对应的模型，直接进行问答体验。



FAQ

如何修改 vLLM 的 api-key?

服务的配置文件目录在 `/var/vllm_service.env` 我们将 `API_KEY` 设置为想要的值。

```
API_KEY=xxh4cVRFPgfAr8ayyy
NV_NUM=4
NV_NUM=4
```

然后重启服务。

```
systemctl restart vllm
```

如何调整模型支持的最大上下文数量？

当前vLLM服务使用systemd部署，需修改service启动命令以实现上下文等服务参数调整。配置文件位于 `/etc/systemd/system/vllm.service`，我们编辑该文件的 `--max-model-len` 参数，然后重启服务即可实现参数调整

```
[Unit]
Description=vLLM Serve Service
After=network.target

[Service]
Type=simple
User=root
Group=root
ExecStartPre=/usr/local/bin/generate_api_key.sh
EnvironmentFile=/var/vllm_service.env
ExecStart=vllm serve /root/model-path/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B --served-model-name DeepSeek-R1-Distill-Qwen-32B --dtype=float16 --max-model-len=32768 --api-key ${API_KEY} --port=8000 --tensor-parallel-size ${NV_NUM}
Restart=on-failure
RestartSec=10

[Install]
WantedBy=multi-user.target
```

重新启动服务

```
systemctl daemon-reload
systemctl restart vllm
```

基于DeepSeek和Chroma构建个性化知识库的最佳实践

前言——私有知识库作用简介

大模型私有知识库，作为大语言模型技术与企业、组织自有数据深度融合的创新知识管理及应用解决方案，能够为特定用户群体提供更为精准、专业且安全的知识服务。具体来讲，它是借助大语言模型搭建而成，专门为特定组织或个人定制的知识存储与检索系统。此系统会对组织内部的专业知识、业务数据、历史文档等各类信息进行深度整合与精细化处理，从而构建出独一无二的专属知识集合。依托大模型强大的语言理解与生成能力，用户能够在此基础上实现高效的知识查询与问答交互。

在本教程中，我们将为您介绍详细介绍一种基于开源框架的私有知识库搭建方案。利用该方案，您可以在本地便捷地搭建起相应的私有知识库，大幅提升文本检索能力。

最佳实践

教程使用配置说明：C7通用型云主机

cpu 8核
内存 32G

优势

知识准确性：让模型访问定制的信息，从而提高回答的准确性和可靠性。

可解释性：检索过程可以明确指出回答所依据的信息来源，增强了回答的可解释性。

减少幻觉：降低了语言模型生成无事实依据内容（即“幻觉”）的可能性。

一、DeepSeek自部署

请参考在[天翼云使用Ollama运行 DeepSeek的最佳实践-7b版-弹性云主机-最佳实践-AIGC实践 - 天翼云](#) --自定义部署DeepSeek。

二、私有知识库构建

1. 安装向量数据库

- 检查python环境是否存在，不存在可参考安装

检查是否存在python

```
python --version
```

安装python环境

```
apt install python3
```

```
root@vm:~/deepseek-r1-1.5b# python --version
Python 3.10.12
```

- 安装向量数据库及与 ollama 交互的组件

安装 Chroma 向量数据库，基于langchain开源框架解析文本

```
pip install langchain_chroma
```

安装 Ollama 交互库

```
pip install langchain_ollama
```

安装 langchain_community 库

最佳实践

```
pip install langchain langchain_community
```

```
root@vm:~/deepseek-r1-1.5b# pip install langchain_chroma
Collecting langchain_chroma
  Downloading langchain_chroma-0.2.1-py3-none-any.whl (11 kB)
Collecting numpy<2.0.0,>=1.22.4
  Downloading numpy-1.26.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.2 MB)
  18.2/18.2 MB 104.7 kB/s eta 0:00:00
Collecting chromadb!=0.5.10,!0.5.11,!0.5.12,!0.5.4,!0.5.5,!0.5.7,!0.5.9,<0.7.0,>=0.4.0
  Downloading chromadb-0.6.3-py3-none-any.whl (611 kB)
  611.1/611.1 KB 63.3 kB/s eta 0:00:00
Collecting fastapi<1,>=0.95.2
  Downloading fastapi-0.115.8-py3-none-any.whl (94 kB)
  94.8/94.8 KB 72.5 kB/s eta 0:00:00
Collecting langchain-core!=0.3.0,!0.3.1,!0.3.10,!0.3.11,!0.3.12,!0.3.13,!0.3.14,!0.3.2,!0.3.3,!0.3.4
  Downloading langchain_core-0.3.34-py3-none-any.whl (412 kB)
  413.0/413.0 KB 91.5 kB/s eta 0:00:00
Collecting overrides>=7.3.1
  Downloading overrides-7.7.0-py3-none-any.whl (17 kB)
Collecting typing_extensions>=4.5.0
  Downloading typing_extensions-4.12.2-py3-none-any.whl (37 kB)
Collecting opentelemetry-instrumentation-fastapi>=0.41b0
  Downloading opentelemetry_instrumentation_fastapi-0.51b0-py3-none-any.whl (12 kB)
Collecting orjson>=3.9.12
  Downloading orjson-3.10.15-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (130 kB)
  130.3/130.3 KB 82.6 kB/s eta 0:00:00
Collecting chroma-baculdlib==0.7.6
```

```
root@vm:~/deepseek-r1-1.5b# pip install langchain_ollama
Collecting langchain_ollama
  Downloading langchain_ollama-0.2.3-py3-none-any.whl (19 kB)
Requirement already satisfied: langchain-core<0.4.0,>=0.3.33 in /usr/local/lib/python3.10/dist-packages (from langchain_ollama)
Collecting ollama<1,>=0.4.4
  Downloading ollama-0.4.7-py3-none-any.whl (13 kB)
Requirement already satisfied: jsonpatch<2.0,>=1.33 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.4.0,>=0.3.33)
Requirement already satisfied: typing-extensions>=4.7 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.4.0,>=0.3.33)
Requirement already satisfied: pydantic<3.0.0,>=2.5.2 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.4.0,>=0.3.33)
Requirement already satisfied: langsmith<0.4,>=0.1.125 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.4.0,>=0.3.33)
Requirement already satisfied: tenacity!=8.4.0,<10.0.0,>=8.1.0 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.4.0,>=0.3.33)
Requirement already satisfied: PyYAML>=5.3 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.4.0,>=0.3.33)
Requirement already satisfied: packaging<25,>=23.2 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.4.0,>=0.3.33)
Requirement already satisfied: httpx<0.29,>=0.27 in /usr/local/lib/python3.10/dist-packages (from ollama<1,>=0.4.4->langchain_ollama)
Requirement already satisfied: certifi in /usr/lib/python3/dist-packages (from httpx<0.29,>=0.27->ollama<1,>=0.4.4->langchain_ollama)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.10/dist-packages (from httpx<0.29,>=0.27->ollama<1,>=0.4.4->langchain_ollama)
```

```
root@vm:~# pip install langchain langchain_community
Collecting langchain
  Downloading langchain-0.3.18-py3-none-any.whl (1.0 MB)
  1.0/1.0 MB 41.5 kB/s eta 0:00:00
Collecting langchain_community
  Downloading langchain_community-0.3.17-py3-none-any.whl (2.5 MB)
  2.5/2.5 MB 57.6 kB/s eta 0:00:00
Requirement already satisfied: langsmith<0.4,>=0.1.17 in /usr/local/lib/python3.10/dist-packages (from langchain) (0.3.8)
Collecting langchain-text-splitters<1.0.0,>=0.3.6
  Downloading langchain_text_splitters-0.3.6-py3-none-any.whl (31 kB)
Collecting aiohttp<4.0.0,>=3.8.3
  Downloading aiohttp-3.11.12-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.6 MB)
  1.6/1.6 MB 413.3 kB/s eta 0:00:00
Requirement already satisfied: numpy<2,>=1.26.4 in /usr/local/lib/python3.10/dist-packages (from langchain) (1.26.4)
Collecting async-timeout<5.0.0,>=4.0.0
  Downloading async_timeout-4.0.3-py3-none-any.whl (5.7 kB)
Requirement already satisfied: PyYAML>=5.2 in /usr/local/lib/python3.10/dist-packages (from langchain) (6.0.2)
```

最佳实践

2. 安装文本解析插件，这里以pdf进行示例

```
pip install pdfplumber
```

```
root@vm:~/hp# pip install pdfplumber
Collecting pdfplumber
  Using cached pdfplumber-0.11.5-py3-none-any.whl (59 kB)
Collecting pypdfium2>=4.18.0
  WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None, status=None)) after
  able)': /packages/be/d9/a2f1ee03d47fbeb48bcfde47ed7155772739622cfadf7135a84ba6a97824/pypdfium2
  Downloading pypdfium2-4.30.1-py3-none-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (2.9 MB)
  _____ 2.9/2.9 MB 53.3 kB/s eta 0:00:00
Collecting Pillow>=9.1
  Downloading pillow-11.1.0-cp310-cp310-manylinux_2_28_x86_64.whl (4.5 MB)
  _____ 4.5/4.5 MB 47.3 kB/s eta 0:00:00
Collecting pdfminer.six==20231228
  Downloading pdfminer.six-20231228-py3-none-any.whl (5.6 MB)
  _____ 5.6/5.6 MB 44.2 kB/s eta 0:00:00
Collecting cryptography>=36.0.0
  Downloading cryptography-44.0.0-cp39-abi3-manylinux_2_28_x86_64.whl (4.2 MB)
  _____ 4.2/4.2 MB 13.2 kB/s eta 0:00:00
```

3. 基于私有数据开展模型对话

```
root@vm:~/hp# python3 ollama_private_vector_data_create.py
Number of requested results 4 is greater than number of elements in index 2, updating n_results = 2
检索到的文档摘要: <think>
嗯，我现在要总结一下这些文档中的主要主题。首先，我得仔细阅读每一段内容，找出它们之间的联系和区别。

第一段 talking about high generation能力，也就是生成文本的能力，包括快速生成高质量的中文文章、报告、故事等。强调语言流畅、逻辑连贯，并且可以根据不同的性

第二段是领域专业性，比如针对代码专门优化训练，提供准确专业的回答支持行业应用。这可能意味着DeepSeek - Coder在代码开发中特别强。

第三段讲可扩展性，模型架构设计灵活，适应数据和计算资源的变化。这说明模型能够随着使用情况不断优化，保持性能稳定。

接下来，应用场景部分分为智能写作、智能客服、代码开发和知识问答。这些都是用户可能遇到的具体问题解决方法，比如生成文案、处理客服请求、编写代码、获取信息等。

DeepSeek的模型家族包括 DeepSeek-Coder、DeepSeek-Llama和 DeepSeek-R1。DeepSeek-Coder专注于代码领域，而 DeepSeek-Llama在通用任务上表现好，DeepSeek-R1则有一定

特点部分提到强大的语言理解能力，并且具备在不同场景下的性能提升。这可能包括通过训练数据进行优化，提高模型的适应能力和扩展性。

综上所述，文档主要讲的是DeepSeek在自然语言处理和相关领域的应用，包括生成、问答、代码开发等多个方面，重点放在模型的专业性、可扩展性和多样化应用场景上。
</think>
```markdown
深度探索 (DeepSeek) 技术总结
1. 主要主题总结
```

图中使用的模型对话脚本如下，仅供参考：

```
from langchain_community.document_loaders import PDFPlumberLoader
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain_chroma import Chroma
from langchain_ollama import OllamaEmbeddings, ChatOllama
from langchain_core.output_parsers import StrOutputParser
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.runnables import RunnablePassthrough

加载 PDF 文档
file="DeepSeek.pdf"
loader = PDFPlumberLoader(file)
docs = loader.load()
```

# 分割文档，分割参数会影响推理准确性

## 最佳实践

```
text_splitter = RecursiveCharacterTextSplitter(chunk_size=500, chunk_overlap=0)
all_splits = text_splitter.split_documents(docs)

创建向量存储
local_embeddings = OllamaEmbeddings(model="deepseek-r1:1.5b")
vectorstore = Chroma.from_documents(documents=all_splits, embedding=local_embeddings)

初始化模型
model = ChatOllama(
 model="deepseek-r1:1.5b",
)

定义摘要提示模板
summary_prompt = ChatPromptTemplate.from_template(
 "总结这些检索到的文档中的主要主题: {docs}"
)

将传入的文档转换成字符串的形式
def format_docs(docs):
 return "\n\n".join(doc.page_content for doc in docs)

创建摘要链
summary_chain = {"docs": format_docs} | summary_prompt | model | StrOutputParser()

定义问题
question = "DeepSeek 项目的目的是什么？"

检索相关文档
docs = vectorstore.similarity_search(question)

生成摘要
summary = summary_chain.invoke(docs)
print("检索到的文档摘要: ", summary)

定义提示模板
RAG_TEMPLATE = """
你是一个问答助手。请使用以下检索到的上下文信息来回答问题。如果你不知道答案，请直接说不知道。答案最多使用三句话，并保持简洁。

<context>
{context}
</context>

回答以下问题:

{question}
"""
```

## 最佳实践

```
rag_prompt = ChatPromptTemplate.from_template(RAG_TEMPLATE)

获取检索器
retriever = vectorstore.as_retriever()

创建问答链
qa_chain = (
 {"context": retriever | format_docs, "question": RunnablePassthrough()}
 | rag_prompt
 | model
 | StrOutputParser()
)

运行问答链
answer = qa_chain.invoke(question)
print("问题的答案：", answer)
```

综上，您成功构建了一个定制化的私有知识库检索模型。

### 注意

对应文本数据需要与运行脚本在同一目录。

```
root@vm:~/hp# ls
DeepSeek.pdf ollama_private_vector_data_create.py
root@vm:~/hp#
```

## 基于OpenWebUI+Ollama+DeepSeek实现大模型私有知识库零代码构建的最佳实践

### 前言——私有知识库作用简介

大模型私有知识库是结合大语言模型技术与企业或组织自有数据的一种知识管理和应用方案，旨在为特定用户群体提供更精准、专业且安全的知识服务。

大模型私有知识库是基于大语言模型构建的，专门服务于特定组织或个人的知识存储和检索系统。它整合组织内部的专业知识、业务数据、历史文档等，形成专属知识集合，借助大模型的强大语言理解和生成能力，为用户提供高效的知识查询和问答服务。

本教程提供一种基于开源框架的私有知识库搭建方案。如果需要构建企业级服务且对数据安全有需求，则需相关企业对该服务进行定制，以完成私有知识库的搭建。

本文介绍的方法采用了开源框架自带的数据库服务，但该方法不保证相关数据在开源框架下的安全性。请务必在上传前对敏感数据进行脱敏处理。

本教程使用的GPU云主机规格如下所示，用于部署 DeepSeek-r1:70b 模型，配置仅供参考。

cpu 32核  
内存 128G

# 最佳实践

GPU: NVIDIA A10\*2 (48GB)

如需体验其它模型版本，请参考[在天翼云使用O1lama运行 DeepSeek的最佳实践-7b版-弹性云主机-最佳实践-AIGC实践 - 天翼云](#) --自定义部署DeepSeek--步骤二：规格选型，选择合适的云主机。

## 优势

知识准确性：让模型访问定制的信息，从而提高回答的准确性和可靠性。

可解释性：检索过程可以明确指出回答所依据的信息来源，增强了回答的可解释性。

减少幻觉：降低了语言模型生成无事实依据内容（即“幻觉”）的可能性。

## 一、环境准备

1. 参考[在天翼云使用O1lama运行 DeepSeek的最佳实践-7b版-弹性云主机-最佳实践-AIGC实践 - 天翼云](#) --快速体验DeepSeek，准备相关设备及模型资源。
2. 待模型启动后，通过访问<http://{公网ip地址}:3000>进入 OpenWebUI 界面。按照界面提示完成账号注册，开启私有知识库搭建之旅。

01



## 二、私有数据准备

搭建私有知识库需提前准备相应的数据材料。OpenWebUI 支持多种文本格式，包括但不限于：

- Markdown：轻量级标记语言格式，便于快速书写和阅读，能清晰呈现文本结构，常用于文档撰写、笔记记录等场景。

# 最佳实践

- PDF: 通用文档格式, 可保留文档原始排版、字体、图像等元素, 广泛应用于正式文档、书籍、报告的分发与存储。
- TXT: 最基本的文本格式, 仅包含文本内容, 无复杂格式设置, 适用于纯文本信息的存储和传输。
- HTML: 超文本标记语言格式, 用于创建网页, 包含文本内容和网页元素标记信息, 可被浏览器解析显示。
- docx: 是 Microsoft Word 的文档格式, 能包含丰富的文本格式、图表、样式等内容, OpenWebUI 可以借助相关的文档解析库来提取其中的文本信息, 用于知识库的构建和查询。

对于结构简单的纯文本材料, 可直接通过 OpenWebUI 进行解析。若需解除纯文本外其他文件格式的支持情况, 请查阅 OpenWebUI 官方文档。

## 三、私有知识库搭建教程

1. 登录 OpenWebUI 后, 点击左上角 “工作空间”。



2. 在 “工作空间” 中找到 “知识库” 选项, 点击右侧的 “+” 号新建知识库。

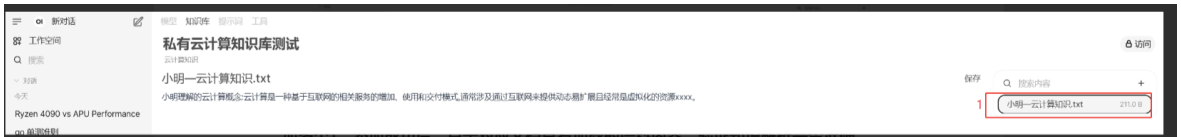


3. 在新建知识库页面, 填写知识库名称、描述, 设置可见性等信息, 完成后点击 “创建知识库”。



# 最佳实践

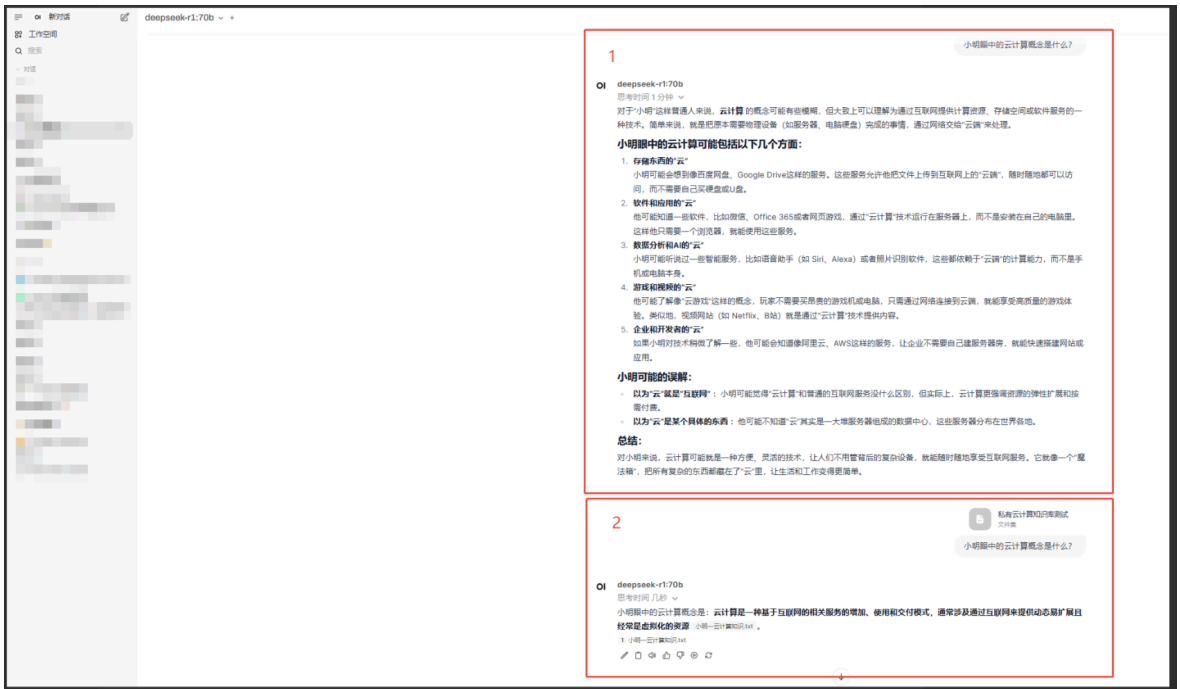
- 新建知识库成功后，再次点击右侧的“+”号，选择要添加的知识文档（可一次性添加多个）。添加成功后，点击对应文档查看加载的语料内容，验证知识解析是否准确。



# 最佳实践

## 四、私有知识模型推理验证

1. 点击左上角“新对话”，选择已部署的推理模型。



# 最佳实践

- 在输入对话内容时，以“#”号开头，可选择对应知识库或知识文档作为推理参考。



- 若不想继续使用某个知识库，可按照特定操作流程将其移除，具体操作可参考相关图示。



通过以上步骤，您即可完成一个简单的私有知识库搭建，赶紧动手试试吧！

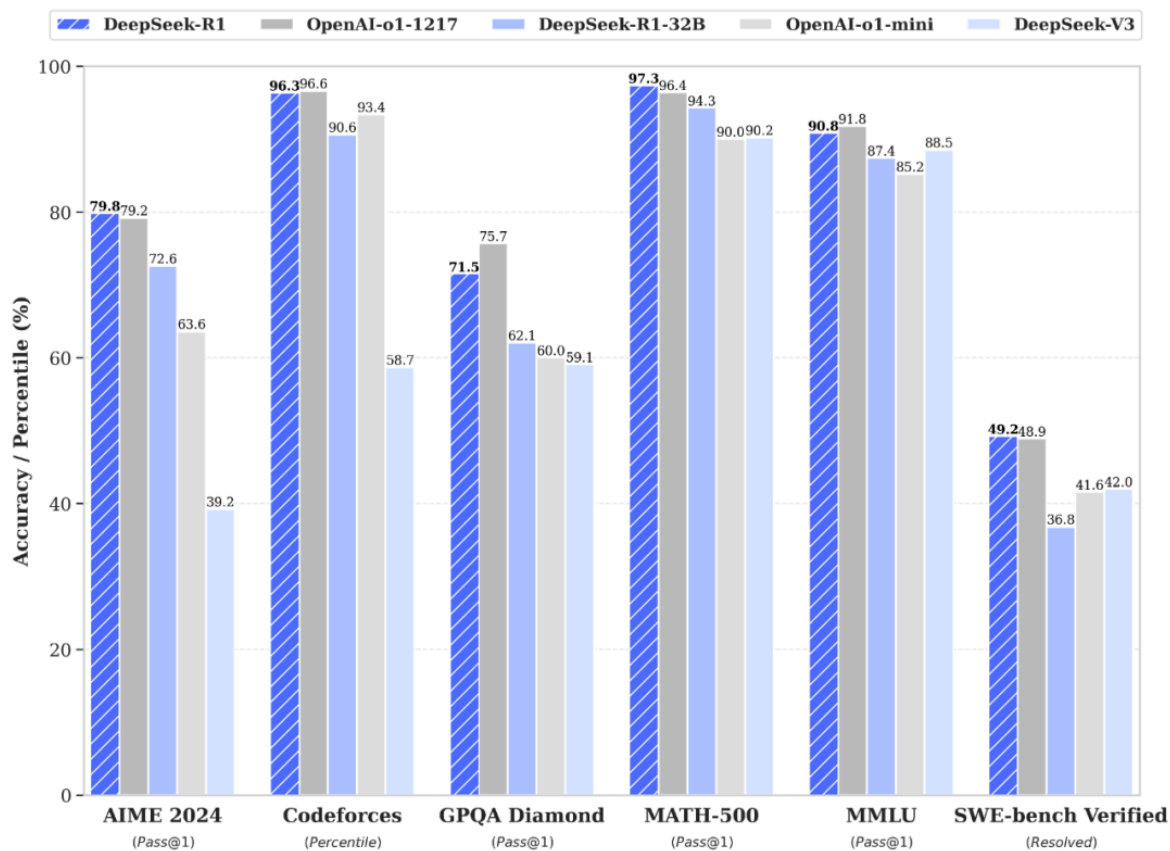
## 五、推荐使用场景

- 智能客服训练 AI 回答公司政策、产品 FAQ，减少人工客服负担。
- 企业内部搜索让员工直接向AI提问，无需翻遍公司Wiki或文档库。
- 智能助手帮助销售整理客户信息，协助HR处理面试反馈，助力研发自动生成技术文档。
- 智能营销策划协助市场团队分析市场趋势和竞争对手数据，生成针对性的营销活动方案，提升营销效果。
- 财务智能分析工具帮助财务人员自动完成财务报表分析，预测财务风险，为企业决策提供数据支持。
- 生产流程优化系统利用AI实时监测生产设备运行状况，预测设备故障，提前安排维护，减少生产中断。
- 培训内容生成助手，为培训部门快速生成员工培训课件、在线课程内容，满足不同岗位的培训需求。
- 供应链管理智能助手，根据销售数据、库存水平和供应商信息，辅助优化采购计划。

## 基于LLaMA-Factory微调DeepSeek-R1-Distill-Qwen-7B最佳实践

### DeepSeek简介

DeepSeek 是一个基于 Transformer 架构的大型语言模型 (LLM)，由深度求索 (DeepSeek) 公司开发。它能够处理自然语言理解、生成、翻译、问答等多种任务。DeepSeek-R1 在后训练阶段大规模使用了强化学习技术，在仅有极少标注数据的情况下，极大提升了模型推理能力。在数学、代码、自然语言推理等任务上，性能比肩 OpenAI o1 正式版。



DeepSeek公司在开源 DeepSeek-R1-Zero 和 DeepSeek-R1 两个参数量超过660B 模型的同时，通过 DeepSeek-R1 的输出，蒸馏了 6 个小模型开源给社区，其中32B和70B模型在多项能力上实现了对标 OpenAI o1-mini 的效果。

## 最佳实践

	AIME 2024 pass@1	AIME 2024 cons@64	MATH- 500 pass@1	GPQA Diamond pass@1	LiveCodeBench pass@1	CodeForces rating
GPT-4o-0513	9.3	13.4	74.6	49.9	32.9	759.0
Claude-3.5-Sonnet-1022	16.0	26.7	78.3	65.0	38.9	717.0
o1-mini	63.6	80.0	90.0	60.0	53.8	1820.0
QwQ-32B	44.0	60.0	90.6	54.5	41.9	1316.0
DeepSeek-R1-Distill-Qwen-1.5B	28.9	52.7	83.9	33.8	16.9	954.0
DeepSeek-R1-Distill-Qwen-7B	55.5	83.3	92.8	49.1	37.6	1189.0
DeepSeek-R1-Distill-Qwen-14B	69.7	80.0	93.9	59.1	53.1	1481.0
DeepSeek-R1-Distill-Qwen-32B	72.6	83.3	94.3	62.1	57.2	1691.0
DeepSeek-R1-Distill-Llama-8B	50.4	80.0	89.1	49.0	39.6	1205.0
DeepSeek-R1-Distill-Llama-70B	70.0	86.7	94.5	65.2	57.5	1633.0

### 模型微调简介

大语言模型微调技术通过调整预训练模型参数，使其适应特定任务需求。其核心价值在于让通用的人工智能工具变得更“专业”和“实用”，就像把一个会做各种菜的厨师培训成专精川菜的大厨。它能让企业用较低成本获得适合自身需求的AI能力，例如：

1. **降低成本**：无需从头训练昂贵的大模型，只需在现有通用模型基础上“补充学习”企业内部的业务知识、行业术语，就能快速打造专属AI助手，成本大幅降低。
2. **精准解决问题**：通用AI可能懂诗词绘画，但不懂医疗诊断或法律合同。通过微调，AI能深度掌握特定领域的专业知识，例如快速分析病历、自动生成符合行业规范的报告。
3. **保护数据隐私**：企业可将内部数据安全地用于微调，既让AI掌握核心业务知识，又避免敏感数据直接暴露在公共模型中。
4. **推动普惠应用**：中小型企业也能通过这项技术享受AI红利，例如教育机构定制学习辅导AI，工厂打造设备维护专家系统，真正让AI进入百行千业。

在下文中，我们将采用两个具体实践案例来说明模型微调的实际效果：

1. **基于天翼云文档数据的微调**。利用企业自身的文档数据构建微调数据集，仅使用单台多GPU卡的云主机即可完成模型微调训练。经过微调后的模型能够在企业内部知识领域提供准确的回答。
2. **基于医学计算数据的微调**。利用医学领域的专业知识数据构建微调数据集，仅使用单台多GPU卡的云主机即可完成模型微调训练。经过微调后的模型能够在识别与分析病例时取得更高的计算准确度。

### 快速微调DeepSeek

LLaMA-Factory是一个开源项目，旨在为用户提供简便的方式来训练和部署大型语言模型（LLM）。这里，我们以蒸馏小模型DeepSeek-R1-Distill-Qwen-7B模型为例，讲解如何在天翼云使用LLaMA-Factory运行与微调DeepSeek模型，读者可以根据自己实际需求选择模型和云主机配置。

# 最佳实践

## 步骤一：创建GPU云主机

### 1. 进入创建云主机页面。

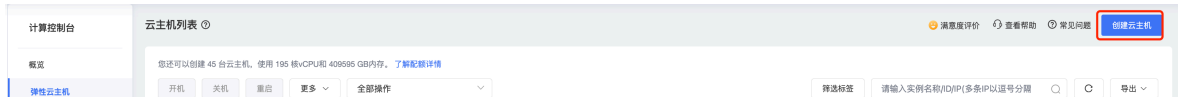
- a. 点击天翼云门户首页的“控制中心”，输入登录的用户名和密码，进入控制中心页面。



- b. 单击“服务列表>弹性云主机”，进入主机列表页。



- c. 单击“创建云主机”，进入弹性云主机创建页。



# 最佳实践

## 2. 进行基础配置

- 根据业务需求配置“计费模式”、“地域”、“企业项目”、“虚拟私有云”、“实例名称”、“主机名称”等。
- 选择规格。此处选择"CPU架构"为"X86"、"分类"为"GPU加速/AI加速型"、"规格族"为"GPU计算加速型pi7"、"规格"为"pi7.4xlarge.4"

### 注意

本案例仅做微调流程演示，选择的机型规格仅支持7B模型的少量数据微调，如果您的数据量较大，请选择显存更高的机型规格。

- 选择镜像。“镜像类型”选择“镜像市场”，在云镜像市场中选择预置了DeepSeek-R1-Distill-Qwen-7B模型的DeepSeek-LlamaFactory-Ubuntu22.04镜像。

## DeepSeek-LlamaFactory模型微调

预装DeepSeek-R1-Distill-Qwen-7B模型及LLaMA-Factory微调框架，支持快速开展DeepSeek模型微调实践。推荐配置：内存≥32G、显存≥24G

操作系统：Ubuntu

架构：X86

服务商：天翼云科技有限公司

推荐配置：16核32GB以上

费用 **¥0 / 月**

**立即使用**

## 3. 网络及高级配置

设置网络，包括"网卡"、"安全组"，同时配备"弹性IP"用于下载和访问模型；设置高级配置，包括"登录方式"、"云主机组"、"用户数据"。

### 注意

安全组需要放行22端口(用于ssh)和7860端口(用于访问web页面)。

## 4. 确认并支付

### 步骤二：微调DeepSeek模型

#### 1. 启动LLaMA-Factory服务

在完成云主机开通后，即可通过ssh连接到到云主机命令行。模型及LLaMA-Factory服务均已放置在/root目录下，且已完成配套环境安装，可直接运行。

#### # 进入目录

```
cd /root/LLaMA-Factory
```

#### # 启动webui服务 (也可通过nohup放到后台启动)

# 最佳实践

llamafactory-cli webui

在成功启动LLaMA-Factory服务后，您将可以通过 `http://{公网ip地址}:7860/` 访问到相关页面。

## # 访问服务

`http://{公网ip地址}:7860/`

## 2. 配置基础大模型和微调方法

在成功进入LLaMA-Factory页面后，您首先需要按照如图的内容进行页面语言，模型名称和模型路径的设置。

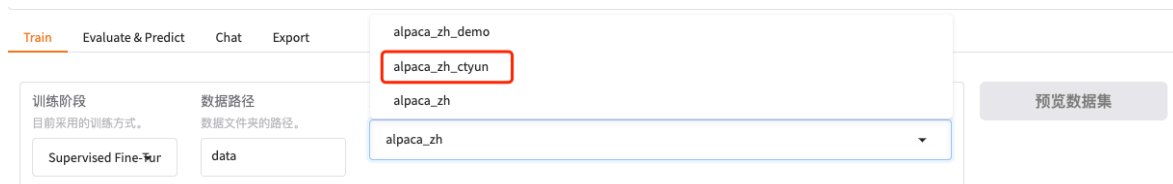
### 注意

模型预置在 `/root/DeepSeek-R1-Distill-Qwen-7B` 目录下。



## 3. 配置微调数据集

在Train页面内，您需要配置用于微调训练的数据集。镜像内预置了alpaca格式的天翼云示例数据，您也可参考示例数据修改 `/root/LLaMA-Factory/data/` 目录下的 `alpaca_zh_ctyun.json` (微调数据集) 和 `dataset_info.json` (数据集信息) 两个文件，改用您自己的私有数据。



镜像内置两份数据集，单条内容示例如下：

`data/alpaca_zh_ctyun.json`(天翼云文档数据集)

```
{
 "instruction": "天翼云的通用型弹性云主机有哪些规格？",
 "input": "",
 "output": "<think>用户询问通用型云主机规格，首先需要明确通用型的定位是平衡计算和内存资源。根据产品文档，通用型以s7系列为代表，应当列举不同后缀规格及其配置差异，最后补充适用场景说明。</think>\n通用型弹性云主机有多个规格，例如s7.small、s7.medium、s7.large、s7.xlarge等，每种规格提供不同的vCPU和内存配置，适合多种业务场景。"
```

# 最佳实践

```
}
```

data/alpaca\_en\_medcalc\_bench\_max2k.json(医学计算数据集MedCalc-Bench)

```
{
```

```
 "instruction": "A 16-year-old female adolescent was referred to our hospital with severe hypertension (systolic pressure 178 mmHg), which was first detected 7 months prior to presentation during a routine a...";
```

```
 "input": "What is the patient's Creatinine Clearance using the Cockcroft-Gault Equation in terms of mL/min? You should use the patient's adjusted body weight in kg instead of the...";
```

```
 "output": "<think>The formula for computing Cockcroft-Gault is given by $CrCl = ((140 - age) * adjusted\ weight * (gender_coefficient)) / (serum\ creatinine * 72)$, where the gender_coefficient is 1 if male, and 0.85 if female..."
```

```
}
```

## 4. 配置相关训练参数

在Train页面内，还有大量可修改的微调训练参数。如果您对他们还不够了解，可暂时不进行修改。其中，训练轮数与样本量级关联性较大，如果您的微调数据量很少，则可能需要设置较大的训练轮数，才能有效果。

The screenshot shows the 'Train' configuration page with the following settings:

- 训练阶段 (Training Stage):** Supervised Fine-Tune
- 数据路径 (Data Path):** data
- 数据集 (Dataset):** alpaca\_zh
- 学习率 (Learning Rate):** 5e-5
- 训练轮数 (Training Steps):** 3.0 (highlighted with a red box)
- 最大梯度范数 (Maximum Gradient Norm):** 1.0
- 最大样本数 (Maximum Number of Samples):** 100000
- 计算类型 (Compute Type):** bf16

# 最佳实践

## 5. 启动微调训练

点击最下方的开始按钮，即可基于上面选择的基础模型和微调数据集，启动模型微调训练。页面下方会显示实时的训练进度，训练日志和loss变化情况。

预览命令 保存训练参数 载入训练参数 开始 中断

输出目录  
保存结果的路径。  
train\_2025-02-10-17-47-25

配置路径  
保存训练参数的配置文件路径。  
2025-02-10-17-47-25.yaml

设备数量  
当前可用的运算设备数。  
1

DeepSpeed stage  
多卡训练的 DeepSpeed stage。  
none

使用 DeepSpeed offload (会减慢速度)。  
 使用 offload

Running 50/100: 0:01:10 < 0:01:10

0  100

[INFO][2025-02-10 17:52:23] configuration\_utils.py:694 >> loading configuration file /root/DeepSeek-R1-Distill-Qwen-7B/config.json

```
[INFO][2025-02-10 17:52:23] configuration_utils.py:768 >> Model config Qwen2Config { "_name_or_path": "/root/DeepSeek-R1-Distill-Qwen-7B",
 "architectures": ["Qwen2ForCausalLM"],
 "attention_dropout": 0.0,
 "bos_token_id": 151643,
 "eos_token_id": 151643,
 "hidden_act": "silu",
 "hidden_size": 3584,
 "initializer_range": 0.02,
 "intermediate_size": 18944,
 "max_position_embeddings": 131072,
 "max_window_layers": 28,
 "model_type": "qwen2",
 "num_attention_heads": 28,
 "num_hidden_layers": 28,
 "num_key_value_heads": 4,
 "rms_norm_eps": 1e-06,
 "rope_scaling": null,
 "rope_theta": 10000,
 "sliding_window": null,
 "tie_word_embeddings": false,
 "torch_dtype": "bfloat16",
 "transformers_version": "4.48.3",
 "use_cache": true,
 "use_mrope": false,
 "use_sliding_window": false,
 "vocab_size": 152064 }
```

损失

如果您的训练样本较多，单张A10显卡的24G显存很容易因为无法承载，而导致报错"CUDA out of memory"。此时，您需要将云主机变配到显存更大的机型规格，并开启DeepSpeed stage3进行模型参数分片，如下图所示。

输出目录  
保存结果的路径。  
train\_2025-02-11-15-44-14

配置路径  
保存训练参数的配置文件路径。  
2025-02-11-15-44-14.yaml

设备数量  
当前可用的运算设备数。  
2

DeepSpeed stage  
多卡训练的 DeepSpeed stage。  
3

使用 DeepSpeed offload (会减慢速度)。  
 使用 offload

截断长度  
输入序列分词后的最大长度。  
2048

批处理大小  
每个 GPU 处理的样本数量。  
1

梯度累积  
梯度累积的步数。  
8

4  131072    1  1024    1  1024

## 最佳实践

同时，您还需要修改默认DeepSpeed配置中的部分参数，以保证训练正常进行。修改/root/LLaMA-Factory/cache/ds\_z3\_config.json中的如下内容：

```
"zero_optimization": {
 "stage": 3,
 "overlap_comm": true,
 "contiguous_gradients": true,
 "sub_group_size": 1e9,
 "reduce_bucket_size": "auto",
 "stage3_prefetch_bucket_size": "auto",
 "stage3_param_persistence_threshold": "auto",
 "stage3_max_live_parameters": 1e4, //降低该参数以减少显存占用
 "stage3_max_reuse_distance": 1e4, //降低该参数以减少显存占用
 "stage3_gather_16bit_weights_on_model_save": true
}
```

### 步骤三：测试模型微调效果

LLaMA-Factory框架提供了模型推理的能力，可直接进入Chat页面加载模型进行对话。

# 最佳实践

## 1. 测试基础大模型效果

保持模型名称和模型路径不变，不设置检查点路径，在Chat页面直接点击加载模型，即可测试基础大模型的效果。

本案例使用prompt="天翼云弹性云主机包含哪些规格"进行测试。

基础DeepSeek-R1-Distill-Qwen-7B大模型给出了通用但是错误的回答。

The screenshot shows a web interface for testing LLMs. At the top, there are navigation tabs: Train, Evaluate & Predict, Chat (selected), and Export. Below this is a configuration panel with the following settings:

- 语言: zh
- 模型名称: DeepSeek-R1-7B-Distill
- 模型路径: /root/DeepSeek-R1-Distill-Qwen-7B
- 微调方法: lora
- 检查点路径: (empty)
- 量化等级: none
- 量化方法: bitsandbytes
- 对话模板: deepseek3
- RoPE 插值方法: none
- 加速方式: auto

Below the configuration panel, there are two buttons: 加载模型 (Load Model) and 卸载模型 (Unload Model). The status bar indicates "模型已加载, 可以开始聊天了!" (Model loaded, ready to chat!).

The chat interface shows a prompt: "天翼云弹性云主机包含哪些规格" (Which specifications does the Elastic Cloud Host of Tianyi Cloud include?). The response is:

扩展性: 根据负载自动调整资源。

7. 使用场景

- 适合企业级应用, 如电商、云计算平台、大数据平台。

8. 价格

- 分定价方案: 按月、按年或按小时计费, 企业版价格较高。

9. 售后服务

- 技术支持: 电话、邮件、在线聊天。
- 售后保障: 365天全天候支持, 快速响应和修复。

总结

天翼云弹性云主机 ideal for high-traffic, mission-critical applications, offering robust security, scalability, and performance. 个人用户应选择更经济的方案。

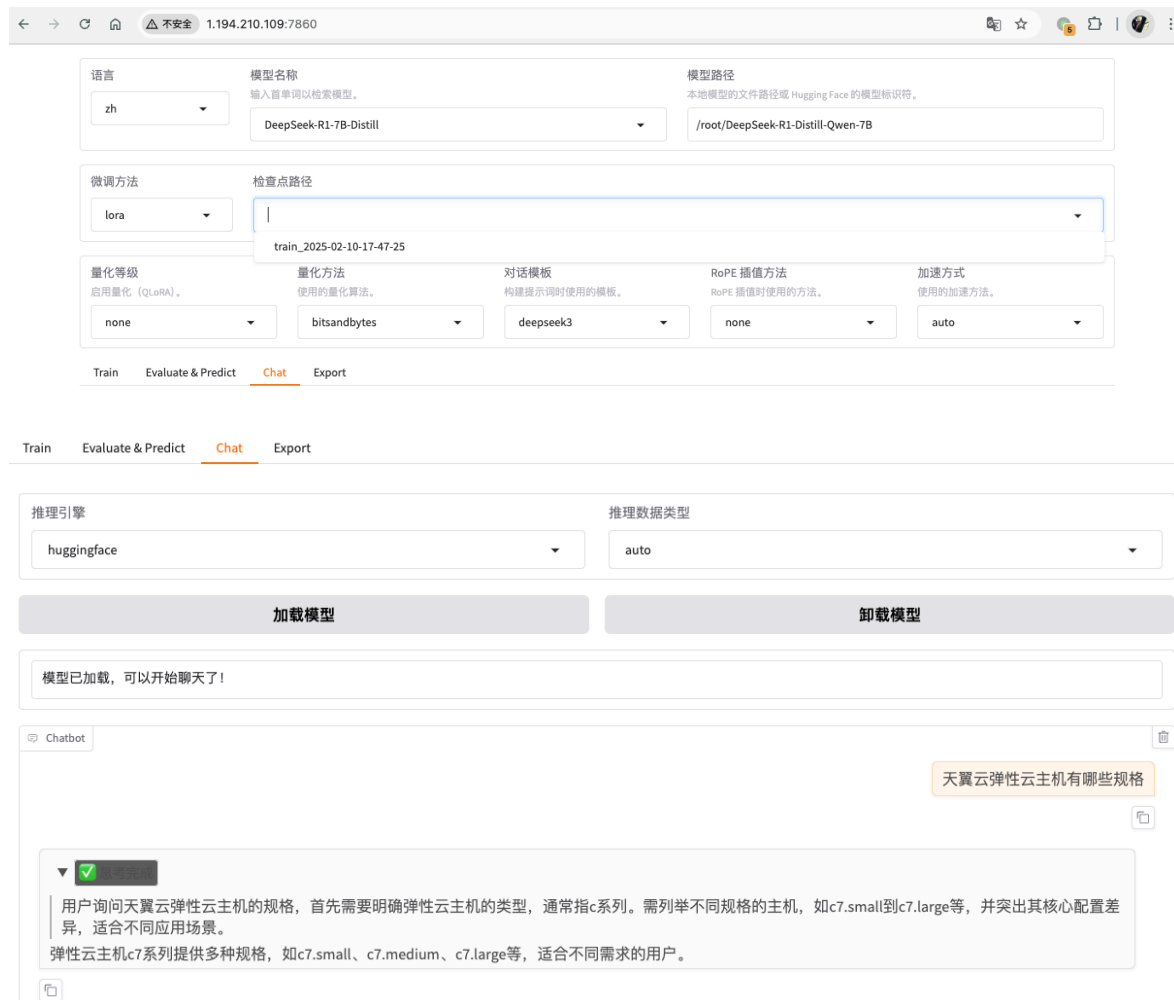
# 最佳实践

## 2. 测试微调大模型的效果

本案例采用了lora的微调方法，因此需将检查点路径设置为步骤二中完成训练的lora小模型。

本案例使用prompt="天翼云弹性云主机包含哪些规格"进行测试。

微调后的大模型能够给出符合微调数据集的回答。



### 从纯净版ubuntu系统构建微调环境

除了使用预装镜像外，您也可以从任意ubuntu云主机开始，按以下步骤搭建出LLaMA-Factory微调环境，并下载任意您需要的模型。

#### 步骤一：安装Nvidia显卡驱动与CUDA库

# 安装nvidia显卡驱动(如已选择带驱动镜像，则忽略)

```
apt update
```

```
apt install ubuntu-drivers-common alsa-utils
```

# 查看推荐的驱动版本

```
ubuntu-drivers devices
```

# 安装推荐版本

```
apt install nvidia-driver-550
```

# 重启一次云主机

```
reboot
```

# 安装cuda-12.2

```
wget https://developer.download.nvidia.com/compute/cuda/12.2.0/local_installers/cuda_12.2.0_535.54.03_linux.run
```

# 注意取消对nvidia driver的安装

```
sh cuda_12.2.0_535.54.03_linux.run
```

# 安装完成后在~/.bashrc中写入环境变量

# 添加CUDA可执行文件路径

```
export PATH=/usr/local/cuda-12.2/bin${PATH:+:${PATH}}
```

# 添加CUDA库文件路径

```
export LD_LIBRARY_PATH=/usr/local/cuda-12.2/lib64${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}
```

# 使环境变量生效

```
source ~/.bashrc
```

## 步骤二：安装git与下载模型

# 安装git和lfs

```
apt update
```

```
apt install git
```

```
curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | bash
```

```
apt install git-lfs
```

```
git lfs install
```

# 下载模型

# 国内用modelscope源速度最快

```
git clone https://www.modelscope.cn/deepseek-ai/DeepSeek-R1-Distill-Qwen-7B.git
```

# 清理掉不用的.git目录下lfs大文件版本数据(能节省一半存储空间)

```
git lfs prune -f
```

## 步骤三：安装LLaMA-Factory及相关训练环境

# 安装miniconda

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

# 下载完成后手动安装

```
sh Miniconda3-[版本号]-Linux-x86_64.sh
```

# 使环境变量生效

```
source ~/.bashrc
```

# 安装LLaMA-Factory

```
git clone --depth 1 https://github.com/hiyouga/LLaMA-Factory.git
```

```
cd LLaMA-Factory
```

```
pip install -e "[torch,metrics]" -i https://pypi.tuna.tsinghua.edu.cn/simple
```

```
pip install deepspeed -i https://pypi.tuna.tsinghua.edu.cn/simple
```

```
pip install flash-attn --no-build-isolation -i https://pypi.tuna.tsinghua.edu.cn/simple
```

## 可选步骤1：安装vllm推理环境

```
创建新的虚拟环境(vllm和llama-factory依赖库版本不同, 无法共存)
```

```
conda create -n vllm_env python=3.12
```

```
conda activate vllm_env
```

```
安装vllm用于推理
```

```
pip install vllm bitsandbytes -i https://pypi.tuna.tsinghua.edu.cn/simple
```

```
启动vllm (可参考vllm官方文档vllm serve命令)
```

## 可选步骤2：安装open-webui可视化界面

```
拉取官方docker镜像(需自行配置网络代理)
```

```
docker pull ghcr.io/open-webui/open-webui:main
```

```
启动容器(对照vllm启动参数, 自行配置API_KEY, URL参数)
```

```
docker run -itd --network host -v /root/volume/open-webui:/app/backend/data -
```

```
e PORT=3000 -e ENABLE_SIGNUP=true -e OPENAI_API_BASE_URL=http://127.0.0.1:8000/v1 -
```

```
e OPENAI_API_KEY=${API_KEY} -e HF_ENDPOINT=https://www.modelscope.cn --name open-webui --
```

```
restart always ghcr.io/open-webui/open-webui:main
```

## 基于CPU的AI推理加速技术在天翼云EMR云主机上的应用

### 1. 背景

#### 1.1 大模型算力需求

大模型因其参数规模巨大、结构复杂, 通常需要强大的计算资源来支持其推理过程, 这使得算力成为大模型应用的关键要素。信通院的数据资料显示云端推理的市场潜力很大, AI大模型落地应用数量正在不断增加, 市场对大模型推理算力的需求也呈现爆发式增长态势。

近年来, CPU制程和架构的提升以及英特尔AMX指令集的面世带来了算力的快速提升。英特尔对大模型推理等多个AI领域持续深入研究, 提供全方位的AI软件支持, 兼容主流AI软件且提供多种软件方式提升CPU的AI性能。目前, 已有充分的数据显示CPU可以用于大模型推理。

#### 1.2 CPU算力用于大模型推理的适用场景

CPU适用于以下大模型推理场景,

场景1: 大模型推理需要的内存超过了单块GPU的显存容量, 需要多块或更高配GPU卡, 采用CPU方案, 可以降低成本;

场景2: 应用请求量小, GPU利用率低, 采用CPU推理, 资源划分的粒度更小, 可有效降低起建成本;

场景3: GPU资源紧缺, CPU更容易获取, 且可以胜任大模型推理。

## 2. 英特尔 AI 推理加速技术

### 2.1 英特尔 AI 硬件基础

#### 第三代英特尔® 至强® 可扩展处理器 (Ice Lake) :

最大支持40个物理核，最大支持内存频率3200MT/s，最大支持64条PCIe 4.0。

搭载英特尔® 高级矢量扩展 512 (英特尔® AVX-512) 技术。

可用：天翼云第七代云主机 (s7/c7/m7)。

#### 第四代英特尔® 至强® 可扩展处理器 (Sapphire Rapids) :

最大支持60个物理核，最大支持内存频率4800MT/s，最大支持80条PCIe 5.0。

搭载英特尔® 高级矩阵扩展 (Advanced Matrix eXtensions)技术。

AMX 支持BF16和INT8，通过加速矩阵乘法运算，极大提升AI应用的性能。

可用：天翼云第八代云主机 (s8r)。

#### 第五代英特尔® 至强® 可扩展处理器 (Emerald Rapids) :

最大支持64个物理核，最大支持内存频率5600MT/s，L3 cache高达5MB/core，最大支持80条PCIe 5.0。

搭载英特尔® 高级矩阵扩展 (Advanced Matrix eXtensions)技术。

AMX 支持BF16和INT8，通过加速矩阵乘法运算，极大提升AI应用的性能。

相较于第四代处理器，第五代处理器具有更大的带宽，在大语言模型推理领域可带来最高1.5倍的性能提升。

可用：天翼云第八代云主机 (c8e)。

### 2.2 英特尔 AI 软件工具

英特尔提供三个层面的AI软件支持：



主流开源社区 (PyPI, Anaconda, Intel, Apt, Yum, Docker) 和生态 (Hugging Face)都可获取

#### 2.2.1 Intel® Extension for PyTorch\*

Intel® Extension for PyTorch\* (旧称Intel PyTorch Extension, IPEX) 能够优化AI推理任务在英特尔硬件上的性能，支持AVX-512与AMX。

## 最佳实践

### 2.2.2 xFasterTransformer

xFasterTransformer (xFT) 是由英特尔官方开源的AI推理框架，为大语言模型 (LLM) 在英特尔®至强®硬件平台上提供了一种深度优化的解决方案。xFT支持AVX-512与AMX，并可在AMX上获得极致的推理性能。xFT目前支持的接口与模型如下：

Models	Framework		Distribution
	PyTorch	C++	
ChatGLM	✓	✓	✓
ChatGLM2	✓	✓	✓
ChatGLM3	✓	✓	✓
Llama	✓	✓	✓
Llama2	✓	✓	✓
Baichuan	✓	✓	✓
QWen	✓	✓	✓
SecLLM(YaRN-Llama)	✓	✓	✓
Opt	✓	✓	✓
Deepseek-coder	✓	✓	✓

(更新时间：2024-04-09)

xFT还可支持多种数据类型，满足用户对不同精度的需求。xFT目前支持的数据类型包括：

FP16 / BF16 / INT8 / W8A8 / INT4 / NF4 / BF16\_FP16 / BF16\_INT8 / BF16\_W8A8 / BF16\_INT4 / BF16\_NF4 / W8A8\_INT8 / W8A8\_INT4 / W8A8\_NF4 (更新时间：2024-04-09)

xFT部署简便，您只需要一键安装插件，并在程序中加入几行代码，即可使用xFT对您的模型进行加速。具体实践可参考本文第3节的内容。

## 3. 英特尔 AI 软件工具在天翼云主机上的应用实践

本节内容主要介绍基于Intel® Extension for PyTorch\*与xFasterTransformer，能够在天翼云主机上取得性能指标，以及具体实施方法。

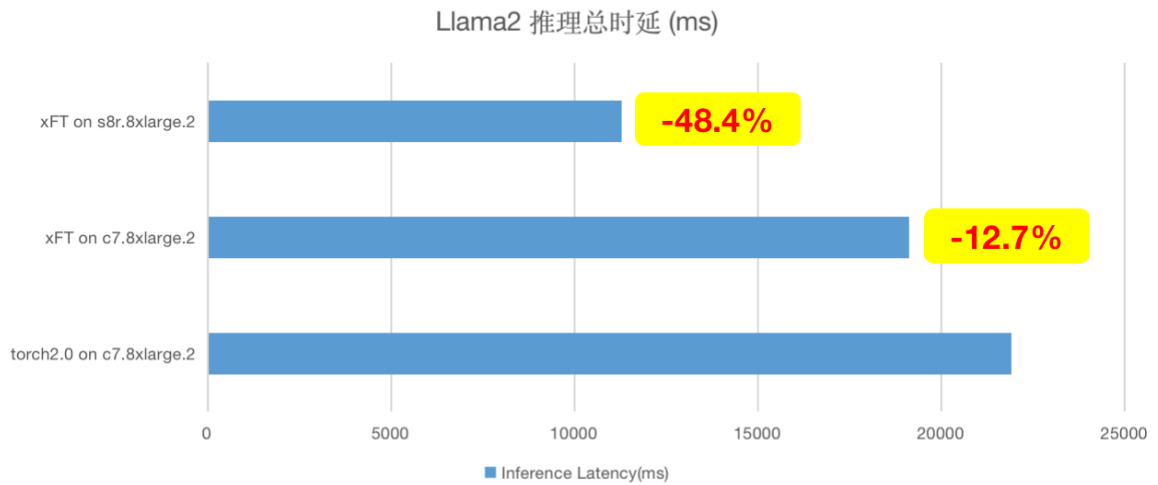
### 3.1 AI 推理性能指标

#### 3.1.1 LLaMA2性能指标

##### 基本参数

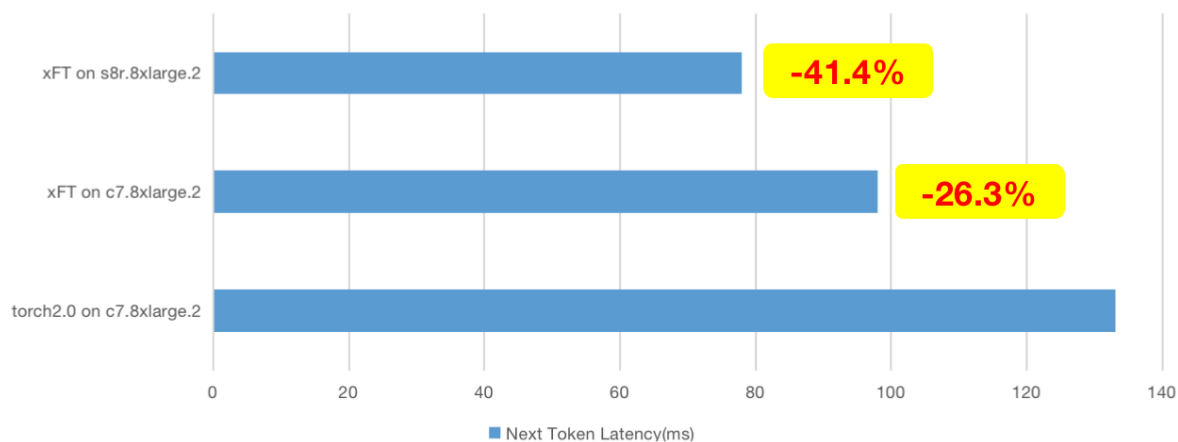
- vcpu数: 32
- 内存: 64GB
- 硬盘: 通用型SSD
- 模型: Llama-2-7b-chat
- 输入token个数: 128
- 输出token个数: 128
- batch size: 1
- 数据类型: torch2.0(BF16), xFT(FP16)
- 测试机型: c7.8xlarge.2 | 32核 | 64G (numa node 1), s8r.8xlarge.2 | 32核 | 64G (numa node 2)

##### 性能数据



## 最佳实践

llama2 生成token平均时延 (ms)



xFT在推理总时延（Inference Latency）和生成token时延（Next Token Latency）上都有明显的优化。

### 3.1.2 StableDiffusion性能指标

#### 基本参数

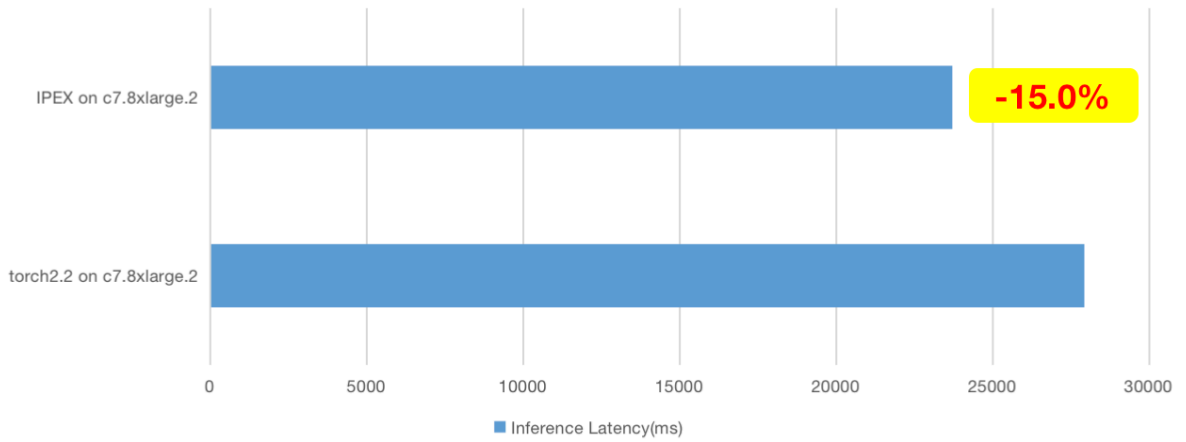
- vcpu数: 32
- 内存: 64GB
- 硬盘: 通用型SSD
- 模型: stable-diffusion-2-1
- 输入token个数: 6
- 输出图片尺寸: 512 x 512
- 推理Steps: 20
- batch size: 1
- 数据类型: torch2.0(FP32), IPEX(FP32)

#### 性能数据

IPEX在推理总时延上有一定的性能提升。

# 最佳实践

StableDiffusion 单张图片推理时延 (ms)



### 3.1.3 EMR云主机LLM性能指标

#### 注意

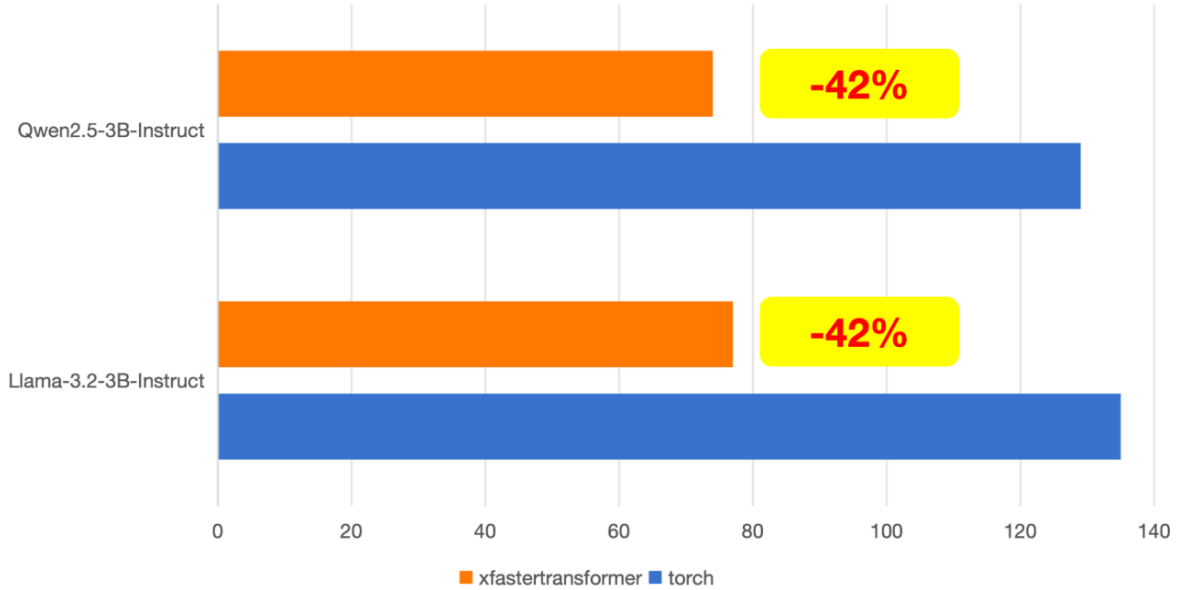
在天翼云EMR云主机正式上线后，我们更新性能指标如下。

#### 基本参数

- vcpu数: 16
- 内存: 64GB
- 硬盘: 通用型SSD
- 输入token个数: 128
- 输出token个数: 128
- batch size: 1
- 数据类型: torch2.0(BF16), xFT(BF16)

## 性能数据

生成token平均时延(不包含首词) (ms)



xFT在生成token时延（Next Token Latency）上有明显的优化，在16vcpu的机型上能使用主流3B规格模型达到低于80ms/token的生成速度，足以满足业界需求。

### 3.1.4 EMR云主机StableDiffusion性能指标

#### 注意

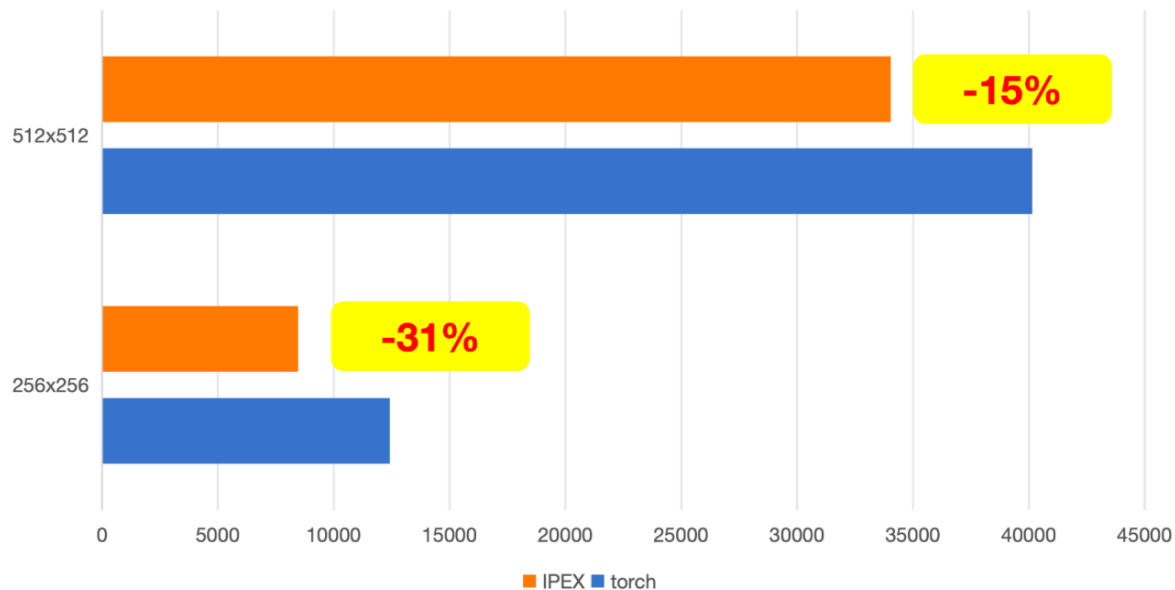
在天翼云EMR云主机正式上线后，我们更新性能指标如下。

#### 基本参数

- vcpu数: 16
- 内存: 64GB
- 硬盘: 通用型SSD
- 模型: stable-diffusion-2-1
- 输入token个数: <16
- 推理Steps: 20
- batch size: 1
- 数据类型: torch(FP32), IPEX(FP32)

## 性能数据

单张图片推理时延 (ms)



IPEX在推理总时延上有一定的提升，在本示例16vcpu和20steps配置下能够将256x256大小图片生成时延压缩到10秒内，512x512大小图片生成时延压缩到35秒内，具有一定的实用价值。

### 3.2 AI 软件工具实施方法（基于一键部署镜像）

为了方便您进行性能测试，天翼云联合英特尔制作了一键部署的云主机镜像，内置torch2.0环境，Intel® Extension for PyTorch\*与xFasterTransformer软件，以及Llama-2-7b-chat与stable-diffusion-2-1模型。您可直接开通云主机并选择对应镜像进行体验。

镜像支持的云主机规格与资源池信息如下表：

#### 注意

内存需在32G及以上。

处理器型号	云主机规格	资源池
英特尔第三代至强可扩展处理器 AVX-512	s7 / c7 / m7	西南-西南1 华东-华东1
英特尔第四代至强可扩展处理器AMX	s8r	华东-华东1 -可用区2
英特尔第五代至强可扩展处理器AMX	c8e	华东-华东1 -可用区2

# 最佳实践

## 注意

镜像内置的是天翼云七代主机(s7/c7/m7)特殊编译版本的xfastertransformer，如果您使用的是天翼云八代机(s8r)，请在xFT环境下手动更新安装新版本，否则版本不兼容，可参考下方的示例代码。

```
conda activate xFT
pip install xfastertransformer==1.4.0 -i https://pypi.tuna.tsinghua.edu.cn/simple
```

镜像名称：应用镜像 - ubuntu - Intel-AI推理加速，如图所示：



## 3.2.1 xFasterTransformer (xFT)实施方法

### 3.2.1.1 模型转换为xFT格式

在使用xFT框架前，需要对模型进行格式转换。xFT提供了转换命令，您可修改和执行/root/tools/xft\_trans.py脚本。

```
import xfastertransformer as xft
输入huggingface模型路径
HF_DIR = "/root/models/Llama-2-7b-chat-hf"
输出xft模型路径
XFT_DIR = "/root/models/Llama-2-7b-chat-xft"
xft.LlamaConvert().convert(HF_DIR, XFT_DIR)
```

在执行前，需要进入conda运行环境(执行过一次即可)

```
conda activate xFT
python xft_trans.py
```

### 3.2.1.2 查看numa节点分布

```
numactl -H
```

以下输出代表32个vcpu均在同一个numa node下：

## 最佳实践

```
(xFT) root@ecm-d23e:~/tools# numactl -H
available: 1 nodes (0)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
node 0 size: 63790 MB
node 0 free: 48631 MB
node distances:
node 0
0: 10
```

以下输出代表vcpu0-15分布在numa node0, vcpu16-31分布在numa node1:

```
(xFT) root@ecm-4295:~# numactl -H
available: 2 nodes (0-1)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
node 0 size: 31634 MB
node 0 free: 2157 MB
node 1 cpus: 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
node 1 size: 32251 MB
node 1 free: 2701 MB
node distances:
node 0 1
0: 10 20
1: 20 10
```

### 3.2.1.3 检查测试参数

性能测试所需的参数均在、root/tools/llama\_benchmark.py中, 您可按需进行修改, 对比开启和关闭xFT的效果, 以及不同token长度的效果等。

```
tools > llama_benchmark.py
10 # llama2 hf格式模型地址, 默认模型将以bf16精度加载
11 MODEL_PATH = "/root/models/Llama-2-7b-chat-hf"
12 # 是否使用xFT
13 USE_XFT = True
14 # 使用xFT时需要提前转换模型, 参考xft_trans.py, 默认模型将以fp16精度加载
15 MODEL_PATH_XFT = "/root/models/Llama-2-7b-chat-xft"
16 # 输入TOKEN数, 预制的输入明文参见prompt.json, 可选: 22, 32, 64, 128, 256, 512, 1024, 2016, 2017, 2048, 4096, 8192
17 TOKEN_IN = 128
18 # 输出新增TOKEN数 (总数=TOKEN_IN+TOKEN_OUT_NEW)
19 TOKEN_OUT_NEW = 128
20 # 模型预热次数, 大于2后指标基本恒定
21 WARMUP = 2
22 # 执行次数(最后取均值)
23 ITERATION = 5
24 # 是否输出明文结果(仅用于debug)
25 PRINT_OUTPUT = False
26 # 读取预制的prompt, 输入只需指定token个数。
27 PROMPT_PATH = "/root/tools/prompt.json"
```

# 最佳实践

## 3.2.1.4 单进程执行

### 注意

如果您的云主机存在不止一个numa node，直接使用单进程方式执行脚本会出现大幅性能波动，您可参考单核和多核的不同实施方法。

在单进程情况下，您可直接执行python脚本：

# 在执行前，需要进入conda运行环境(执行过一次即可)

```
conda activate xFT
```

```
cd /root/tools
```

# 绑定cpu核，性能更稳定

# OMP线程数和绑定的cpu核数保持一致

# 天翼云主机开启了cpu超线程，0-1/2-3号vcpu分别在同一个物理核上，只需要使用其中一个核即可

# vcpu与物理核的对应关系可在/proc/cpuinfo查看每个processor的物理id

```
OMP_THREAD_NUM=16 numactl --all -
```

```
C 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30 python llama_benchmark.py
```

您将会得到以下输出（本示例采用天翼云c7.8xlarge.2 | 32核 | 64G云主机）：

```
(xFT) root@ecm-d23e:~/tools# OMP_THREAD_NUM=16 numactl --all -C 0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30 python llama_benchmark.py
OMP_NUM_THREADS: None
temporary set OMP_NUM_THREADS=16
torch thread num: 16
xFT loading model...
[INFO] SINGLE_INSTANCE MODE.
master node run
===== llama benchmark =====
Input token num 128
Output new token num: 128
Model warmup round: 2
Warmup Iteration 1: total time: 19245.86 ms, new token num: 128
Warmup Iteration 2: total time: 19178.66 ms, new token num: 128
Start benchmark:
Iteration 1: total time: 19096.87 ms, new token num: 128, first token time: 6603.65 ms, throughput(next token): 10.17 tokens/s
Iteration 2: total time: 19133.18 ms, new token num: 128, first token time: 6612.43 ms, throughput(next token): 10.14 tokens/s
Iteration 3: total time: 19131.97 ms, new token num: 128, first token time: 6612.95 ms, throughput(next token): 10.14 tokens/s
Iteration 4: total time: 19090.90 ms, new token num: 128, first token time: 6601.22 ms, throughput(next token): 10.17 tokens/s
Iteration 5: total time: 19109.18 ms, new token num: 128, first token time: 6603.84 ms, throughput(next token): 10.16 tokens/s

===== Final Performance =====
Use xFT: True
Model dtype: fp16
Input token num 128
Output new token num: 128
Inference Avg Latency: 19112.42 ms
First token Avg Latency: 6606.82 ms
Next token Avg Latency: 97.70 ms
Throughput(next token): 10.24 tokens/s
```

如果想要和加速前的性能进行对比，您可以修改脚本内的USE\_XFT=False，然后再次执行。

您也可使用测试脚本查看模型生成的结果：

# 在执行前，需要进入conda运行环境(执行过一次即可)

```
conda activate xFT
```

```
cd /root/tools
```

# 修改prompt后执行

```
python llama_run.py
```

## 最佳实践

### 3.2.1.5 多进程执行

在多numa node场景下，推荐使用多进程获得更好的性能指标。您需要根据上文3.2.1.2节 `numactl -H`查到的信息，调整`/root/tools/llama_benchmark_multirank.sh`脚本中的相关内容。具体信息在脚本中均有详细注释。

#### 注意

当前多进程脚本只支持`USE_XFT=True`的模式，关闭xFT加速时无法利用多进程。

# 在执行前，需要进入conda运行环境(执行过一次即可)

```
conda activate xFT
```

```
cd /root/tools
```

```
bash llama_benchmark_multirank.sh
```

在numa node分配正确的情况下，天翼云s8r.8xlarge.2 | 32核 | 64G云主机可达到如下指标：

```
===== Final Performance =====
Use xFT: True
Model dtype: fp16
Input token num 128
Output new token num: 128
Inference Avg Latency: 11331.35 ms
First token Avg Latency: 1245.55 ms
Next token Avg Latency: 78.80 ms
Throughput(next token): 12.69 tokens/s
=====
```

对比来看，如果此时坚持使用单进程模式，带来了跨numa node的性能损耗，则只能达到如下指标：

```
===== Final Performance =====
Use xFT: True
Model dtype: fp16
Input token num 128
Output new token num: 128
Inference Avg Latency: 13058.03 ms
First token Avg Latency: 1051.49 ms
Next token Avg Latency: 93.80 ms
Throughput(next token): 10.66 tokens/s
=====
```

可以明显看到性能下降。

### 3.2.2 Intel® Extension for PyTorch\*(IPEX)实施方法

OPEX的应用和性能测试可直接按单进程执行：

# 在执行前，需要进入conda运行环境(执行过一次即可)

```
conda activate ipex
```

```
python sd_benchmark.py
```

您将会得到如下输出：



## 最佳实践

```
从huggingface下载llama-2-7b-chat模型
```

```
Intel® Extension for PyTorch*
```

```
pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cpu
```

```
pip install intel-extension-for-pytorch
```

```
pip install onecccl_bind_pt --extra-index-url https://pytorch-extension.intel.com/release-
whl/stable/cpu/us/
```

```
pip install transformers accelerate diffusers
```

```
从huggingface下载stable-diffusion-2-1模型
```

```
注意sd模型仓库内放了太多不同版本的模型，实际只需要下载其中一版即可。
```

```
只下载小文件
```

```
GIT_LFS_SKIP_SMUDGE=1 git clone https://huggingface.co/stabilityai/stable-diffusion-2-1
```

```
然后根据需要下载所需文件
```

```
cd stable-diffusion-2-1
```

```
git lfs pull --include=" pytorch_model.bin"
```

```
git lfs pull --include=" diffusion_pytorch_model.bin"
```

```
/root/tools/stable_diffusion_ipex/pipeline.py下载来自
```

```
https://github.com/huggingface/diffusers/blob/main/examples/community/stable_diffusion_ipex.
py
```

### 基于Open WebUI 实现DeepSeek API 调用及外部大模型API 接入最佳实践

#### 使用场景

大模型API提供了接口以便用户更好地进行DeepSeek管理和应用开发。用户在开通天翼云DeepSeek云主机后，利用Open WebUI API即可实现对外提供API调用，同时也可以非常方便的实现外部大模型API接入使用。

本教程使用的GPU云主机规格如下所示，用于部署DeepSeek-r1:7b模型，配置仅供参考。

cpu 16核

内存 64G

GPU: NVIDIA A10\*1 (24GB)

如需体验其它模型版本，请参考[在天翼云使用O1llama运行 DeepSeek的最佳实践-7b版-弹性云主机-最佳实践-AIGC实践 - 天翼云](#) --自定义部署DeepSeek--[步骤二：规格选型](#)，选择合适的云主机。

#### 准备

参考[在天翼云使用O1llama运行 DeepSeek的最佳实践-7b版-弹性云主机-最佳实践-AIGC实践 - 天翼云](#) --快速体验DeepSeek，准备相关设备及模型资源。

#### 认证

OpenWebUI 使用Bearer Token认证保证客户端请求的合法性，其Token为API Key中JWT令牌(Json Web Token)。获取API key方式如下：

1. 登录Open WebUI。

## 登录 Open WebUI

电子邮箱

输入您的电子邮箱

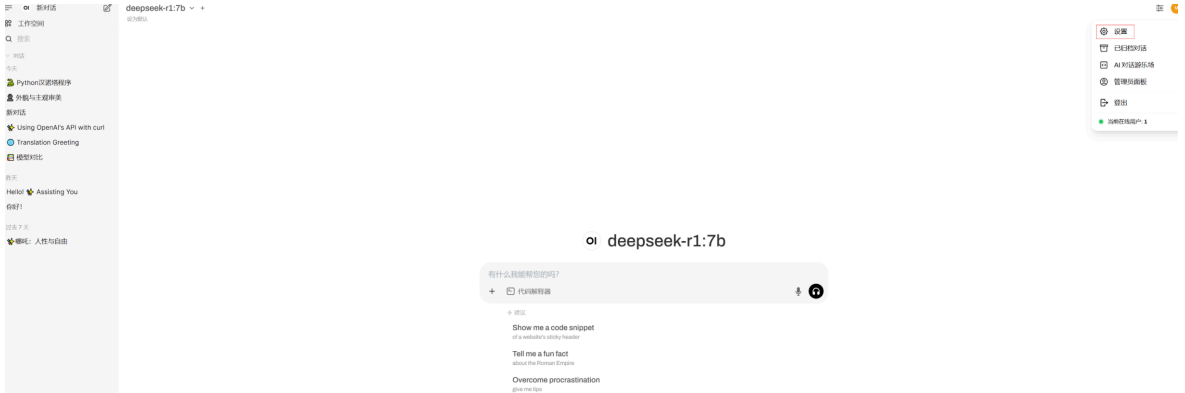
密码

输入您的密码

登录

# 最佳实践

2. 选择设置->账户，选择获取或者新建API KEY。



API接口

1. 列举模型

URL: GET /api/models

描述: 列出可用的模型列表，并提供相关模型的基本信息。

例子:

```
curl -H "Authorization: Bearer API_KEY" http://公网IP:3000/api/models
```

# 最佳实践

## 2. 对话补全

URL: POST /api/chat/completions

描述: 根据输入的上下文, 来让模型补全对话内容。

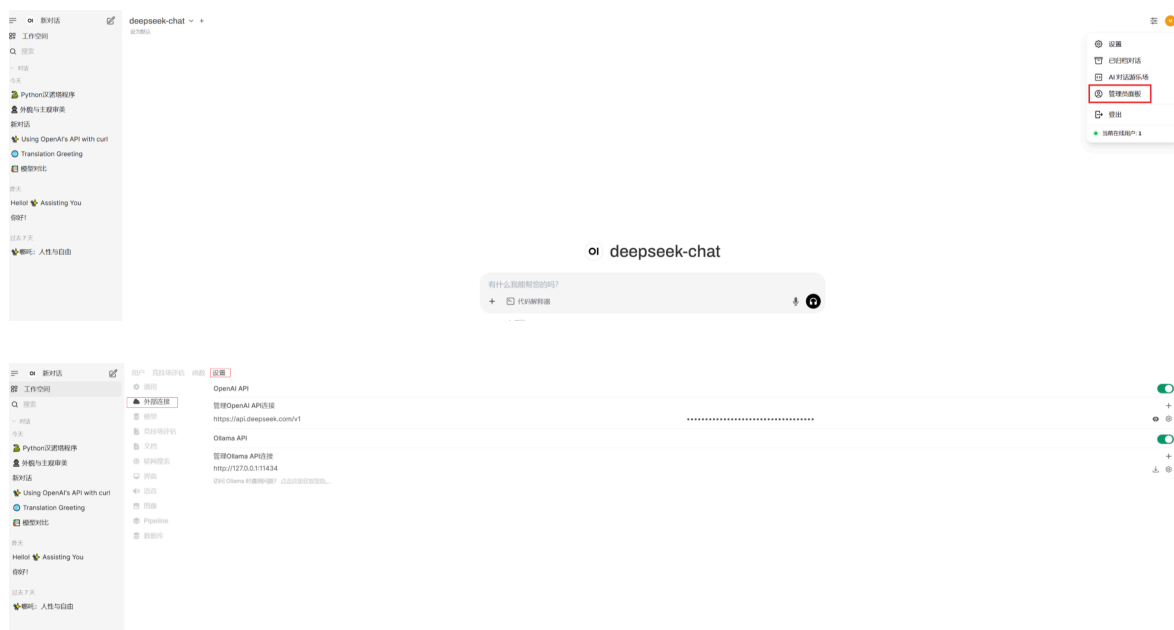
例子:

```
curl -L -X POST 'http://公网IP:3000/api/chat/completions' \
-H 'Content-Type:application/json' \
-H 'Accept:application/json' \
-H 'Authorization:BearerAPI_KEY' \
-d '{
 "model":"deepseek-r1:7b",
 "messages": [
 {
 "content":"用英语翻译一下你好",
 "role":"user"
 }
],
 "stream":false
}'
```

使用外部大模型

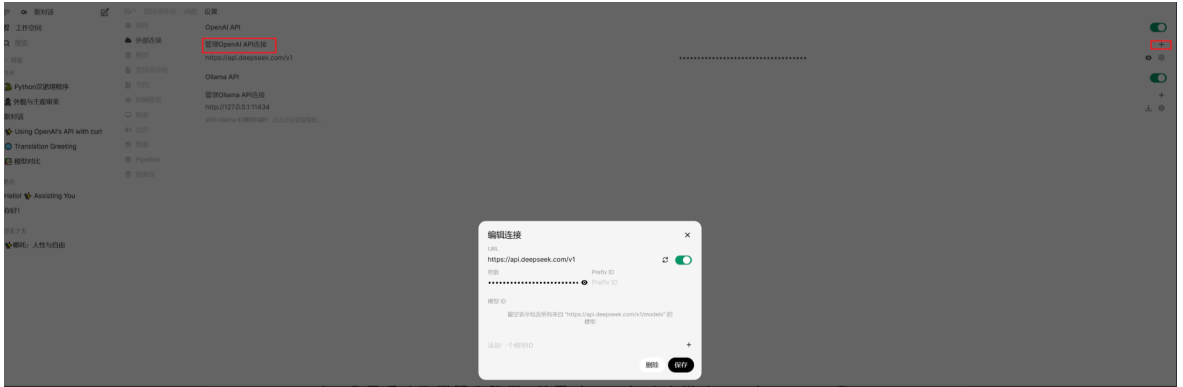
Open WebUI通过云主机方式部署完自己的DeepSeek后, 可以直接嵌入使用外部大模型, 嵌入后可以通过WEB、API进行模型使用。配置方式如下:

1. 登录Open WebUI。
2. 选择管理员面板->设置->外部连接。



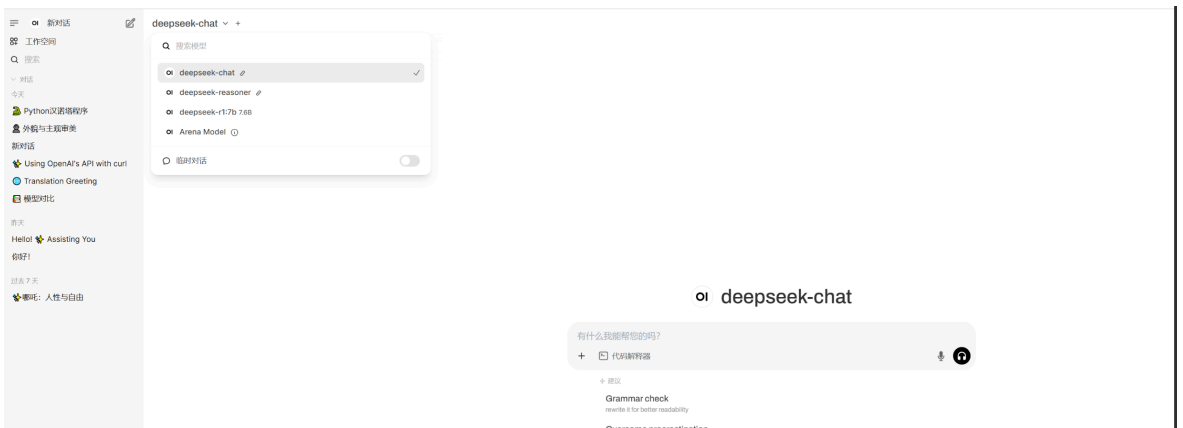
# 最佳实践

3. 选择OpenAI API开启，点击管理OpenAI API连接，新增外部大模型连接，输入外部模型的URL和API KEY。



4. 点击保存。

5. 查看对话新增的大模型，其中DeepSeek-chat和DeepSeek-reasoner为DeepSeek官网新增模型。



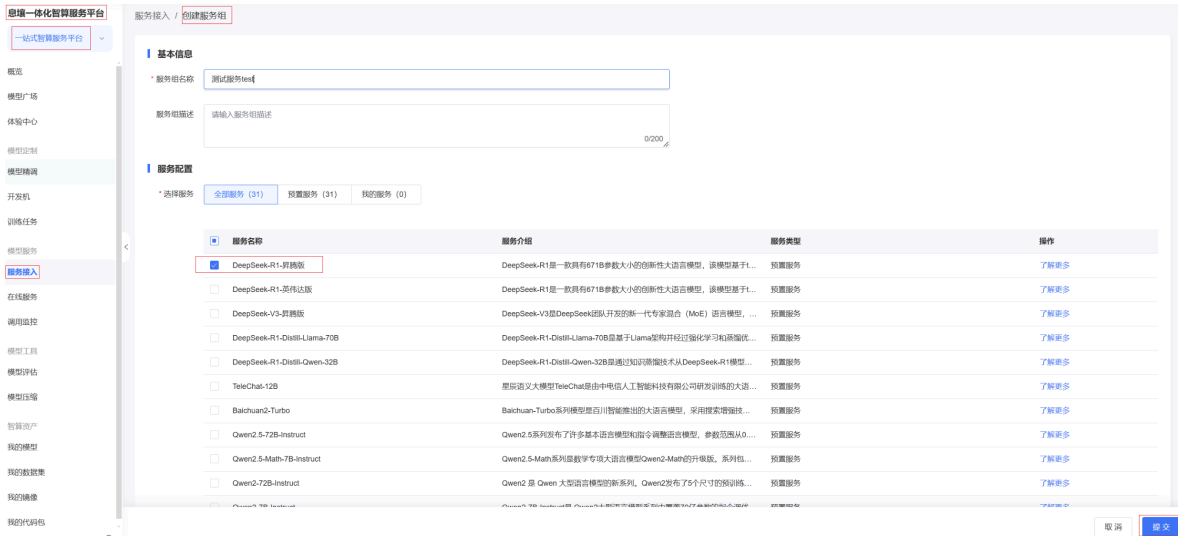
使用天翼云息壤一体化智算服务平台

## 准备工作

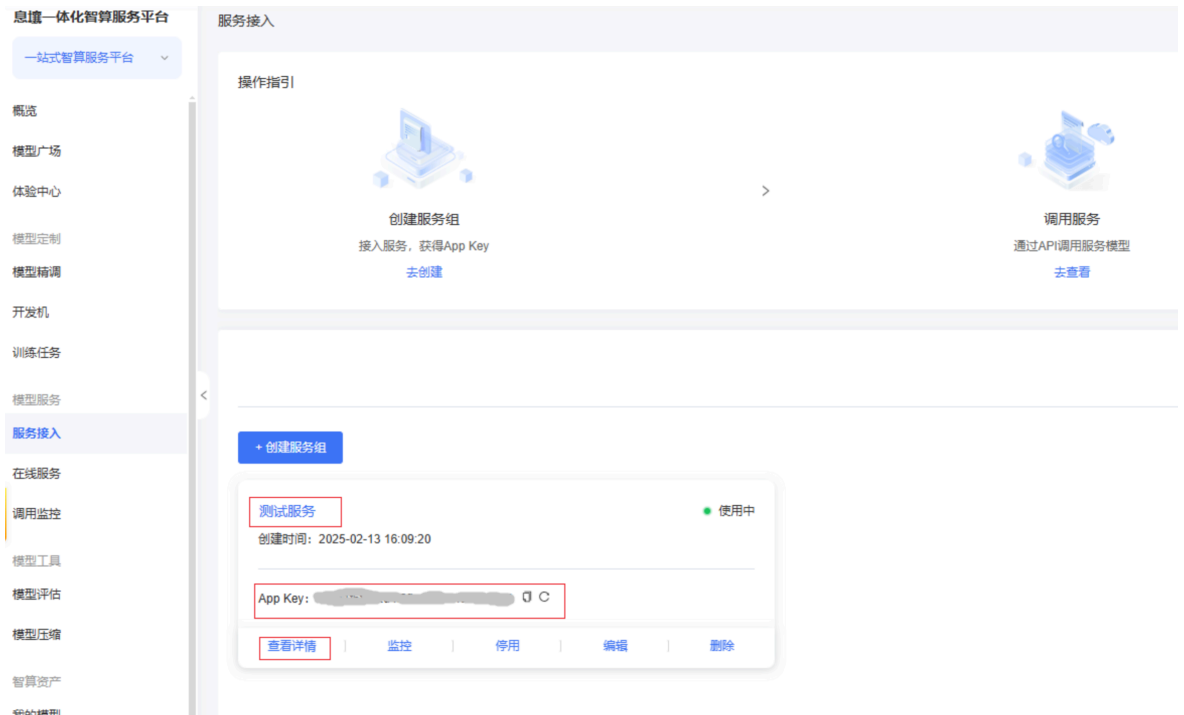
使用天翼云息壤提供的大模型不仅需要提供API KEY，还需要提供对应的模型ID。其获取方式如下：

# 最佳实践

1. 登录天翼云官网，选择**息壤一体化智算服务平台**->一站式智算服务平台->服务接入->创建服务组->DeepSeek-R1-昇腾版，点击提交。



2. 选择测试服务，获App key作为API KEY。选择查看详情->了解更多。



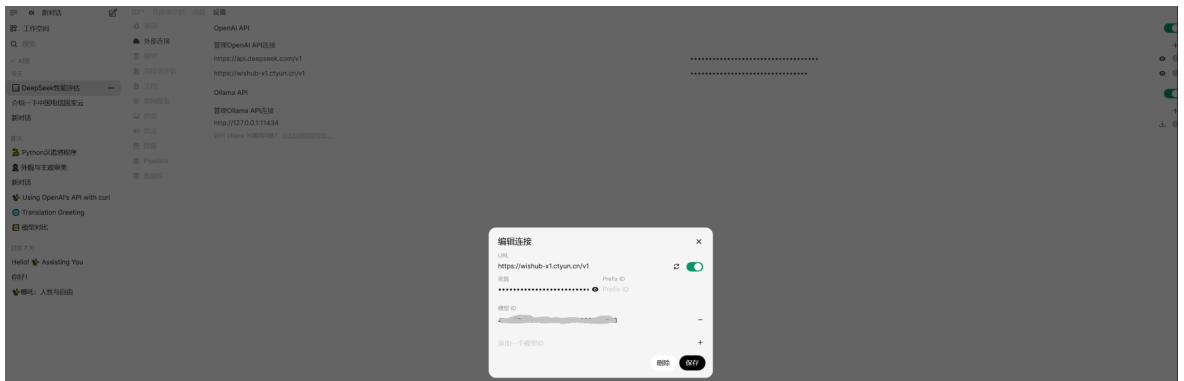
# 最佳实践

3. 选择API文档，获取模型ID和终端地址。



## Open WebUI 接入

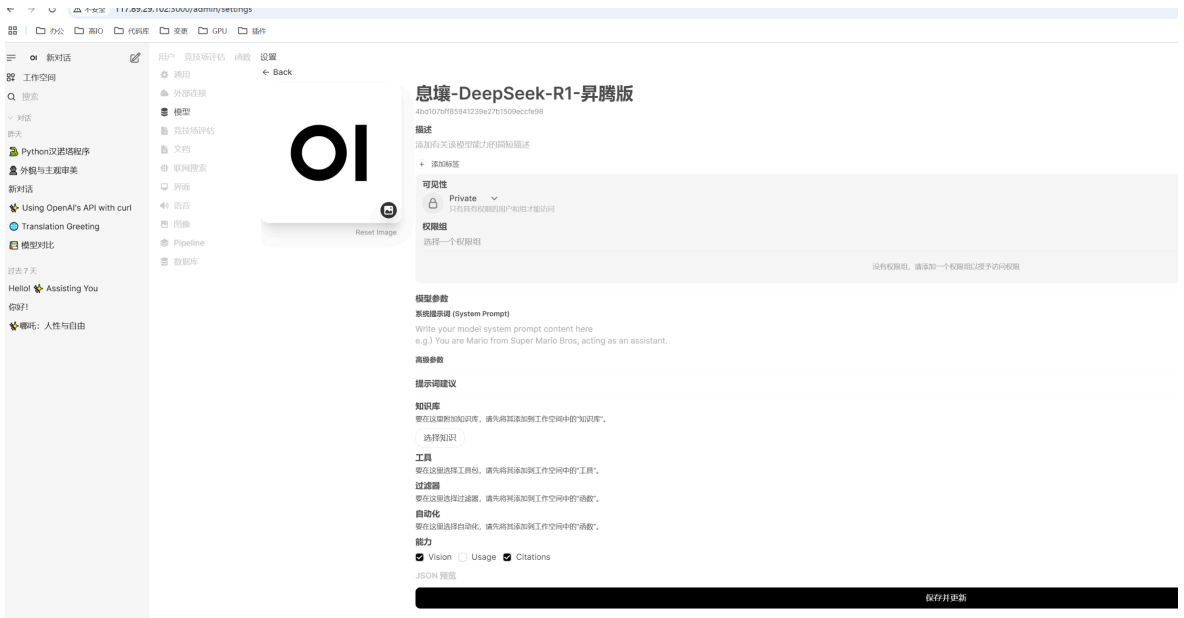
1. 登录Open WebUI并选择管理员面板->设置->外部连接。
2. 选择OpenAI API开启，点击管理OpenAI API连接，新增息壤大模型终端地址、API KEY和模型ID。终端地址需要增加V1版本。



3. 点击保存。

# 最佳实践

## 4. 点击模型，修改模型ID为易理解的模型。



## 5. 查看对话新增的大模型，其中息壤-DeepSeek-R1-昇腾版为新增模型。



### 注意

请妥善保管您的API KEY，不要在公共场合泄露。

## AnythingLLM 协同 Ollama 启用 DeepSeek 大模型最佳实践

### 推荐使用场景

工具	推荐场景	不适用场景
AnythingLLM	企业文档问答、私有知识库管理、需严格权限控制的团队协作	快速测试模型、轻量级个人聊天需求
Open WebUI	个人或开发者快速体验模型、多模型切换测试、多模态交互(如语音/图像)	复杂文档管理、企业级安全与协作需求

## 最佳实践

### 注意

建议使用天翼云提供的 DeepSeek 镜像创建云主机，减少安装过程中可能遇到的问题。镜像选择请参考[在天翼云使用Ollama运行DeepSeek的最佳实践-7B等版本-弹性云主机-最佳实践-AIGC实践 - 天翼云](#)。

# 最佳实践

操作步骤：

## 一、云主机端口配置更新

1. 进入云主机详情页，选择安全组选项，添加新的开放端口。

建议选择新加 3001 端口。添加完成后，可在安全组规则列表中查看新添加的 3001 端口规则，确认其状态为正常生效。



# 最佳实践

方向	类型	优先级	授权策略	协议	端口范围/ICMP类型	远端	描述	操作
入方向	IPv4	1	允许	TCP	3001	0.0.0.0/0		删除 修改

2. 进入控制台，远程登陆即将安装AnythingLLM的云主机。

实例名称/ID	镜像	安全防护	状态	标签	可用区	企业项目	IPv4地址	IPv6地址	CPU架构	规格	操作
...	DeepSeek-R1-7...	未防护	运行中		可用区3	default		--	X86计算	pi7.Axlargt GPU: NVIC	远程登录 更多

## 二、AnythingLLM 服务部署（docker 方式）

1. 获取 AnythingLLM docker image 文件。

由于国内无法访问 docker hub 网站，您需要修改 docker 的服务配置文件，添加国内可访问的源，以便能够顺利拉取 AnythingLLM docker image 文件。操作步骤如下：

- 向 docker 配置文件中添加国内可访问的 docker 镜像源
- 重启 docker 服务
- 拉取 AnythingLLM docker image

# Docker 守护进程（daemon）的配置文件。添加国内可访问镜像源。

```
sudo vi /etc/docker/daemon.json
```

```
重启 docker 服务
```

```
sudo systemctl restart docker
```

```
确认 docker 服务正常
```

```
sudo systemctl status docker
```

```
拉取 anythingllm 镜像
```

# 最佳实践

docker pull mintplexlabs/anythingllm

```
root@ecm-aigc-test:~# sudo vi /etc/docker/daemon.json
root@ecm-aigc-test:~# sudo systemctl restart docker
root@ecm-aigc-test:~# sudo systemctl status docker
● docker.service - Docker Application Container Engine
 Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
 Active: active (running) since Tue 2025-02-18 14:49:44 CST; 11s ago
 TriggeredBy: ● docker.socket
 Docs: https://docs.docker.com
 Main PID: 363483 (dockerd)
 Tasks: 39
 Memory: 28.5M
 CPU: 242ms
 CGroup: /system.slice/docker.service
 └─363483 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
 └─363691 /usr/bin/docker-proxy -proto tcp -host-ip 0.0.0.0 -host-port 3000 -container-ip 172.17.0.2 -container-port 3001
 └─363702 /usr/bin/docker-proxy -proto tcp -host-ip :: -host-port 3000 -container-ip 172.17.0.2 -container-port 3001

Feb 18 14:49:42 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:42.806234391+08:00" level=info msg="Starting up"
Feb 18 14:49:42 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:42.806535786+08:00" level=info msg="OTEL tracing is not configur
Feb 18 14:49:42 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:42.853239755+08:00" level=info msg="[graphdriver] using prior st
Feb 18 14:49:42 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:42.855934725+08:00" level=info msg="Loading containers: start."
Feb 18 14:49:43 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:43.594220275+08:00" level=info msg="Default bridge (docker0) is
Feb 18 14:49:43 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:43.988487598+08:00" level=info msg="Loading containers: done."
Feb 18 14:49:44 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:44.030943927+08:00" level=info msg="Docker daemon" commit=4c9b3b
Feb 18 14:49:44 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:44.030997550+08:00" level=info msg="Daemon has completed initial
Feb 18 14:49:44 ecm-aigc-test dockerd[363483]: time="2025-02-18T14:49:44.044714842+08:00" level=info msg="API listen on /run/docker.so
Feb 18 14:49:44 ecm-aigc-test systemd[1]: Started Docker Application Container Engine.
root@ecm-aigc-test:~# sudo docker pull mintplexlabs/anythingllm
Using default tag: latest
latest: Pulling from mintplexlabs/anythingllm
3713021b0277: Pull complete
4f4fb700ef54: Pull complete
2e34c2dfb39a: Pull complete
d36c412a557a: Pull complete
4e3ea4c2625c: Pull complete
0e60e62892da: Pull complete
6dca9380efe7: Pull complete
578bda425106: Pull complete
c3d83711a1ff: Pull complete
a03b8dfb9685: Pull complete
1e822b21773f: Pull complete
2330397bbdc8: Pull complete
ac3c5f050331: Pull complete
48dc98216779: Pull complete
Digest: sha256:db45b3f9242b87e650e46081b81fb9004b88971c2e53cc85047200df7e110ee8
Status: Downloaded newer image for mintplexlabs/anythingllm:latest
docker.io/mintplexlabs/anythingllm:latest
```

daemon.json 文件添加的内容为:

```
{
 "registry-mirrors": ["http://hub.geekery.cn", "http://hub.littlediary.cn",
 "http://docker.rainbond.cc"]
}
```

daemon.json 文件如下, 仅供参考:

```
"registry-mirrors": ["http://hub.geekery.cn", "http://hub.littlediary.cn", "http://docker.rainbond.cc"]
```

## 2. 启动 AnythingLLM 服务。

- 首先查看下载的 AnythingLLM 容器镜像的 tag
- 启动 AnythingLLM 服务
- 查看 AnythingLLM 服务状态

# 查看当前的容器镜像列表

```
docker image list
```

# 启动 AnythingLLM 服务, 下述 PROT 参数需要与第一部分开放的端口相对应

# 最佳实践

```
docker run -d -e PORT=3001 -e OLLAMA_BASE_URL=http://127.0.0.1:11434 -
e ENABLE_SIGNUP=true -e ENABLE_OPENAI_API=False --add-host=host.docker.internal:
host-gateway --net=host -v anythingLLM:/app/backend/data --name anythingLLM --
restart always mintplexlabs/anythingllm:latest
查看 AnythingLLM 服务状态
docker ps -a
```

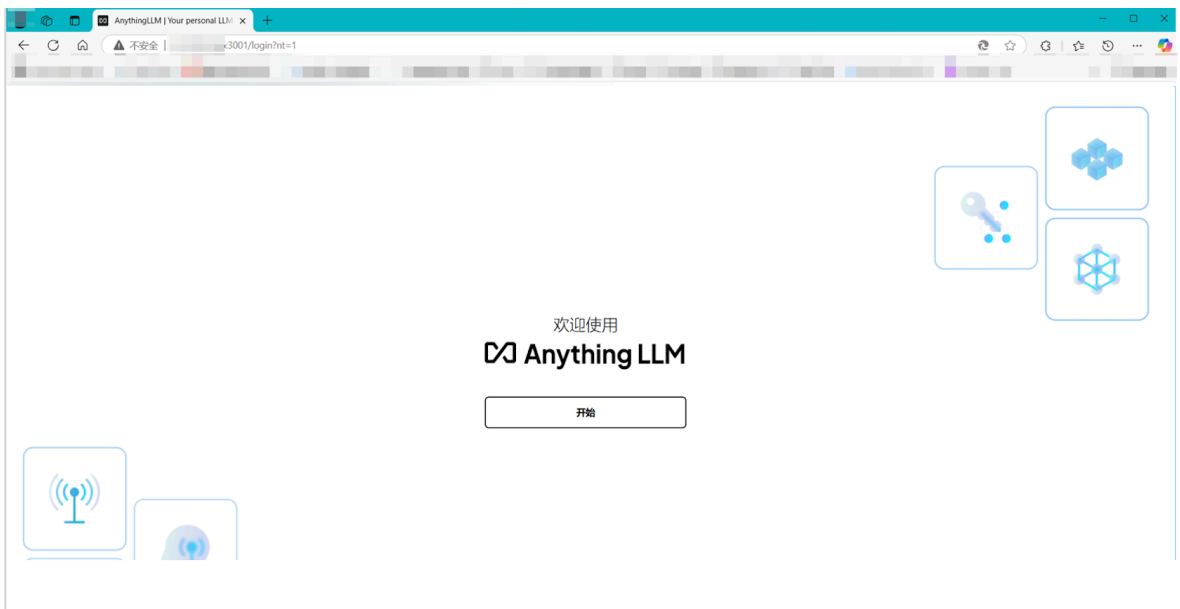
```
root@ecm-aigc-test:~# docker image list
REPOSITORY TAG IMAGE ID CREATED SIZE
mintplexlabs/anythingllm latest 753644541603 2 hours ago 2.51GB
ghcr.io/open-webui/open-webui main a088eea70396 13 days ago 4.27GB
```

```
root@ecm-aigc-test:~# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
9f8e1f3c021854b71b09f5c66d067006509d0aa1d03a26fd93d48fcs mintplexlabs/anythingllm:latest /bin/bash /usr/local- 9 seconds ago Up 8 seconds (healthy) anythingLLM
d72207b0f739 ghcr.io/open-webui/open-webui:main "bash start.sh" 33 minutes ago Up 26 minutes (healthy) open-webui
```

## 三、AnythingLLM 前端页面配置

### 1. 公网访问 AnythingLLM。

在浏览器输入 `http://{your_ip}:3001` (your\_ip 这台云主机的eip)，会出现如下页面



# 最佳实践

## 2. 选择 Ollama 作为模型加载项。

Search LLM providers

- HuggingFace**  
Access 150,000+ open-source LLMs and the world's AI community
- Ollama**  
Run LLMs locally on your own machine.
- Novita AI**  
Reliable, Scalable, and Cost-Effective for LLMs from Novita AI
- LM Studio**  
Discover, download, and run thousands of cutting edge LLMs in a few clicks.
- Local AI**  
Run LLMs locally on your own machine.

**Ollama Model**

deepseek-r1.7b

Choose the Ollama model you want to use for your conversations.

**Max Tokens**

4096

Maximum number of tokens for context and response.

Hide advanced settings ^

**Ollama Base URL**

http://127.0.0.1:11434

Enter the URL where Ollama is running.

**Performance Mode**

Base (Default)

Choose the performance mode for the Ollama model.

**Ollama Keep Alive**

Forever

Choose how long Ollama should keep your model in memory before unloading.  
[Learn more >](#)

**Auth Token**

Enter a Bearer Auth Token for interacting with your Ollama server. Used **only** if running Ollama behind an authentication server.

Ollama Auth Token

3. 用户设置，按页面提示操作，注册账号，跳过用户调查。

## 用户设置

配置你的用户设置。

将有多少用户使用此实例？

只有我

我的团队

common.adminUsername

用户名必须至少为 6 个字符，并且只能包含小写字母、数字、下划线和连字符，不含空格。

管理员账户密码

密码必须至少包含 8 个字符。

默认情况下，你将是唯一的管理员。完成入职后，你可以创建和邀请其他人成为用户或管理员。不要丢失你的密码，因为只有管理员可以重置密码。

## 欢迎使用 AnythingLLM

帮助我们为你的需求打造 AnythingLLM。可选。

你的电子邮件是什么？

you@gmail.com

你将如何使用 AnythingLLM？

用于工作

用于个人使用

其他

你是如何听说 AnythingLLM 的？ (Optional)

Reddit, Twitter, GitHub, YouTube 等 - 让我们知道你是如何找到我们的!

跳过调查

4. 创建工作区。

## 创建你的第一个工作区

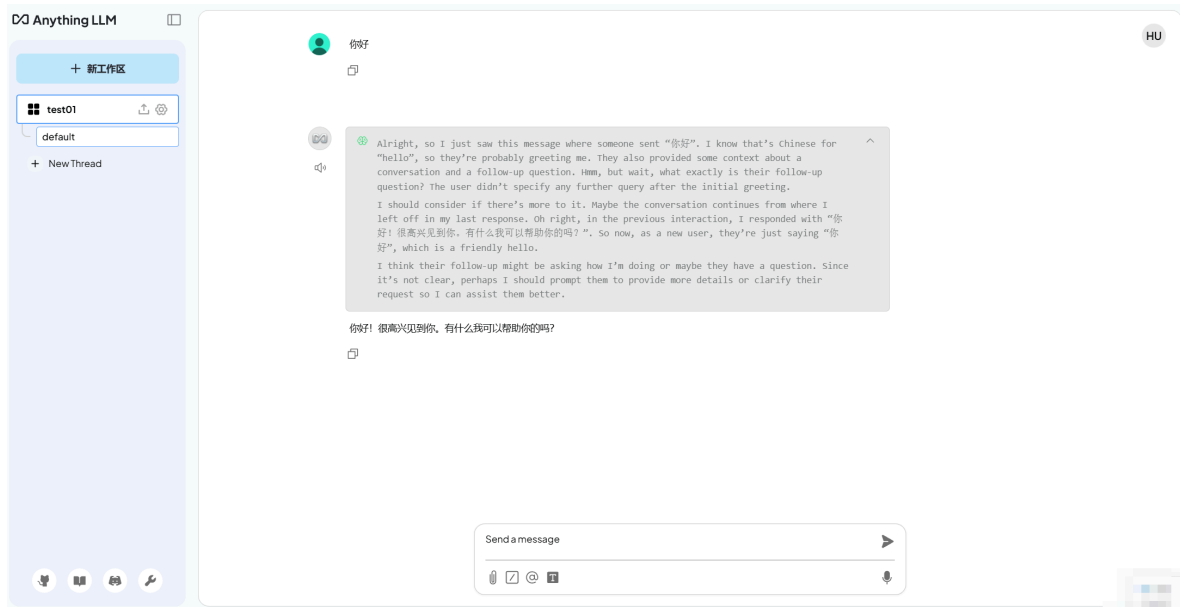
创建你的第一个工作区并开始使用 AnythingLLM。



工作区名称

test01|

5. 点击工作区 test01 打开对话框，输入聊天内容。

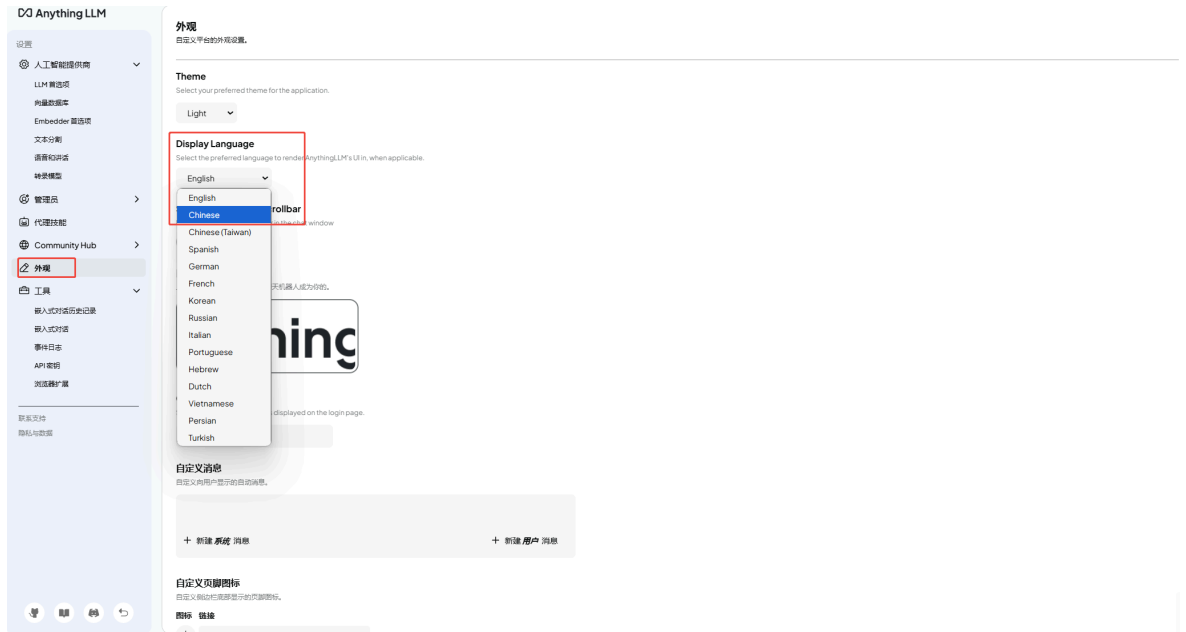


## 四、AnythingLLM 个性化探索

下面给出了一些个性化探索案例，您如果想要了解更多，请访问 [AnythingLLM 社区](#)。

# 最佳实践

1. 设置 AnythingLLM 默认语言格式为中文。参考下图进行设置。



# 最佳实践

## 2. AnythingLLM 启用联网搜索服务。

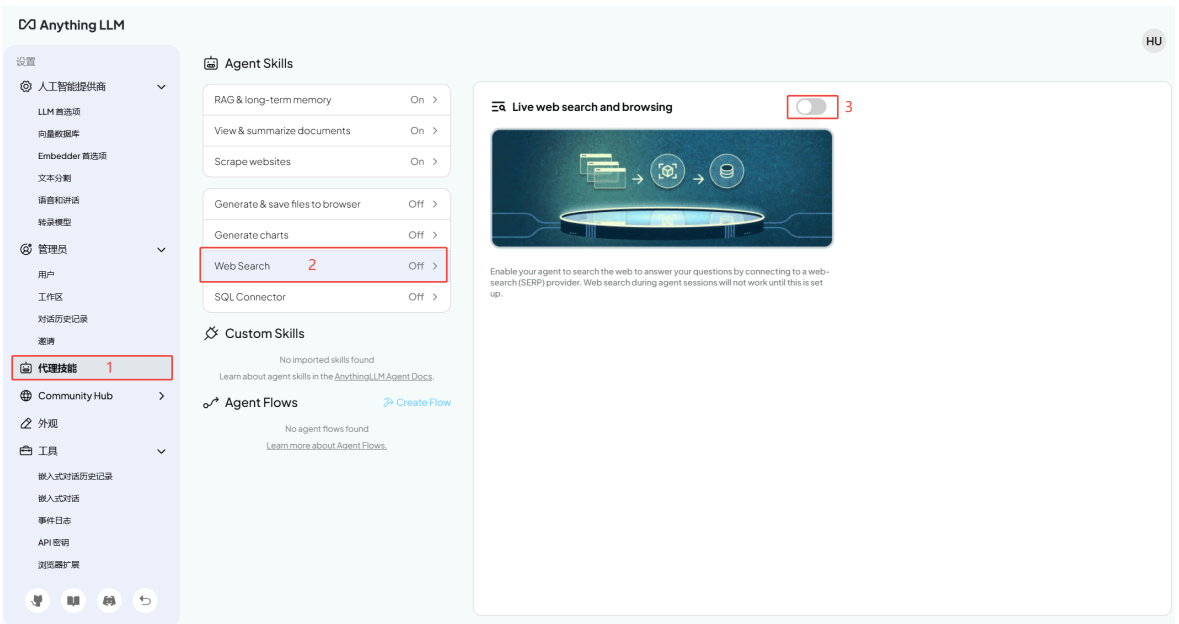
启用 AnythingLLM 的联网搜索服务，需要您提供一个联网搜索引擎 api key 来访问Web资源，目前 AnythingLLM 支持的搜索引擎可以参考如下步骤查看。

### a. 打开 AnythingLLM 的联网搜索功能

- 首先点击 AnythingLLM 的全局设置按钮。



- 参考下图，找到“代理技能”，选择 agent skills 中的 Web Search，点击步骤3，开启对应功能。



### b. 获取 Google 搜索引擎的 ID 以及 API key

# 最佳实践

- 打开下图中步骤3对应的网页，按提示注册或登录账号、命名搜索引擎、选择搜索范围（可输入指定网页或搜索全网），以及选择是否启用图片搜索功能等操作。完成上述设置并通过人机验证后，点击创建，按提示获取搜索引擎的 ID 以及 API key。

The screenshot displays the configuration interface for AI providers. On the left is a navigation sidebar with categories: 人工智能提供商 (AI Providers), 管理员 (Admin), 代理技能 (Agent Skills), Community Hub, 外观 (Appearance), and 工具 (Tools). The main content area is divided into several sections:

- AI Providers:** A list of features with toggle switches: RAG & long-term memory (On), View & summarize documents (On), Scrape websites (On), Generate & save files to browser (Off), Generate charts (Off), Web Search (On), and SQL Connector (Off).
- Custom Skills:** A section with a search icon and the text "No imported skills found".
- Agent Flows:** A section with a search icon and the text "No agent flows found".
- Live web search and browsing:** A section with a toggle switch (1) that is turned on. Below it is a diagram showing a search process. Text below the diagram reads: "Enable your agent to search the web to answer your questions by connecting to a web-search (SERP) provider. Web search during agent sessions will not work until this is set up."
- Google Search Engine:** A section with a dropdown menu (2) showing "Google Search Engine". Below it is a link (3) that says "You can get a free search engine & API key from Google here."
- Search engine ID and Programmatic Access API Key:** Two input fields at the bottom, one containing "Google Search Engine Id" and the other "Google Search Engine API Key".

## 创建新的搜索引擎

请先提供一些关于您的引擎的基本信息。创建之后，您将能够进一步自定义引擎的配置（语言、区域等）。 [了解详情](#)

命名搜索引擎

搜索引擎名称

test01

搜索什么内容? ⓘ

在特定网站或网页中搜索

您可以添加以下任意内容：

单个网页：www.example.com/page.html

整个网站：www.mysite.com/\*

网站的某些部分：www.example.com/docs/\* 或

www.example.com/docs/

整个网域：\*.example.com

输入网站或网页

添加

在整个网络中搜索

搜索设置 ⓘ

图片搜索

安全搜索



进行人机身份验证



reCAPTCHA  
隐私权 - 使用条款

点击“创建”即表示您同意 [服务条款](#)。

创建

- 创建成功后，得到如下图示内容：



### 新的搜索引擎已创建

复制以下代码，并将其粘贴到网站的 `<body>` 部分，也就是您希望在其中显示搜索框和搜索结果的位置。

```
<script async src="https://cse.google.com/cse.js?cx=[ID]">
</script>
<div class="gcse-search"></div>
```



[预览](#)

[自定义](#)

- 进入自定义页面，获取搜索引擎的 ID。

## 概览

### 基本

搜索引擎名称

test01 

说明

[添加说明](#)

代码

[获取代码](#)

搜索引擎 ID

7  76 

公开网址



- 随后下滑到页面底部，点击程序化访问模块中的“开始使用”按钮。

## 成员

[删除](#)[添加](#)

成员	角色	已验证
<input type="checkbox"/> 	所有者	是

第 1-1 行 (共 1 行) < >

## 广告

公益组织、经认证的教育机构和政府机构可使用无广告的免费版可编程搜索引擎。 [了解详情](#)

可以通过改为按搜索查询付费，停用广告功能。开始使用 [Programmable Search Element Paid API](#) 设置结算方法并生成 API 密钥。

输入 Programmable Search Element Paid API 密钥可启用相关功能，留空可停用该功能。

Paid Element API 密钥

[添加 API 密钥](#)

## 程序化地访问

Custom Search JSON API  
每天的查询数上限为 10,000 次。

[开始使用](#)

- 跳转到如下页面，点击获取密钥：

## Custom Search JSON API: 简介



发送反馈

本文档可帮助您熟悉 Custom Search JSON API 及其用法。


### 前期准备

#### 创建可编程搜索引擎

通过调用 API，用户可针对可编程搜索引擎的现有实例发出请求。因此，在使用 API 之前，您需要在**控制台**中创建一个。按照**教程**中的说明详细了解不同的配置选项。创建之后，您可以在**概览页的基本部分**找到**搜索引擎 ID**。这是 API 使用的 `cx` 参数。

#### 使用 API 密钥向 Google 标识您的应用

Custom Search JSON API 需要使用 API 密钥。API 密钥是向 Google 标识客户端的一种方式。

- 可编程搜索引擎 (免费版) 用户: 

在您获得 API 密钥后，您的应用便可将查询参数 `key=yourAPIKey` 附加到所有请求网址中。API 密钥可以安全地嵌入到网址中，无需任何编码。

- 按照提示输入api名称，查看api使用协议并按照您的想法选择是否接受：

## Enable Custom Search API

Enter new project name

AnythingLLMTest

I agree that my use of any [services and related APIs](#) is subject to compliance with the applicable [Terms of Service](#).

Yes  No

BACK

CANCEL

NEXT

- 按照提示操作后得到api调用的key。

You're all set!

You're ready to start developing with Custom Search API

YOUR API KEY

AIz [REDACTED] j6Cc

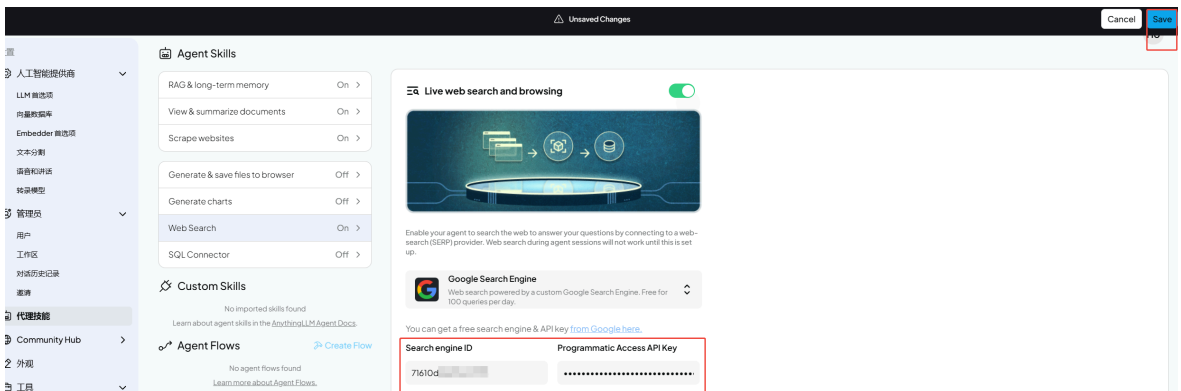


**i** To improve your app's security, restrict this key's usage in the [API Console](#).

DONE

c. 启用 AnythingLLM 的联网搜索功能

- 配置 Google Search Engine：回到 AnythingLLM 联网搜索的页面，填入上述步骤获取的 ID 和 API key，点击右上角保存。



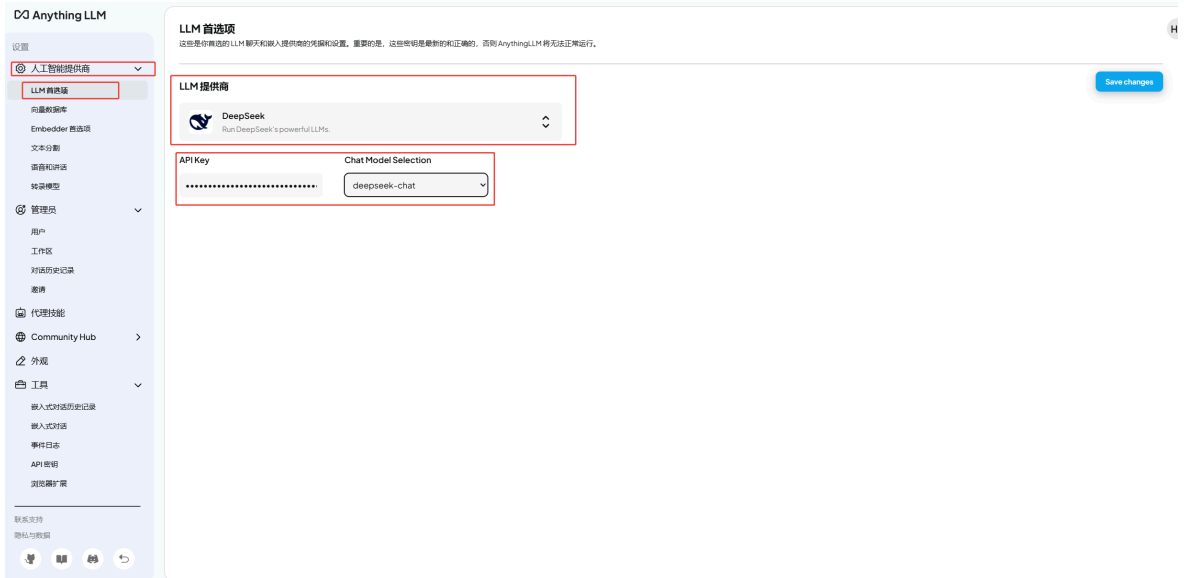
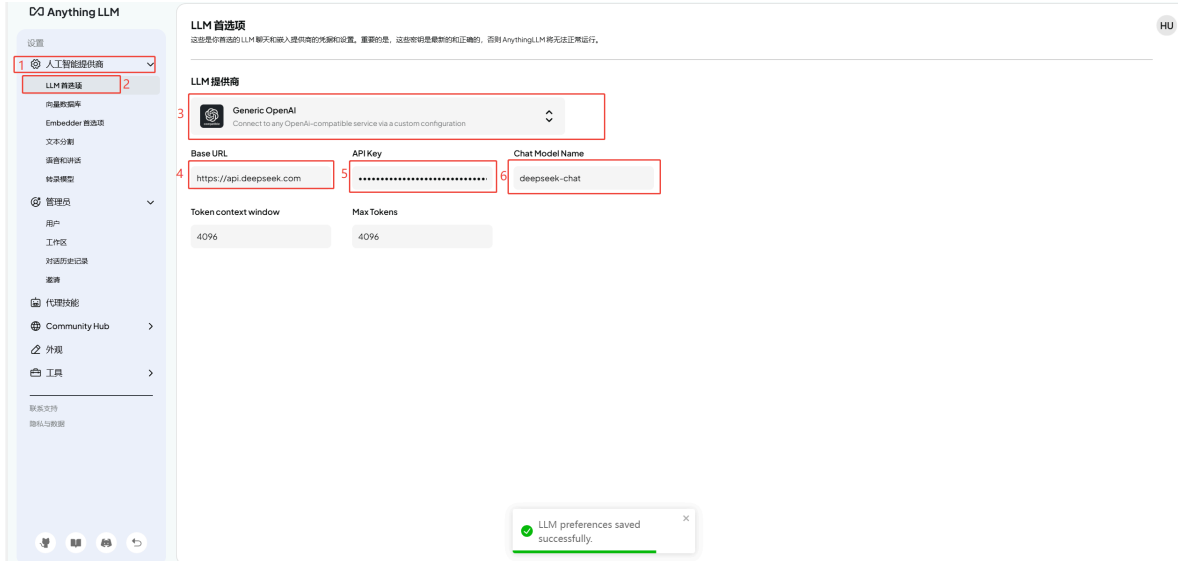
- 保存成功后得到如下提示：
- 在对话框以 @agent 启用联网功能：



# 最佳实践

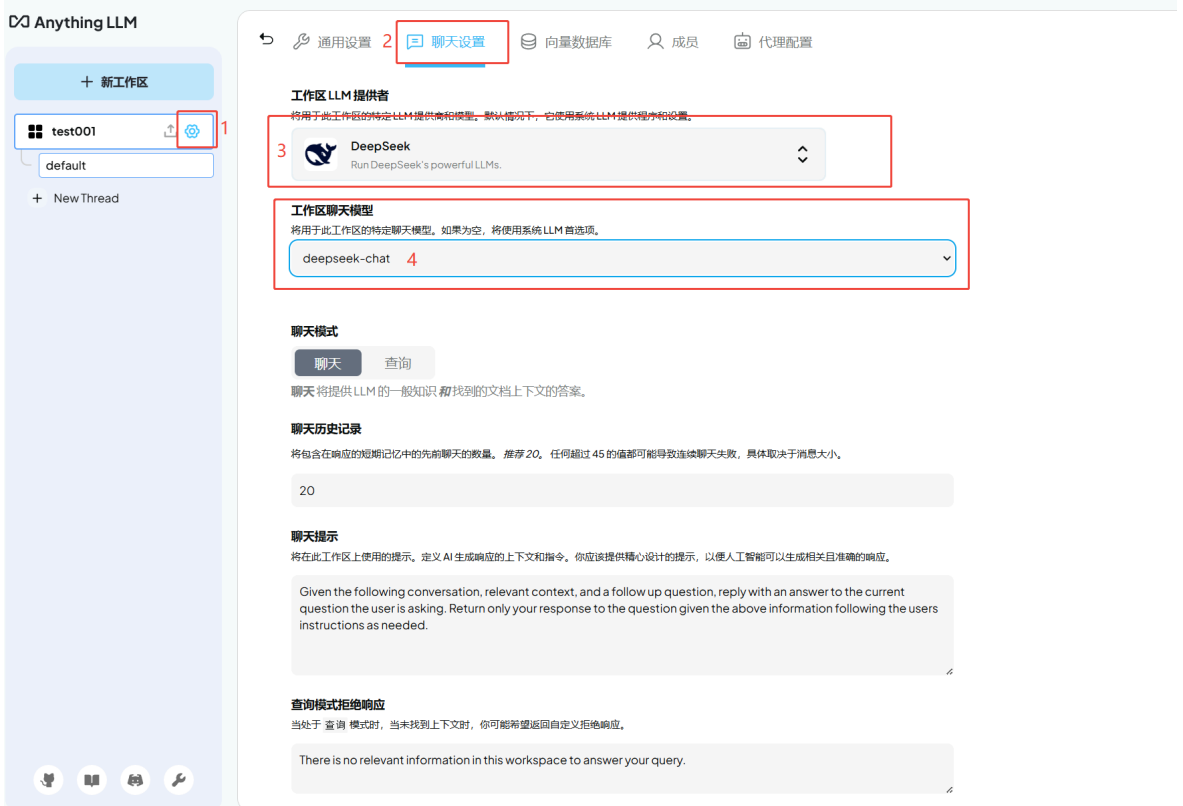
## 3. AnythingLLM 使用 DeepSeek API

- 前往 [DeepSeek API 开放平台](#) 注册一个账号，购买 API 使用资格。
- 参考下述操作步骤，选择 Generic OpenAI 选项，填入 API 认证信息。或者选择 DeepSeek 选项，填入 API 认证信息。



- 到您想要使用的工作区，点击设置，更新当前聊天设置，以启用 DeepSeek API 服务。

# 最佳实践



d. 需要点击下图按钮来保存配置。

# 最佳实践

将用于此工作区的特定聊天模型。如果为空，将使用系统 LLM 首选项。

deepseek-chat

## 聊天模式

聊天

查询

聊天 将提供 LLM 的一般知识 和 找到的文档上下文的答案。

## 聊天历史记录

将包含在响应的短期记忆中的先前聊天的数量。推荐 20。任何超过 45 的值都可能导致连续聊天失败，具体取决于消息大小。

20

## 聊天提示

将在此工作区上使用的提示。定义 AI 生成响应的上下文和指令。你应该提供精心设计的提示，以便人工智能可以生成相关且准确的响应。

Given the following conversation, relevant context, and a follow up question, reply with an answer to the current question the user is asking. Return only your response to the question given the above information following the users instructions as needed.

## 查询模式拒绝响应

当处于 查询 模式时，当未找到上下文时，你可能希望返回自定义拒绝响应。

There is no relevant information in this workspace to answer your query.

## LLM 温度

此设置控制你的 LLM 回答的“创意”程度

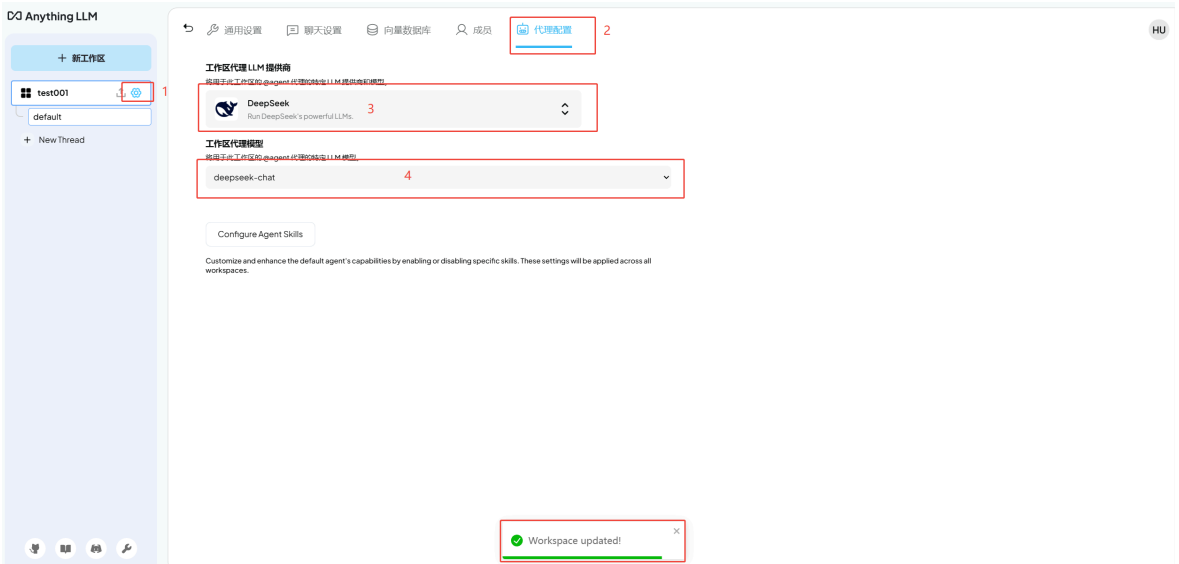
数字越高越有创意。对于某些模型，如果设置得太高，可能会导致响应不一致。

大多数 LLM 都有各种可接受的有效值范围。请咨询你的 LLM 提供商以获取该信息。

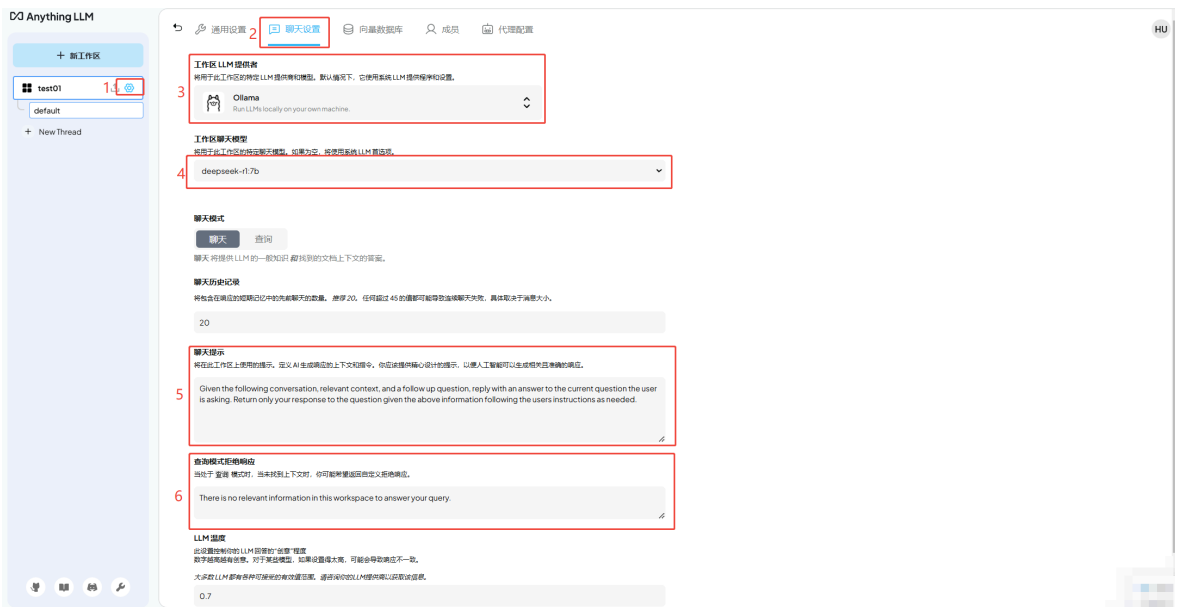
0.7

Update workspace

# 最佳实践



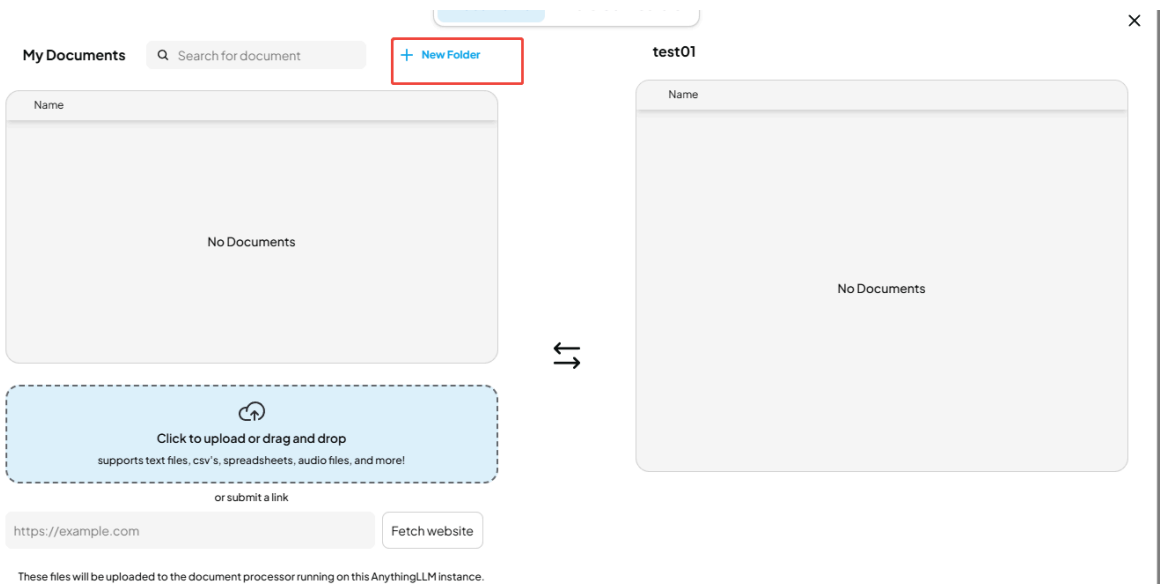
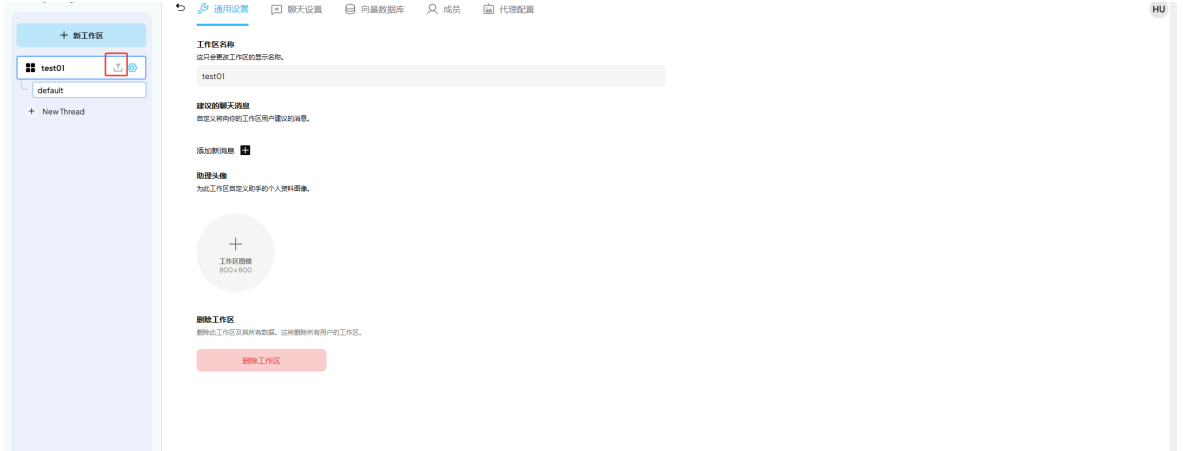
## 4. 调整对话模型及提示词



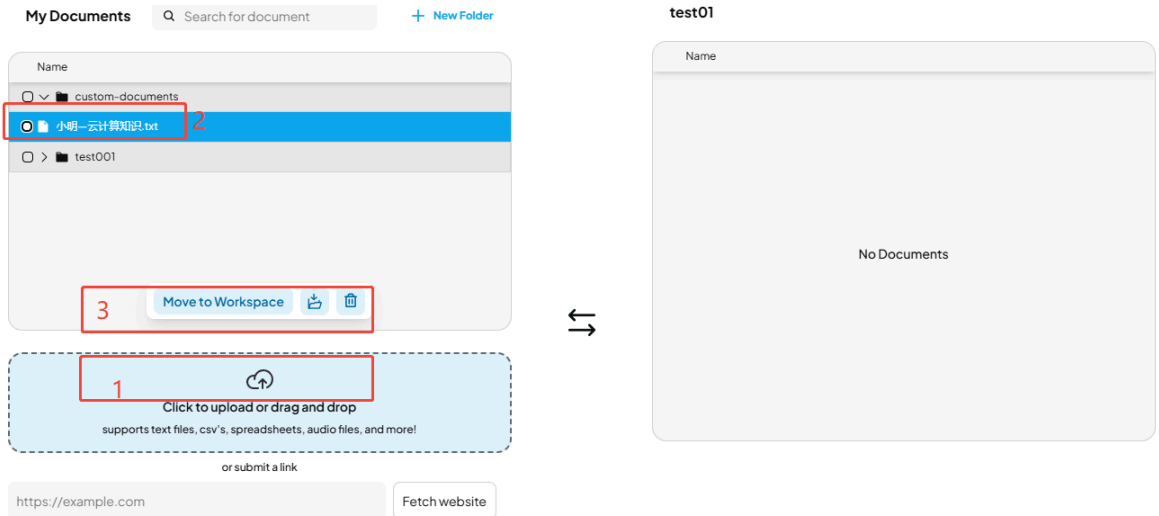
# 最佳实践

## 5. 知识库构建

### a. 上传文件。



# 最佳实践



其中第3步需要第二步选择上传的文件后才会出现按钮，点击“Move to Workspace”，等待出现成功提示即可。



## b. 基于知识库的对话。

首先选择上传了语料的工作区，随后直接提问即可。

# 最佳实践



## CherryStudio 协同 Ollama 启用 DeepSeek 大模型

### 应用场景

使用CherryStudio加载DeepSeek大模型的应用场景如下：

# 最佳实践

1. **学习研究**：学生用其整理复习资料、辅助学术研究、助力论文写作；研究人员用于文献综述、分析问题等。
2. **工作办公**：项目管理中生成报告、分析风险；内容创作时构思创意、撰写文案；代码开发里获取代码示例、解决问题。
3. **生活娱乐**：作为智能聊天伙伴解闷陪伴，为创意活动如绘画等提供灵感启发。
4. **知识管理**：个人可构建知识体系，团队能实现知识共享。

## 注意

建议使用天翼云提供的DeepSeek镜像创建云主机，减少安装过程中可能遇到的问题。镜像选择请参考[在天翼云使用Ollama运行DeepSeek的最佳实践-7B等版本-弹性云主机-最佳实践-AIGC实践 - 天翼云](#)。

## 操作步骤

### 一、云主机端口配置更新

首先，您需要登录天翼云控制中心，对部署DeepSeek大模型的云主机添加新的端口开放操作。通过天翼云控制台进入部署了DeepSeek大模型的云主机详情页，选择安全组，添加新的开放端口，建议选择新加114343端口，该端口是 Ollama 服务默认监听的端口。添加完成后，可在安全组规则列表中查看新添加的11434 端口规则，确认其状态为正常生效。



# 最佳实践

## 添加规则

\* IP版本  IPv4  IPv6

\* 方向  入方向  出方向

\* 授权策略:  允许  拒绝

\* 优先级:

\* 协议

TCP、UDP报文分片后, 分片不带有端口信息, 需要将端口范围指定为1-65535, 不进行端口过滤。

\* 端口范围

\* 源地址:  IP  安全组  前缀列表

描述

方向	类型	优先级	授权策略	协议	端口范围/ICMP类型	远端	描述	操作
入方向	IPv4	1	允许	TCP	11434	0.0.0.0/0		删除 修改

随后进入控制台, 远程登陆即将安装AnythingLLM的云主机。

实例名称/ID	镜像	安全防护	状态	标签	可用区	企业项目	IPv4地址	IPv6地址	CPU架构	规格	操作
DeepSeek-R1-7...	DeepSeek-R1-7...	未防护	运行中		可用区3	default		--	X86计算	pi7.4xlarge GPU: NVIC	远程登录 更多

# 最佳实践

## 二、Cherry Studio桌面软件安装


### 1. 下载Cherry Studio桌面软件

访问[Cherry Studio官网](#)，选择合适的软件版本并下载，本案例中以Cherry-Studio-0.9.27-setup.exe (88.1 MB) - Windows标准安装包为例。

Cherry-Studio-0.9.27-setup.exe (88.1 MB) - Windows标准安装包【备用下载线路2】

### 2. 安装Cherry Studio软件

找到您下载的Cherry Studio安装包，按照提示进行安装。

 Cherry-Studio-0.9.24-setup.exe	2025/2/19 13:44	应用程序	86,689 KB
------------------------------------------------------------------------------------------------------------------	-----------------	------	-----------

## 三、Cherry Studio实现自部署DeepSeek大模型对话

### 1. 云主机ollama服务配置更新

对云主机上的ollama服务配置文件进行更新，添加第一部分中开放的端口。

如果您想同时在OpenWebUI以及其它服务上面启用，建议您使用11434端口，这样不用重启OpenWebUI服务。

# 编辑 ollama.service 服务参数，添加监听所有 ip 设置

vim /etc/systemd/system/ollama.service

添加内容如下：

```
Environment="OLLAMA_HOST=0.0.0.0"
```

```
Environment="OLLAMA_ORIGINS=*"
```

```
[t]
Description=Ollama Service
Wants=network-online.target

[Service]
ExecStart=/usr/local/bin/ollama serve
User=ollama
Group=ollama
Restart=always
RestartSec=1
Environment="PATH=/usr/local/cuda/bin:/usr/local/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin"
Environment="OLLAMA_HOST=0.0.0.0"
Environment="OLLAMA_ORIGINS=*"

[Install]
WantedBy=default.target
```

随后按照以下操作，重启ollama服务。

# 重新加载系统管理器的配置文件，使得ollama.service文件修改生效

```
sudo systemctl daemon-reload
```

# 重启 ollama 服务

```
sudo systemctl restart ollama.service
```

# 查看 ollama 服务监听端口

## 最佳实践

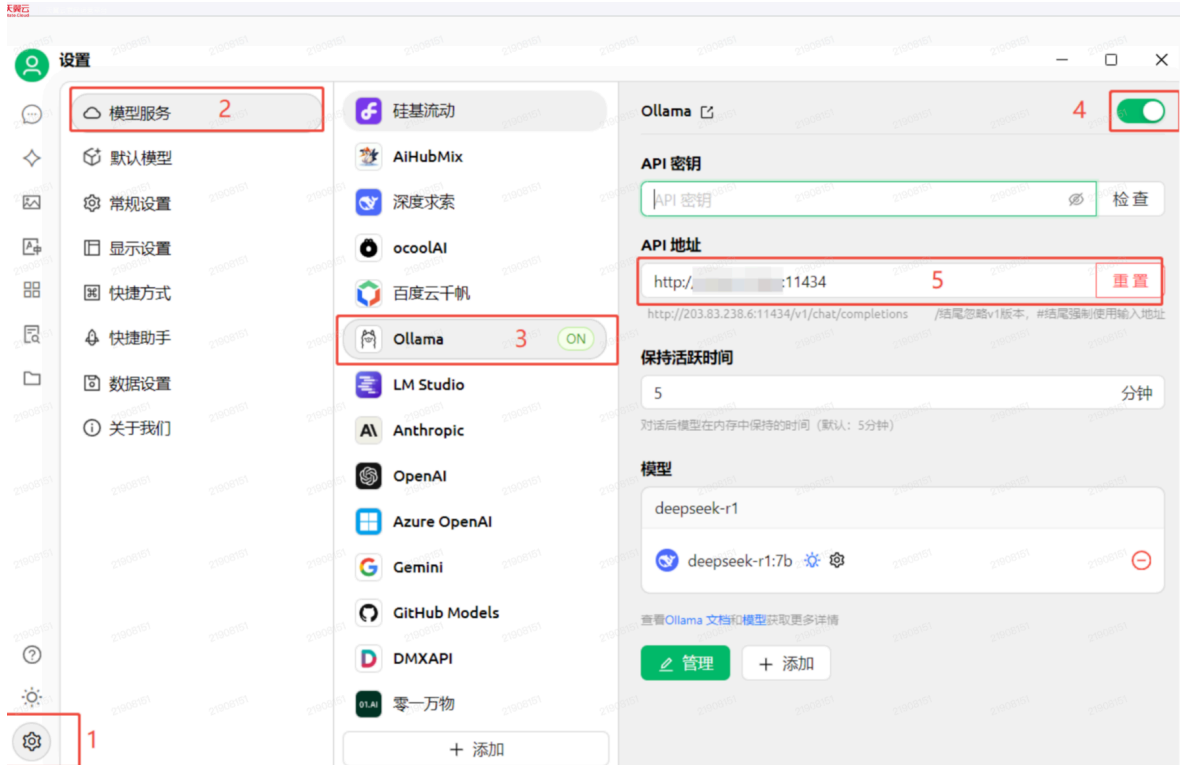
```
sudo netstat -tulnp | grep 11434
```

```
root@ecm-aigc-test:~# vim /etc/systemd/system/ollama.service
root@ecm-aigc-test:~# sudo systemctl daemon-reload
root@ecm-aigc-test:~# sudo systemctl restart ollama.service
root@ecm-aigc-test:~# sudo netstat -tulnp | grep 11434
tcp6 0 0 :::11434 :::* LISTEN 31184/ollama
root@ecm-aigc-test:~# vim /etc/systemd/system/ollama.service
```

# 最佳实践

## 2. Cherry Studio中加载云主机上的ollama服务

首先，打开安装好的Cherry Studio桌面程序，找到设置按钮，请参考下图步骤进行设置，第5步中的ip是您云主机公网的访问ip。



随后点击下方的添加按钮，输入预装的deepseek-r1:7b，点击添加模型。

# 最佳实践

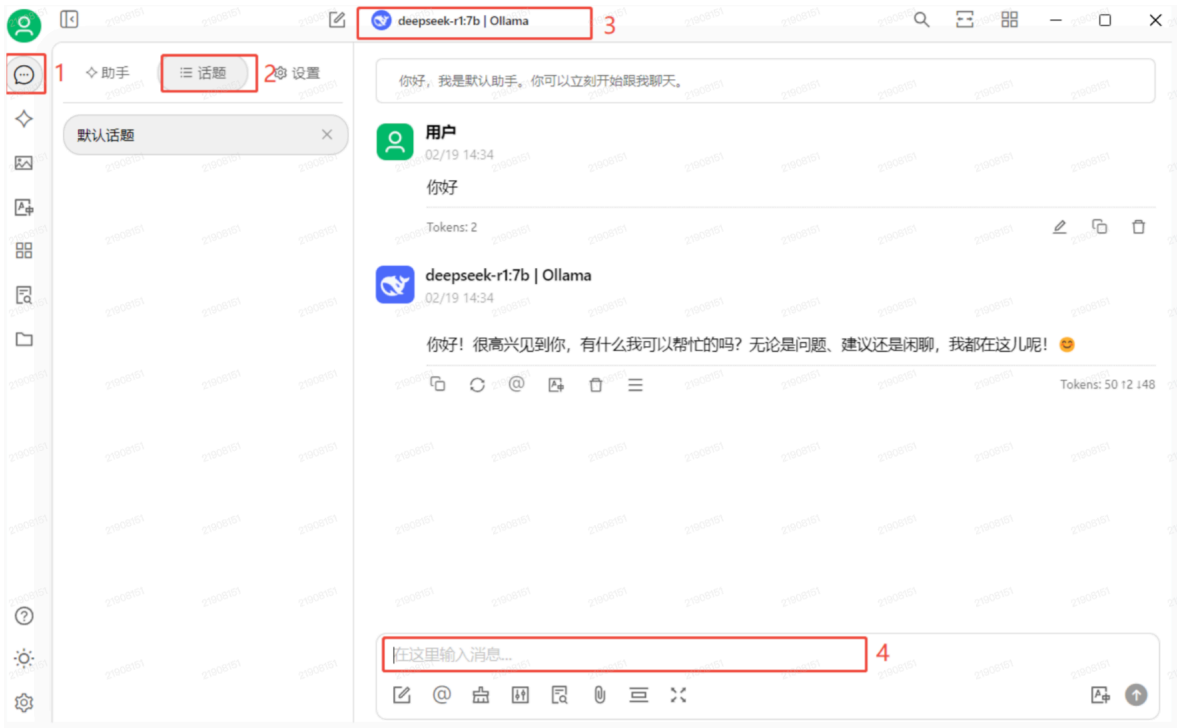
## 添加模型 ✕

\* 模型 ID ?

模型名称 ?

分组名称 ?

最后，按照下图提示进行对话吧。



# 最佳实践

## FAQ

### 1. 如何基于Ollama使用其它大模型？

请参考ollama模型站支持的大模型，在云主机上面下载您想要下载的大模型。

```
ollama pull $mode-name
```

```
root@ecm-aigc-test:~# ollama pull deepseek-r1:1.5b
pulling manifest
pulling aabd4debf0c8... 0% |
```

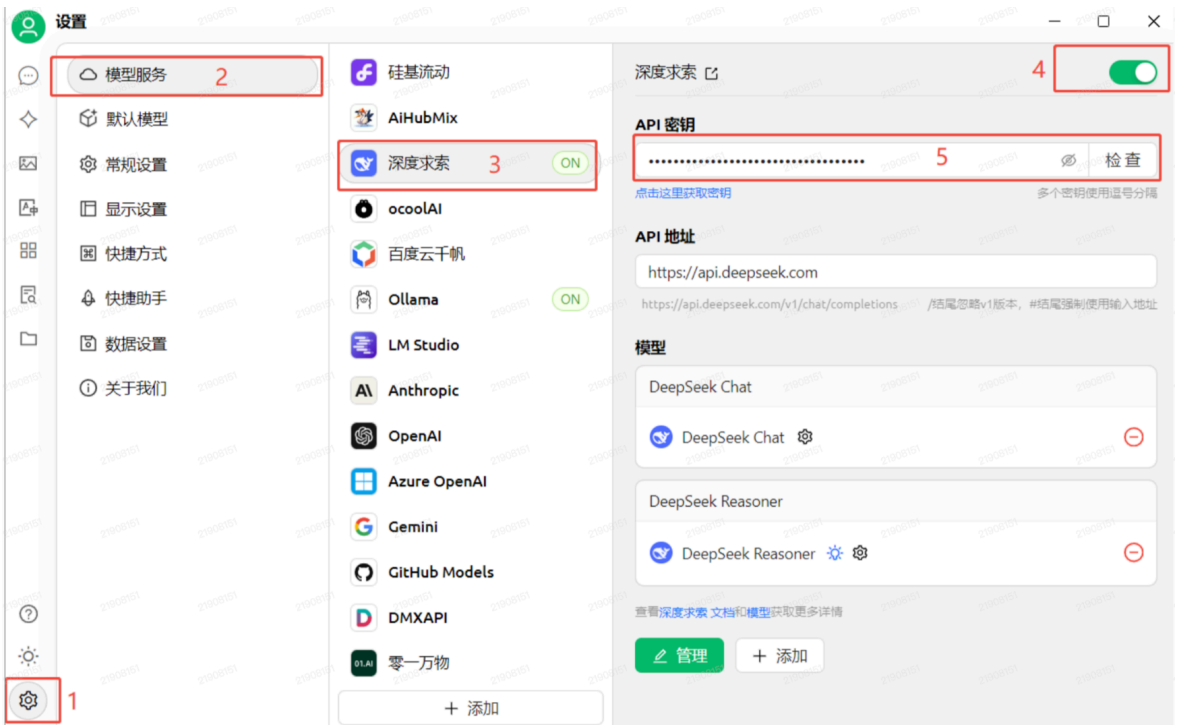
随后在Cherry Studio上面选择启用对应的大模型，相关操作参考下图。



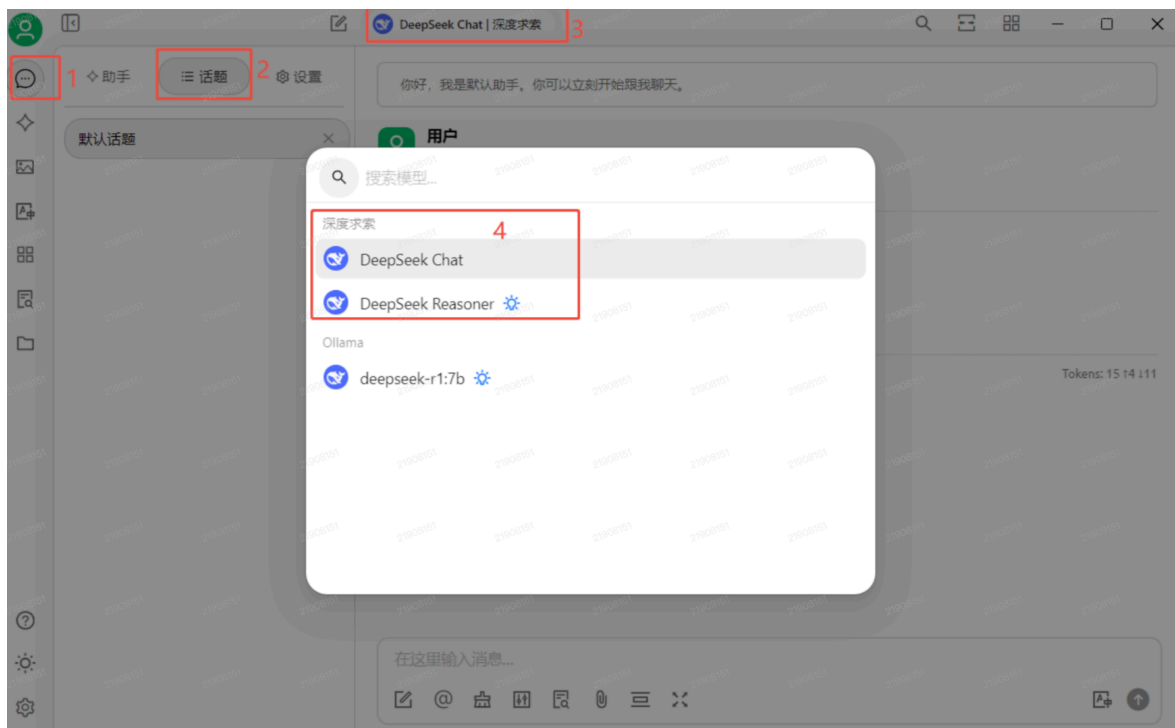
# 最佳实践

## 2. 如何配置外部大模型？

参考下图步骤，填入您获取的其它大模型调用的API密钥，启用相应大模型服务。



随后，在对话框中选择对应的大模型，即可使用对应大模型服务。



### 3. 如何启用联网搜索功能？

Cherry Studio目前已经支持的联网模型服务商有：

- Google Gemini
- 腾讯混元
- 智谱AI

注意

如果您想了解更多的Cherry Studio功能，请访问[Cherry Studio文档](#)。

## 天翼云CPU实例部署DeepSeek-R1模型最佳实践

### 1 背景

#### 1.1 大模型算力需求

大模型因其参数规模庞大、结构复杂，通常需要强大的计算资源来支持其推理过程，这使得算力成为大模型应用的核心要素。随着DeepSeek-R1模型的问世，各行各业纷纷展开了关于如何接入大模型能力的广泛调研与探索，市场对大模型推理算力的需求呈现出爆发式增长的趋势。例如在医疗、金融、零售等领域，企业迫切希望通过接入DeepSeek大模型来提升决策效率和业务能力，从而推动行业的创新发展。在这一背景下，算力的供给和优化成为推动大模型落地应用的重要因素。

# 最佳实践

近年来，CPU制程和架构的提升以及**英特尔® 高级矩阵扩展AMX**（Advanced Matrix Extensions）加速器的面世带来了算力的快速提升。英特尔对大模型推理等多个AI领域持续深入研究，提供全方位的AI软件支持，兼容主流AI软件且提供多种软件方式提升CPU的AI性能。目前，已有充分的数据显示CPU完全可以用于大模型推理场景。

## 1.2 CPU算力用于大模型推理的适用场景

CPU适用于以下大模型推理场景：

场景1: 大模型推理需要的内存超过了单块GPU的显存容量，需要多块或更高配GPU卡，采用CPU方案，可以降低成本；

场景2: 应用请求量小，GPU利用率低，采用CPU推理，资源划分的粒度更小，可有效降低起建成本；

场景3: GPU资源紧缺，CPU更容易获取，且可以胜任大模型推理。

## 2 天翼云EMR实例DeepSeek-R1-Distill-Qwen-7B蒸馏模型部署实践

本节内容主要介绍如何在天翼云EMR实例上，基于Intel® xFasterTransformer加速库和vllm推理引擎完成模型部署，并展示相关性能指标。

### 2.1 服务部署

为了方便用户使用，天翼云联合英特尔制作了一键部署的云主机镜像，内置DeepSeek-R1-Distill-Qwen-7B模型、vLLM推理框架、xFt加速库以及open-webui前端可视环境。您可在天翼云控制台选择下列资源池和镜像，开通云主机进行体验。

类型	可用资源
可用资源池	华东-华东1-az2、华东-华东1-az3
可用规格	c8e系列（最低内存需求32G）
可用镜像	DeepSeek-vLLM-英特尔AMX推理加速（CPU） 

完成云主机开通后，推理服务会在5分钟内自动启动，您无需进行任何其他操作。

### 注意

如需在云主机外访问服务，您需要绑定弹性IP，并在安全组内放行22/3000/8000端口。

## 2.2 模型使用

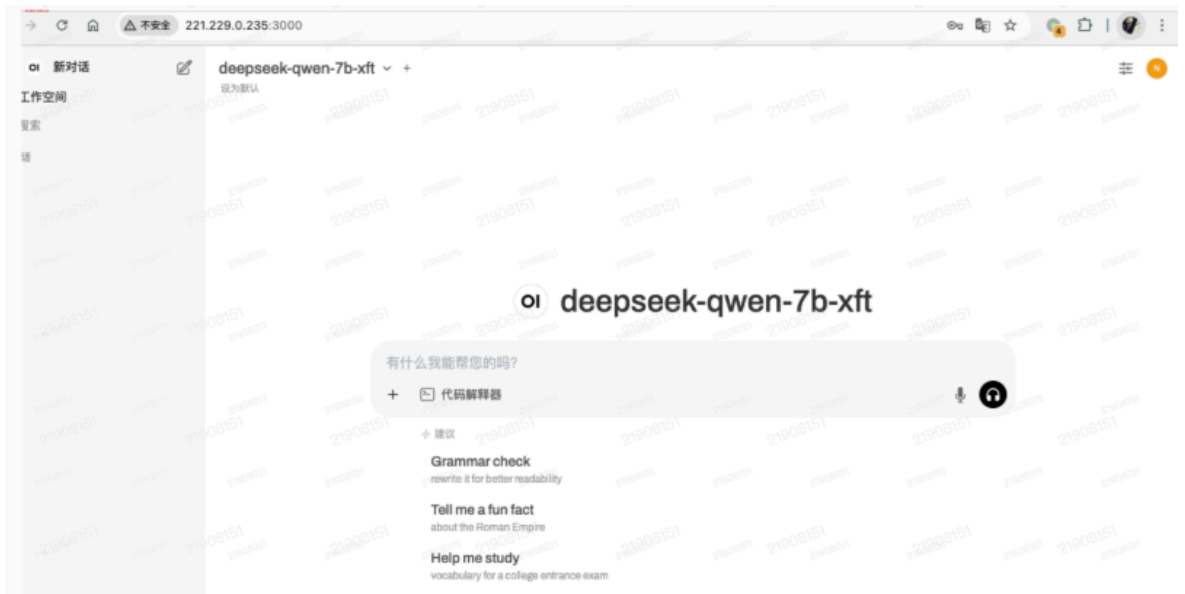
### 2.2.1 open-webui 前端使用

镜像已内置open-webui，并已完成和vllm的连接配置，可直接通过以下地址进行访问：

http://[弹性IP]:3000/

#### 注意

1. 首次打开页面时，您需要先完成管理员注册，以进行后续的用户管理。注册数据均保存在云实例的/root/volume/open-webui目录下。
2. 如果首次打开对话页面时没有模型可供选择，请您稍等几分钟让模型完成加载即可。



### 2.2.2 vllm api 调用

镜像内置vllm服务可直接通过如下地址访问：

# 根路径

http://[弹性IP]:8000/

# 查询现有模型

http://[弹性IP]:8000/v1/models

# 其他api接口参阅vllm文档

#### 注意

vllm服务配置有API\_KEY，您可在云实例的/root/recreate\_container.sh文件开头查看到当前值，并可进行修改以确保服务安全。

# 最佳实践

## 2.3 性能指标

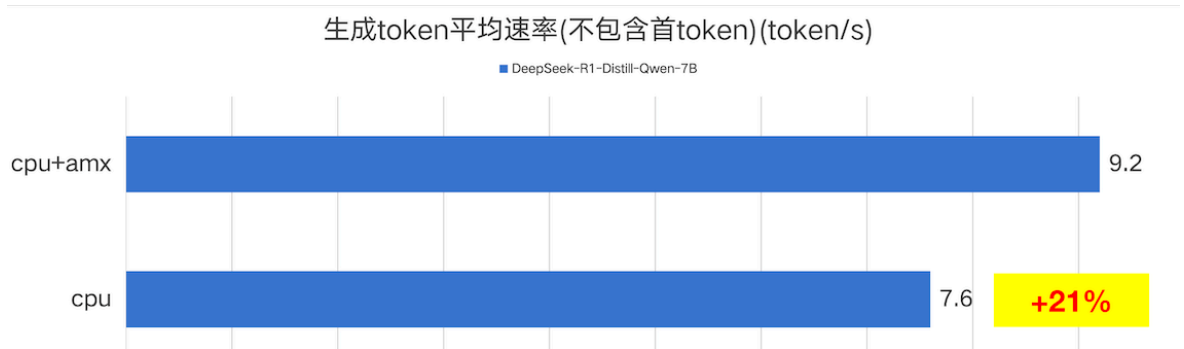
借助于英特尔AMX的加速能力，本推理服务能够取得显著的性能提升，天翼云完成测试并给出参考指标如下：

### 基本参数

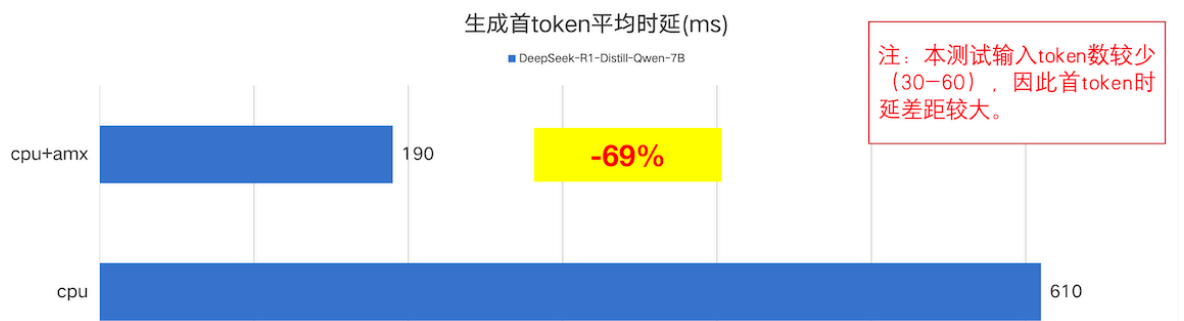
- vcpu数：24（物理核12）
- 内存：64GB
- 硬盘：60G 通用型SSD
- 模型：DeepSeek-R1-Distill-Qwen-7B (bf16)
- batch size: 1
- 输入token个数：30-60
- 输出token个数：256

### 性能数据

平均token生成速率：



首token时延：



在天翼云c8e系列24vcpu云主机上，启用AMX加速能力后，DeepSeek 7B蒸馏模型（BF16）推理速度能够超越9token/s，满足日常使用需求。

## 3 基于英特尔®至强®6处理器部署满血版DeepSeek-R1 671B实践

### 3.1 性能指标

DeepSeek R1 671B 满血版模型以其卓越的性能，为用户带来了极致的效果体验，不过其部署成本也不容小觑。若采用传统的 GPU 部署方式，需要 8 - 16 张 GPU 才能提供足够的支持，这无疑大幅增加了硬件购置、能耗以及维护等方面的成本。

在这样的背景下，天翼云基于英特尔®提供的至强®6处理器服务器进行了DeepSeek R1 671B满血版Q4\_K\_M模型的部署尝试，测试结果如下：

- 1-instance 1-socket:

平均吞吐性能9.7~10 token/s

- 2-instance 1-socket:

平均7.32 token/s和7.38token/s，共14.7token/s

从上面测试数据可以看到，采用单实例单socket部署下，DeepSeek R1 671B满血版模型可达到平均9.7 ~ 10 token/s的吞吐量，而在双实例部署模式中，总体吞吐量提升至14.7 token/s。单颗CPU系统的吞吐性能可以达到普通用户正常使用的需要。

### 3.2 英特尔®至强®6处理器简介

英特尔®至强®CPU 为 DeepSeek R1 671B 模型的部署提供了一个极具竞争力的方案。英特尔®至强®CPU 具备支持 T 级超大内存的能力，这使得它在权重存储与加载方面表现高效。对于像 DeepSeek R1 671B 这样的超大模型，其所需的显存容量在多卡 GPU 配置下才能满足，而英特尔®至强®CPU 能够凭借其强大的内存支持能力，为该模型提供良好的运行环境。

此外，DeepSeek R1 模型采用的 MOE (Mixture of Experts) 结构，通过参数稀疏化的方式，使得在单 token 推理时仅需激活少量专家参数。这种特性显著降低了推理过程中的算力要求，与 CPU 的计算特点相契合，使得模型在 CPU 系统上的运行更加高效。这意味着在英特尔®至强®CPU 上部署 DeepSeek R1 671B 模型，不仅能够充分发挥模型的性能优势，还能有效降低部署成本，避免了对大量 GPU 的依赖。

如需复现以上性能测试结果，请参看附录2

#### 总结

通过本次实践，无论是在天翼云EMR云实例上结合xFasterTransformer部署DS R1 distill Qwen-7B蒸馏模型，还是基于英特尔®至强®6处理器部署满血版DeepSeek-R1 671B模型，均验证了CPU系统在DeepSeek大模型推理上的可行性和符合业界普遍要求的性能表现。CPU系统不仅能够灵活应对不同规模的模型需求，无论是轻量化蒸馏模型还是全功能满血模型，都能高效满足用户场景需求，提供了一种低成本、经济高效的解决方案。

#### 附录1 英特尔®至强®可扩展处理器与AI加速技术

##### 最新英特尔®至强®可扩展处理器产品

英特尔第五代®至强®可扩展处理器（代号 Emerald Rapids）——为AI加速而生

第五代英特尔®至强®处理器以专为AI工作负载量身定制的设计理念，实现了核心架构和内存系统的双重飞跃。其64核心设计搭配高达320MB的三级缓存（每核心由1.875MB提升至5MB），相较于上代缓存容量实现近三倍增长，为大规模并行AI推理提供充裕的本地数据存储空间。与此同时，处理器支持DDR5-5600高速内存，单路最大4TB的容量保证了大数据处理时的带宽和延迟优势。基于这些硬件提升，Emerald Rapids整体性能较上一代提升21%，AI推理性能平均提升42%，在大语言模型推理场景中可实现最高1.5倍的性能加速，同时大幅降低总拥有成本达77%。

## 英特尔® 至强®6处理器（代号 GNR Granite Rapids）——引领CPU AI算力革新

全新GNR处理器专为应对人工智能、数据分析及科学计算等计算密集型任务而设计。该产品在内核数量、内存带宽及专用AI加速器方面均实现重大突破：

- 核心与性能：每CPU配备多达128个性能核心，单路核心数较上一代翻倍，同时平均单核性能提升达1.2倍、每瓦性能提升1.6倍，进一步强化了CPU在大模型推理中的独立处理能力；
- AI加速功能：内置英特尔® 高级矩阵扩展（AMX）新增对FP16数据类型的支持，使得生成式AI和传统深度学习推理任务均能获得显著加速；
- 内存与I/O突破：支持DDR5-6400内存及英特尔首款引入的Multiplexed Rank DIMM (MRDIMM) 技术，有效将内存带宽提升至上一代的2.3倍；同时，高达504MB的三级缓存和低延迟设计确保数据能够更快加载，为复杂模型训练和推理缩短响应时间。

英特尔® 至强® 6处理器不仅通过更多的核心和更高的单线程性能提升了AI大模型推理能力，同时也能够作为机头CPU为GPU和其他加速器提供高速数据供给，进一步缩短整体模型训练时间。在满足混合工作负载需求的同时，其TCO平均降低30%，大模型推理加速最高可达2.4倍。

无论是第五代至强还是全新的至强6处理器，英特尔均通过在核心架构、缓存系统、内存技术和专用AI加速器方面的全面革新，提供了业界领先的AI计算支持。这两款产品为数据中心和高性能计算平台在AI推理、训练以及多样化工作负载下提供了强大而高效的算力保障

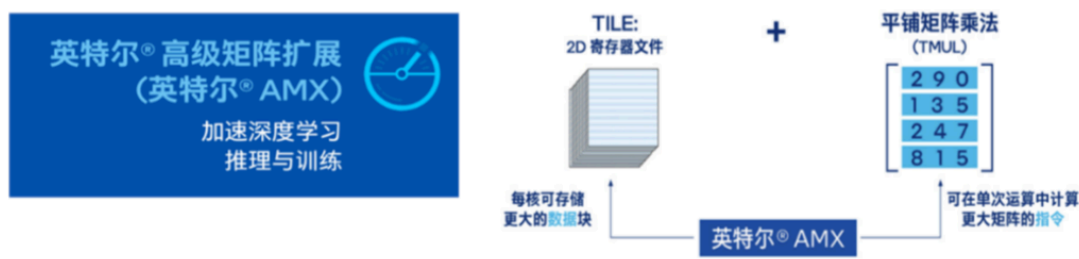


图 1 英特尔高级矩阵扩展（AMX）

## 英特尔全方位的AI软件生态支持

英特尔及其合作伙伴凭借多年AI积累，围绕至强®可扩展处理器打造了完善的软件生态：广泛支持主流开源框架，通过插件优化及多样化开源工具链，使用户在x86平台上能够轻松开发、部署通用AI应用，无需手动调整，同时确保从终端到云的全程安全保护。

此外，至强®处理器内置多种AI加速指令（如AVX-512、AMX），使得任何兼容软件均可直接调用加速功能。开发者可免费下载英特尔分发版工具、库及开发环境，充分利用这些内置加速器应对各类AI管线需求。结合多样化硬件优势与开放生态，英特尔通过经济、可扩展的方案，将AI能力无缝延伸至云端与边缘。

其中，xFasterTransformer (xFT) 是英特尔官方开源的AI推理框架，专为大语言模型在至强®平台上深度优化。xFT不仅支持多种数据精度（FP16、BF16、INT8、INT4），还能利用多CPU节点实现分布式部署，显著提升推理性能并降低成本。其简单的安装和与主流Serving框架（如vLLM、FastChat、MLServer、MindSpore Transformer、PaddlePaddle）的兼容性，帮助用户快速加速大模型应用。在3.1节中基于天翼云EMR云主机和xFasterTransformer加速引擎实现了对与DeepSeek R1蒸馏模型的高效推理部署。

## 英特尔提供AI软件工具 全面兼容主流AI开发框架



主流开源社区 (PyPI, Anaconda, Intel, Apt, Yum, Docker) 和生态 (Hugging Face) 都可获取

图 2 英特尔提供 AI 软件工具 全面兼容主流 AI 开发框架

### 附录2 CPU环境下部署DeepSeek-R1 671B模型实践

#### 环境配置

##### 硬件配置

- CPU: Intel® Xeon® 6980P Processor, 128core 2.00 GHz
- 内存 24\*64GB DDR5-6400
- 存储 1TB NVMe SSD

##### 软件环境

- OS: Ubuntu 22.04.5 LTS
- Kernel: 5.15.0-131-generic
- llama.cpp: github bd6e55b
- cmake: 3.22.0
- gcc/g++: 11.4.0
- Python: 3.12.8
- git: 2.34.1

BIOS#关闭sub NUMA相关配置。

#### 注意

版本是指本测试中服务器上安装的版本，并#要求的最低版本。

# 最佳实践

## 部署步骤

### 1. 安装llama.cpp

下载并安装intel oneapi库

```
wget https://registrationcenter-download.intel.com/akdlm/IRC_NAS/dfc4a434-838c-4450-a6fe-2fa903b75aa7/intel-oneapi-base-toolkit-2025.0.1.46_offline.sh
sh ./intel-oneapi-base-toolkit-2025.0.1.46_offline.sh -a --silent --cli --eula accept
source /opt/intel/oneapi/setvars.sh
```

下载llama.cpp推理框架源码

```
git clone https://github.com/ggml-org/llama.cpp
```

基于oneapi库对llama.cpp进行编译

```
cd llama.cpp
```

```
cmake -B build -DGGML_BLAS=ON -DGGML_BLAS_VENDOR=Intel10_64lp -DCMAKE_C_COMPILER=icx -DCMAKE_CXX_COMPILER=icpx -DGGML_NATIVE=ON
cmake --build build --config Release -j$nproc
```

### 2. 下载模型#件我们直接使用了社区制作的DeepSeek 671B满血模型的Q4量化版，您也可以下载DeepSeek官方BF16版本，并通过 llama.cpp 提供的脚本转换为 GGUF 格式。

社区提供了从 1bit 到 8bit 不同版本的量化选项，具体区别可以参考社区网页。我们选择了使用最受欢迎的 Q4\_K\_M 版本。如果追求最佳效果，建议使用 Q8\_0 版本。

Filename	Quant type	File Size	Split	Description
<a href="#">DeepSeek-R1-Q8_0.gguf</a>	Q8_0	713.29GB	true	Extremely high quality, generally unneeded but max available quant.
<a href="#">DeepSeek-R1-Q4_K_M.gguf</a>	Q4_K_M	404.43GB	true	Good quality, default size for most use cases, <i>recommended</i> .

# 下载unsloth制作的社区版量化模型(注意这里要下载9个文件)

```
mkdir DeepSeek-R1-Q4_K_M
```

```
cd DeepSeek-R1-Q4_K_M
```

# 建议nohup执行，预计至少需要半天时间，同时确保磁盘容量足够400G.

```
wget https://modelscope.cn/models/unsloth/DeepSeek-R1-GGUF/resolve/master/DeepSeek-R1-Q4_K_M/DeepSeek-R1-Q4_K_M-0000{1..9}-of-00009.gguf
```

Q4\_K\_M 版本的文件大小为 404.43GB，下载过程可能会比较耗时。下载完成后，您可以在 DeepSeek-R1-Q4\_K\_M 目录下找到一系列 .gguf 文件，例如 DeepSeek-R1-Q4\_K\_M-00001-of-00009.gguf。

### 3. 模型加载和运#

#启动模型的交互模式(替换路径地址为实际值)

# 最佳实践

```
[llama.cpp路径]/build/bin/llama-cli -m [模型下载路径]/DeepSeek-R1-Q4_K_M-00001-of-00009.gguf -i
```

下面用几个示例展现DeepSeek-R1 671B满血版强大的的reasoning推理能力：

测试模型自我认知：

```
> 你是谁？
<think>
您好！我是由中国的深度求索（DeepSeek）公司开发的智能助手DeepSeek-R1，如您有任何任何问题，我会尽我所能为您提供帮助。
```

验证推理能#的经典“草莓”问题：

```
> How many Rs are there in strawberry?
<think>
To determine how many times the letter "r" appears in the word "strawberry":
1. Write out the word: S T R A W B E R R Y
2. Identify each "r":
 - The third letter is R.
 - The eighth letter is R.
 - The ninth letter is R (since "berry" is spelled B-E-R-R-Y).
Conclusion: There are 3 instances of the letter "r" in "strawberry".
```

“等灯等灯”的意思：

```
> “灯，等灯等灯”是什么意思？
<think>
好的，用户对“灯，等灯等灯”是什么意思，我需要先回想一下这个梗的来源。首先，这可能与英特尔®的那个经典广告曲有关，因为那个旋律非常有名，中文发音可能被谐音成“灯，等灯等灯”。然后可能用户在网络中看到有人使用这个梗，但不确定具体指什么，或者有没有其他含义。

接下来要确认这个谐音是否准确对应英特尔®的广告曲，比如原旋律是否真的是这几个音节。可能需要回忆广告曲的节奏，“灯，等灯等灯”确实对应的是“dēng, dēng dēng dēng dēng”这样的声音，中文里用这几个字来模仿，用户可能想知道这个网络用语的出处和具体用法。

然后要考虑是否有其他可能的解释，比如是否有其他品牌或作品用了类似的音节，但根据常见的网络文化，英特尔®这个应该是最常见的。另外，用户可能是在某个特定上下文看到这个梗，比如评论科技产品、电脑相关的帖子，或者用来幽默地提到电脑性能、广告等。

需要解释这个梗的来源，它是怎么样被网友用来调侃或引用的，同时也要说明，除了字面模仿声音外，可能还带有一些戏谑的语气，比如吐槽电脑性能，或者用来制造幽默效果，比如在DIY电脑时用这个梗来暗示用了英特尔®处理器。

还要注意用户可能对英特尔®不太了解，所以可能需要简要介绍一下英特尔®和他们的广告背景，帮助用户理解为什么这个旋律如此知名。同时要提到广告的名称是“Intel Bong”，并说明它在各种场合中的使用情况，比如电视广告、品牌宣传等。

最后检查有没有遗漏的信息，确保回答准确全面，避免让用户产生其他疑问。比如是否还有其他可能的含义，但目前看来这个是最主要的解释。总结一下，分步骤解释来源、用法、相关文化背景，应该能满足用户的需求。
</think>

“灯，等灯等灯”（或写作“dēng, dēng dēng dēng dēng”）是网络上一个经典梗，源自英特尔®（Intel）标志性的广告音效。

背景解析：
1. 来源：这个音效是英特尔®自1990年代起长期使用的品牌声音标识（原名为“Intel Bong”），它由奥地利音乐人Walter Werzowa创作，通常伴随英特尔®的广告和品牌宣传出现，成为全球消费者对英特尔®的“听觉记忆”。
2. 中文谐音梗：中国网友将这段旋律的节奏音效用拟声词“灯，等灯等灯”来模仿（普通话发音类似“dēng, dēng dēng dēng dēng”），因其魔性洗脑且辨识度极高，逐渐成为网络流行语。
3. 用法与含义：
 - 调侃电子产品：常见于电脑、处理器相关的讨论中，用来暗示“这台设备用了英特尔®芯片”（例如：“这电脑一开机就‘灯等灯等灯’，肯定是英特尔®处理器！”）。
 - 幽默玩梗：结合广告中工程师造型的“蓝人组”（Blue Man Group），网友会用制造科技感或戏谑“技术含量高”的场景。
 - 跨领域二创：被改编成鬼畜视频、手机铃声，甚至有人用这段旋律玩音乐梗（如用《最炫民族风》唱出“灯等灯等灯”）。

扩展知识：
- 英特尔®的广告音效曾被评为“全球最成功的品牌声音之一”，甚至有研究称它能激活大脑对品牌的联想记忆。
- 近年来，英特尔®逐渐淡化这一音效，转而使用更现代的品牌声音，但“灯等灯等灯”作为网络文化符号依然活跃。

下次听到这个旋律，你可能会条件反射地想到电脑开机或英特尔®的“蓝色科技世界”囉~ ☺
```

## 4. 性能及优化

#性能测试(替换路径地址为实际值，并根据CPU实际核数和numa情况指定-C和-m参数，千万不要照抄!)  
export MODEL\_PATH=[模型下载路径]/DeepSeek-R1-Q4\_K\_M-00001-of-00009.gguf  
numactl -C 0-127 -m 0 [llama.cpp路径]/build/bin/llama-cli -m \$MODEL\_PATH -t 128 --temp 0.6 -s 42 --no-cnv --no-kv-offload -ctk q4\_0 -c 8192 -n 128 -p "< | User | >以孤独的夜行者为题写一篇

## 最佳实践

750字的散文，描绘一个人在城市中夜晚漫无目的行走的心情与所见所感，以及夜的寂静给予的独特感悟。< | Assistant | >"

这里使`#numactl`来指定使`#`单路CPU (0-127, 6980P有128核)，以及这`#`路CPU的内存节点 (`numa0`)，避免跨`numa`传输以获取最佳性能。

`llama.cpp`是本地编译的，编译的时候使`#Intel oneAPI`可以有效提升它的性能。英特尔尝试`#`了`oneAPI``#`的`Intel C++`编译器和数学加速库`MKL`，结合`jemalloc`内存管理优化，推理速度可以达到每秒`9.7~10`词元(TPS, `tokens per second`)。

上`#`的实验是在单路CPU上`#`进的，我们`#`在两路CPU上各`#`独`#`启动1个模型实例，总速度可以达到`14.7TPS` (`7.32TPS + 7.38TPS`)。

再进`#`步，英特尔观察到基于现有的`llama.cpp`软件`#`案，在CPU平台没有实现`#`效的专家并`#`和张量并`#`等优化，CPU核`#`利`#`率和带宽资源没有充分发挥出来，6980P的128核`#`运`#`1个模型还有不少性能储备。预计可以继续增加实例数来获得更好的总TPS。

另外，通常情况下，CPU的`SNC (Sub-NUMA Clustering)`设置可以获得更`#`的带宽，但是受限于软件并未优化实现良好匹配，此次实验关闭了`SNC`测试。

以下`#`式的系统配置也有助于提升性能：

- `BIOS``#`关闭`AMP prefetcher`
- `#cpupower`打开CPU的`pstate`性能模式
- 提`#`CPU的`uncore`频率
- 关闭超线程(`Hyper-Threading`)

### 注意

为了加快试验进度，我们限制了词元输出`#`度(`-n 128`)。经过验证，增加输出`#`度(例如`-n 512`)对于生成性能指标的影响不大。

## Open WebUI 快速入门

`Open WebUI`是一个可扩展、功能丰富、用户友好的自托管`WebUI`，它支持各种`LLM`运行程序，包括`Ollama`和`OpenAI`兼容的`API`。并且内置了`RAG (检索增强生成)`推理引擎，使其成为一个功能强大的`AI`部署解决方案。`Deepseek`云主机自带`Open WebUI`，通过`GPU`云主机自带`Open WebUI`，本文简述了如何通过`Open WebUI`快速体验本地和第三方大模型。

### 注意

`GPU`云主机镜像自带的`ollama`监听`127.0.0.1:11434`、`vllm` 监听 `0.0.0.0:8000`、`open-webui` 监听 `0.0.0.0:3000` 端口，云主机默认不对外开放任何端口访问，请按需开放端口访问规则，避免数据泄露。

## 开放云主机端口

云主机安全组需要放行`3000`端口，[添加安全组规则帮助文档](#)。

### 注册管理员账号

`Deepseek`云主机内置可视化界面，云主机启动 5 分钟后，可访问 `http://{your_ip}:3000` (`your_ip` 这台云主机的 `eip`) 体验`DeepSeek`。

# 最佳实践

首次登录页面如下：




点击开始使用，注册管理员账号：




## 基本配置

Open WebUI有强大的设置功能，可根据需要进行设置，点击左下角进入设置页面。

首先在通用设置中，将语言设置为简体中文。

进入管理员设置。

如果您不想开放其他用户注册使用，则需要关闭“允许用户注册”功能。

如果你允许用户注册，还可以设置用户注册后的行为，例如将新用户注册后的默认用户角色设为“用户”或“待激活”等，这些用户需要管理员手动激活：

# 最佳实践

## 模型配置

单机版 Deepseek GPU 云主机在完成部署后，会自动匹配云主机本地模型，刷新进入首页，可以看到，模型下拉列表中，可以选择我们部署的deepseek-r1 模型。

您可以主动设置您的模型连接，以体验任意第三方连接。以ollama和vllm提供的服务为例，进入“管理员面板”->“设置”->“外部链接”，如果使用的vllm则填入您的vllm地址和--api-key，如果使用ollama则填入您的ollama地址，点击保存后即可使用模型。

设置模型可见性。多用户模式下，建议把模型设置为public，如此才能让模型被所有用户看到。进入“管理员面板”->“设置”->“模型”，并点击编辑。



将模型可见性设置为 public 并点击保存。

最后，我们尝试下问一个问题，模型可以正常回答问题。

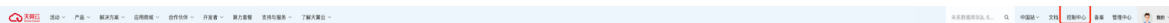


## 天翼云 GPU 云主机构建高性能 Deepseek 集群最佳实践-32B等版本

### 步骤一：创建GPU云主机

#### 1. 进入创建云主机页面

- a. 点击天翼云门户首页的“控制中心”，输入登录的用户名和密码，进入控制中心页面。



- b. 单击“服务列表>弹性云主机”，进入主机列表页。



- c. 单击“创建云主机”，进入弹性云主机创建页。

# 最佳实践

## 2. 进行基础配置

### a. 基础配置

根据业务需求配置“计费模式”、“地域”、“企业项目”、“虚拟私有云”、“实例名称”、“主机名称”等。

### b. 选择规格。

此处选择"CPU架构"为"X86"、"分类"为"GPU加速/AI加速型"、"规格族"为"GPU计算加速型pn8r"、"规格"为"pn8r.16xlarge.4"。

### c. 选择镜像

“镜像类型”选择“镜像市场”，在云镜像市场中选择预置了DeepSeek R1模型的DeepSeek-R1-32B-vLLM-Ubuntu22.04镜像。



### d. 设置云硬盘。

#### \* 存储

存储资源需与云主机在同一可用区，鼠标放置存储类型可查看其支持的可用区。

系统盘   GB

当前配置: 高IO | 150GB 基础IOPS上限3000, IOPS突发上限5000, [了解IOPS计算规则。](#)

[增加一块数据盘](#) 您还可以增加8块数据盘

## 3. 网络及高级配置

设置网络，包括“网卡”、“安全组”，这里，为安全和成本考虑，我们先不设置弹性IP，后续按需开放IP；设置高级配置，包括“登录方式”、“云主机组”、“用户数据”。

## 4. 确认并支付

## 最佳实践

### 步骤二：配置弹性负载均衡

现在，我们已经有了两台 Deepseek-R1-32B 服务器，接下来，我们利用弹性负载均衡构建一个可任意横向扩容的 Deepseek 集群。

#### 1. 配置 vllm 服务

首先需要为所有服务器上的 vllm 服务配置相同的 `--api-key`。

服务的配置文件目录在 `/var/vllm_service.env`，我们将 `API_KEY` 设置为想要的值。

```
API_KEY=xxh4cVRFPgfAr8ayy
NV_NUM=4
NV_NUM=4
```

然后重启服务。

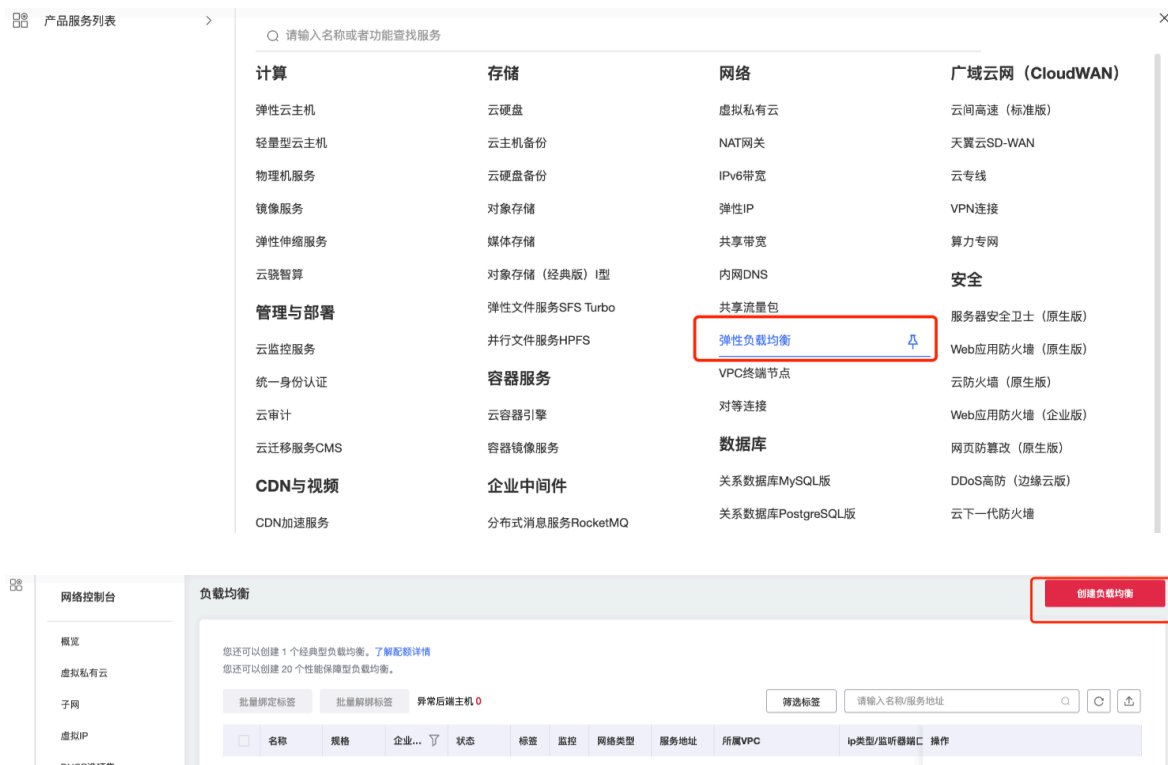
```
systemctl restart vllm
```

# 最佳实践

## 2. 创建弹性负载均衡

我们参考[弹性负载均衡官方文档](#)进行服务创建。

### a. 创建弹性负载均衡



在此例中，我们的Deepseek集群仅用于集群自带的 Open WebUI内网使用，不涉及外网通讯，因此“网络类型”选择“内网”，如果你想将此集群暴露到外网，则选择外网。VPC和子网跟 Deepseek 服务器保持一致。

# 最佳实践

创建负载均衡

地域: 长沙42 温馨提示: 页面左上角切换区域

类型: 经典型 性能保障型 ?

名称: elb-731d ?

网络类型: 外网 内网

所属VPC: default\_vpc\_336c13(10) 查看虚拟私有云

子网: default\_subnet\_336c13

负载均衡虚拟IP: 自动分配 手动分配

企业项目: default ? 如需创建新的企业项目, 您可以点击去创建

描述:

## b. 创建监听器

完成网络负载均衡创建, 接下来配置监听器, 在负载均衡列表页, 点击开始配置。

网络控制台

负载均衡

您还可以创建 0 个经典型负载均衡。了解配额详情  
您还可以创建 20 个性能保障型负载均衡。

批量绑定标签 批量解绑标签 异常后端主机 0

筛选标签 请输入名称/服务地址

名称	规格	企业...	状态	标签	监控	网络类型	服务地址	所属VPC	ip类型/监听器端口	操作
elb-731d	经典型	default	运行中		查看	内网	10.0.0.8...	default_vpc_336c13 10.0.0.0/8(Pv4 主)	开始配置	修改 监听器配置向导 删除 更多

端口类型, 选择“HTTP”, 端口填 8000。

# 最佳实践

负载均衡

### 协议&监听器

\* 名称:  ⓘ

\* 监听器协议/端口:   ⓘ

是否编辑标签:

描述:

---

高级配置: [重定向](#) | [附加HTTP头字段](#) | [访问控制](#)

重定向:  ⓘ

附加HTTP头字段:  通过X-Forwarded-For获取客户端IP

访问控制:

[取消](#) [下一步](#)

进入下一步，创建健康检查。这里我们通过 HTTP 进行健康检查，返回码部分选择 2xx、3xx、4xx。

负载均衡

### 负载方式&健康检查

监听器协议: HTTP

\* 后端主机组名称:  ⓘ

后端协议:

描述:  0/128

\* 负载方式:

会话保持:

\* 健康检查:

\* 健康检查类型:

# 最佳实践

\* 健康检查:

\* 健康检查类型: HTTP

\* 间隔时间 (秒): 5 取值范围1-20940

\* 超时时间 (秒): 2 取值范围2-60

\* 最大重试次数: 2 取值范围1-10

\* 检查路径: / 长度范围1-80 以/开头, 可包含数字、大小写字母或/, 路径中可包含'.' ':' '/'

\* HTTP方法: GET

\* HTTP状态码: http\_2xx @ +2

- http\_2xx ✓
- http\_3xx ✓
- http\_4xx ✓
- http\_5xx ✓

[上一步](#) [立即创建](#)


点击“立即创建”，完成创建。

## c. 为监听器开通GPU云主机白名单

此时监听器列表中，我们可以看到访问Deepseek服务器的主机组地址为“100.89.0.0/16”，我们需要在Deepseek集群的安全组中，为vllm端口配置此地址的白名单。

< 负载均衡

### elb-731d详情



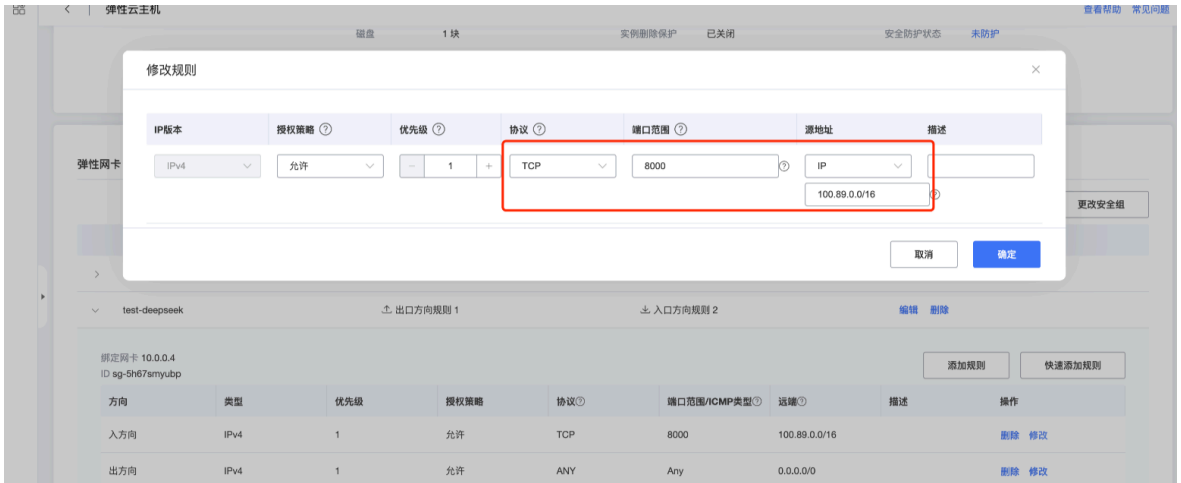
网络类型	内网	名称	elb-731d
类型	经典型	企业项目	default
ID	lb-craoyfthbr	VPC	default_vpc_336c13 10.0.0.0/16(Pv4 主)
服务器地址(Pv4)	10.0.0.8(内)	服务器地址(Pv6)	--
子网	default_subnet_336c13(10.0.0.0/24)	计费模式	免费
创建时间	2025-02-22 17:25:24	到期时间	--
IPv4带宽	--	描述	--

您还可以创建 2 个监听器 [添加](#)

名称	状态	监听器协议/端口	后端主机组	访问后端主机组地址	标签	描述	重定向	创建时间	操作
Deepseek	运行中	IPv4:HTTP: 8000	Deepseek	100.89.0.0/16		--	--	2025-02-22 22:43:27	<a href="#">修改</a> <a href="#">删除</a> <a href="#">更多</a>

为Deepseek云主机所在的安全组新增规则，操作文档详见[添加安全组规则帮助文档](#)。

# 最佳实践

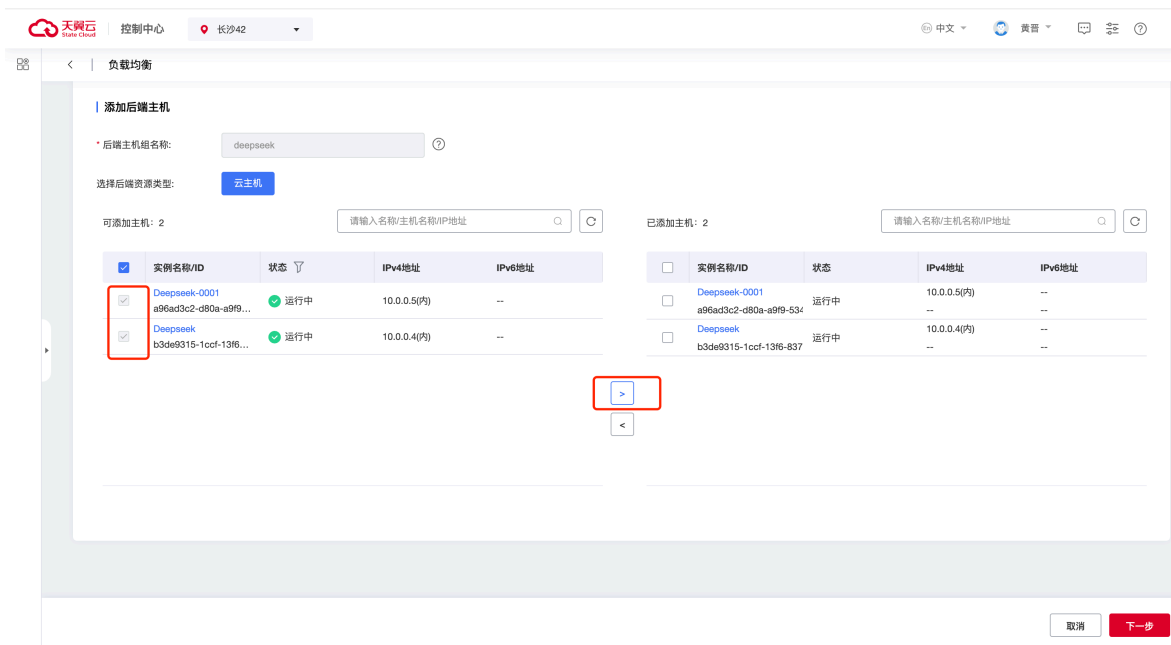


d. 为监听器添加后端云主机

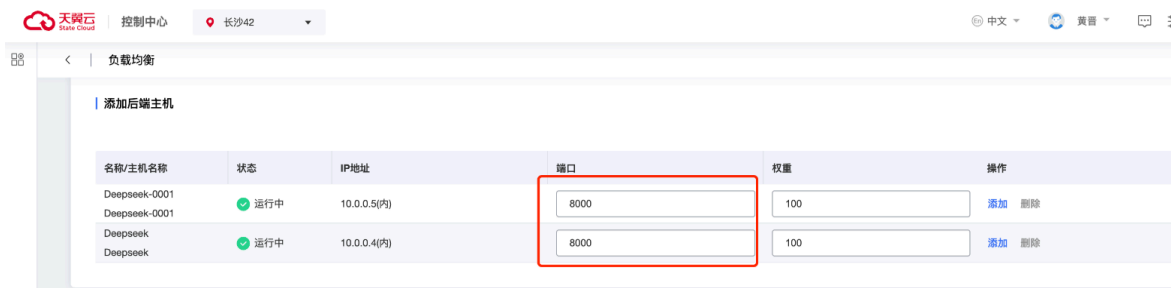


选中主机，然后下一步。

# 最佳实践



设置端口为“8000”。



点击确认即可完成配置。

## e. 验证连通性

监听器页面，显示监控检查正常。



# 最佳实践

通过一台 Deepseek 服务器，测试负载均衡连通性。

```
root@Deepseek:~# telnet 10.0.0.8 8000
Trying 10.0.0.8...
Connected to 10.0.0.8.
Escape character is '^]'.

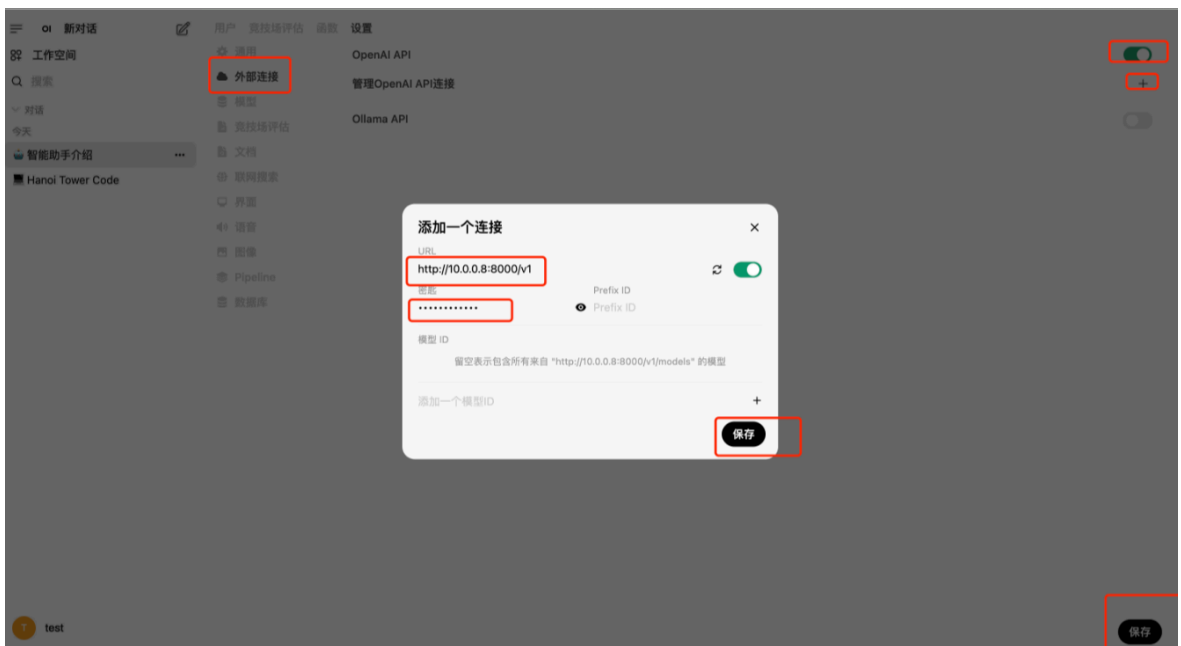
```

## 步骤三：配置 UI 访问能力

为便于公网访问，我们选择一台DeepSeek服务器，利用DeepSeek服务器自带的Open WebUI进行交互。Open WebUI使用文档见[Open WebUI 快速入门](#)。

## 注意

Openai API地址为前面弹性负载均衡的地址，在本例中为“http://10.0.0.8:8000/v1”，密钥为我们刚刚设置的vllm的--api-key。



## 使用云主机启动模板快速部署DeepSeek的最佳实践

### 前置步骤

进入创建云主机启动模板页面

# 最佳实践

1. 点击天翼云门户首页的“控制中心”，输入登录的用户名和密码，进入控制中心页面。



2. 单击“产品服务列表>弹性云主机”，进入计算控制台，单击“云主机启动模板”，进入启动模板列表页。



计算控制台



概览



弹性云主机



云主机启动模板

物理机服务

弹性伸缩

镜像服务

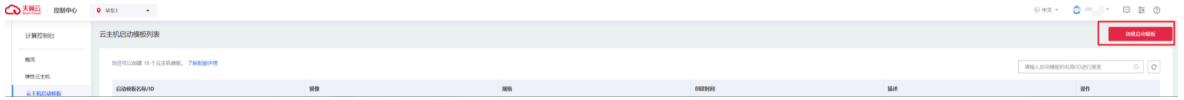
快照

SSH密钥对

云主机组

# 最佳实践

3. 单击“创建启动模板”，进入云主机实例模版创建页。

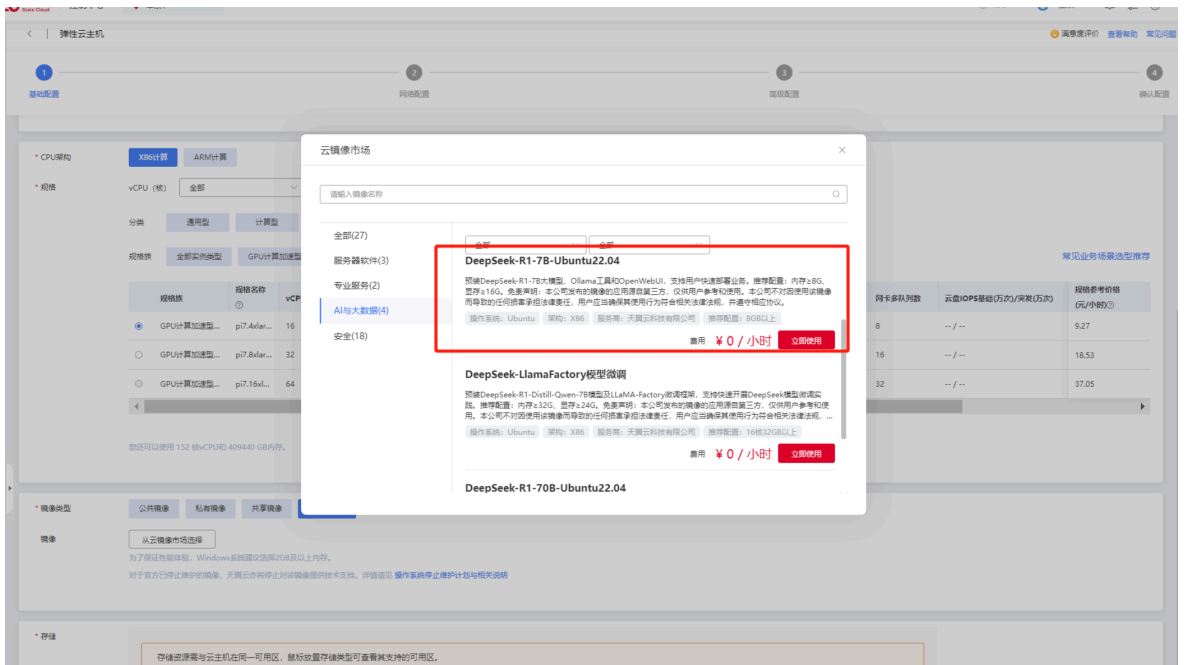


## 大模型服务器启动模板配置

1. 根据业务需求配置“计费模式”、“地域”、“可用区”、“企业项目”、“虚拟私有云”等。
2. 选择规格。此处选择“CPU架构”为“X86”、“分类”为“GPU加速/AI加速型”、“规格族”为“GPU计算加速型pi7”、“规格”为“pi7.4xlarge.4”。
3. 选择镜像。“镜像类型”选择“镜像市场”，在云镜像市场中选择预置了DeepSeek-R1模型的DeepSeek-R1-7B-Ubuntu22.04镜像。

### 注意

本镜像推荐配置：内存≥8G、显存≥16G。



### 注意

目前提供了预装多种量化精度、多种参数量的模型，如您有需求，也可在云镜像市场中进行选择。

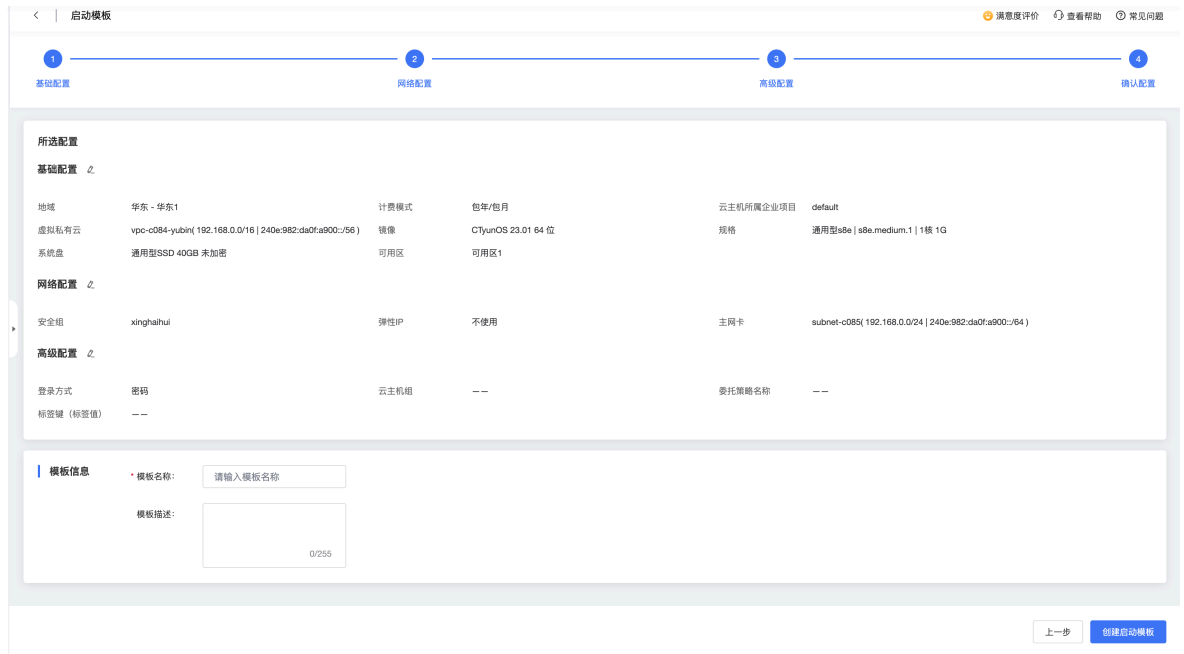
# 最佳实践

## 4. 设置云盘类型和大小。



## 5. 根据需要设置网络，包括"网卡"、"安全组"，同时配备 "弹性IP" 用于下载和访问模型；设置高级配置，包括"登录方式"、"云主机组"、"用户数据"。

## 6. 填写此模板名称，创建启动模板。



## 启动模板使用

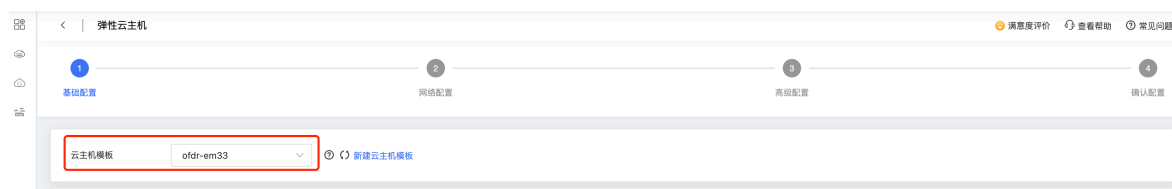
### 1. 单击左侧导航栏“产品服务列表”，选择“计算 > 弹性云主机 > 云主机启动模板”，进入云主机启动模板控制列表页。

# 最佳实践

2. 找到想要使用的云主机启动模板，点击操作中“创建云主机”。



3. 在创建云主机页面将默认选中该条云主机启动模板。创建云主机时，所需参数已根据模板内容自动填充。



## 弹性伸缩搭配GPU云主机水平扩展最佳实践

对比传统部署模式，GPU云主机配合弹性伸缩使用，可以帮助您在推理任务高峰期自动扩容GPU节点减少卡顿，在任务请求低峰期减少GPU服务器规模节约成本，保障业务连续性的同时降低资源、运维成本。

## 操作步骤

### 1. 使用大模型镜像创建包年包月的云主机实例

创建一定数量的包年包月GPU实例，用于后续将实例添加到伸缩组，满足大模型业务模块的非高峰或低谷时间段要求。

- 登录控制中心。
- 根据业务部署需求选择弹性伸缩组及其管理的云主机实例所在地域。
- 单击“计算>弹性云主机”，进入弹性云主机管理控制台。
- 在右上角点击“创建云主机”按钮，进入到弹性云主机创建页面。
- 在弹性云主机创建页面，完成云主机参数配置。创建成功之后如图：



## 说明

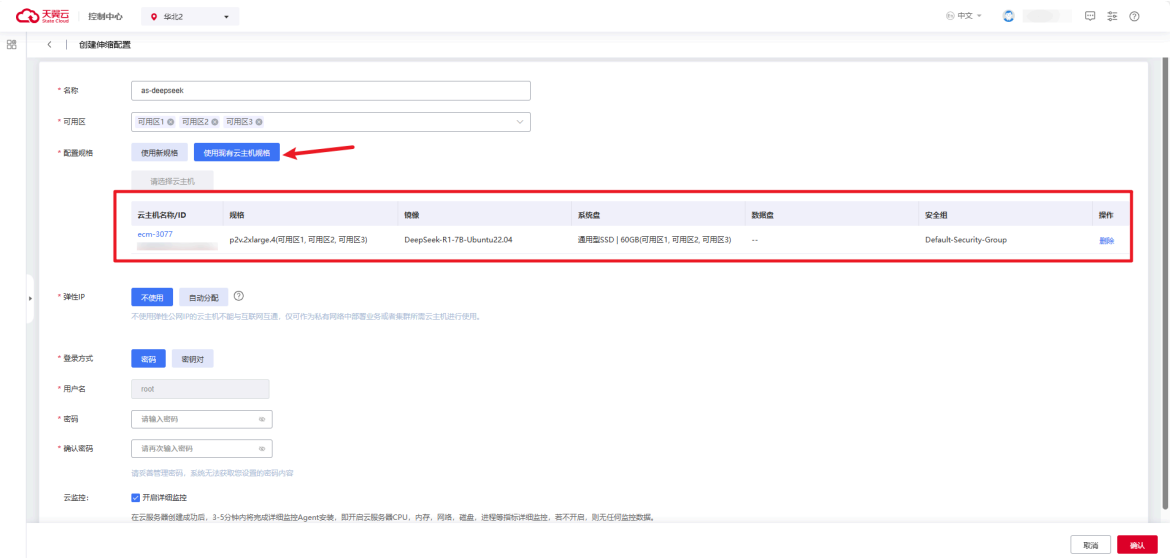
搭建DeepSeek等大模型云主机选型、部署等说明可参考[GPU云主机/弹性云主机：零基础搭建DeepSeek云端环境指南-弹性云主机-DeepSeek专题-实践指南：DeepSeek驱动高效能云生态 - 天翼云](#)。

# 最佳实践

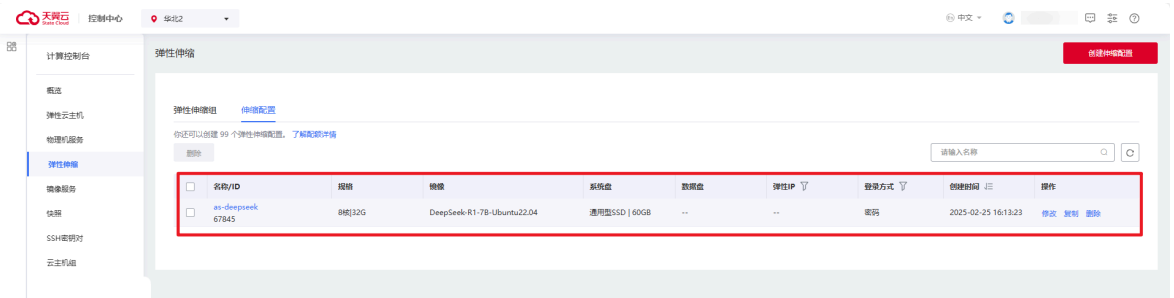
## 2. 创建使用大模型镜像的伸缩配置

为业务创建可用于自动扩容的云主机模板，即弹性伸缩配置。

- 登录控制中心，选择弹性伸缩组所在地域（与前一步骤中创建的GPU云主机需保持一致）。
- 单击“计算>弹性伸缩服务”，进入弹性伸缩管理控制台并切换到伸缩配置页面。
- 点击右上角“创建伸缩配置”，进入“创建伸缩”配置页面。使用步骤1中创建的实例作为模板创建伸缩配置。



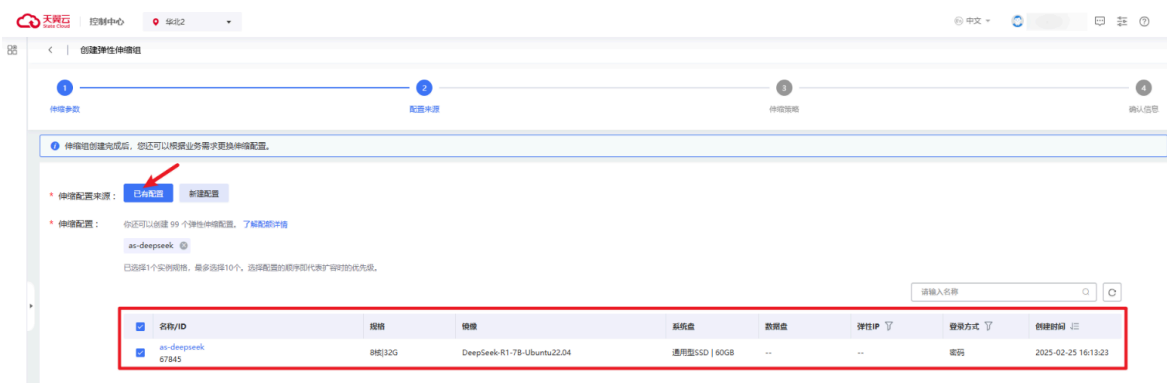
- 单击“确认”，即可创建伸缩配置成功，创建成功如图所示：



## 3. 创建伸缩组并添加云主机实例

为低成本、高稳定性业务集群创建弹性伸缩组，并将创建好的云主机实例添加至此伸缩组内，来满足日常业务的需求。

- 登录控制中心，选择弹性伸缩组所在地域（与前一步骤中创建的GPU云主机需保持一致）。
- 单击“计算>弹性伸缩服务”，进入弹性伸缩管理控制台。
- 在“弹性伸缩组”页签中，点击右上角“创建弹性伸缩组”，进入“创建弹性伸缩组>伸缩参数”配置页面。
- 在“伸缩参数”页面完成基础配置，具体参数配置可参见[创建伸缩组](#)。
- 在“配置来源”页面，选择已有配置，为伸缩组绑定一个或多个配置。

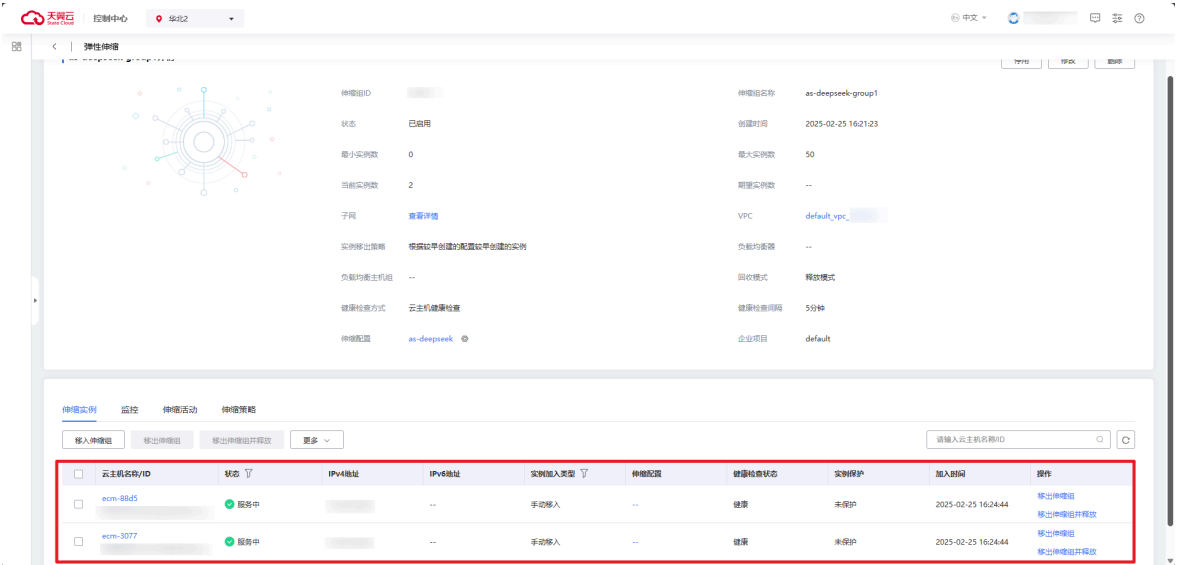


f. 点击“立即创建”伸缩组创建成功，如下图：



g. 伸缩组创建成功之后，进入到伸缩组详情页面，将步骤1中创建包年包月弹性云主机添加至伸缩组内并开启伸缩保护防止实例被移除，用于日常业务需求的满足。具体操作步骤请参见[将实例移入伸缩组](#)。添加成功之后如图：

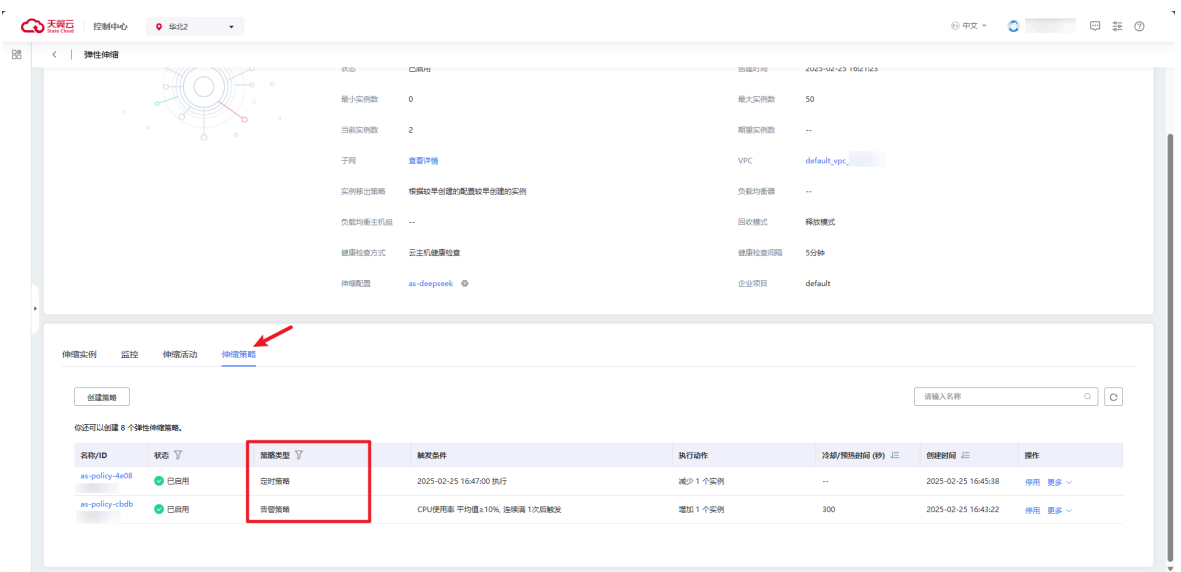
# 最佳实践



## 4. 根据业务场景为伸缩组设置自动伸缩策略

为已创建好的伸缩组设置伸缩策略，来满足突发的业务流量变化，确保业务的稳定运行。

- 进入弹性伸缩管理控制台，点击步骤3创建好的伸缩组名称，进入到伸缩组详情页面。
- 单击“伸缩策略”进入伸缩策略页签，单击“创建策略”按钮，进入到“创建伸缩策略”页面。
- 在“创建伸缩策略”页面中完成策略的配置。您可以根据使用场景选择告警、定时等自动伸缩策略，帮助伸缩组在业务高峰期自动扩容大模型实例、在业务低谷期自动释放实例，保证业务稳定的同时最大程度的优化资源使用成本。具体操作步骤可参见[创建伸缩策略](#)。



## 基于Anything LLM实现云主机外挂Embedding模型、向量数据库的最佳实践指南

### 前言

AnythingLLM 是一个开源项目，旨在简化将本地文档和数据源转化为可检索、可交互知识库的过程。它允许用户利用大语言模型（LLM）的能力，基于自身的私有数据进行智能问答、语义搜索等操作。其技术架构包含以下三方面：

- **Embedding 模型**：使用 Embedding 模型将文本转换为向量表示，以便在向量空间中进行相似度计算。常见的 Embedding 模型如 BGE - M3 等，能够捕捉文本的语义信息，提高检索的准确性。
- **向量数据库**：用于存储和管理文本向量，支持高效的向量搜索。AnythingLLM 可以集成多种向量数据库，如 Chroma、Milvus 等，根据实际需求选择合适的数据库来满足性能和规模的要求。
- **大语言模型（LLM）**：负责根据检索到的相关内容生成自然语言回答。用户可以选择不同的 LLM，如 OpenAI 的 GPT 系列、国产的通义千问、文心一言等，或者使用开源的 LLM 模型。

客户在搭建完具备推理训练基本能力的私有 DeepSeek 服务后，往往需要外挂知识库以达到应答更专业的目的。外挂知识库涉及到 Embedding 模型以及向量数据库这两大部分。本文档结合 GPU 物理机提供的 DeepSeek 服务，结合 Anything LLM，介绍如何使用云主机外挂 Embedding 模型以及向量数据库。

### 准备

- 创建一台物理机并搭建DeepSeek服务，参考[NVIDIA GPU物理机搭建DeepSeek指南（单机版）](#)。
- 安全组放开Anything LLM云主机的弹性IP+端口8000，参考[安全组-物理机-用户指南-安全 - 天翼云](#)。

### 部署Anything LLM

#### 1. 创建通用云主机

创建步骤请参考[创建弹性云主机-弹性云主机-快速入门 - 天翼云](#)，此处以“s8r.large.2”规格，“Ubuntu Server 22.04 64”镜像为例部署。

#### 2. 部署docker软件

##### #备份软件源

```
sudo cp /etc/apt/sources.list /etc/apt/sources.list.bak
```

#替换软件源，在编辑器中删除原内容添加备份源配置

```
sudo vim /etc/apt/sources.list
```

#更新软件包索引

```
sudo apt update
```

#安装 Docker 依赖

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common -y
```

#添加 Docker 官方 GPG 密钥

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
```

#添加 Docker 软件源

```
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

```
sudo apt update
```

#安装 Docker 引擎

```
sudo apt install docker-ce docker-ce-cli containerd.io -y
```

#启动并设置 Docker 服务

```
sudo systemctl start docker
```

```
sudo systemctl enable docker
```

说明

后续使用docker拉取镜像时，可将其配置为国内源以提升镜像拉取速度。

### 3. 部署Anything LLM

```
docker pull mintplexlabs/anythingllm
```

```
export STORAGE_LOCATION=$HOME/anythingllm && \
```

```
mkdir -p $STORAGE_LOCATION && \
```

```
touch "$STORAGE_LOCATION/.env" && \
```

```
docker run -u root -d -p 3001:3001 \
```

```
#指定宿主机核容器内服务到的端口映射，可自主选择
```

```
--cap-add SYS_ADMIN \
```

```
Docker 命令中的一个选项，用于给容器添加额外的 Linux 内核能力
```

```
-v ${STORAGE_LOCATION}:/app/server/storage \
```

```
#服务启动数据宿主机目录和容器目录的映射
```

```
-v ${STORAGE_LOCATION}/.env:/app/server/.env \
```

```
#服务启动环境变量配置文件，宿主机和容器的映射
```

```
-e STORAGE_DIR="/app/server/storage" \
```

```
#容器内的环境变量，指定服务数据在容器内存放目录
```

```
mintplexlabs/anythingllm
```

## 4. 登录Anything LLM

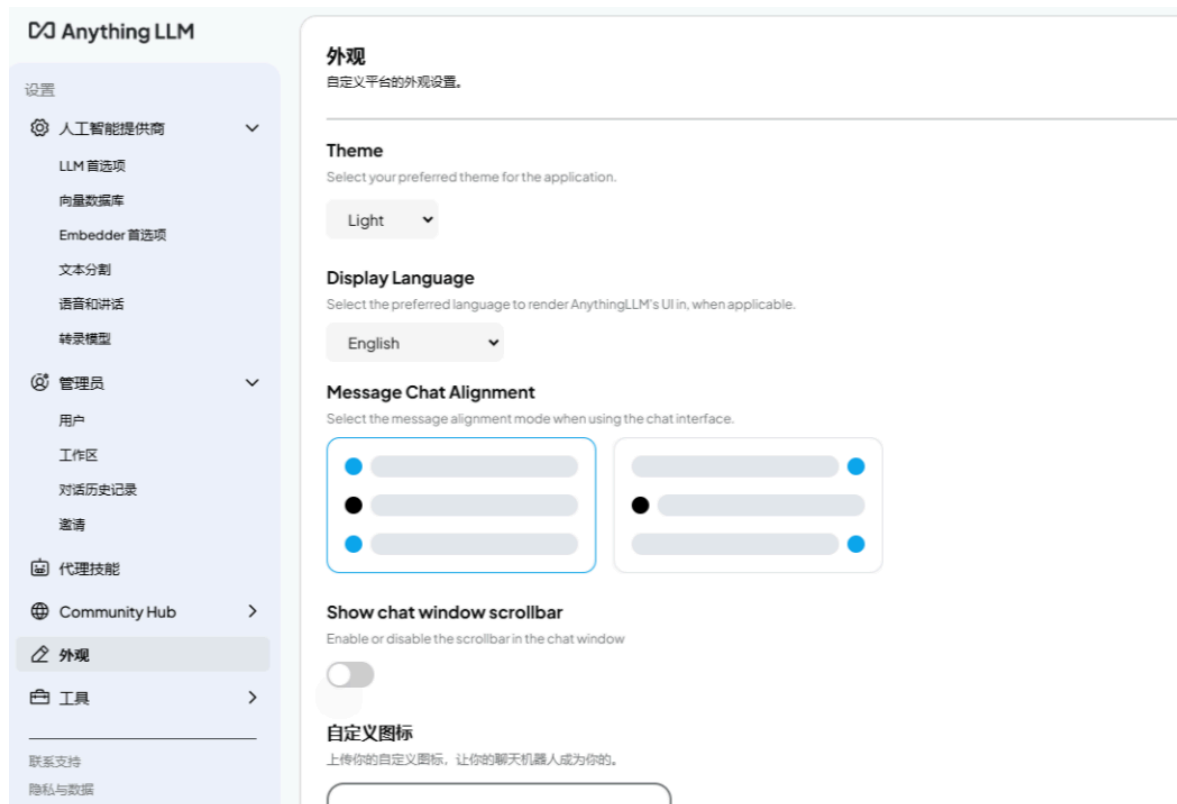
The screenshot displays the Anything LLM web interface. On the left is a sidebar with the title 'Anything LLM' and a '+ 新工作区' (New Workspace) button. Below this are two buttons: '测试GPU云主机' (Test GPU Cloud Instance) and '测试影视' (Test Video). The main area contains a chat conversation:

- DD** 欢迎使用 AnythingLLM, 这是由 Mintplex Labs 开发的开源 AI 工具, 可以将任何东西转换为你可以查询和聊天的训练有素的聊天机器人。AnythingLLM 是一款 BYOK (自带)...
- DD** AnythingLLM 是将强大的 AI 产品 (如 OpenAI, GPT-4, LangChain, PineconeDB, ChromaDB 等) 整合在一个整洁的包中而无需繁琐操作的最简单方法, 可以将你的生...
- DD** AnythingLLM 可以完全在你的本地计算机上运行, 几乎没有开销, 你甚至不会注意到它的存在! 无需 GPU, 也可以进行云端和本地安装。AI 工具生态系统每天都在变得更强...
- [在 GitHub 上创建问题](#)
- DD** 我该如何开始?!
- DD** 很简单, 所有集合都组织成我们称之为“工作区”的桶。工作区是文件、文档、图像、PDF 和其他文件的存储桶, 这些文件将被转换为 LLM 可以理解和在对话中使用的内容。...
- [+ 创建你的第一个工作区](#)
- DD** 这像是一个 AI Dropbox 吗? 那么聊天呢? 它是一个聊天机器人, 不是吗?
- DD** AnythingLLM 不仅仅是一个更智能的 Dropbox。AnythingLLM 提供了两种与你的数据交流的方式:  
查询:  
你的聊天将返回在你的工作区中访问的文档中找到的数据或推论。向工作区添加更多文档会使其更智能!  
对话:  
你的文档和正在进行的聊天记录同时为 LLM 知识做出贡献。非常适合添加基于文本的实时信息或纠正 LLM 可能存在的误解。你可以在聊天过程中切换模式!
- DD** 哇, 这听起来很棒, 让我马上试试!
- DD** 玩得开心!
- [在 GitHub 上加星](#) [联系 Mintplex Labs](#)

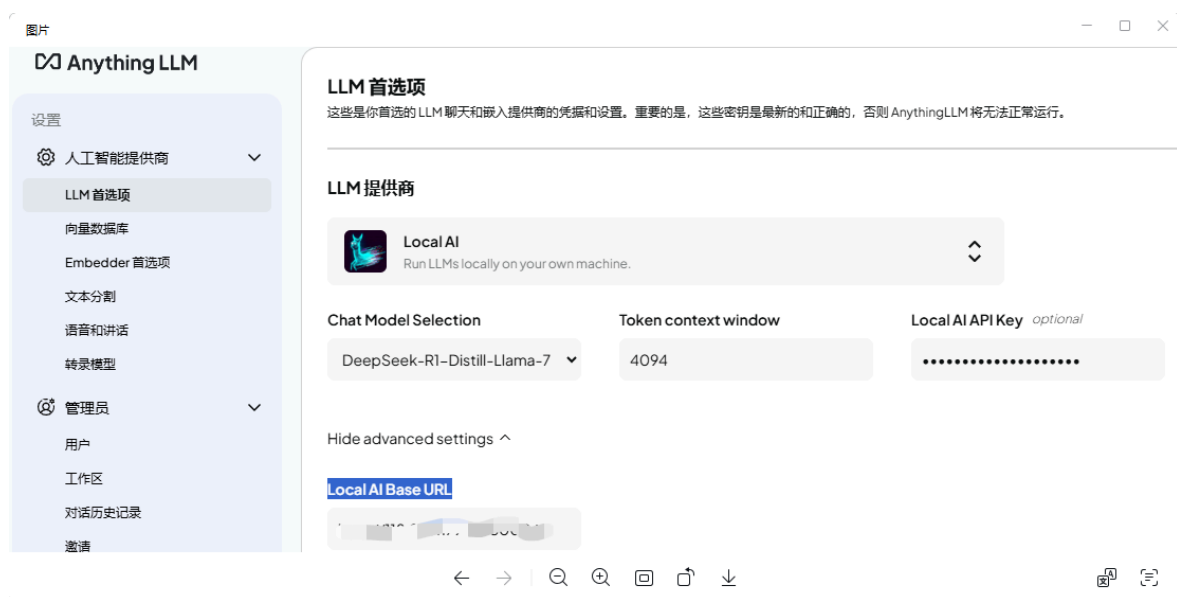
# 最佳实践

接入GPU物理机提供的DS模型

1. 登录Anything LLM，选择设置LLM首选项。



2. 设置“Local AI Base URL”，其中“Local AI Base URL”填充GPU物理机DS地址和端口，然后保存。



## Embedding模型

不同的任务需要不同的模型和数据库支持。如信息检索任务，需要 embedding 模型能准确捕捉文本语义信息，向量数据库具备高效的检索功能；对于推荐系统任务，模型要能挖掘用户和物品的潜在特征，数据库要支持快速的相似性计算。

Anything LLM支持多种Embedding模型引擎，包括AnythingLLM Native Embedder (default)、OpenAI、Azure OpenAI、LocalAi (all)、Ollama (all)、LM Studio (all)、Cohere。下面以LocalAi (all)使用BGE - M3模型为例说明，其他请参考对应软件官网安装使用。

### 1. 创建GPU云主机并开放安全组规则。

BGE - M3 是一款文本嵌入模型，在部署使用时，云主机配置会受任务类型（如推理、训练）、数据规模、并发请求数量等因素影响。如果是训练任务建议使用1张A100，其他任务可根据实际需求创建规格。创建步骤请参考[创建配备GPU驱动的GPU云主机（Linux）-GPU云主机-用户指南-创建GPU云主机 - 天翼云](#)。

安全组放开Anything LLM云主机弹性IP+端口8000，参考[添加安全组规则-弹性云主机-用户指南-安全-安全组-配置安全组规则 - 天翼云](#)。

### 2. 下载模型

#### 2.1.安装依赖包

参考文档在[天翼云使用vLLM运行DeepSeek的最佳实践-32B等版本-自定义部署DeepSeek-步骤三：手动部署DeepSeek -> 1.安装依赖包](#)。

#### 2.2.下载BGE - M3模型

将以下内容保存为 model\_download.py 文件，参数 cache\_dir 为模型的下载路径，您可以按需修改，需确保存储空间足够存放模型。

```
from modelscope import snapshot_download
model_dir = snapshot_download('BAAI/bge-m3', cache_dir='/root/model-path', revision='master')
```

运行该文件，启动模型下载。

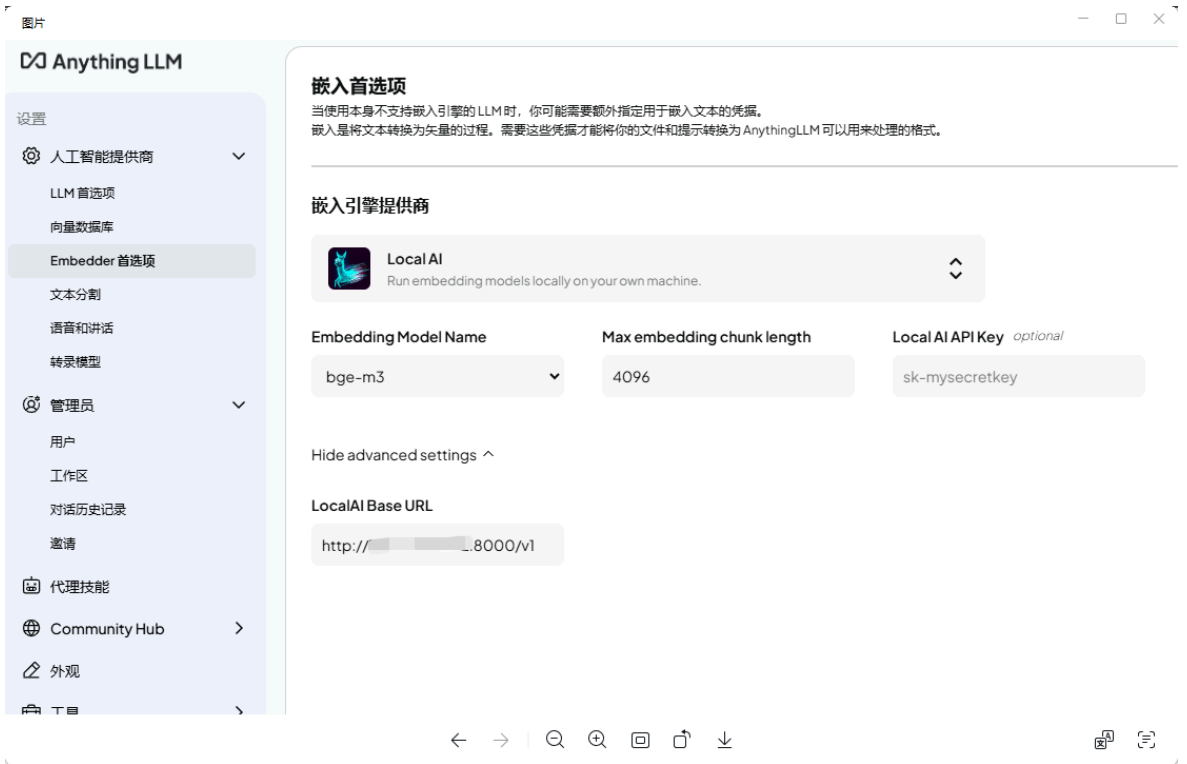
```
python model_download.py
```

#### 2.3.运行模型

```
vllm serve BAAI/bge-m3 --served-model-name bge-m3 --gpu-memory-utilization 0.95 --enforce
```

# 最佳实践

## 3. 配置Embedding模型引擎



## 4. 使用Embedding模型引擎



# 最佳实践

## 向量数据库

Anything LLM支持多种向量数据库，包含LanceDB (default)、Astra

DB、Pinecone、Chroma、Weaviate、Qdrant、Milvus、Zilliz。以下为各个向量数据库的对比描述：

向量数据库	开源/闭源	主要特点	支持语言	部署方式	性能优化	索引算法	使用场景
LanceDB	开源	基于 Rust，支持多模态数据，零拷贝、自动版本控制	Python、Java、TypeScript	本地/云	GPU 加速	HNSW、IVF等	AI 应用、多模态数据检索
Astra DB	闭源（云服务）	高可用性、弹性扩展，支持多模态数据	Python、Java、TypeScript等	云服务	分布式优化	HNSW等	需要高可用性和弹性扩展的应用
Pinecone	闭源（云服务）	高性能向量搜索，支持 GPU 和 TPU 加速	Python、Java等	云服务	GPU/TPU 加速	HNSW、IVF等	大规模向量搜索
Chroma	开源	适用于语义搜索和相似性匹配，支持单机和分布式部署	Python、Java等	本地/云	并行查询优化	HNSW等	AI 应用、语义搜索
Weaviate	开源	云原生，支持多模态数据，模块化设计	Python、Java等	本地/云	高效查询优化	HNSW等	多模态数据检索、推荐系统
Qdrant	开源	高效向量检索，支持加密和安全访问控制	Python、Java等	本地/云	并行查询优化	HNSW等	高性能向量检索
Milvus	开源	分布式向量数据库，支持大规模数据查询	Python、Java、++等	本地、云	GPU 加速	HNSW、IVF等	人工智能、推荐系统
Zilliz	开源（云服务）	基于 Milvus，深度优化，支持云原生部署	Python、Java等	云服务	高性能优化	HNSW等	云原生向量检索

下面以Chroma为例进行说明，其他请参考对应软件官网安装使用。

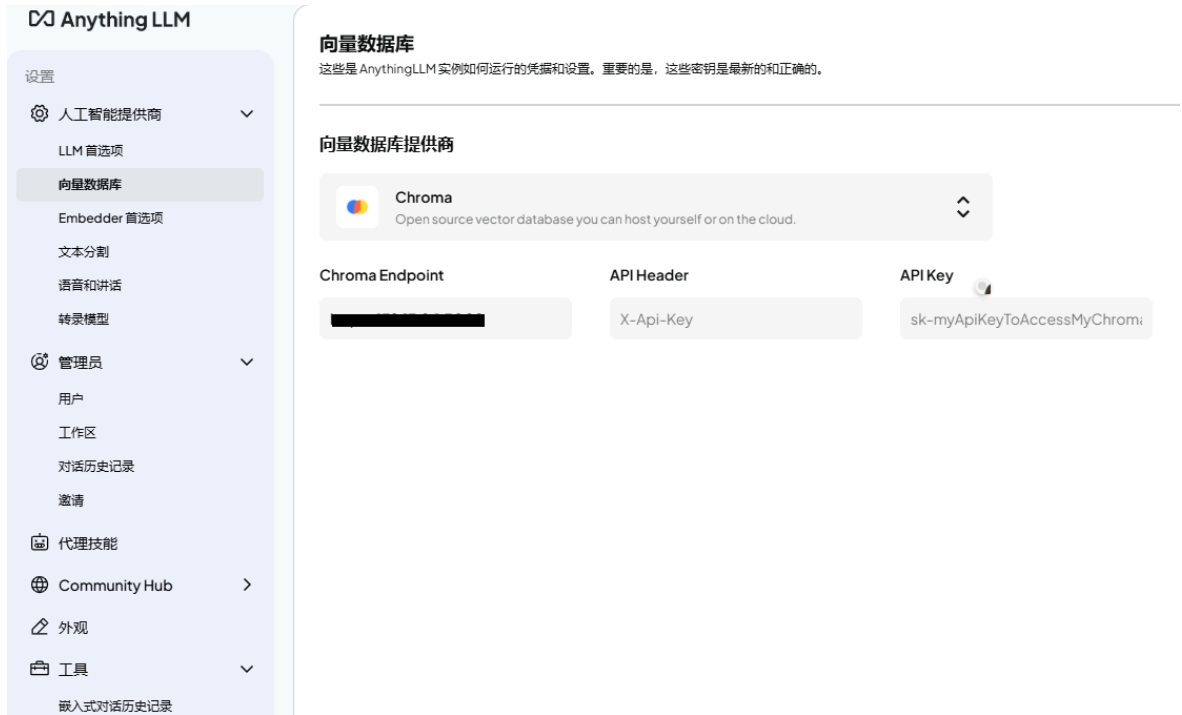
1. 创建一台挂载XSSD云盘的本地盘云主机/通用云主机，并增加Anything LLM云主机弹性IP+端口8000 的安全组规则。创建步骤请参考[创建弹性云主机-弹性云主机-快速入门 - 天翼云](#)，增加安全组规则参考[添加安全组规则-弹性云主机-用户指南-安全-安全组-配置安全组规则 - 天翼云](#)。

## 2. 安装向量数据库

```
docker pull chromadb/chroma:latest
```

```
docker run -d --name chroma -p 8000:8000 chromadb/chroma:latest
```

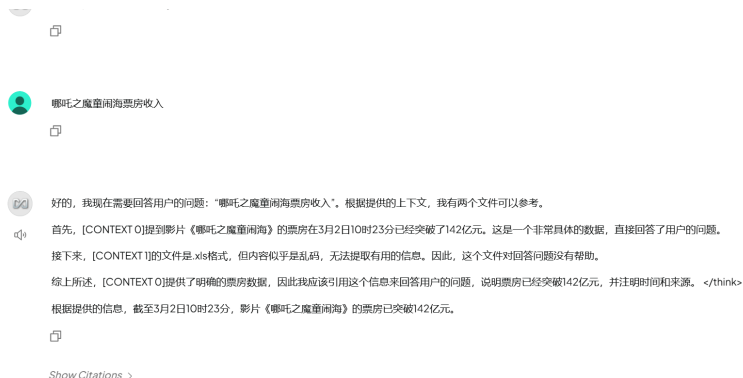
## 3. 配置向量数据库



The screenshot shows the 'AnythingLLM' settings interface. On the left is a navigation sidebar with categories like '设置' (Settings), '人工智能提供商' (AI Providers), '向量数据库' (Vector Database), 'Embedder 首选项' (Embedder Preferences), '文本分割' (Text Splitting), '语音和讲话' (Voice and Speech), '转录模型' (Transcription Models), '管理员' (Admin), '代理技能' (Agent Skills), 'Community Hub', '外观' (Appearance), and '工具' (Tools). The '向量数据库' section is active, displaying the following configuration:

- 向量数据库** (Vector Database): A dropdown menu is set to 'Chroma', with the description 'Open source vector database you can host yourself or on the cloud.'
- Chroma Endpoint**: A text input field containing a redacted URL.
- API Header**: A text input field containing 'X-API-Key'.
- API Key**: A text input field containing 'sk-myApiKeyToAccessMyChrom:'.





## 大模型推理性能测试实践

### 一、前言

本文介绍三种大模型评测工具，包括EvalScope、LLMPerf和vLLM Benchmark，总结每种测评工具的测试实践，并对三种测评工具在核心功能、性能指标、典型应用场景等方面进行了对比。

### 二、EvalScope

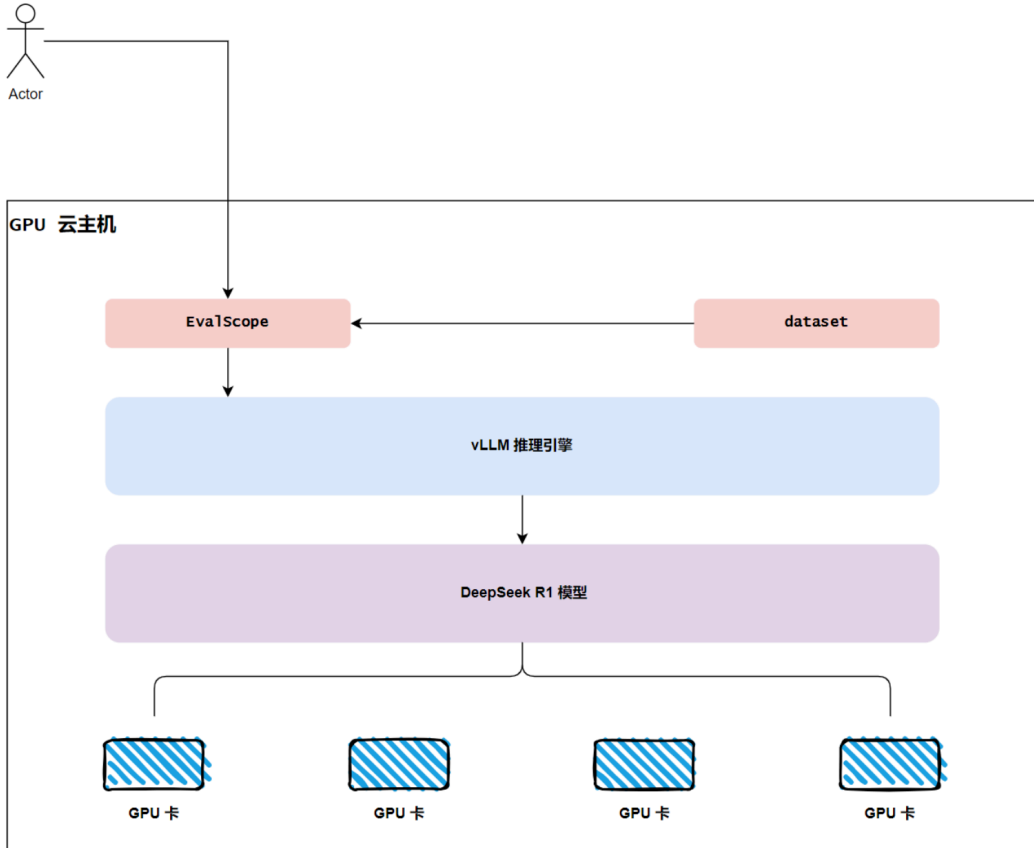
#### 1. 前言

EvalScope是魔搭社区官方推出的模型评测与性能基准测试框架，内置多个常用测试基准和评测指标，如MMLU、CMMLU、C-Eval、GSM8K、ARC、HellaSwag、TruthfulQA、MATH和HumanEval等；支持多种类型的模型评测，包括LLM、多模态LLM、Embedding模型和Reranker模型。

下面介绍基于EvalScope评测DeepSeek R1模型，涵盖基本评测和模型推理性能压测两大场景。

## 2. 测试环境配置

如下图所示为使用EvalScope测评工具对DeepSeek R1模型进行测试的拓扑图，其中推理引擎选择vLLM。



安装EvalScope测评工具的步骤如下所示：

```
pip install evalscope
```

## 3. 测试实践

- 基本评测场景

本文采用指定模型API地址（`api_url`）和API Key（`api_key`），评测部署的模型API服务，指定`eval-type`参数为`service`。

测试命令示例：

```
evalscope eval \
--model DeepSeek-R1-Distill-Qwen-32B \
--api-url http://127.0.0.1:8000/v1/chat/completions \
--api-key Vb7h4cVRFPgfAr8a \
--eval-type service \
--datasets gsm8k \
```

## 最佳实践

```
--limit 10
```

测试结果:

Model	Dataset	Metric	Subset	Num	Score	Cat.0
DeepSeek-R1-Distill-Qwen-32B	gsm8k	AverageAccuracy	main	10	0.9	default

- 推理性能压测场景

使用openqa数据集进行推理性能压测。

openqa可以模拟真实的问答情况，更接近现实场景，其中包括了超3000条中文对话数据，可以有效评估模型在实际应用中的推理性能。

测试命令示例:

```
evalscope perf \
 --url "http://127.0.0.1:8000/v1/chat/completions" \
 --parallel 1 \
 --model qwen2.5 \
 --number 15 \
 --api openai \
 --dataset openqa \
 --stream
```

要准确统计Time to First Token (TTFT)指标，需要在请求中包含--stream参数。

测试结果如下所示:

Percentile results:

Percentile	TTFT (s)	TPOT (s)	Latency (s)	Input tokens	Output tokens	Throughput(tokens/s)
10%	0.0202	0.0027	0.1846	41	50	270.8324
25%	0.0209	0.0028	0.2861	44	83	290.0714
50%	0.0233	0.0028	0.7293	49	250	335.644
66%	0.0267	0.0029	0.9052	50	308	340.2603
75%	0.0437	0.0029	0.9683	53	325	341.947
80%	0.0438	0.003	1.0799	58	376	342.7985
90%	0.0439	0.0032	1.2474	62	424	345.5268
95%	0.0463	0.0033	1.3038	66	431	348.1648
98%	0.0463	0.0035	1.3038	66	431	348.1648

## 最佳实践

99%	0.0463	0.0037	1.3038	66	431	348.1648	
+-----+-----+-----+-----+-----+-----+-----+-----+							
+-----+-----+-----+-----+-----+-----+-----+-----+							

使用openqa可以有效评估模型在处理实际自然语言问题时的具体响应时间，如上展示了模型在不同百分位下的TFTT(首token时间)、TPOT(生成每个token需要的时间)、Latency(延迟时间)、Input tokens(输入token)、Output tokens(输出token)、Throughput(吞吐量)等指标。

此外，使用Speed Benchmark工具可以重点测试模型在不同输入长度下的推理速度。以下为测试生成2048 tokens时的速度，输入长度分别为1、6144、14336、30720。

测试命令示例：

```
evalscope perf \
--parallel 1 \
--url http://127.0.0.1:8000/v1/completions \
--model DeepSeek-R1-Distill-Qwen-32B \
--log-every-n-query 1 \
--connect-timeout 60000 \
--read-timeout 60000 \
--max-tokens 2048 \
--min-tokens 2048 \
--api openai \
--dataset speed_benchmark \
--api-key Vb7h4cVRFPgfAr8a
```

参数说明如下所示：

- parallel: 指定并发数，默认为1个；
- url: 指定待测试系统的服务地址；
- model: 指定待测试的模型名称；
- log-every-n-query: 指定日志打印间隔；
- connect-timeout: 指定网络连接超时时间，单位为毫秒；
- read-timeout: 指定网络读取超时时间，单位为毫秒；
- max-tokens: 指定输出最大token值；
- min-tokens: 指定输出最小token值；
- api: 指定服务API类型；
- dataset: 指定数据集；
- api-key: 指定认证的API key值。

测试结果：

+-----+-----+-----+-----+			
Prompt Tokens	Speed(tokens/s)	GPU Memory(GB)	
+-----+-----+-----+-----+			
1	44.58	0.0	
6144	42.47	0.0	
14336	34.55	0.0	

| 30720 | 31.72 | 0.0 |

从测试结果可以看出，随着输入token数量增加，模型推理速度下降。

- 定制化推理性能压测场景

我们对Speed Benchmark 工具代码进行了修改，增加了输入token和repeat参数，用户可以在压测命令中可以直接指定需要测试的输入token长度和repeat值，不需要去修改代码重新配置环境，改动代码如下所示：

1) 修改数据集文件：evalscope/perf/plugin/datasets/speed\_benchmark.py。注释掉之前的INPUT\_LENGTH和REPEAT，然后替换初始化代码和build\_messages方法

```
def __init__(self, query_parameters: Arguments):
 super().__init__(query_parameters)
 self.input_length = getattr(query_parameters, 'dataset_input_length',
[1, 6144, 14336, 30720])
 # 从参数获取重复次数，默认值为 2
 self.repeat = getattr(query_parameters, 'dataset_repeat', 2)

def build_messages(self) -> Iterator[List[Dict]]:
 for input_len in self.input_length:
 for _ in range(self.repeat):
 yield self.create_query(input_len)
```

2) 修改参数加载文件evalscope/perf/arguments.py，修改如下

#在Arguments中添加对应字段

```
dataset_input_length=args.dataset_input_length,
```

```
dataset_repeat=args.dataset_repeat
```

#在add\_argument函数里添加：

```
parser.add_argument(
 '--dataset-input-length',
 nargs='+',
 type=int,
 default=[1, 6144, 14336, 30720],
 help='The length of the input, default [1, 6144, 14336, 30720]')
parser.add_argument('--dataset-
repeat', type=int, default=2, help='The repeat times of the dataset')
```

安装修改后的evalscope工具后，执行测试的示例如下所示：

```
evalscope perf \
--parallel 1 \
--url http://127.0.0.1:8000/v1/completions \
--model DeepSeek-R1-Distill-Qwen-32B \
--log-every-n-query 1 \
--connect-timeout 60000 \
--read-timeout 60000
```

```
--max-tokens 2048 \
--min-tokens 2048 \
--api openai \
--dataset speed_benchmark \
--api-key Vb7h4cVRFPgfAr8a \
--dataset-input-length 256 1024 2048 \
--dataset-repeat 5
```

当我们传入多个dataset-input-length值时会同时测试多个不同的输入长度的样本，总的请求数为dataset-input-length输入的样本个数乘以dataset-repeat传入的值。

## 三、LLMPerf

### 1. 前言

LLMPerf（大型语言模型性能测试）是评估大语言模型（LLM）在生产环境中表现的核心评测体系。本文基于真实业务场景提炼最佳实践方法论，为开发、运维与测试团队提供可靠的测试方案，系统性保障服务的高吞吐、低延迟和稳定性。

### 2. 测试环境配置

#### 测试脚本下载

LLMPerf由 Ray 项目团队开发，在github上开源，可直接下载。

```
git clone https://github.com/ray-project/llmperf
```

#### 工具核心功能：

##### 性能评估

- **负载测试：**LLMPerf可以对大语言模型（如Llama 2、GPT-3等）的吞吐量和延迟性能进行测量。它通过负载测试来评估模型在不同负载下的响应延迟和生成吞吐量，帮助用户了解模型在实际应用中的性能表现。
- **准确性测试：**该工具还可以进行正确性测试，衡量模型输出的准确性，确保模型在生成内容时能够保持较高的质量。

##### 参数化测试

LLMPerf允许用户更改输入和输出大小等参数，以匹配不同的应用场景。这意味着用户可以根据自己的具体需求和工作负载，灵活地调整测试参数，从而获得更具针对性的性能评估结果。

##### 结果分析

LLMPerf能够提供详细的性能指标分析，包括响应时间、吞吐量、准确性等，帮助用户深入了解模型的性能特点。

##### 兼容性

LLMPerf支持主流的大语言模型，如OpenAI、Anthropic、TogetherAI等。这使得用户可以在一个统一的框架下对不同来源的模型进行性能比较和评估。

##### 安装测试工具

```
git clone https://github.com/ray-project/llmperf
cd llmperf
pip install -e .
```

#如下载速度较慢可以使用国内源进行加速

```
pip install -e . -i https://pypi.tuna.tsinghua.edu.cn/simple
```

## 配置环境变量

根据不同的API服务，需要设置不同的密钥信息和服务地址。例如对于OpenAi（vLLM类推理框架）：

```
export OPENAI_API_KEY="your api-key"
export OPENAI_API_BASE="https://api.endpoints.anyscale.com/v1"
```

## 3. 测试实践

### 性能测试

使用token\_benchmark\_ray.py脚本运行测试，主要参数如下：

- `--model`：被测的模型名称（如DeepSeek-R1-Distill-Qwen-32B）。
- `--mean-input-tokens`：输入token的均值。
- `--mean-output-tokens`：输出token均值。
- `--stddev-input-tokens`：输入token的标准差。
- `--stddev-output-tokens`：输出token的标准差。
- `--num-concurrent-requests`：并发请求数。

示例：

```
python token_benchmark_ray.py \
 --model "DeepSeek-R1-Distill-Qwen-32B" \
 --mean-input-tokens 256 \
 --stddev-input-tokens 0 \
 --mean-output-tokens 256 \
 --stddev-output-tokens 0 \
 --max-num-completed-requests 200 \
 --timeout 600 \
 --num-concurrent-requests 10 \
 --results-dir "result" \
 --llm-api openai \

```

## 最佳实践

```
--additional-sampling-params '{}'
```

用vLLM部署DeepSeek-R1-Distill-Qwen-32B模型进行测试，测试结果主要指标如下所示：

```
~~~~~  
-- TTFT(Time To First Token)  
   mean = 3.942 s  
-- Prefill-Throughput (tokens/second)  
   mean = 10391.209 tokens/second  
-- TPOT(Time Per Output Token)  
   mean = 0.030 s  
-- Decoder-Throughput  
   mean = 665.873 tokens/second  
~~~~~  
Number Of Errored Requests: 0
Overall Output Throughput: 454.10329598217925
Number Of Completed Requests: 180
Completed Requests Per Minute: 17.108177128730524
~~~~~
```

测试结果还包含一些不同分位的指标，这里只展示关键指标，其主要含义如下：

- TTFT（首token时间）：请求开始到收到第一个token的时间。
- Prefill-Throughput：表示模型处理输入Prompt的速率。
- TPOT：表示生成相邻token之间的间隔时间。
- Decoder-Throughput：模型的输出速率。

### 准确性测试

使用llm\_correctness.py脚本可进行文本转换任务，测试模型准确性，其主要测试模型能否准确将文本转换为数字。

示例：

```
python llm_correctness.py \  
  --model "DeepSeek-R1-Distill-Qwen-32B" \  
  --max-num-completed-requests 100 \  
  --num-concurrent-requests 5
```

用vLLM部署DeepSeek-R1-Distill-Qwen-32B模型进行准确性测试，主要测试模型的文本转化能力，测试结果如下所示：

```
Results for llm correctness test for DeepSeek-R1-Distill-Qwen-32B queried with the openai api.  
Errors: 0, Error rate: 0.0  
Mismatched: 0, Mismatch rate: 0.0  
Completed: 35  
Completed without errors: 35
```

可以看到模型的文本转换请求全部成功，没有失败的请求。

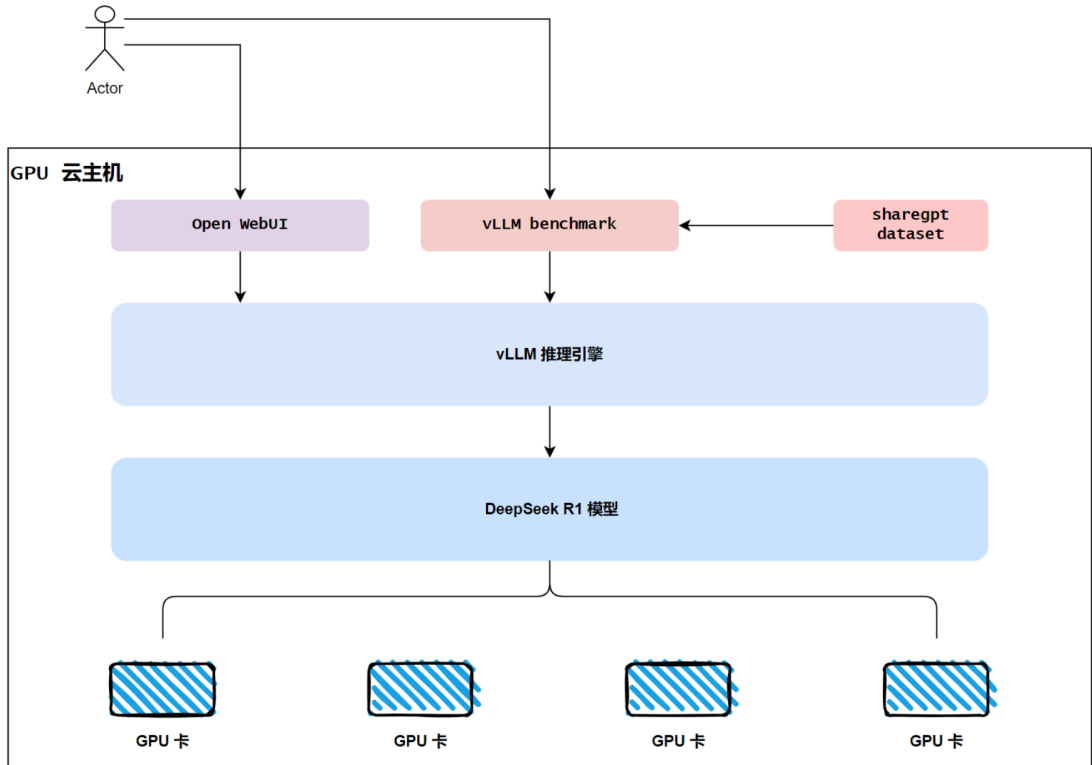
## 四、vLLM Benchmark

### 1. 前言

vLLM Benchmark 是 vLLM 框架用于评估和优化其推理性能的标准化测试工具集，核心作用是为大语言模型（LLM）的推理服务提供量化指标，帮助开发者优化模型部署策略。

## 2. 测试环境配置

如下图所示为测试环境拓扑，其中GPU云主机配置有4块GPU卡。部署DeepSeek的模型，安装vLLM推理引擎服务并启动；启动Open WebUI服务来接入到vLLM，实现从浏览器客户端登录系统进行问答；vLLM Benchmark用于进行性能压力测试，sharegpt dataset数据集用于性能压测过程的输入数据。



可以通过如下命令来安装vLLM Benchmark工具脚本：

```
git clone https://github.com/vllm-project/vllm.git
cd vllm/benchmarks/
```

# 最佳实践

## 3. 测试实践

使用vLLM Benchmark进行性能测试时可以指定sharegpt数据集。

Sharegpt数据集的所有问题均来自于真实的用户提问，更贴近实际应用场景，可以全面、真实地评估模型在实际应用场景中的表现。具体的指标和测试方法见下文。

如下所示为性能压测命令：

```
python benchmark_serving.py --model /root/model-path/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B --backend vllm --dataset-name sharegpt --dataset-path /root/ShareGPT_V3_unfiltered_cleaned_split.json --max-concurrency 1 --num-prompts 50
```

命令参数解释如下：

- model: 指定模型路径；
- backend: 指定后端推理引擎类型；
- dataset-name: 指定数据集类型；
- dataset-path: 指定数据集路径；
- max-concurrency: 指定并发数，默认为1；
- num-prompts: 指定总的请求数；

对应的测试结果如下所示：

```
===== Serving Benchmark Result =====
Successful requests:                50
Benchmark duration (s):             229.88
Total input tokens:                 11748
Total generated tokens:             10162
Request throughput (req/s):         0.22
Output token throughput (tok/s):    44.21
Total Token throughput (tok/s):    95.31
-----Time to First Token-----
Mean TTFT (ms):                    95.08
Median TTFT (ms):                  64.17
P99 TTFT (ms):                    208.99
-----Time per Output Token (excl. 1st token)-----
Mean TPOT (ms):                    22.27
Median TPOT (ms):                  22.26
P99 TPOT (ms):                    22.45
-----Inter-token Latency-----
Mean ITL (ms):                     22.26
Median ITL (ms):                   22.03
P99 ITL (ms):                      23.67
=====
```

指标说明：

测试结果中的具体指标含义如下所示：

- Output token throughput(tok/s)：输出 Token 吞吐量，单位为 tok/s，是衡量大模型服务生成效率的核心指标，指系统每秒能够生成的输出 token 数量。它直接反映模型的文本生成速度和系统的整体处理能力，是评估推理成本和服务商业价值的关键依据。
- TTFT: Time to First Token，首 Token 延迟，是衡量大模型服务响应速度的核心指标，具体指从用户发送请求到系统生成第一个输出。
- TPOT: 指排除首 Token 延迟后，生成每个后续输出 token 的平均耗时。计算公式为：总解码时间 / 输出 token 数量。例如，若 TPOT 为 50ms，则每秒可生成 20 个 token。
- ITL: 连续两个输出 token 之间的间隔时间，单位毫秒，反映生成稳定性。

## 指定输入输出token长度

当使用的数据集为random类型的时候，执行vLLM Benchmark压测命令的时候可以指定输入长度和输出长度，测试命令如下：

```
python vllm/benchmarks/benchmark_serving.py \  
--model /root/model-path/deepseek-ai/DeepSeek-R1-Distill-Qwen-32B \  
--backend vllm \  
--dataset-name random \  
--random-input-len 256 \  
--random-output-len 256 \  
--max-concurrency 1 \  
--num-prompts 50
```

进行上述测试时如果使用vllm启动的时候添加了--served-model-name参数以及--api-key参数，测试的时候需要在上述命令中加入--served-model-name内容，并且在环境变量中添加OPENAI\_API\_KEY=yourkey。

## 五、三种测评工具对比

### 1. 核心功能对比

- vLLM Benchmark
  - 专注于LLM推理引擎性能评测，支持OpenAI接口兼容的框架（如Qwen系列模型）
  - 内置TTFT（首Token延迟）、TPOT（后续Token时延）、吞吐量等核心指标
  - 支持长上下文压力测试（如LongAlpaca-12k数据集）
- EvalScope
  - 大模型多维度评估框架，覆盖性能测试（Perf）、效果评测（Accuracy）和合规性检测
  - 提供标准化测试集（如MMLU、CMMLU）和自定义数据集扩展能力，支持多种测评场景
  - 集成生产级压力测试功能，支持并发请求模拟与实时监控（通过Wandb可视化）
- LLMPeef
  - 开源基准测试工具，专攻API级性能评估（延迟、吞吐量、请求成功率）
  - 支持正确性验证（如数字格式转换准确性测试）

## 2. 性能指标侧重

工具	核心指标	测试场景差异
vLLM Benchmark	首Token延迟 (TTFT)、Token吞吐量、显存占用率	重推理引擎的硬件资源利用率，常用于GPU云主机环境下的引擎选型
EvalScope	综合吞吐量、QPS、端到端延迟、多模态任务准确率	适配生产环境压力测试，支持API服务与本地模型的混合评测
LLMPerf	Token间延迟、首Token延迟 (TTFT)、请求吞吐量、错误率	强调高并发模拟能力 (如10+并发请求)，适合API服务的SLA验证

## 3. 技术特性差异

- 部署复杂度
  - vLLM Benchmark: 从vLLM的github代码仓库源克隆
  - EvalScope: 提供Docker镜像和Python CLI工具链，支持快速接入ModelScope社区模型和数据集
  - LLMPerf: 依赖Ray分布式框架，需修改Tokenizer配置适配私有化模型
- 测试数据灵活性
  - vLLM Benchmark: 使用ShareGPT\_V3、LongAlpaca等结构化数据集，也可使用随机数据自定义输入长度
  - EvalScope: 允许自定义输入长度 (--dataset-input-length) 和样本重复次数
  - LLMPerf: 支持动态调整输入Token分布参数 (--mean-input-tokens)
- 结果可视化
  - vLLM Benchmark/EvalScope: 支持Wandb面板实时监控测试进度
  - LLMPerf: 生成CSV格式原始数据，需二次处理

## 4. 典型应用场景

- vLLM Benchmark
  - 推理引擎选型 (如对比vLLM、LMDeploy、TensorRT-LLM的吞吐量)
  - 长上下文支持能力验证 (如测试16k+Token输入的显存溢出风险)
- EvalScope
  - 生产环境服务容量规划 (如通过--parallel 模拟多路并发)
  - 多模态模型效果-性能平衡分析 (如Embedding模型响应延迟与召回率关联性)
- LLMPerf
  - 云API服务商性能对比 (如Anthropic Claude vs OpenAI GPT-4 Turbo)
  - 模型部署前的SLA合规性检查 (如验证P99延迟是否达标)

## 5. 工具选型建议

- **LLMPerf**: 功能比较单一, 适合API基准测试以及快速对比多个LLM API性能场景, 聚焦于LLM API性能(如延迟、吞吐量、请求成功率)和正确性测试(输出格式与内容验证);
- **EvalScope**: 功能全面且灵活, 适合复杂评测的需求场景, 支持多维度评估, 包括模型能力、性能压测、端到端RAG评测、多模态模型评测等。适合综合评估模型在学术研究、工业部署中的表现, 支持自定义评测指标和复杂场景(如竞技场模式、长文本生成测试);
- **vLLM Benchmark**: 主要针对vLLM推理引擎的性能优化(如PagedAttention技术效果、多卡扩展性)。专为优化vLLM框架的部署参数设计, 适合需要调整批次大小、显存利用率等参数以实现高吞吐量推理的场景。

## 六、附录

[vLLM官方代码仓库](#)

[EvalScope官方文档](#)

[LLMPerf官方代码仓库](#)

[DeepSeek提示库](#)

## Open WebUI 及vLLM版本升级最佳实践

### 一、Open WebUI 升级指南

#### 1. 查看现有版本

```
docker exec -it open-webui bash -c "cat /app/package.json|grep version"
```

```
root@ecm-372c:~# docker exec -it open-webui bash -c "cat /app/package.json|grep version"
  "version": "0.5.10",
root@ecm-372c:~#
```

# 最佳实践

## 2. 版本升级

(1) 登录云主机，下载最新容器镜像

```
docker pull ghcr.nju.edu.cn/open-webui/open-webui:v0.6.5
```

```
[root@ecm-bf9f ~]# docker pull ghcr.nju.edu.cn/open-webui/open-webui:v0.6.5
v0.6.5: Pulling from open-webui/open-webui
8a628cdd7ccc: Pull complete
2a47a8c4fd5c: Pull complete
9c538fc35491: Pull complete
47bbb0afa7fe: Pull complete
782acb99e453: Pull complete
4f4fb700ef54: Pull complete
b7915144f9c7: Pull complete
633be3c6bab0: Pull complete
266a80d83771: Pull complete
7b4a3fa111d1: Pull complete
693caf783e3a: Downloading [=====] 637.9MB/1.043GB
ec0cdf90a948: Download complete
39845d3298e5: Download complete
55245328d5ea: Download complete
87ea629f5ffc: Download complete
```

(2) 删除旧镜像

查看当前容器镜像

```
[root@ecm-bf9f ~]# docker image ls
REPOSITORY                                TAG      IMAGE ID      CREATED      SIZE
ghcr.nju.edu.cn/open-webui/open-webui    v0.6.5   ba71477c0bbc  2 weeks ago  4.59GB
ghcr.nju.edu.cn/open-webui/open-webui    cuda     ab45a42f3e76  2 months ago  8.78GB
```

删除旧容器镜像

```
docker image rm ghcr.nju.edu.cn/open-webui/open-webui:cuda
```

```
[root@ecm-bf9f ~]# docker image rm ghcr.nju.edu.cn/open-webui/open-webui:cuda
Untagged: ghcr.nju.edu.cn/open-webui/open-webui:cuda
Untagged: ghcr.nju.edu.cn/open-webui/open-webui@sha256:72e3a276586471d5188fb9277d012eeefb91e3e47e971cbc8d98396590dadf36
Deleted: sha256:ab45a42f3e76083fd3cd5f3a0def94bedb45d20fa512e22a83f5d51ca13993b7
Deleted: sha256:844554aaf616051a469258924b499c186e993e6b8bd1b7692f9c7bbf3c88afc8
Deleted: sha256:ebd4eaa516b8f2db2d5491c38538d19e29e0564260b3f288b223eee1904379b
Deleted: sha256:5b2392a345c11d990b8b7f4460db3f248b272866a287665db637eb61a2c638f8
Deleted: sha256:4f804b08c4c03b1c7cbdc79be7a2de648815f8c7a6e52f3eeb85b3eba7888471
Deleted: sha256:edcfa083c4cf051f6d0d5d3dd0c785347630004f6fe2922f53b12c3cbabc818e
Deleted: sha256:7b2e8194700d7d0f8629017ad7c06143a32d3e725d2eaf6f63f9f29262891e12
Deleted: sha256:75f11ef19a0ec655d8c77e829b76ba483ed563b65aba45d6532bb6840bc6ae42
Deleted: sha256:d3373f275461f0f2b452a32a80f9e1917f643559debd16e0dd1fd193a3f449a3
Deleted: sha256:57dab2b0c9806d399b06a4886d8f4655c264c90f3353a62714d8342b0fd91585
Deleted: sha256:6b4710dee645ddd2f834e405fc13b4d6eb98b78e7e04da8480e069893730f9a1
Deleted: sha256:29cd39632ff67290565056bc2b1d979c7c1cd6e23a6b87a15a09afe451a397f8
Deleted: sha256:9961f67f49aa0c506794b96672e88779ce6923adad2c47b9aeedc f63e94a6a17
Deleted: sha256:1a283ab5951ff9954c2898c76df19f044c50cddd29d7a5acc28fe78ac2f55e01
Deleted: sha256:60922dbb5c4ead2daa8c28c22529016875364b1830af541b07efd7a7fab2e05f
Deleted: sha256:e5cccb7a4f590bb889bbe78736dff1a7227ac4e75b3ba0dd9acab43e0860d8a6a
Deleted: sha256:7914c8f600f532b7adbd0b003888e3aa921687d62dbe2f1f829d0ab6234a158a
```

查看更新后的容器镜像



## 最佳实践

```
# pip3 install vllm==0.8.5
```

(2) 查看是否升级成功

```
# 如果是pip安装的执行
```

```
pip show vllm
```

```
# 如果是pip3安装的执行
```

```
# pip3 show vllm
```

```
[root@ecm-bf9f ~]# pip3 show vllm
Name: vllm
Version: 0.8.5
Summary: A high-throughput and memory-efficient inference and serving engine for LLMs
Home-page:
Author: vLLM Team
Author-email:
License:
Location: /usr/local/python3.10.16/lib/python3.10/site-packages
Requires: aiohttp, blake3, cachetools, cloudpickle, compressed-tensors, depyf, einops, fastapi, filelock,
orcer, mistral_common, msgspec, ninja, numba, numpy, openai, opencv-python-headless, opentelemetry-api, op
ntions-ai, outlines, partial-json-parser, pillow, prometheus-fastapi-instrumentator, prometheus_client, pr
ay, requests, scipy, sentencepiece, tiktoken, tokenizers, torch, torchaudio, torchvision, tqdm, transforme
Required-by:
[root@ecm-bf9f ~]# vi /etc/systemd/system/vllm.service
[root@ecm-bf9f ~]# /usr/local/python3.10.16/bin/vllm -v
INFO 05-01 09:08:49 [__init__.py:243] No platform detected, vLLM is running on UnspecifiedPlatform
0.8.5
[root@ecm-bf9f ~]#
```