

云计算导论实践----虚拟化实验

3.1 应用实践 1：计算虚拟化实践

云计算的实验环境要求一般比较复杂，最好在有数据中心硬件设备的支持下完成。考虑具体实验条件和环境，本书中采用嵌套虚拟化技术，尽量把复杂的实验拆分为小的实验，让读者使用配置高一点的笔记本计算机或台式计算机即可完成，希望通过实验可以提升大家对云计算理解，熟悉实现云计算的一些技术细节。

本实验中我们使用 VMware 和 SUSE linux 给大家演示如何创建和使用虚拟机。首先在自己的计算机上安装 VMware 虚拟化软件，安装一个操作系统为 SUSE Linux 虚拟机，该虚拟机我们已经给大家准备好了，就是一个标准的虚拟机文件（OVA 或 OVF 格式），也就是前面说的虚拟机被代码（code）化了，此处我们下载或复制该文件到本地即可，这实际上就是虚拟机的冷迁移过程。启动该虚拟机后，在此虚拟机上使用嵌套虚拟化技术创建 KVM 或 XEN 虚拟机，相关软件简要介绍如下。

VMware 成立于 1998 年，是最早专注于虚拟化商业软件并成功的公司。从虚拟化软件兴起之始，它就是这个市场的霸主，甚至在早期很多人的认知中将 VMware 等同于虚拟化。直到随着公有云的兴起，VMware 开始受到 Xen 和 KVM 等开源项目以及微软 Azure/HyperV 的挑战。VMware 应用较为广泛的虚拟化产品有 VMware Workstation 和 VMware ESXI。VMware Workstation 是桌面级虚拟化产品，运行在 Windows、Linux 和 macOS 操作系统上，是 type-2 hypervisor。VMware ESXI 是服务器级（裸金属架构）的虚拟化软件，它直接运行在硬件平台上，是 type1 hypervisor，很多大公司的私有云都是用它来搭建的。

SUSE 是德国 SuSE Linux AG 公司发行维护的 Linux 发行版，第一个版本出现在 1994 年年初，2004 年这家公司被 Novell 公司收购。SUSE Linux 是一种可适应任何环境的操作系统，允许开发人员和管理员在本地、云端和边缘部署应用服务，并且专门针对性能、安全性和可靠性进行了优化。SUSE 也是全球范围内企业级开源解决方案，专注于企业级 Linux、企业容器管理和边缘解决方案，通过与合作伙伴和社区合作，帮助客户随时随地在任意场景进行创新——无论是在数据中心、云端还是边缘环境。SUSE 让“开源”重新“开放”，使客户能够应对创新挑战，并能够在未来发展其 IT 战略和解决方案。

Xen 是由剑桥大学计算机实验室开发的一个开源项目，可以直接运行在计算机硬件上，并且可以并发地支持多个客户操作系统。Xen 能够支持多种处理器，如 x86、PowerPC 和 ARM 等，所以它可以运行在较多种类的设备上。目前，Xen 支持的客户操作系统有 Linux、NetBSD、FreeBSD、Solaris、Windows 和其他一些常用的操作系统。Xen 包含 3 种基本组件，分别是 hypervisor、Dom 0 和 Dom U，如图 3-1（a）所示。

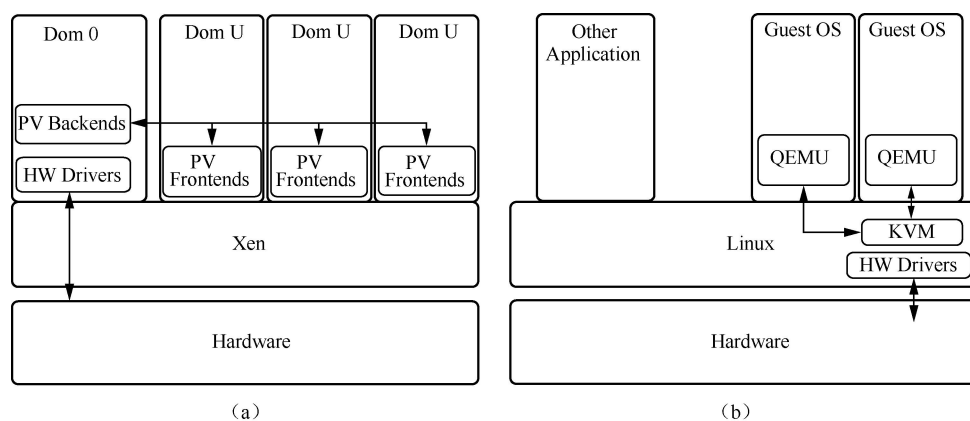


图 3-1 Xen 和 KVM 虚拟化示意

Dom 0 运行在 hypervisor 之上，是 Xen 的管理员，具有直接访问硬件和管理其他客户操作系统的权限。在 Dom 0 中，存在两个基本的驱动程序，分别是网络驱动程序和块存储驱动程序。网络驱动程序能够直接与本地的网络硬件进行通信，从而处理来自客户操作系统的网络请求。类似地，块存储驱动程序能够与本地的存储设备通信，处理客户操作系统的数据读写请求。Dom U 运行于 hypervisor 之上，是 Xen 虚拟环境中的客户虚拟机。Dom U 上运行的客户操作系统有两类，一种是半虚拟化客户机，另一种是完全虚拟化客户机。

这种将 Linux 内核对 CPU、内存管理等核心部分排除在外，由 hypervisor 接手的方式一开始就受到 Linux 内核开发人员的抵制，致使将 Linux 内核作为 Dom 0 对 Xen 的支持部分一直不能合入 Linux 内核社区。直到 2010 年，在采用了基于内核的 PVOSs 方式大量重写了 Xen 代码之后，才被勉强融合入 Linux 内核社区。2011 年，从 Linux 内核 2.6.37 版本开始，正式支持 Xen Dom 0。美国思杰（Citrix）的 XenServer 就是基于 Xen 架构的一款优秀的服务器虚拟化平台。

KVM 全称是 kernel-based virtual machine，中文意思基于内核的虚拟机，是采用硬件虚拟化技术的全虚拟化解决方案。它是以 kernel module 的形式加载于 kernel，与 kernel 融为一体，将 Linux kernel 本身变成了一个 hypervisor。随着 2008 年 9 月 RedHat 公司对 KVM 开源项目和开发人员的收购，RedHat 开始在其 RHEL 发行版中集成了 KVM 取代之前的 Xen。得益于与 Linux 天然一体以及 RedHat 的倾力打造，KVM 已经成为最主流的 hypervisor（KVM 现在是 OpenStack 的默认 hypervisor）

KVM 虚拟化中每一个客户机就是一个 qemu 进程，客户机中的一个 vCPU（虚拟处理器）对应 qemu 进程中的一个执行线程，如图 3-1（b）所示。因新的 VirtIO 驱动框架可以提供高效的准虚拟化支持，并且因为其开源的特点，所以被国内大部分云提供商广泛使用。

1. 实验环境和工具软件准备

对于该实验，我们建议的硬件是 Intel I5 CPU，或与它同一等级或更高配的 CPU），内存必须在 8 GB 以上，磁盘空闲容量在 300 GB 以上。

实验需要的 VMware Workstation 15（或更新版本）、TinyCore 11、SUSE Enterprise Server 11 等软件家可以通过对应的官网下载，也可以从本书配套资源中下载。

2. 实验过程

步骤 1：准备虚拟机 SUSE-11-majun.ova 文件，可以通过网络下载或者直接通过 U 盘复制到本地。

步骤 2：安装并启动 VMware，选择打开虚拟机，选择前面的虚拟机文件，在弹出的对话框中，设置新虚拟机的名称和存储路径，单击“导入”按钮，如图 3-2 所示。



图 3-2 导入虚拟机界面

步骤 3：等待导入完毕。导入虚拟机成功界面如图 3-3 所示，这里记住用户名 root 和密码 Admin@1234。



图 3-3 导入虚拟机成功

步骤 4: 单击图 3-15 中的“开启此虚拟机”按钮, 这时出现虚拟机启动界面, 如图 3-4 所示。此处菜单第一项为 KVM 环境, 第三项为 XEN 环境, 因为 KVM 环境和 XEN 环境不能同时启动, 所以大多数 Linux 系统默认使用 KVM 虚拟机核心。本实验所用虚拟机同时配置了 KVM 环境和 Xen 环境, 因此在启动时通过菜单选择具体环境。

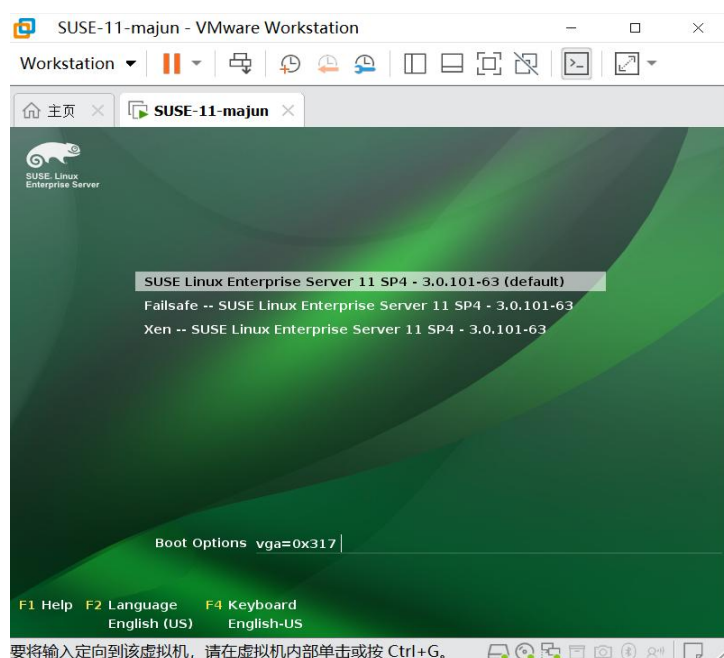


图 3-4 虚拟机启动界面

步骤 5: 选择第一项菜单启动 (默认 KVM 环境), 等待启动完成。之后输入用户名和密码进行登录, 完成后进入 SUSE 系统, 如图 3-5 所示。

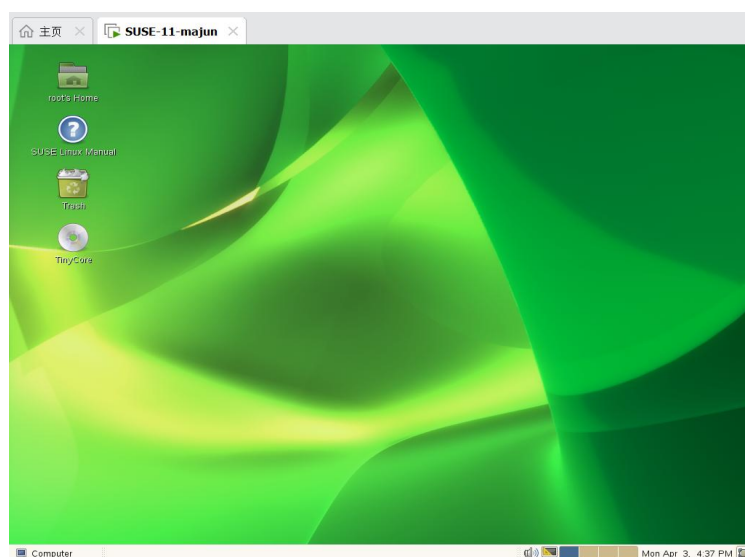


图 3-5 启动并登录成功界面

步骤 6: 在 SUSE 桌面上单击鼠标右键, 在弹出的菜单中选择“Open in Terminal”, 打开一个 Terminal 窗口, 在窗口中输入命令 `virt-manager` 启动虚拟机管理程序, 如图 3-6 所示。

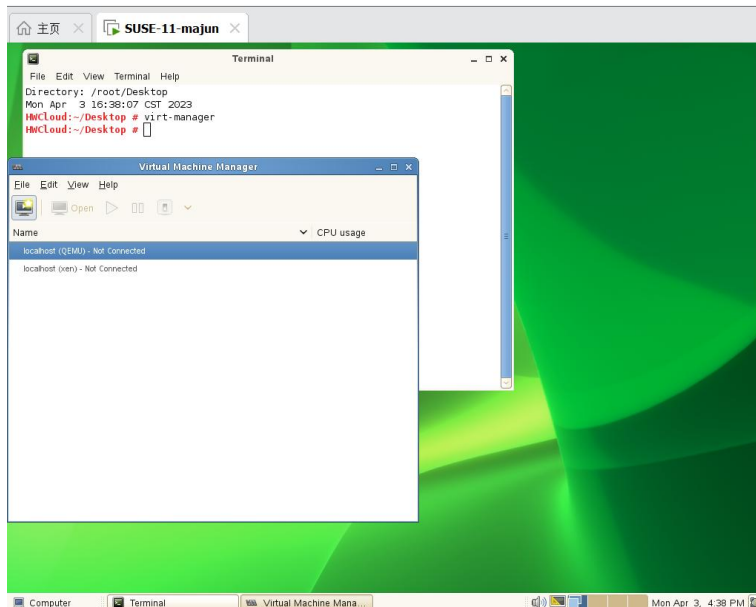


图 3-6 打开终端窗口并启动虚拟机管理程序界面

步骤 7：用鼠标右键单击第一个 localhost（QEMU）项目，在弹出的菜单中选择“Connect”，如图 3-7 所示。显示 connected 后就可以单击“Name”图标来创建 KVM 虚拟机了。在创建虚拟机之前，我们双击“localhost（QEMU）”，先检查一下虚拟机的配置，得到的 KVM 虚拟机基本参数如图 3-8 所示。

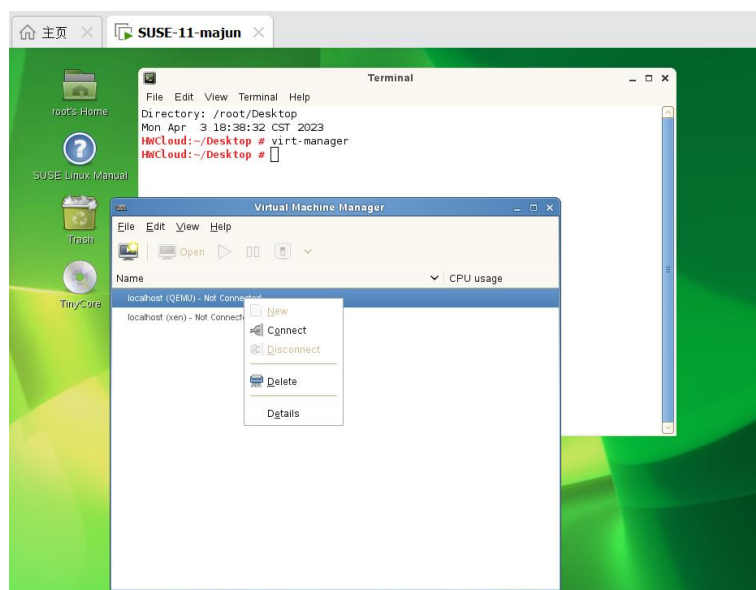


图 3-7 KVM 连接 QEMU

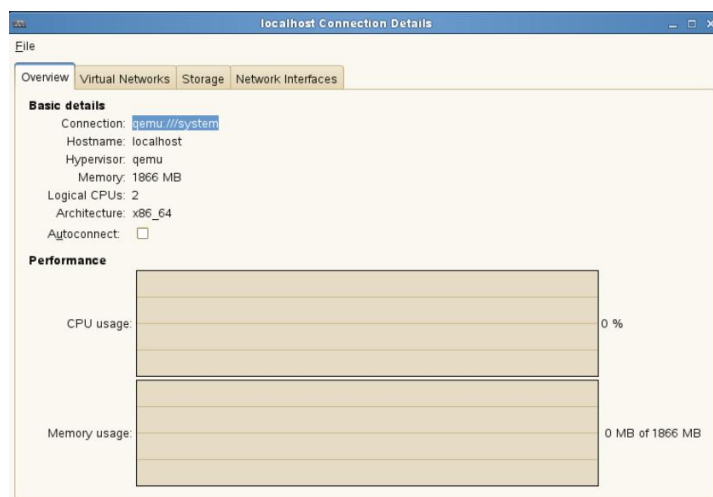


图 3-8 KVM 虚拟机基本参数

步骤 8: 单击“Name”图标，开始创建虚拟机。在弹出的“Create a Virtual Machine”对话框中（如图 3-9 所示）单击“Forward”。在“Install an Operating System”对话框中（如图 3-10 所示）选择“I need to install an operating system”选项，单击“Forward”按钮。在“Type of Operating System”对话框中（这里不展示界面，读者按步骤操作即可），选择“RedHat Enterprise Linux 3”（“Other”也可以），然后单击“Forward”按钮。

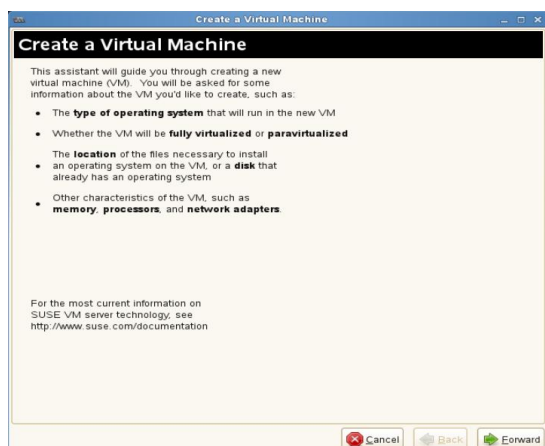


图 3-9 准备创建虚拟机

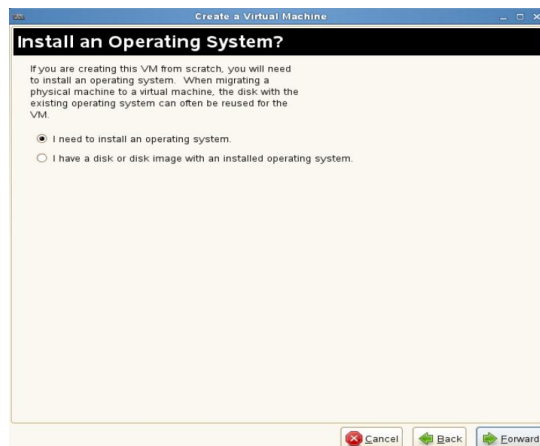


图 3-10 安装操作系统界面

步骤 9: 在图 3-11 所示的“Summary”对话框中可以再次检查和修改该虚拟机的各种参数，此处我们单击“Disks”链接。在弹出的“Disks”对话框中单击“CD-Rom”按钮，并在弹出的“Virtual Disk”对话框中单击“OK”按钮。之后，返回“Disks”对话框，单击“Apply”按钮并返回“Summary”对话框界面，单击“OK”按钮后，进入图 3-12 所示的 TinyCore Linux 启动界面，选择第一项并等待启动完成。



图 3-11 “Summary” 对话框

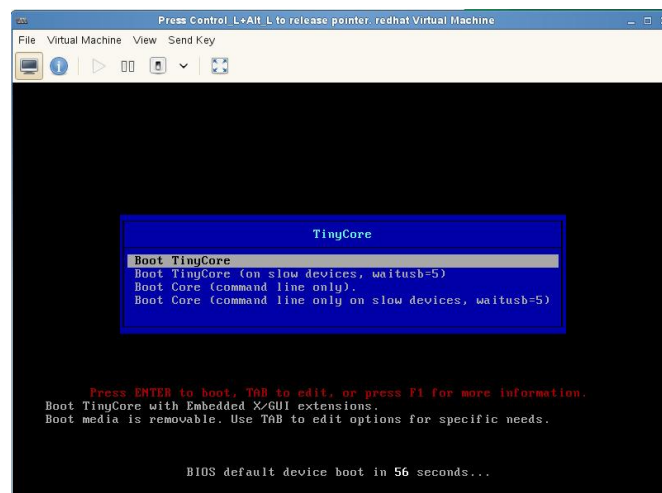


图 3-12 TinyCore Linux 启动界面

步骤 10: 启动成功后，在 TinyCore Linux 桌面单击鼠标左键，在弹出的菜单依次选择 “Applications” → “Terminal”，进入该 TinyCore Linux 的终端模式，如图 3-13 所示。这是，我们可以使用常用的 Linux 命令操作了。至此，一个嵌套的 KVM 虚拟机创建完成。

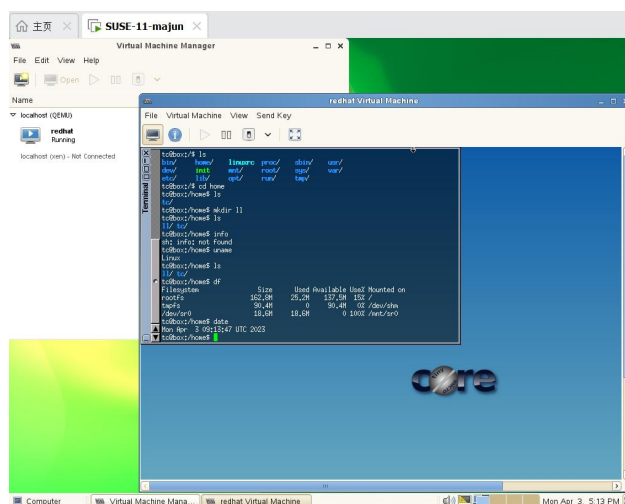


图 3-13 虚拟机启动成功

步骤 11: 我们在 SUSE 虚拟机的终端窗口中输入 `virsh-list` 命令, 在命令窗口中查看目前运行的虚拟机信息, 如图 3-14 所示。读者要明白, 通过 GUI 管理程序完成的工作完全也可以通过命令或 API 来完成。

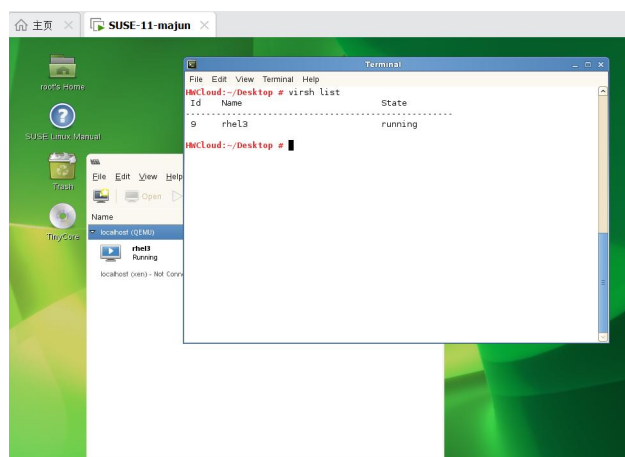


图 3-14 通过命令查看虚拟机信息

通过这个实验我们应该明白, CPU 虚拟化解解决的是程序指令集合的有序执行问题。虚拟化后一个物理 CPU 可以同时执行若干个程序, 这类似于过去分时, 只是高层视角和抽象理论不一样了, 管理方式不一样。现在更多是基于多核和多线程技术, 而内存虚拟化解解决的是虚拟机执行程序时, 如何将按照传统方式去使用物理机内存的问题, 即应用程序看到“连续的虚拟机内存”时, 虚拟机内存如何映射到物理机内存的。

3.2 应用实践 2: 网络虚拟化实践

此实验的目标是学习和理解前面 SDN 理论, 理解虚拟机之间如何通过虚拟的网络设备进行通信, 理解虚拟机如何利用虚拟网络设备和外部物理网络设备通信。同时, 读者还要理解什么是 SDN, 如何通过命令或 API 完成虚拟网络设备的创建和使用。此处我们通过 KVM 创建虚拟机, 通过 OpenSwitch 演示网络虚拟化的各种命令。

由于该实验比较复杂，对计算机要求比较高，建议使用 CPU 为 Intel i7 UPC，或同等水平或更高配的 CPU，内存在 8 GB 以上，硬盘建议为固态硬盘可用空间在 300 GB 以上。由于本实验的目标是学习和理解网络虚拟化原理，因此我们不建议使用最新版本的软件，尤其华为已经宣布不再支持 eNSP 的升级更新了，大家实验中需要对软件版本严格对应，否则效果不理想。

首先，准备需要的软件和工具。大家可以从网络上下载相关软件和工具，也可以从本书配套资源中下载相关软件和工具。本实验所需软件如下。

- eNSP V100R003C00SPC100 Setup.zip，可以模拟外部物理网络。
- netvirtexp.ova，安装了网络虚拟化软件的虚拟电脑。
- VirtualBox-6.0.24-139119-Win.exe，Windows 操作系统下虚拟化软件，虚拟电脑用此软件导入。

- VirtualBox-6.1.26-145957-Win.exe，用此软件消除无法设置 CPU 虚拟指令 bug。
- WinPcap_4_1_3.exe，网络抓包软件。
- Wireshark-win64-3.4.4.exe，网络封包分析软件。
- TinyCore-11.0.iso，微型 RedHat 操作系统，用此软件完成网络实验。

1. 搭建环境（用管理员权限）

- (1) 安装 eNSP 依赖软件 VirtualBox-6.0.24-139119-Win.exe。
- (2) 安装 eNSP 依赖软件 Wireshark-win64-3.4.4.exe。
- (3) 安装 eNSP 依赖软件 WinPcap_4_1_3.exe。
- (4) 重启计算机后解压和安装 eNSP_Setup.exe，默认安装即可，检查启动是否正常。

*启动 eNSP 后，如果出现图 3-15 所示的提示不影响本实验效果。eNSP 我们后面会用到。

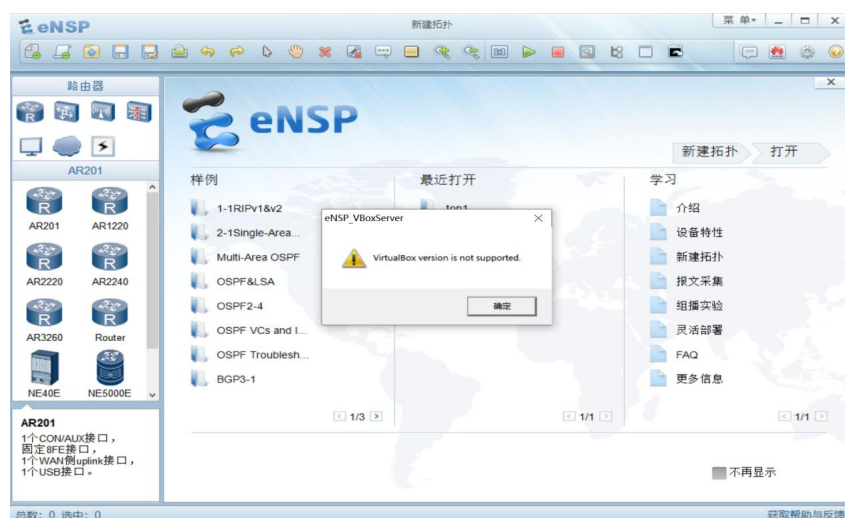


图 3-15 启动 eNSP

- (5) 准备已经配好 kvm-hypervisor 和 openSwitch 运行环境的虚拟计算机，此处安装

netvirtexp.ova，具体步骤如下。

步骤 1：先安装 VirtualBox-6.0.24-139119-Win.exe。

步骤 2：将 netvirtexp.ova 虚拟机导入，如图 3-16 所示。



图 3-16 导入虚拟机 netvirtexp.ova

步骤 3：查看 VM 配置，修改 cpu 数量为 4 个，内存为 4 GB.

步骤 4：查看 VM 设置中的“系统”选项中的“处理器”页面，启动嵌套硬件 CPU 虚拟化选项，如图 3-33 所示。

步骤 5：如果无法选择，则执行步骤 6。

步骤 6：执行以下命令，再次检查，如果还不行，卸载前面的版本，安装另一版本 VirtualBox-6.1.26-145957-Win.exe，然后再执行以下的命令，再检查。

```
cd C:\Program Files\Oracle\VirtualBox
VBoxManage.exe list vms
VBoxManage.exe modifyvm "vm" --nested-hw-virt on
```

步骤 7：再次安装 VirtualBox-6.0.24-139119-Win.exe，检查图 3-17 中的选项是否已经选上。



图 3-17 处理器选项启用嵌套虚拟化

步骤 8: 检查以下各种参数，并根据警告信息修改相关参数，之后启动虚拟机。

步骤 9: 如果启动后出错 `xrandr:cannot find output "Virtual1"`，则卸载 `virtualbox`，重新执行步骤 1~步骤 4。

2. 实验 1: 使用默认的虚拟网卡和网桥和外部通信

打开 `virtualBox` 选择前面导入的虚拟机，检查配置如图 3-18。在网卡 1 的配置项选择“桥接网卡”（即桥接宿主机物理网卡），并将“高级”选项中的混杂模式设为“全部允许”，如图 3-19 所示。



图 3-18 检查虚拟机配置



如图 3-19 网络配置

启动虚拟电脑。这里的启动比较慢，请耐心等待启动完成，之后用以下用户登录（建议用 root 用户登录），进入 Terminal 窗口，执行/xxx.sh 脚本，如图 3-20 所示。

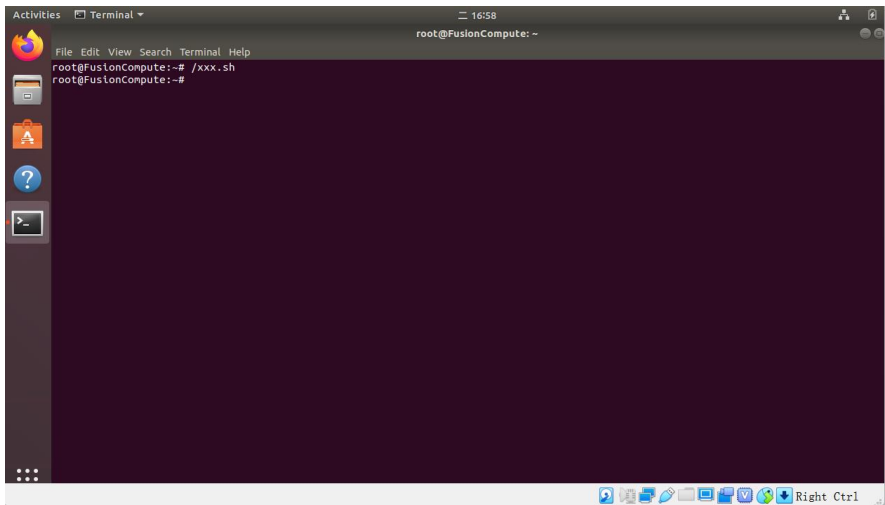


图 3-20 登录成功（root 用户）

```
huawei:Admin@1234
root:Admin@1234
```

③ 输入命令 ifconfig 查看虚拟机网络环境，执行结果如图 3-21 所示。

```
ifconfig
virt-manager
```

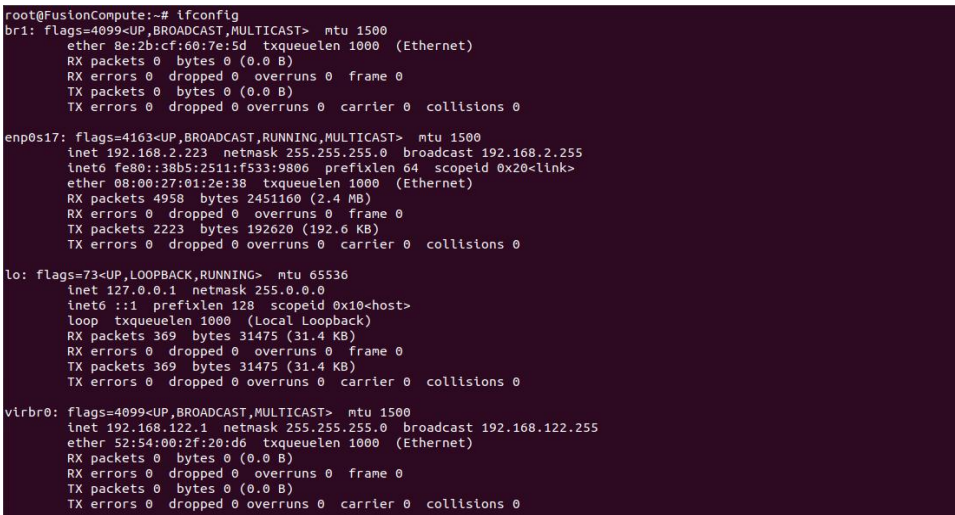


图 3-21 ifconfig 的执行结果

图 3-37 所示结果中的参数含义如下。

Lo: 本地回环接口

enp0s17: Ubuntu 操作系统识别到的物理网卡，已拿到物理网卡同一网段 DHCP 分

配的 IP 地址。

virbr0 和 br1：虚拟机中已经配置好的虚拟网桥，其中，virbr0 是给 KVM 虚拟机准备的网桥。

④ 和前面计算虚拟化的实验相同，使用 virt-manager 创建虚拟机 testvm，界面如图 3-22 所示。

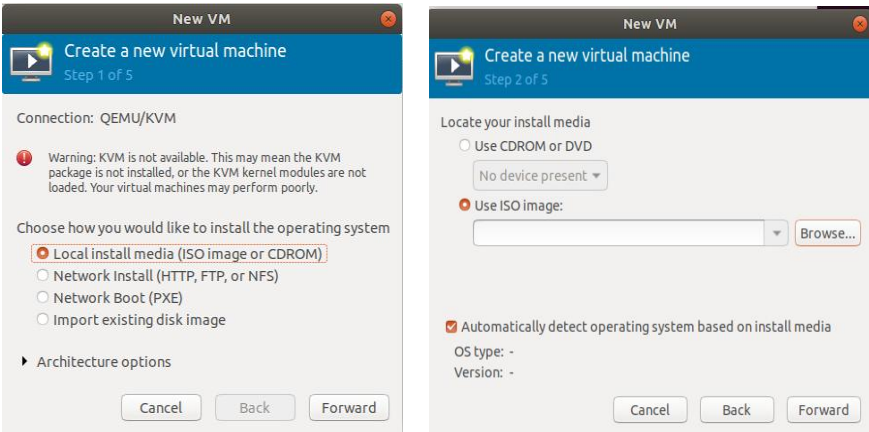


图 3-22 创建虚拟机

⑤ 安装媒介选择桌面上的 Tiny-Core-current.iso，内存设置为 128 MB，CPU 数 1 个，硬盘容量为 2 GB，虚拟机名称设为“testvm”，如图 3-23 和如图 3-24 所示。启动该嵌套的虚拟机，从启动菜单中选择第 3 项“Boot Core (command only)”，等待启动成功，如图 3-25 所示。



图 3-23 安装媒介选择桌面上的 TinyCore-current.iso

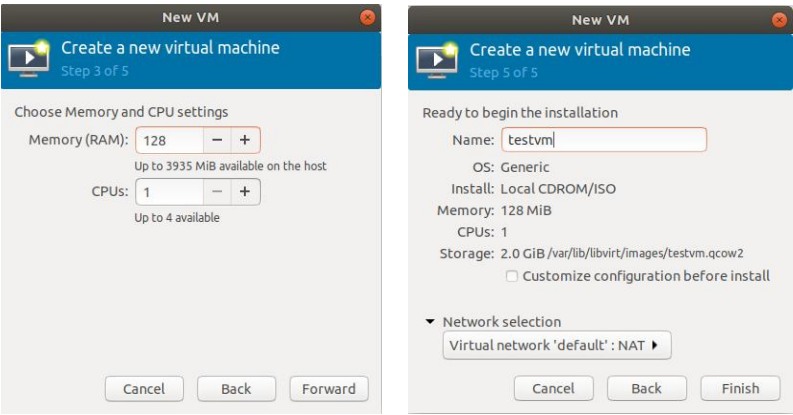


图 3-24 设置虚拟机参数

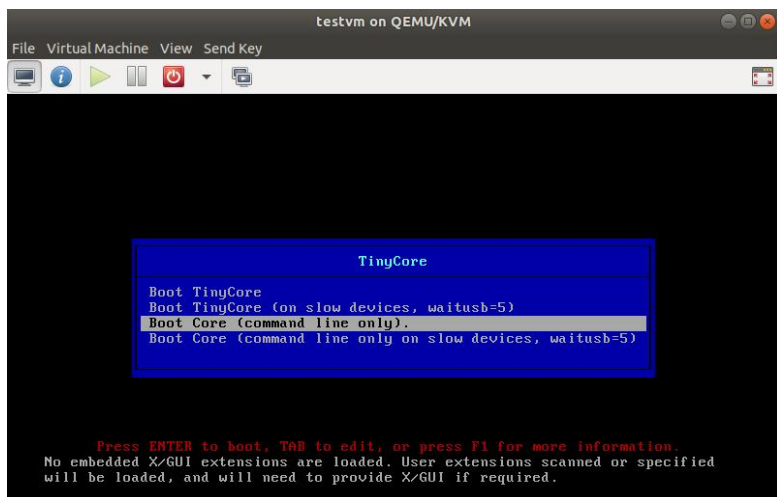


图 3-25 启动 TiinyCore Linux

⑥ 在嵌套虚拟机中输入“ifconfig”命令，查看 testvm 虚拟机的网络信息。可以发现，发现它有一个虚拟网卡 eth0，已经获得宿主机给它分配的一个 IP 地址（这里不展示具体页面）。

⑦ 使用以下命令测试网络，运行结果如图 3-26 所示。发现虚拟机可以 ping 通人民邮电出版和官网，也可以从互联网（信通社区上）下载文件。虚拟机通过虚拟网卡和外界通信如图 3-26 所示。这说明实验创建的虚拟机通过虚拟网卡可以访问外部物理网络，请思考一下它的通信原理。

```
ping www.ptpress.com.cn
```

```
https://book.cww.net.cn/book/details?id = 52497
```

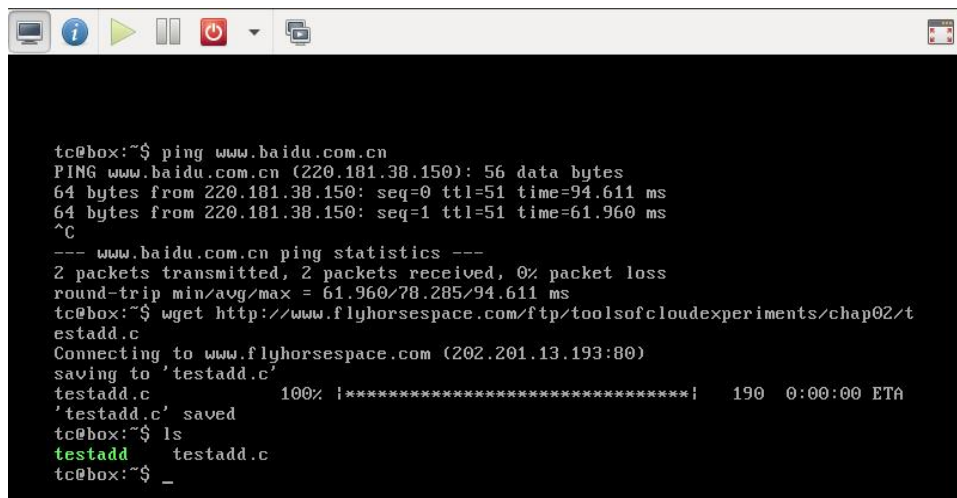


图 3-26 虚拟机通过虚拟网卡和外界通信

到此，我们将嵌套的虚拟机通过它的虚拟网卡，桥接到宿主机（该宿主机也是虚拟的）的虚拟网桥 virbr0，再通过此虚拟宿主机的虚拟网桥桥接到物理宿主机的物理网卡，通过该物理无线网卡完成数据进出。实验 1 结束。

下面通过实验 2，我们进一步完成网络虚拟化探索。

3. 实验 2：子网设计和 VLAN 划分

本实验中我们创建 4 台虚拟机 vm01~vm04，创建 2 台虚拟交换机，测试虚拟机的连通性，之后将这 4 台虚拟机划分到 2 个 VLAN 中，测试子网隔离。本实验的网络拓扑如图 3-27 所示，注意，读者在自己的计算机上做这个实验时，MAC 地址肯定是不同的。

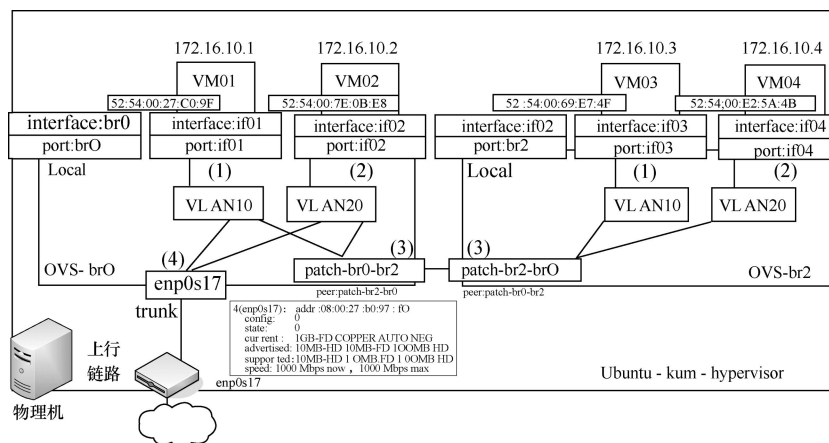


图 3-27 实验 2 的网络拓扑

① 使用以下命令添加两台虚拟交换机 br0 和 br2，创建相应的虚拟端口和虚拟网卡，运行结果如图 3-44 所示。

```
ovs-vsctl add-br br0
ovs-vsctl add-port br0 if01 -- set interface if01 type=internal
ovs-vsctl set port if01 VLAN_mode=access
ovs-vsctl add-port br0 if02 -- set interface if02 type=internal

ovs-vsctl set port if02 VLAN_mode=access
ovs-vsctl add-br br2
ovs-vsctl add-port br2 if03 -- set interface if03 type=internal
ovs-vsctl set port if03 VLAN_mode=access
ovs-vsctl add-port br2 if04 -- set interface if04 type=internal
ovs-vsctl set port if04 VLAN_mode=access
ovs-vsctl show
```

上述结果中各参数含义如下。

Bridge: 网桥名称，即虚拟交换机名。

port: 交换机上的端口名。

interface: 对应于某端口的虚拟网卡名。

type: 接口类型。

internal: 默认 access 只允许一个 wlan 数据通过 vlan0, [vlan0 表示不打标记]。在虚

拟网络中，vlan 默认 vlan0,vlan0 表示不打标记，vlan1 打标记 tag。在物理网络中，vlan 默认是 vlan1。

patch: 默认 trunk，允许多个 vlan 数据通过。虽然图 3-28 中未出现此参数，但它也是一个读者必须掌握的内容。

```
root@FusionCompute:~# ovs-vsctl show
e436b05b-210c-468a-9d94-fd63c6c419ed
  Bridge "br2"
    Port "br2"
      Interface "br2"
        type: internal
    Port "if04"
      Interface "if04"
        type: internal
    Port "if03"
      Interface "if03"
        type: internal
  Bridge "br0"
    Port "if01"
      Interface "if01"
        type: internal
    Port "br0"
      Interface "br0"
        type: internal
    Port "if02"
      Interface "if02"
        type: internal
  ovs_version: "2.9.8"
root@FusionCompute:~#
```

图 3-28 创建虚拟交换机、虚拟端口和虚拟网卡

② 使用以下命令显示 ovs 上添加的端口信息和对应的网卡信息，运行结果如图 3-45 所示。

```
ovs-vsctl list-ports br0
ovs-vsctl list-ports br2
ip add show if01
ip add show if02
ip add show if03
ip add show if04
```

```
root@FusionCompute:~# ovs-vsctl list-ports br0
if01
if02
root@FusionCompute:~# ovs-vsctl list-ports br2
if03
if04
root@FusionCompute:~# ip add show if01
9: if01: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether da:0d:9c:c2:94:67 brd ff:ff:ff:ff:ff:ff
root@FusionCompute:~# ip add show if02
10: if02: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether e6:78:d3:af:61:54 brd ff:ff:ff:ff:ff:ff
root@FusionCompute:~# ip add show if03
11: if03: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 7a:bf:5d:dc:96:cb brd ff:ff:ff:ff:ff:ff
root@FusionCompute:~# ip add show if04
12: if04: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether c6:95:01:a8:86:23 brd ff:ff:ff:ff:ff:ff
root@FusionCompute:~#
```

图 3-29 虚拟交换机上的端口和对应的网卡信息

③ 创建虚拟机 vm01~vm04，并将它们绑定到对应的虚拟网卡上。创建过程同实验 1，这里不再讲。选择网卡时请选择对应的网卡，例如 vm01 选择 if01，模式(Source mode)选择“Passthrough”，如图 3-30 所示。

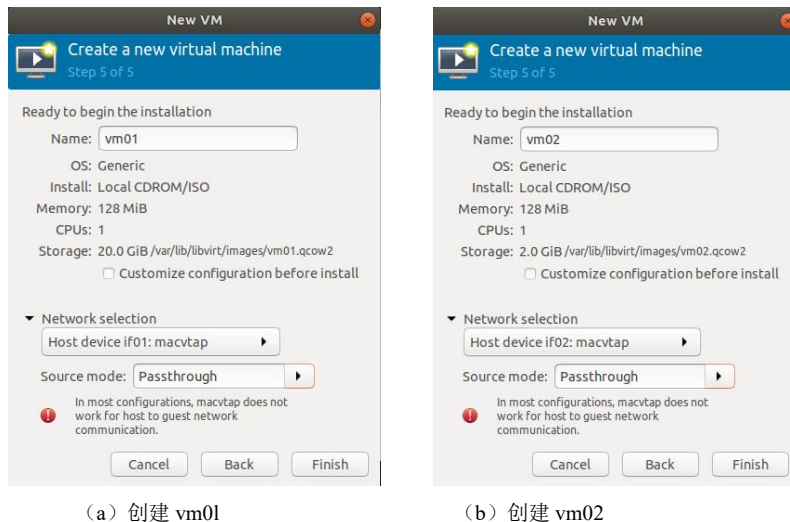


图 3-30 创建虚拟机 vm01~vm04

④ 启动虚拟机并选择“command only”，然后用以下命令给虚拟机配置 IP 地址，其中，vm01~vm04 的 IP 地址分别配置为 172.16.10.1~172.16.10.4。以虚拟机 vm01 为例，它的配置结果如图 3-31 所示，其他参考此结果。

```
sudo ifconfig eth0 172.16.10.1 netmask 255.255.255.0
sudo ifconfig eth0
```

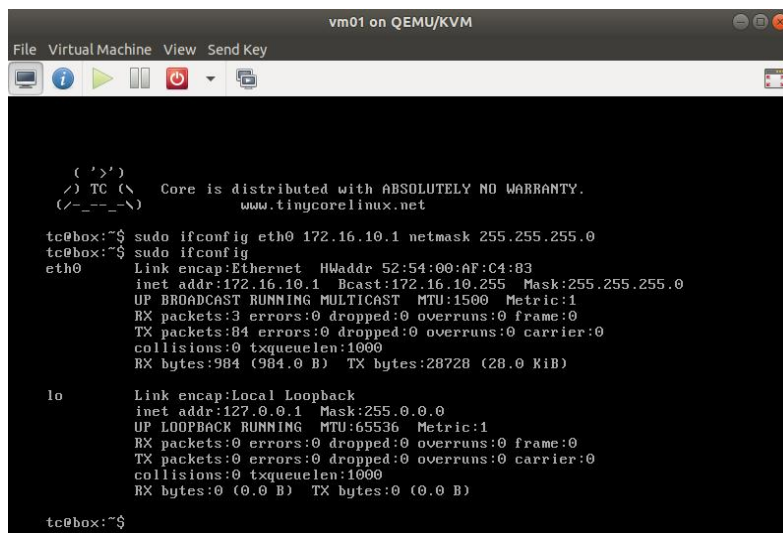


图 3-31 vm01 的配置结果

⑤ 测试连通性，在没有划分 VLAN，并且虚拟机交换机 br0 和 br2 没有互连的情况下，vm04 和 vm01 是无法通信的，但 vm04 和 vm03 可以 ping 通，读者可以自行测试。

⑥ 使用以下命令查看宿主机 ovs br0 和 br2 的连接情况（类似看物理交换机的端口插线），运行结果如图 3-32 所示。

```
ovs-ofctl show br0
Ovs-ofctl show br2
```

```

root@FusionCompute:~# ovs-ofctl show br0
DPPT_FEATURES_REPLY (xid=0x2): dpid:0000825896fd9e41
n_tables:254, n_buffers:0
capabilities: FLOW_STATS TABLE_STATS PORT_STATS QUEUE_STATS ARP_MATCH_IP
actions: output enqueue set_vlan_vid set_vlan_pcp strip_vlan mod_dl_src mod_dl_dst mod_nw_src mod_nw_dst mod_nw_tos mod_tp_src mod_tp_dst
1(tf01): addr:52:54:00:af:c4:83
config:
state:
speed: 0 Mbps now, 0 Mbps max
2(tf02): addr:52:54:00:c8:d1:52
config:
state:
speed: 0 Mbps now, 0 Mbps max
LOCAL(br0): addr:82:58:96:fd:9e:41
config:
PORT_DOWN
state:
LINK_DOWN
speed: 0 Mbps now, 0 Mbps max
DPPT_GET_CONFIG_REPLY (xid=0x4): frags=normal miss_send_len=0
root@FusionCompute:~# ovs-ofctl show br2
DPPT_FEATURES_REPLY (xid=0x2): dpid:00003aade761354d
n_tables:254, n_buffers:0
capabilities: FLOW_STATS TABLE_STATS PORT_STATS QUEUE_STATS ARP_MATCH_IP
actions: output enqueue set_vlan_vid set_vlan_pcp strip_vlan mod_dl_src mod_dl_dst mod_nw_src mod_nw_dst mod_nw_tos mod_tp_src mod_tp_dst
1(tf03): addr:52:54:00:df:ec:2b
config:
state:
speed: 0 Mbps now, 0 Mbps max
2(tf04): addr:52:54:00:11:1b:36
config:
state:
speed: 0 Mbps now, 0 Mbps max
LOCAL(br2): addr:3a:ad:e7:61:35:4d
config:
PORT_DOWN

```

图 3-32 br0 和 br2 的连接情况

可以看到, br0 和 br2 上分别连接了 2 个虚拟网卡, 以及每个虚拟网卡的 MAC 地址。对比一下就会发现, 在宿主机中的 if01 网卡的 MAC 地址和虚拟机 vm01 中网卡 eth0 的 MAC 地址是相同的, 这说明虚拟机 vm01 识别出的网卡就是宿主机中我们创建的虚拟网卡 if01, 该网卡有连接到虚拟交换机的 if01 端口上。

⑦ 连接两台虚拟交换机, 并划分 VLAN, 然后测试各虚拟机的连通性。

两台虚拟交换机之间进行流量交互需要连接 patch 类型接口, patch 接口默认允许所有 VLAN 数据通过, 而 patch 接口互联需要设置 peer (对端)。我们参考图 3-27 所示网络拓扑, 先完成 VLAN 划分和交换机连接后, 再使用以下命令测试连通性。得到的结果如图 3-33 所示, 和预期结果一致, VLAN 隔离成功, 两台虚拟交换机可以互通。

```

ovs-vsctl add port if01 tag 10
ovs-vsctl add port if02 tag 20
ovs-vsctl add port if03 tag 10
ovs-vsctl add port if04 tag 20
ovs-vsctl add-port br0 patch-br0-br2 -- set interface patch-br0-br2 type=patch
options:peer=patch-br2-br0
ovs-vsctl add-port br2 patch-br2-br0 -- set interface patch-br2-br0 type=patch
options:peer=patch-br0-br2

```

```

vm01 on QEMU/KVM
File Virtual Machine View Send Key
tc@box:~$ ping 172.16.10.2
PING 172.16.10.2 (172.16.10.2): 56 data bytes
^C
--- 172.16.10.2 ping statistics ---
2 packets transmitted, 0 packets received, 100% packet loss
tc@box:~$ ping 172.16.10.3
PING 172.16.10.3 (172.16.10.3): 56 data bytes
64 bytes from 172.16.10.3: seq=0 ttl=64 time=0.073 ms
64 bytes from 172.16.10.3: seq=1 ttl=64 time=3.926 ms
^C
--- 172.16.10.3 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 3.926/5.993/0.073 ms
tc@box:~$ ping 172.16.10.4
PING 172.16.10.4 (172.16.10.4): 56 data bytes
^C
--- 172.16.10.4 ping statistics ---
0 packets transmitted, 0 packets received, 100% packet loss
tc@box:~$

```

(a) vm01的ping结果

```

vm02 on QEMU/KVM
File Virtual Machine View Send Key
tc@box:~$ ping 172.16.10.1
PING 172.16.10.1 (172.16.10.1): 56 data bytes
^C
--- 172.16.10.1 ping statistics ---
2 packets transmitted, 0 packets received, 100% packet loss
tc@box:~$ ping 172.16.10.3
PING 172.16.10.3 (172.16.10.3): 56 data bytes
^C
--- 172.16.10.3 ping statistics ---
4 packets transmitted, 0 packets received, 100% packet loss
tc@box:~$ ping 172.16.10.4
PING 172.16.10.4 (172.16.10.4): 56 data bytes
64 bytes from 172.16.10.4: seq=0 ttl=64 time=7.813 ms
64 bytes from 172.16.10.4: seq=1 ttl=64 time=2.156 ms
64 bytes from 172.16.10.4: seq=2 ttl=64 time=1.237 ms
^C
--- 172.16.10.4 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 1.237/3.735/7.813 ms
tc@box:~$

```

(b) vm02的ping结果

图 3-33 虚拟交换机连接后互通性测试

⑧ 使用以下命令查看虚拟机交换机的连接情况,以及 4 台虚拟机的 MAC 地址表和各端口连接的 VLAN 信息。这里展示虚拟交换机各端口连接信息,如图 3-34 所示。

```
ovs-vsctl show      #两个虚拟交换机各端口信息
```

```
ovs-appctl fdb/show br0  #虚拟交换机各端口对应的 vlan 信息
```

```
root@FusionCompute:~# ovs-appctl fdb/show br0
port  VLAN  MAC  Age
2      20    52:54:00:c8:d1:52  18
3      20    52:54:00:11:1b:36   7
1      10    52:54:00:af:c4:83   6
3      10    52:54:00:df:ec:2b   3
root@FusionCompute:~# ovs-appctl fdb/show br2
port  VLAN  MAC  Age
3      20    52:54:00:c8:d1:52  21
2      20    52:54:00:11:1b:36  10
3      10    52:54:00:af:c4:83   9
1      10    52:54:00:df:ec:2b   2
root@FusionCompute:~#
```

图 3-34 虚拟交换机各端口连接信息

在图 3-50 中,我们应该注意到 port 3 对应的就是两个交换机互连的端口,可以同时允许 VLAN 10 和 VLAN 20 的数据通过。至此,实验 2 结束。

4. 实验 3: 连接外部模拟网络

在本实验中,我们需要使用华为的 eNSP 模拟出一个外部网络,然后通过桥接技术,将前面的虚拟内部网络连接到外部网络,实践虚拟机和外部网络的通信技术。

①打开 eNSP 软件,并创建图 3-35 所示网络拓扑(网络配置命令这里不讲,读者自己学习),配置对应的 IP 地址,并启动交换机、PC1、PC2。

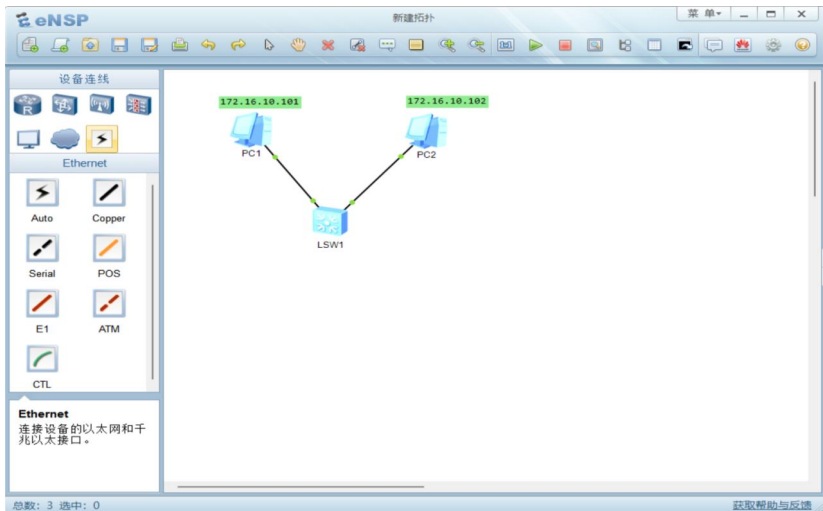


图 3-35 实验 3 网络拓扑

② 使用以下命令显示交换机 LSW1 的 VLAN 信息,并测试 PC1 和 PC2 的连通性。运行结果如图 3-36 和图 3-37 所示。

```
disp vlan
ping 172.16.10.102
```

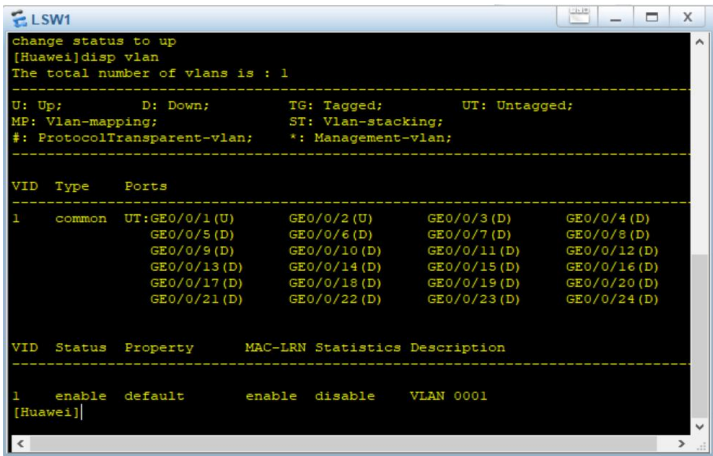


图 4-36 LSW1 的 VLAN 信息

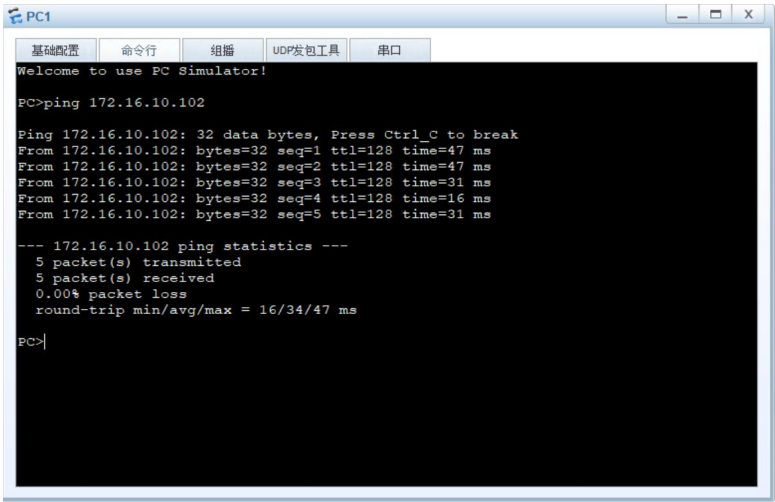


图 3-37 PC1 和 PC2 连通结果

③ 使用以下命令在交换机上查看物理网络 MAC 地址和 VLAN 信息，运行结果如图 4-38 所示。

dis mac-address

```

[Huawei]disp mac-address
MAC address table of slot 0:
-----
MAC Address    VLAN/  PEVLAN CEVLAN Port          Type    LSP/LSR-ID
                VSI/SI
-----
5489-98d6-38fa 1          -    -    GE0/0/1    dynamic  0/-
5489-98f0-124c 1          -    -    GE0/0/2    dynamic  0/-
-----
Total matching items on slot 0 displayed = 2

```

图 3-38 交换机上物理网络的 MAC 地址和 VLAN 信息

默认情况下，物理网络的 VLAN 为 vlan1。可以看出，交换机的端口 1 和端口 2 分别连接了 PC1 和 PC2。

④ 使用以下命令划分 VLAN，其中，PC1 属于 vlan10，PC2 属于 vlan20。运行结果如图 3-39 所示。


```

vlan batch 10 20
interface g0/0/1
port link-type access
port default vlan 10
q
int g0/0/2
p l a
p d v 20
disp vlan

```

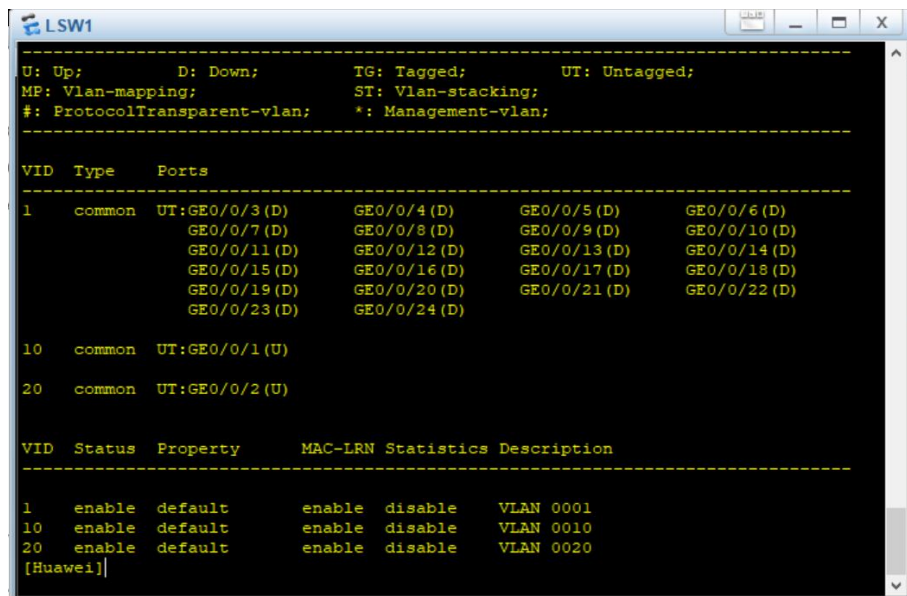


图 3-39 划分 VLAN

⑤ 在 PC1 的命令行窗口中使用以下命令 ping PC2, 测试 PC1 和 PC2 是否隔离成功。运行如图 4-40 所示。

```
ping 172.16.10.102
```

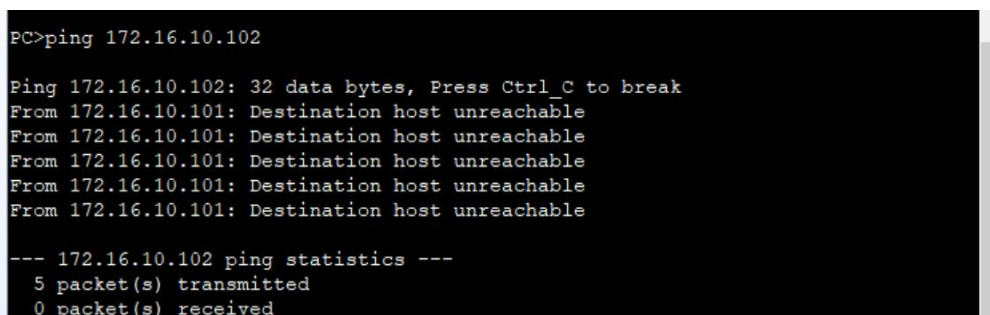


图 4-40 PC1 和 PC2 隔离成功

⑥在 virtualBox 依次选择“管理菜单”→“主机网络管理”，增加一块虚拟网卡

VirtualBox Host-Only Ethernet Adapter #2，如图 3-41 所示。



图 3-41 给 VirtualBox 增加一块虚拟网卡

⑦ 在虚拟机 vm 的设置中，将网卡 1 的连接方式设置为“仅主机（Host-Only）网络”，绑定 VirtualBox Host-Only Ethernet Adapter #2 虚拟网卡，并将“混杂模式”设置为“全部允许”，并选择接入网线复选框，如图 3-42 所示。



图 3-42 设置 vm 的网卡 1

⑧ 在 eNSP 软件中，增加一个桥接工具，并设置桥接工具绑定 VirtualBox Host-Only Ethernet Adapter #2 虚拟网卡，如图 3-43 所示。注意，这里要重新安装 WinPcap_4_1_3 软件。

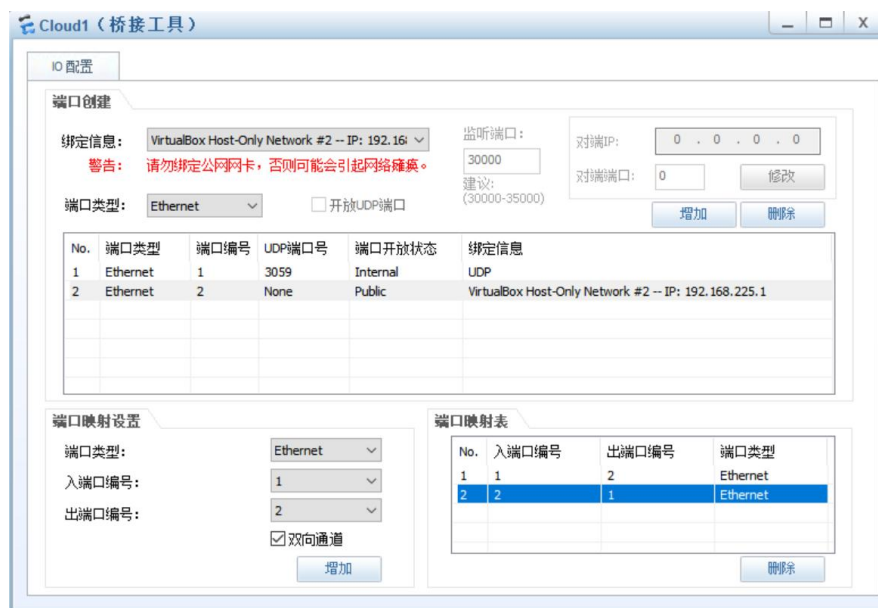


图 3-43 增加桥接工具

⑨ 使用以下命令记得要在模拟的物理网络中配置 LSW1 的 24 口为 trunk 模式，否则无法通信。

```
sys
int g0/0/24
p lt
p t a v 10 20
```

⑩ 测试虚拟机 vm01~vm04 中虚拟网络和模拟物理网络的连通性，运行结果如图 3-44 和图 3-45 所示。

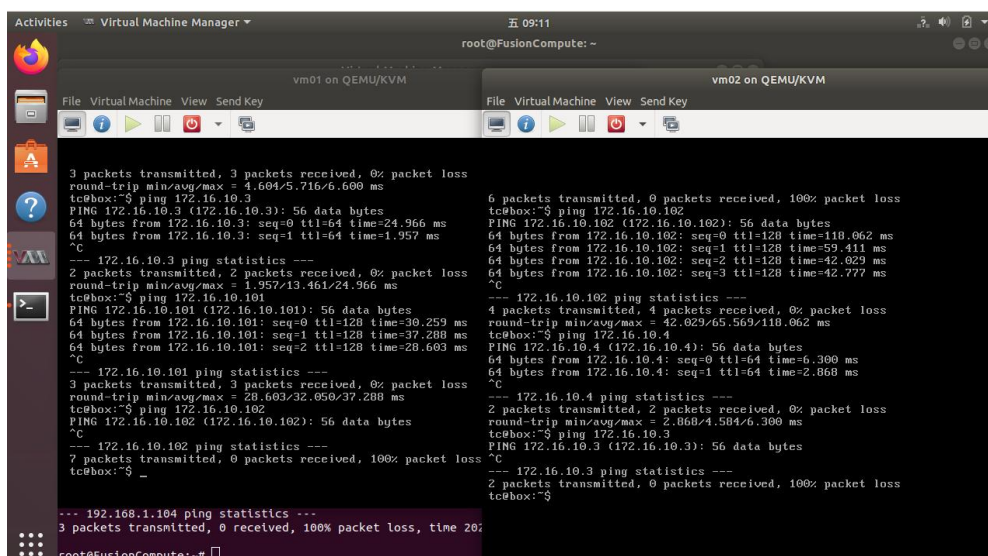


图 3-44 虚拟网络访问物理网络

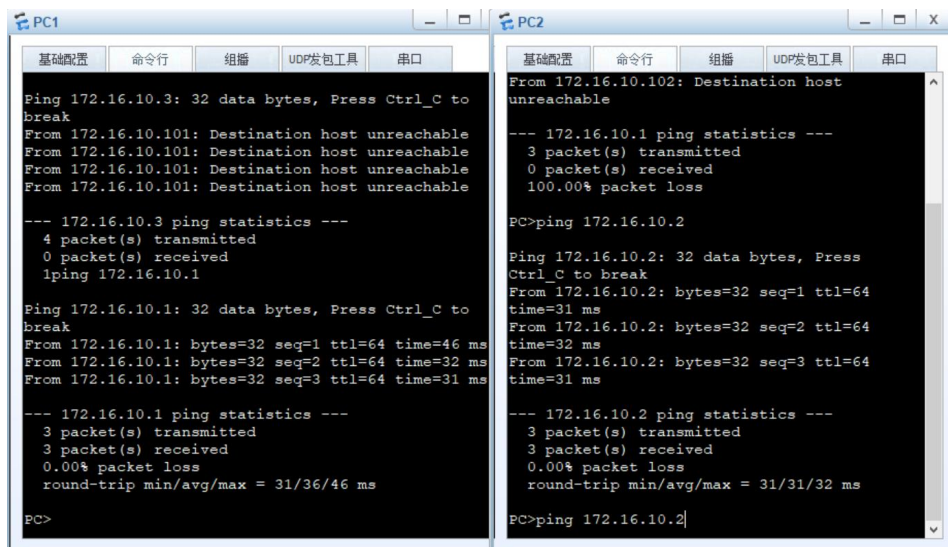


图 3-45 物理网络访问虚拟网络

⑪ 本实验最后的网络拓扑结构如图 3-46 所示。

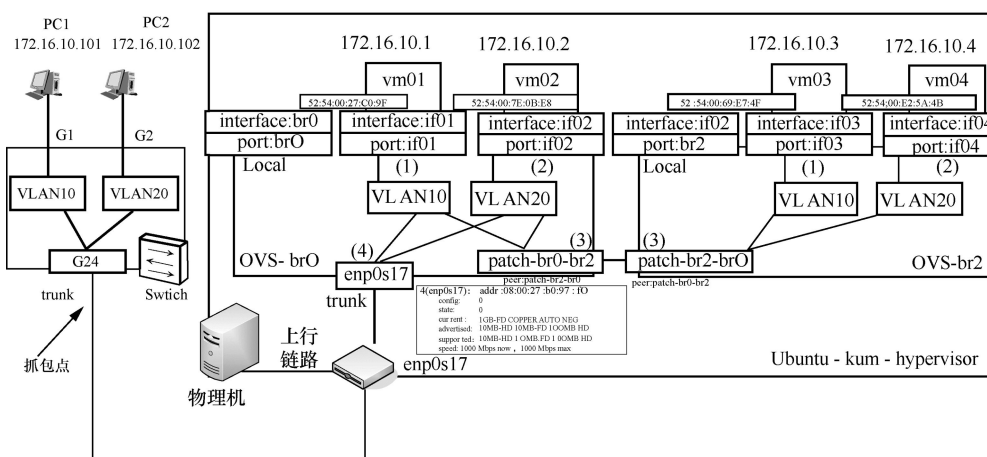


图 3-46 实验 3 最终的网路拓扑

至此，实验 3 结束。感兴趣的读者可以在 eNSP 的交换机上添加三层路由转发，针对不同 VLAN 设置 IP 地址，给虚拟机和 eNSP 中计算机添加默认网关，并测试全网连通性。

通过本实验我们应该明白，通过桥接方式就可以划分虚拟子网，并且可以和外部网络通信。实际上这也是各大公有云底层网络的通信原理。

3.3 应用实践 3：存储虚拟化实验（以 NAS 为例）

FreeNAS 是一种基于 FreeBSD 的嵌入式开源的、基于网络的访问存储（NAS）操作系统，遵循开源 BSD 许可证，其常用的网络结构如图 3-47 所示。NAS 和基于块存储的网络存储系统 SAN 不一样，它是一个针对文件存储和共享存储进行优化的网络存储系统，FreeNAS 提供基于浏览器的图形配置界面和内置的网络协议，可提供对多个操作系统的网络存储访服务。FreeNAS 还提供了一个插件系统，可通过安装其他软件来

扩展内置功能。

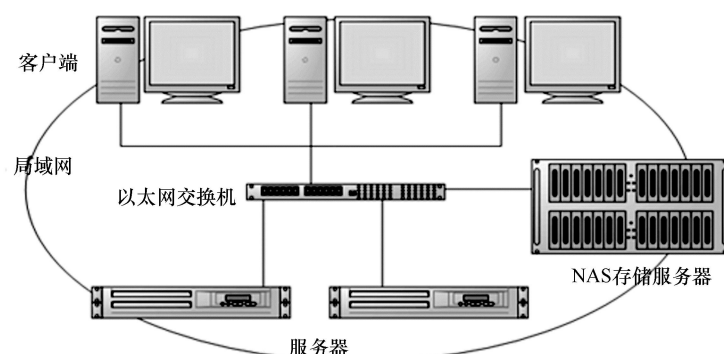


图 3-47 FreeNAS 常用的网络结构

本实验中我们通过嵌套虚拟化来完成模拟物理环境中业务网络和存储网络，这样就可以在个人计算机完成实验内容和知识点的学习。首先，在宿主机上安装 VMware 上虚拟化软件，然后安装 3 台虚拟机：一台虚拟机安装 FreeNAS 系统，用于模拟存储网络中一个提供 NAS 存储系统的物理服务器；另外两台虚拟机模拟业务网络中的应用服务器，其中一台是 Windows 服务器和一台是 Linux 服务器。之后按照实验流程完成让 NAS 服务器为 Windows 服务器和 Linux 服务器提供网络存储服务的实验，拓扑结构如图 3-48 所示。

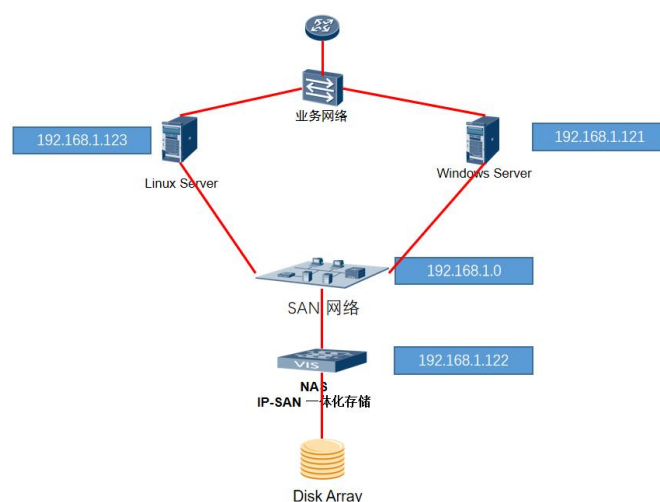


图 3-48 存储实验拓扑图

一、准备软硬件环境

1. 准备一台内存在 8G 以上、双核 4 线程以上台式机或笔记本电脑
2. FreeNAS 软件 FreeNAS-11.0-RELEASE.iso
3. Windows7 软件 cn_windows_7_ultimate_with_sp1_x64_dvd_u_677408.iso
4. VMwareWorkstation 软件 VMwareWorkstation16.1.2.zip
5. CentOS 软件 CentOS-7-x86_64-DVD-1708.iso
6. 网络拓扑结构如图 3-66 所示：

二、安装 FreeNAS 系统

- 1. 安装 VMWareStation 软件（略）
- 2. 在 VMWareStation 安装 FreeNAS 虚拟机, 选择 FreeNAS-11.0-RELEASE.iso 镜像, 如图 3-49、图 3-50 所示。

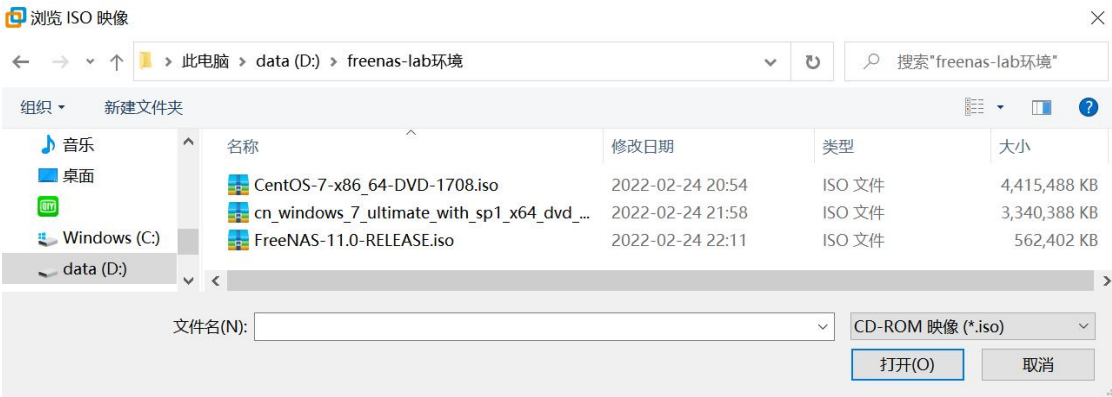


图 3-49 选择 iso 镜像

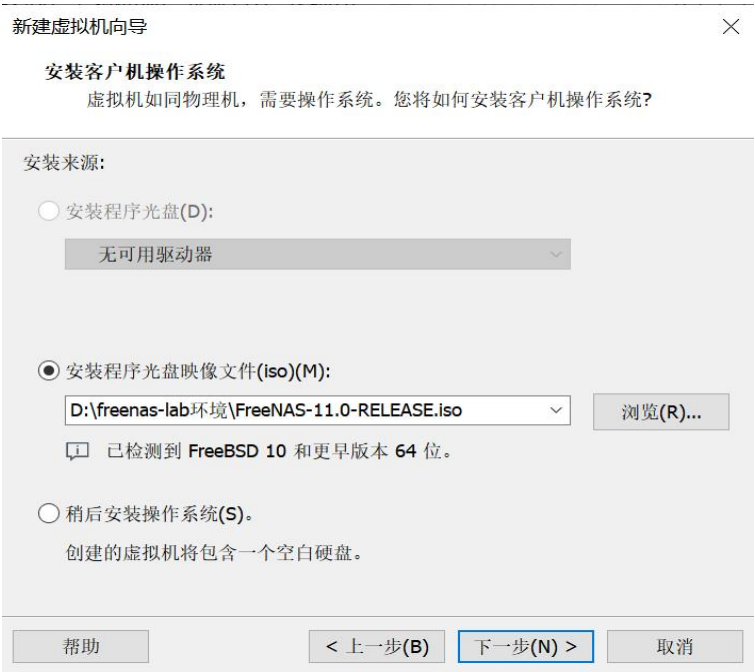


图 3-50 安装 freeNAS 镜像

- 3. 修改虚拟机名称, 选择存储位置, 如图 3-51 所示。

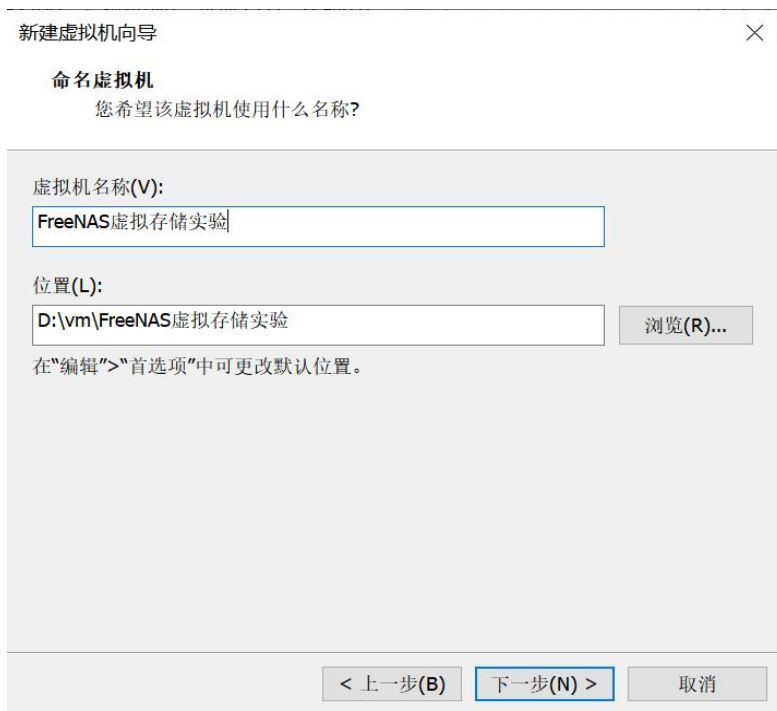


图 3-51 修改虚拟机名称

4. 内存选择 1G 或 2G，cpu 选择 1-2 个即可，网络选择桥接，如图 3-52 所示
5. 接下来两步默认，到选择磁盘时，选择“创建新虚拟磁盘”，如图 3-53 所示
6. 磁盘容量为 20G，选择“存储为单个文件”，如图 3-54 所示
7. 下一步选择合适的目录存储磁盘文件，然后选择“完成”，虚拟机的配置如图 3-55 所示：

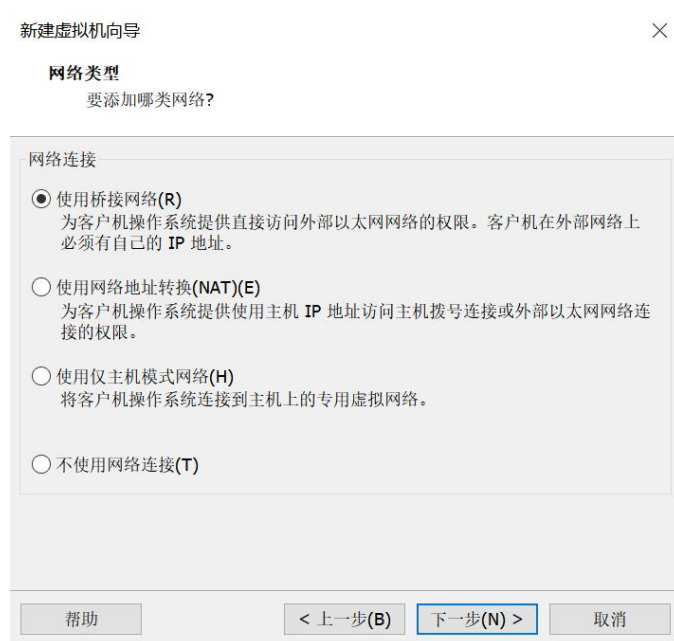


图 3-52 选择桥接网络

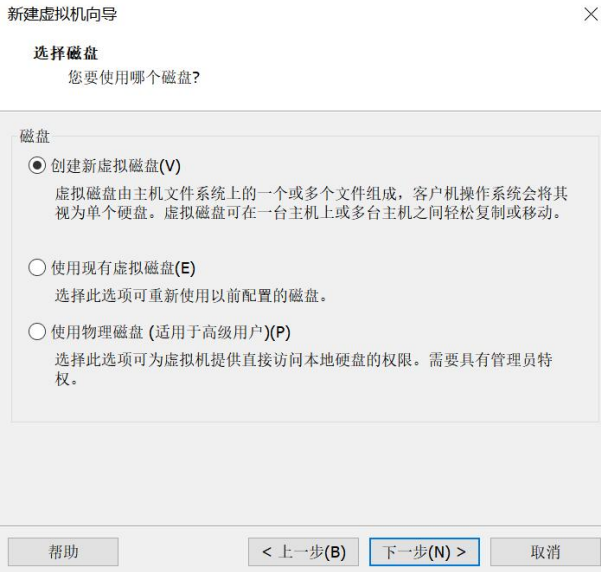


图 3-53 创建新虚拟磁盘

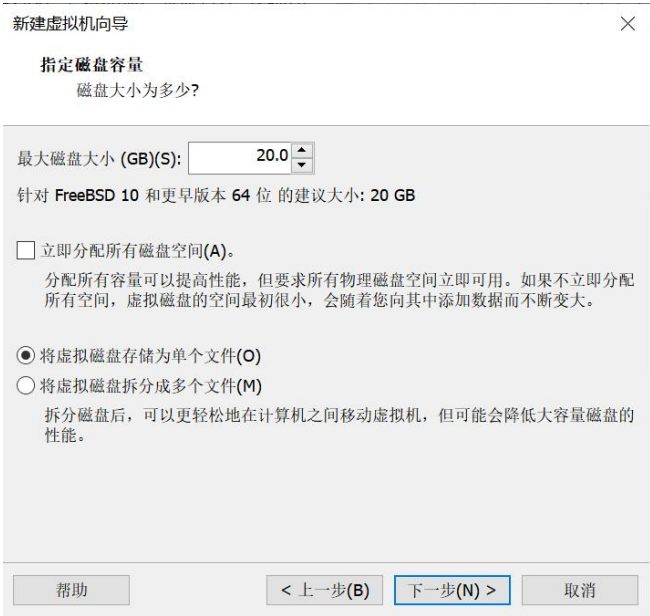


图 3-54 虚拟磁盘存储为单个文件



图 3-55 freeNAS 虚拟机配置

8. 单击“开启此虚拟机”，开始安装系统，如图 3-56。

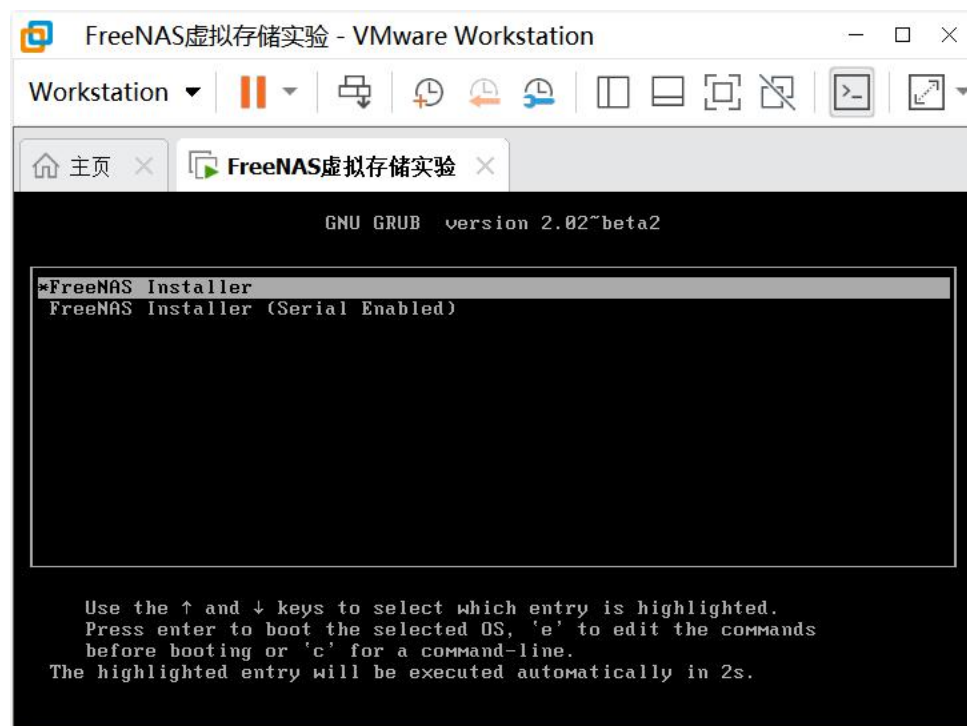


图 3-56 安装 freeNAS 系统

9. 用 tab 键选择 Yes，用空格键选择 da0 磁盘，开始安装，如图 3-57 所示。

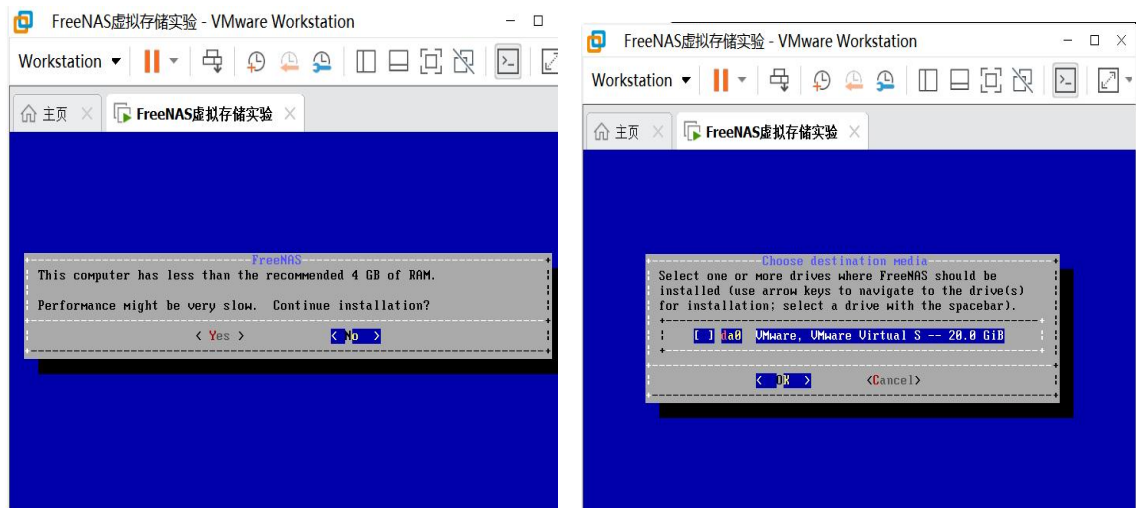


图 3-57 选择安装系统的磁盘

10. 输入 root 用户密码，记住密码，此处我们输入 Huawei@1234，然后选择 bios 启动，如图 3-58 所示。

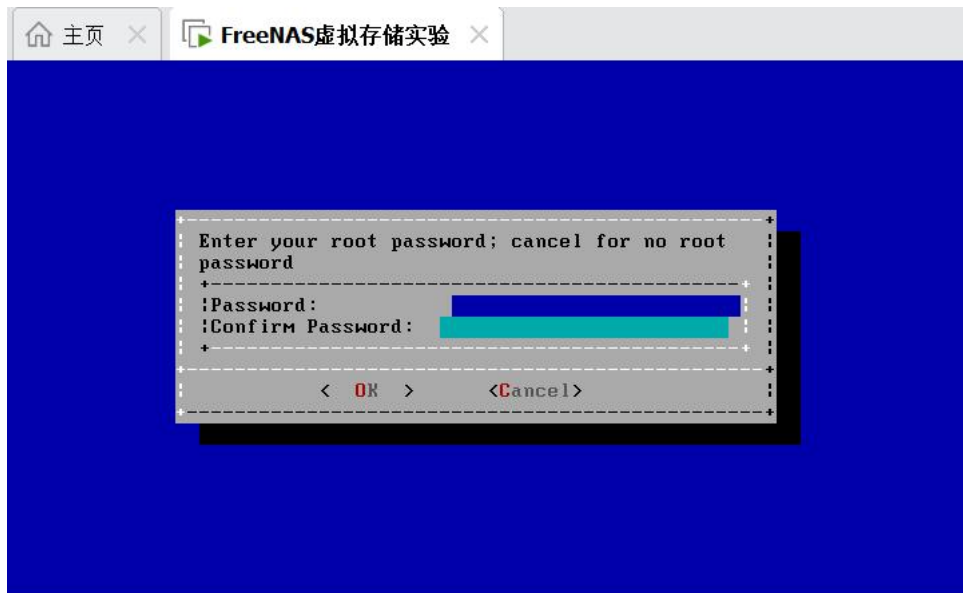


图 3-58 设置 root 密码

11. 选择 reboot, 然后 normal 启动, 启动完成后如图 3-59 所示。
12. 选择 9, 然后 ping 一下宿主机, 看是否连通, 如图 3-60 所示。
13. 在宿主机上打开浏览器, 输入图 13 中的 url, 输入 root 用户和密码 Huawei@1234, 登录后如图 3-61 所示。

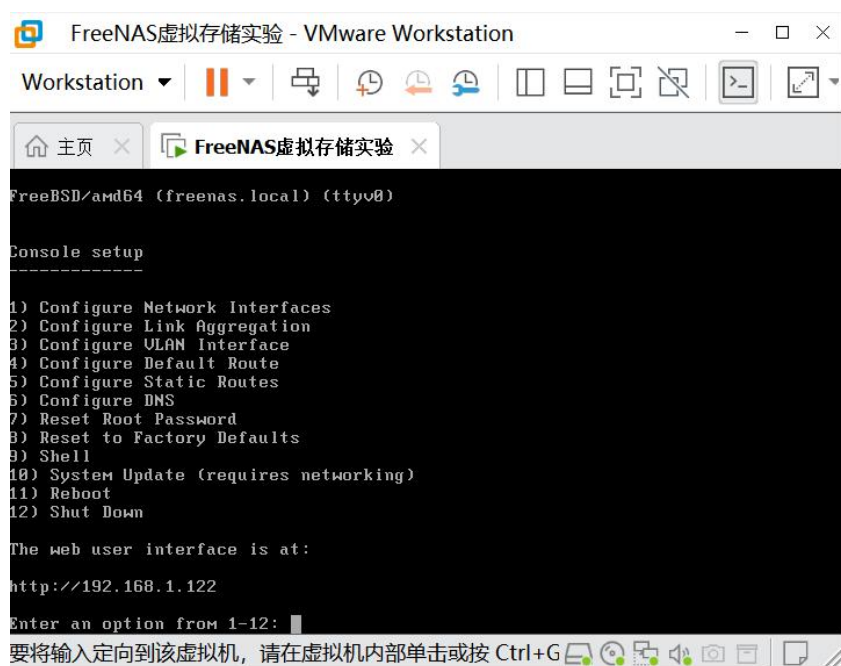


图 3-59 重新启动成功

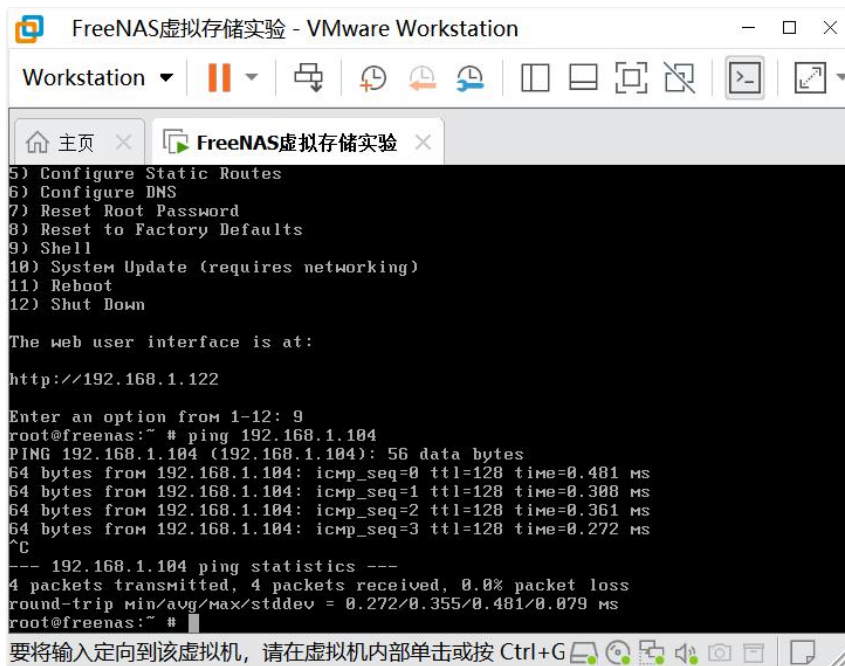


图 3-60 测试和宿主机的连通性

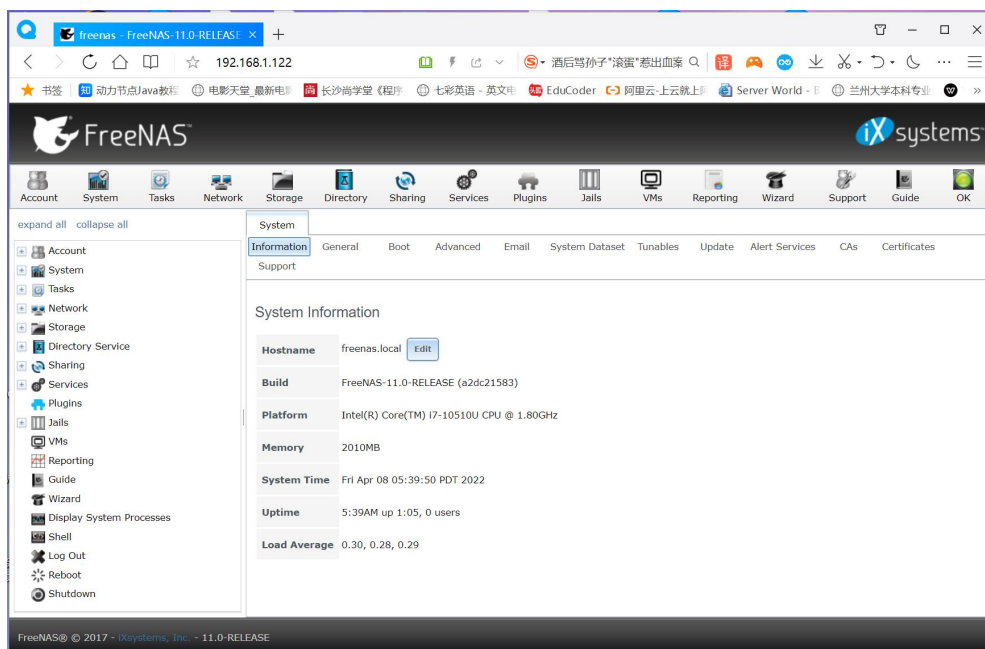


图 3-61 FreeNAS 安装成功

三、安装 Windows7 系统虚拟机（模拟业务服务器一）

- 1.在 VMwareStation 安装 windows7 虚拟机，配置如图 3-62 所示，安装过程参考 freeNAS，此处略。
- 2.完成安装后，开启虚拟机，启动后如图 3-63 所示。
- 3.打开控制面板，关闭防火墙，如图 3-64 所示。
- 4.打开 CMD 窗口，测试和宿主机、FreeNAS 虚拟机的连通性，如图 3-65 所示。

```
ping 192.168.1.104
ping 192.168.1.121
```

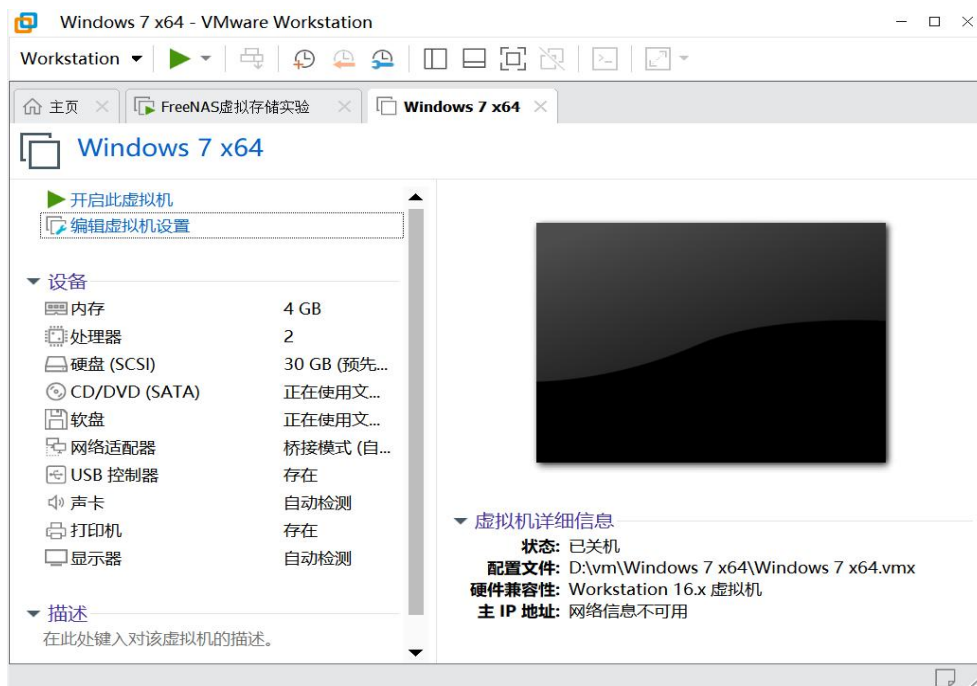


图 3-62 windows7 虚拟机参赛配置

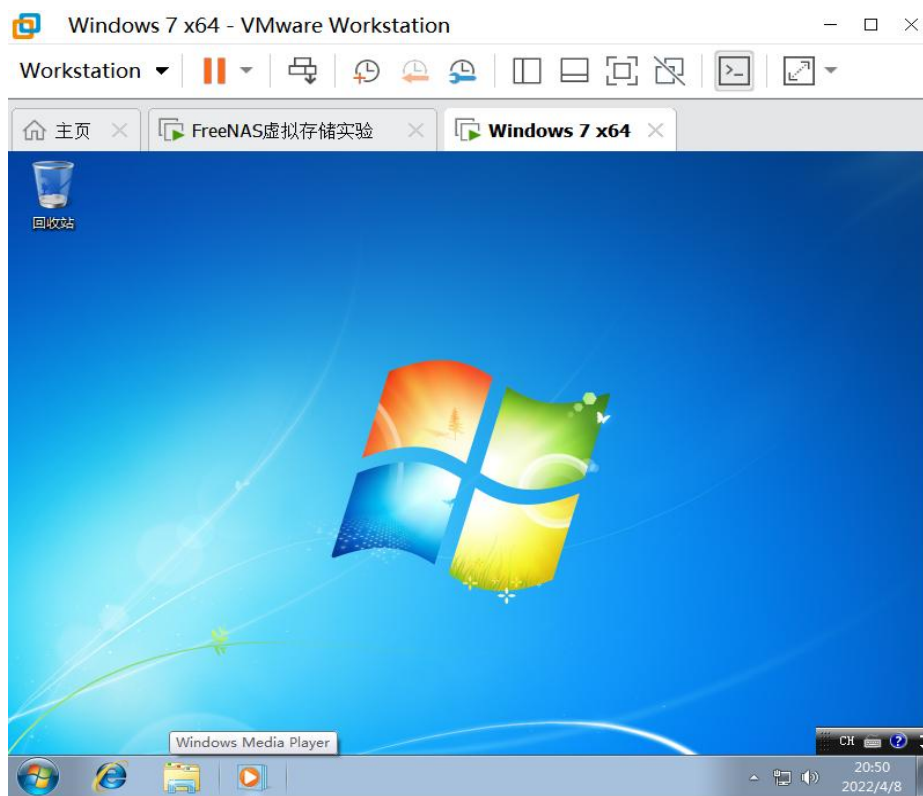


图 3-63 windows7 系统安装成功

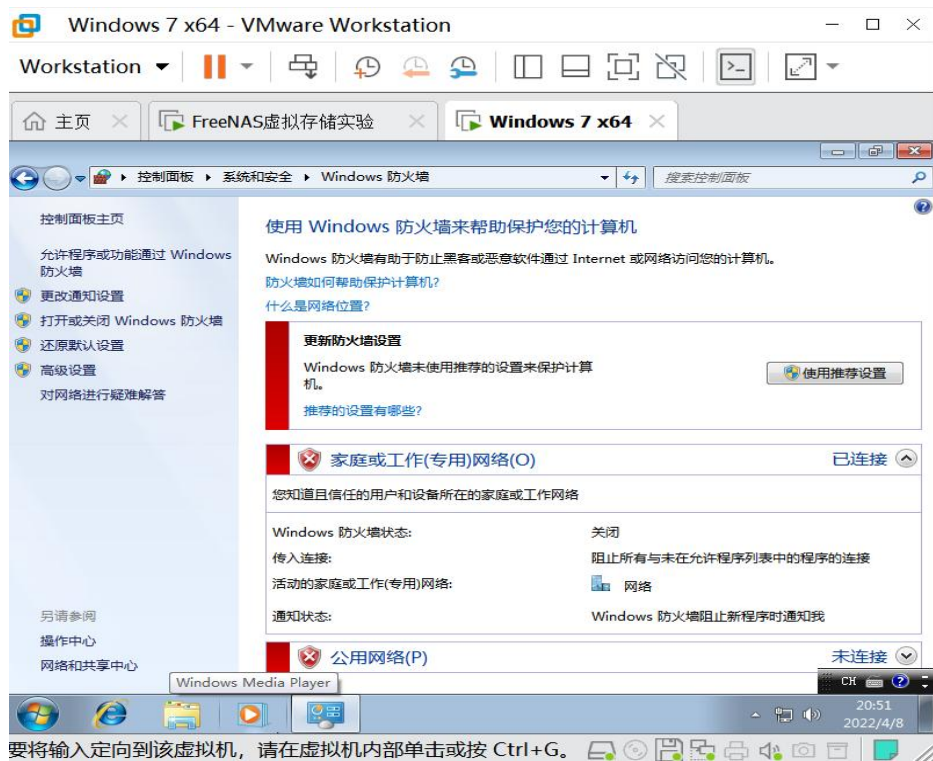


图 3-64 关闭防火墙

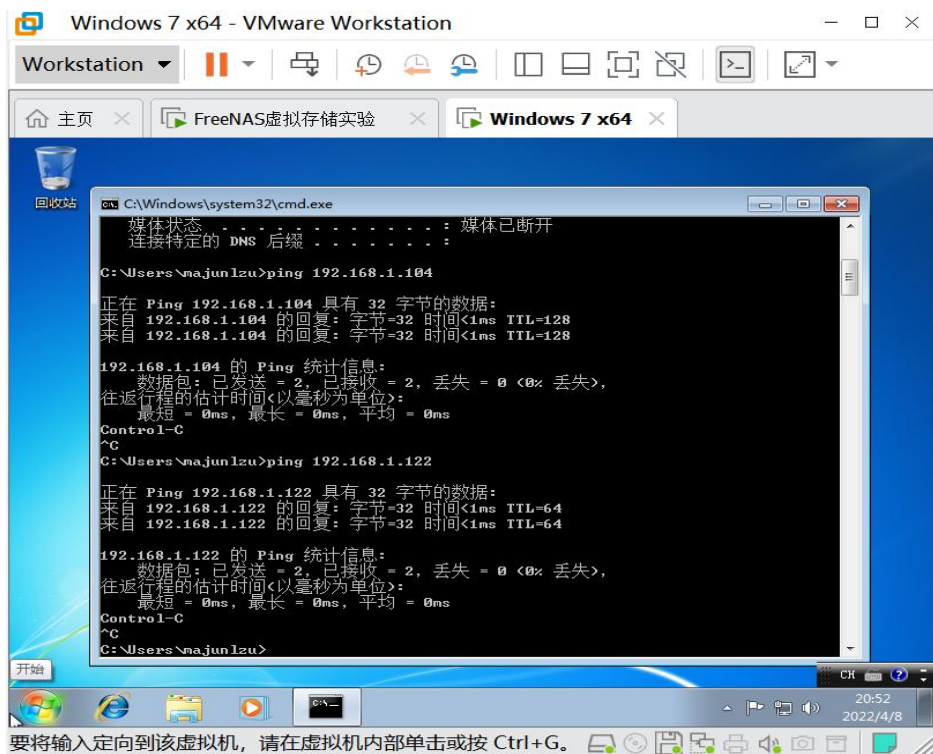


图 3-65 测试和宿主机、freeNAS 虚拟机的连通性

四、安装 Centos7 linux 系统虚拟机（模拟业务服务器二）

1.在 VMwareStation 安装 Centos7 Linux 虚拟机，配置如图 3-66 所示，安装过程略，需要记住用户和密码，我们设置用户 root 和 majunlzu，密码都为 Huawei@1234。

2.安装成功后启动后如图 3-67 所示。



图 3-66 Centos7-Linux 虚拟机配置

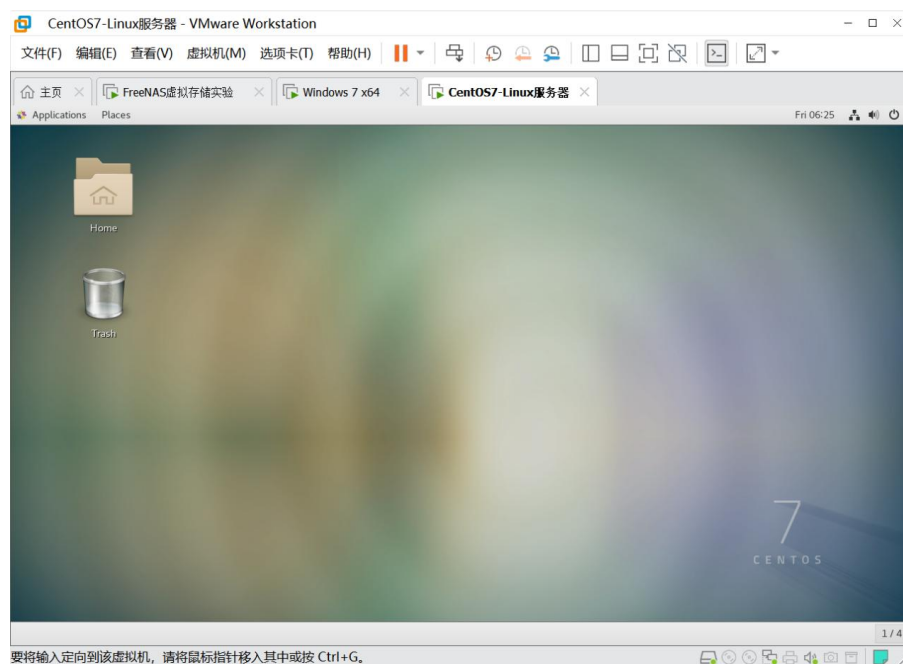


图 3-67 安装 centos7 成功

3. 打开 terminal 窗口, 运行 ifconfig 命令查看 IP 地址, 如图 3-68 所示。

```
majun@localhost:~  
File Edit View Search Terminal Help  
[majun@localhost ~]$ ifconfig  
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.1.123 netmask 255.255.255.0 broadcast 192.168.1.255  
    inet6 fe80::b99d:617f:586e:c9c8 prefixlen 64 scopeid 0x20<link>  
    ether 00:0c:29:ae:ed:49 txqueuelen 1000 (Ethernet)  
    RX packets 5468 bytes 7302736 (6.9 MiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 2501 bytes 171046 (167.0 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1 (Local Loopback)  
    RX packets 0 bytes 0 (0.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 0 bytes 0 (0.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500  
    inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255  
    ether 52:54:00:11:61:1f txqueuelen 1000 (Ethernet)  
    RX packets 0 bytes 0 (0.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 0 bytes 0 (0.0 B)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
[majun@localhost ~]$
```

图 3-68 查看 IP 地址

4.测试和宿主机以及其它虚拟机的连通性，如图 3-69 所示。

```
majun@localhost:~  
File Edit View Search Terminal Help  
[majun@localhost ~]$ ping 192.168.1.104  
PING 192.168.1.104 (192.168.1.104) 56(84) bytes of data.  
64 bytes from 192.168.1.104: icmp_seq=1 ttl=128 time=0.466 ms  
64 bytes from 192.168.1.104: icmp_seq=2 ttl=128 time=0.177 ms  
64 bytes from 192.168.1.104: icmp_seq=3 ttl=128 time=0.260 ms  
^C  
--- 192.168.1.104 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2000ms  
rtt min/avg/max/mdev = 0.177/0.301/0.466/0.121 ms  
[majun@localhost ~]$ ping 192.168.1.121  
PING 192.168.1.121 (192.168.1.121) 56(84) bytes of data.  
64 bytes from 192.168.1.121: icmp_seq=1 ttl=128 time=0.491 ms  
64 bytes from 192.168.1.121: icmp_seq=2 ttl=128 time=0.265 ms  
64 bytes from 192.168.1.121: icmp_seq=3 ttl=128 time=0.319 ms  
^C  
--- 192.168.1.121 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2001ms  
rtt min/avg/max/mdev = 0.265/0.358/0.491/0.097 ms  
[majun@localhost ~]$ ping 192.168.1.122  
PING 192.168.1.122 (192.168.1.122) 56(84) bytes of data.  
64 bytes from 192.168.1.122: icmp_seq=1 ttl=64 time=0.426 ms  
64 bytes from 192.168.1.122: icmp_seq=2 ttl=64 time=0.227 ms  
64 bytes from 192.168.1.122: icmp_seq=3 ttl=64 time=0.180 ms  
^C  
--- 192.168.1.122 ping statistics ---  
3 packets transmitted, 3 received, 0% packet loss, time 2000ms  
rtt min/avg/max/mdev = 0.180/0.277/0.426/0.108 ms  
[majun@localhost ~]$
```

图 3-69 测试连通性

五、模拟 FreeNAS 添加物理存储设备并提供给各业务服务器

1. 关闭 FreeNAS 虚拟机
2. 单击“编辑虚拟机设置”，单击“添加...”，选择“硬盘”，如图 3-70 所示。

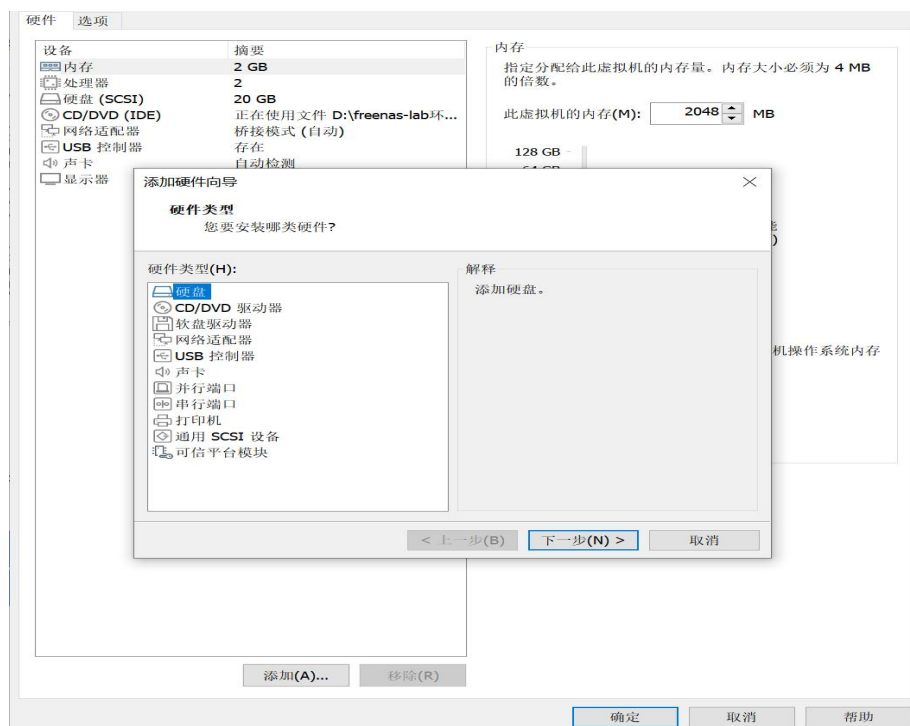


图 3-70 添加 1 块硬盘

3. 选择创建新虚拟磁盘，模拟物理环境中增加物理磁盘，如图 3-71 和图 3-72 所示。

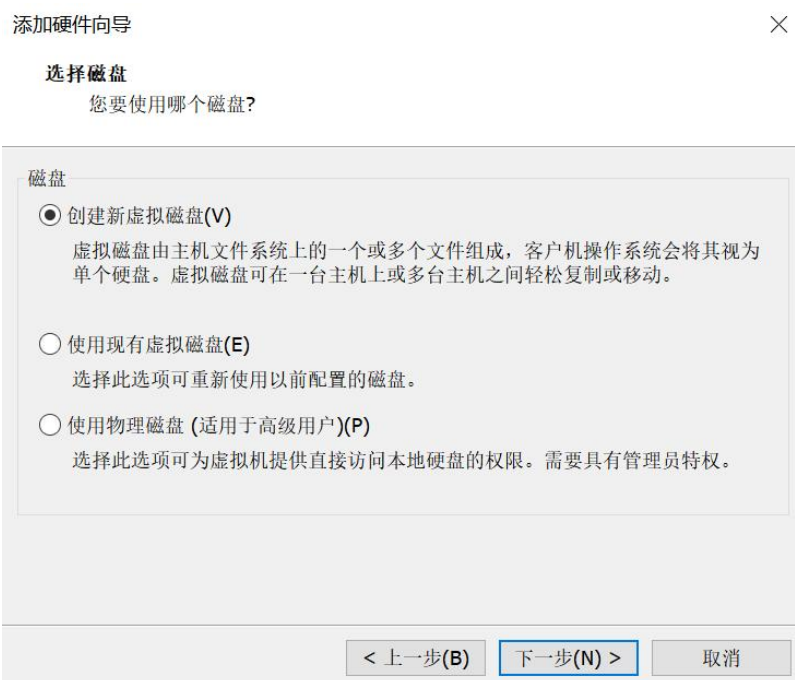


图 3-71 创建一块新的虚拟磁盘

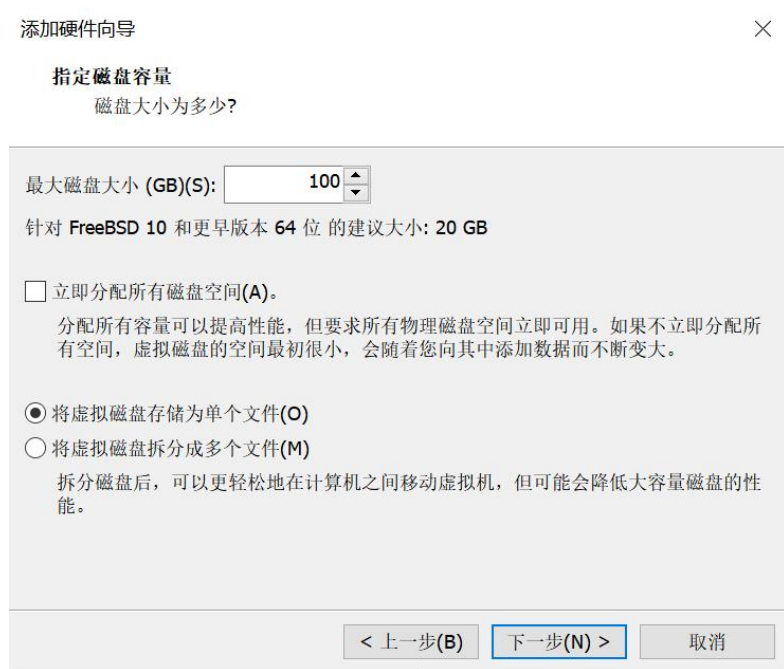


图 3-72 磁盘大小 100G

4. 重新启动 FreeNAS 虚拟机，然后准备完成以下操作：
 - (1) 在存储中划分 50G-LUN-给 windows 服务器使用
 - (2) 在存储中划分 50G-LUN-给 linux 服务器使用
5. 在浏览器中打开 freeNAS 的 url 地址 <http://192.168.1.122>，输入用户名和密码登录系统
6. 选择“Storage”图标，单击“volume manager”，导入新加的磁盘，如图 3-73 所示。
7. 单击“add volume”，稍等，等添加成功，如图 3-74。

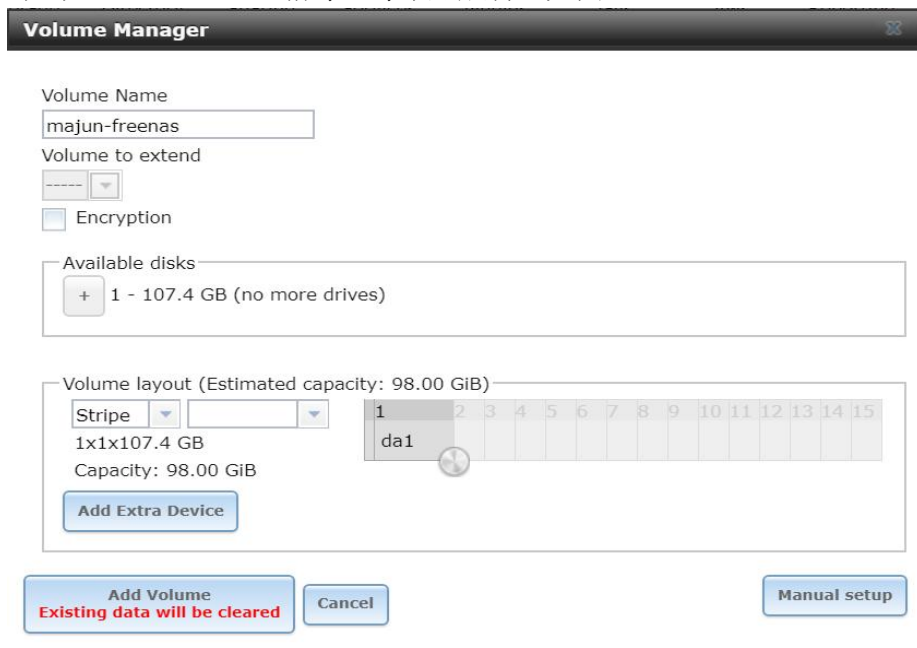


图 3-73 导入新加的磁盘

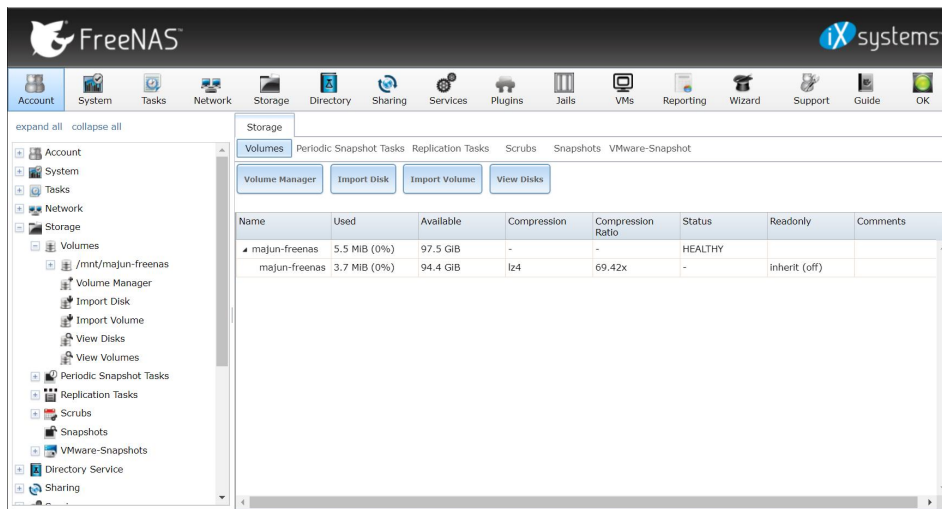


图 3-74 添加磁盘卷成功

8. 创建一个 50G 的 LUN 分享给 windows，选择“majun-freenas”磁盘，在窗口下方单击“zvol”图标，如图 3-75 所示，填写相关信息。

9. 选择 sharing，选择 block（iSCSI）块存储，添加 portal，如图 3-76 所示。

10. 在 freeNAS 中，启动 iscsi 服务，如图 3-77 所示。

11. 在 windows7 虚拟机中，打开控制面板，打开管理工具，双击“iSCSI 发起程序”，如图 3-78，如果没有运行，则单击确定让服务运行，服务运行起来后，再双击“iSCSI 发起程序”，出现配置 iSCSI 界面，选择“配置”标签，可看到发起程序名称栏目，如图 3-79 所示。

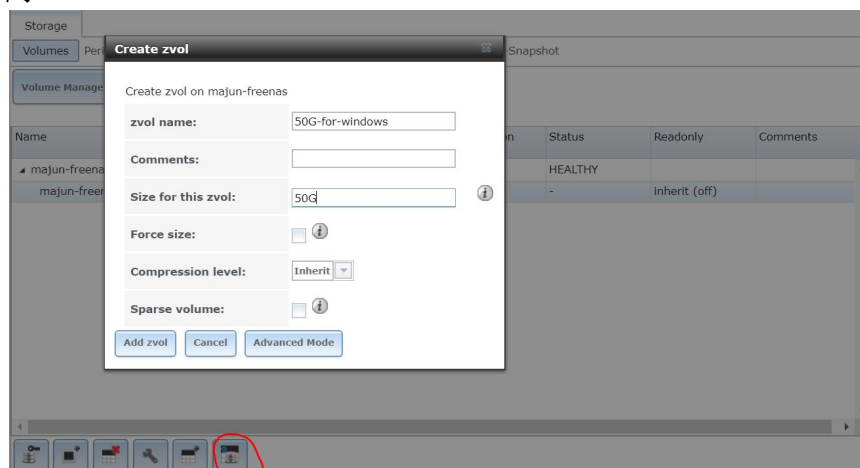


图 3-75 创建 50G-LUN

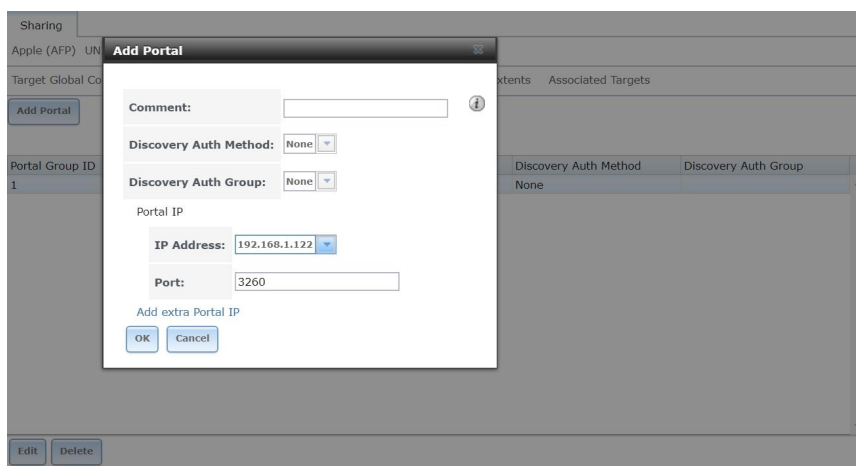


图 3-76 添加 Portal，默认情况下，端口为 3260

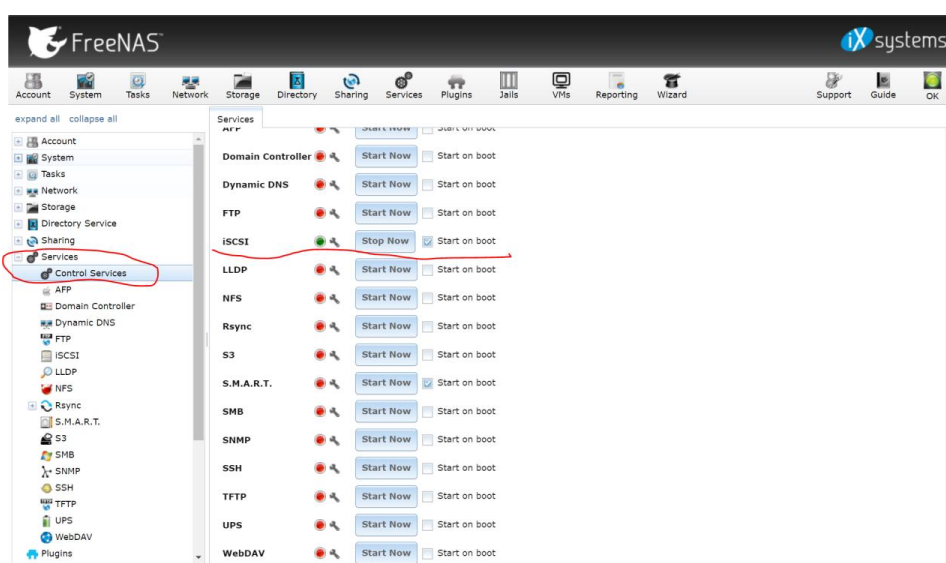


图 3-77 FreeNAS 中启动 iSCSI 服务

12.修改并复制 windows 服务器 iSCSI 中的发起程序名称如图 3-79 所示。

13.返回 freeNAS WEB 界面，再 block (iSCSI) 标签下选择“Initiators”，然后单击“add initiator”按钮，如图 3-80 所示，粘贴或输入刚才的 iSCSI 的发起程序名。

14.添加一个 Target，填入相应信息，如图 3-81 所示。

15.添加一个 extent，填入相应信息，代表准备共享给 windows 使用，如图 3-82 所示。

16.添加 Associated Target，选择创建好 target，映射给 windows 使用，如图 3-83 所示。

17.回到块处存储，我们的准备工作已经完成，如图 3-84 所示。

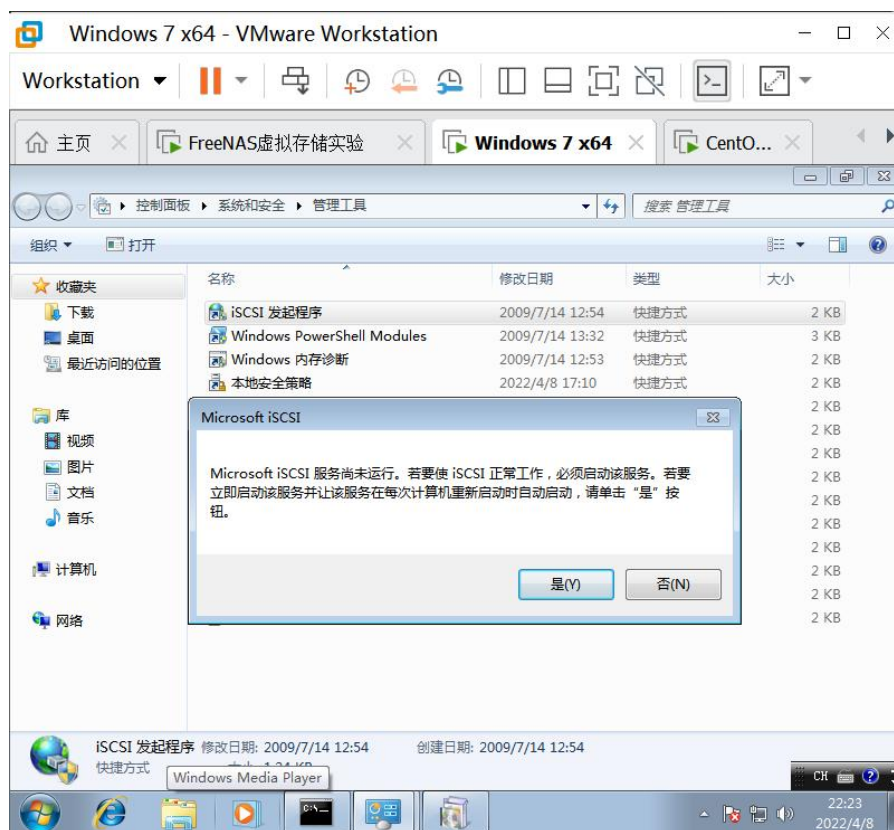


图 3-78 Windows 中运行 iSCSI 发起程序

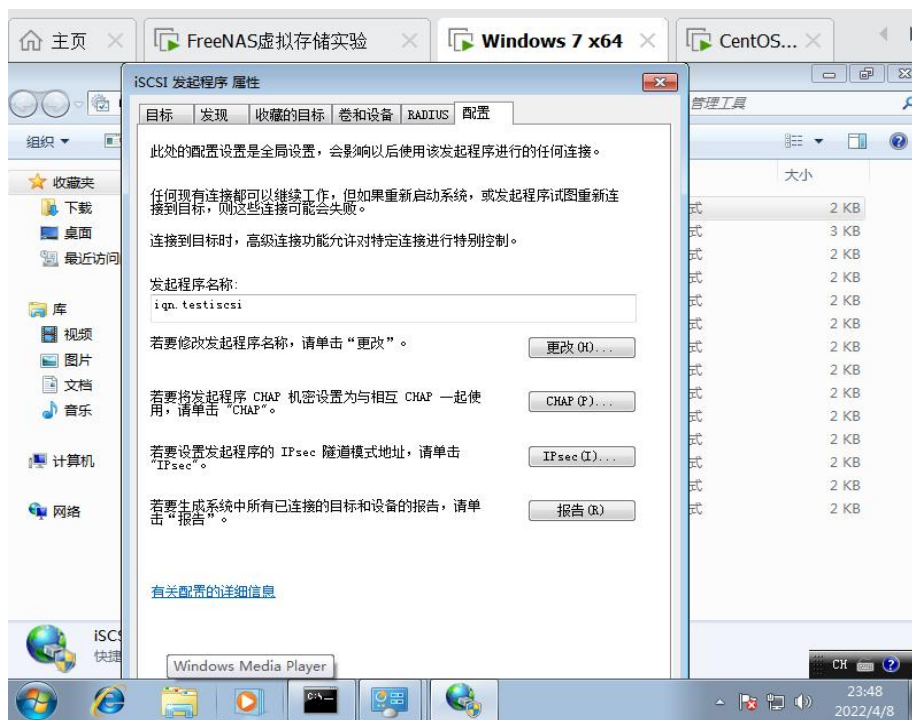


图 3-79 Windows 中修改和记录发起程序名称

Sharing			
Apple (AFP) UNIX (NFS) WebDAV Windows (SMB) Block (iSCSI)			
Target Global Configuration Portals Initiators Authorized Access Targets Extents Associated Targets			
Add Initiator			
Group ID	Initiators	Authorized network	Comment
1	iqn.testiscsi	ALL	

图 3-80 FreeNAS 中添加 Initiator

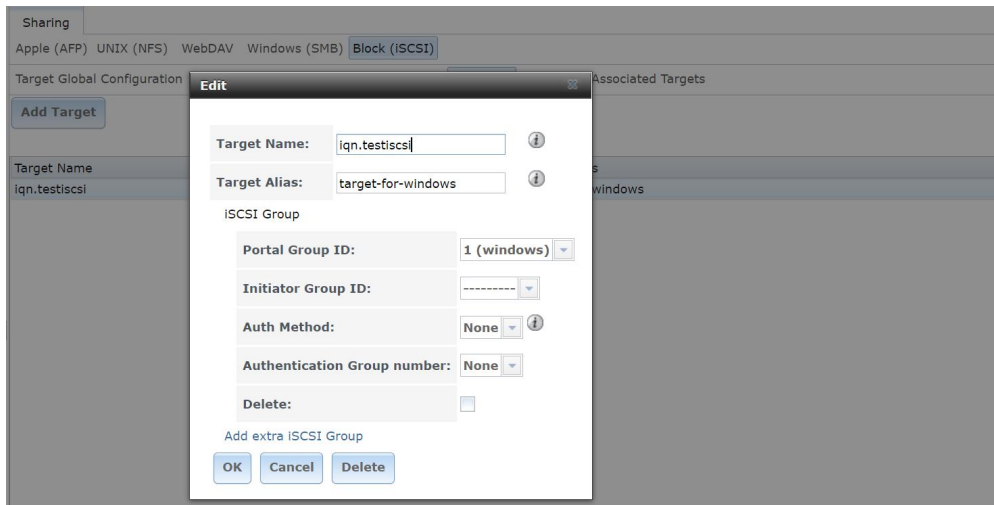


图 3-81 FreeNAS 中添加 target

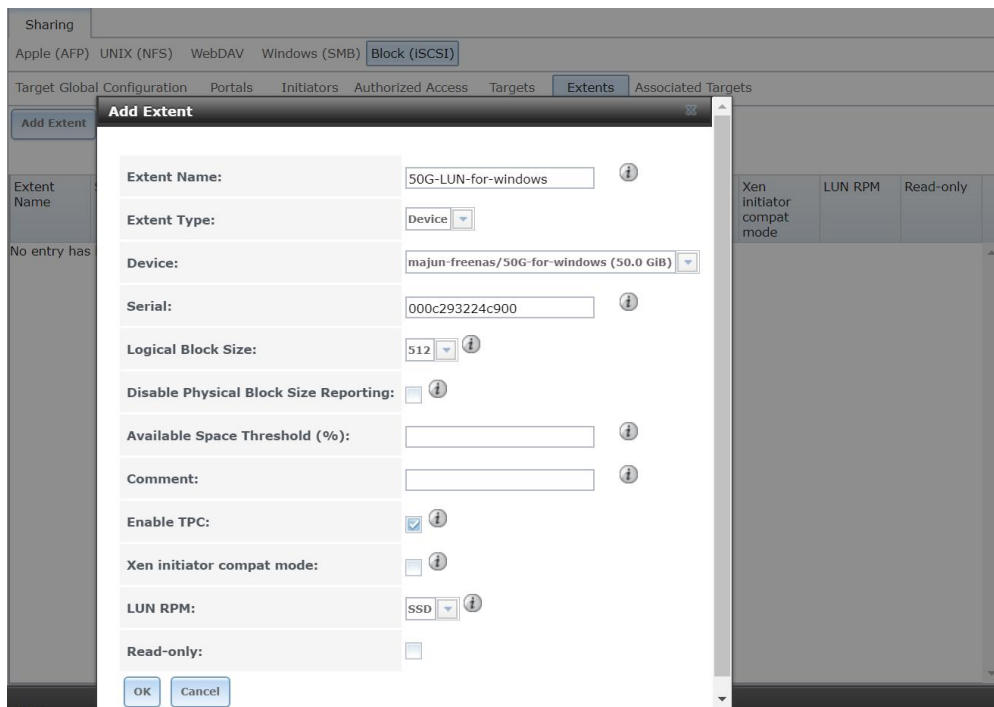


图 3-82 添加 extent

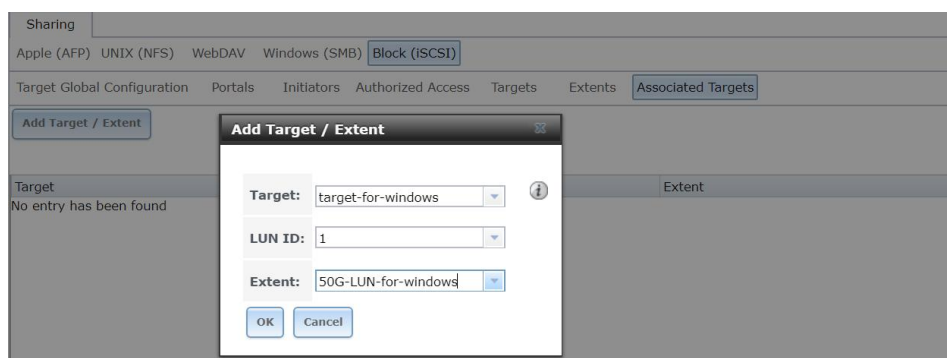


图 3-83 FreeNAS 中添加 Associated Target

Sharing		
Apple (AFP) UNIX (NFS) WebDAV Windows (SMB) Block (iSCSI)		
Target Global Configuration Portals Initiators Authorized Access Targets Extents Associated Targets		
Add Target / Extent		
Target	LUN ID	Extent
iqn.testiscsi	1	LUN-for-windows

图 3-84 成功创建 LUN

18.回到 windows7 系统，在目标标签下单文本框中输入 freeNAS 虚拟机的 IP 地址 192.168.1.122，然后单击“快速连接”按钮，如图 3-85 所示，系统找到 freeNAS 提供连接 iqn.testiscsi,然后连接。

19.打开主菜单，右键单击“管理”，如图 3-86 所示。

20.然后选择“磁盘管理”，系统弹出发现一块新“磁盘”，该磁盘就是通过 iscsi 接口发现的通过 ip 网络协议可以使用的磁盘，如图 3-87 和图 3-88 所示。

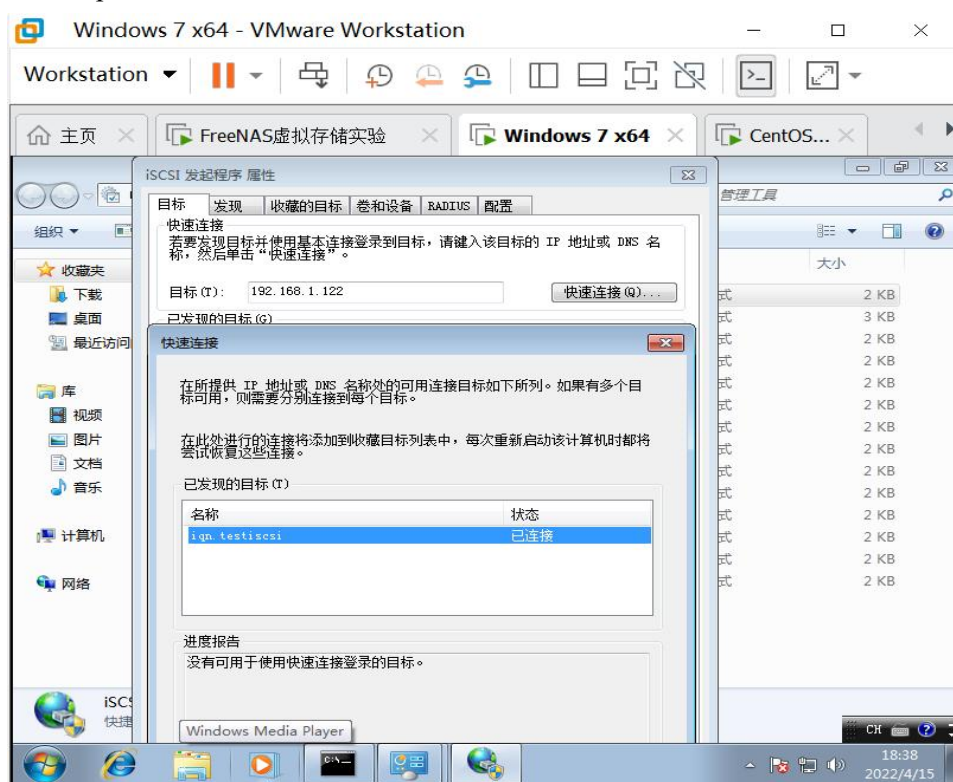


图 3-85 连接成功

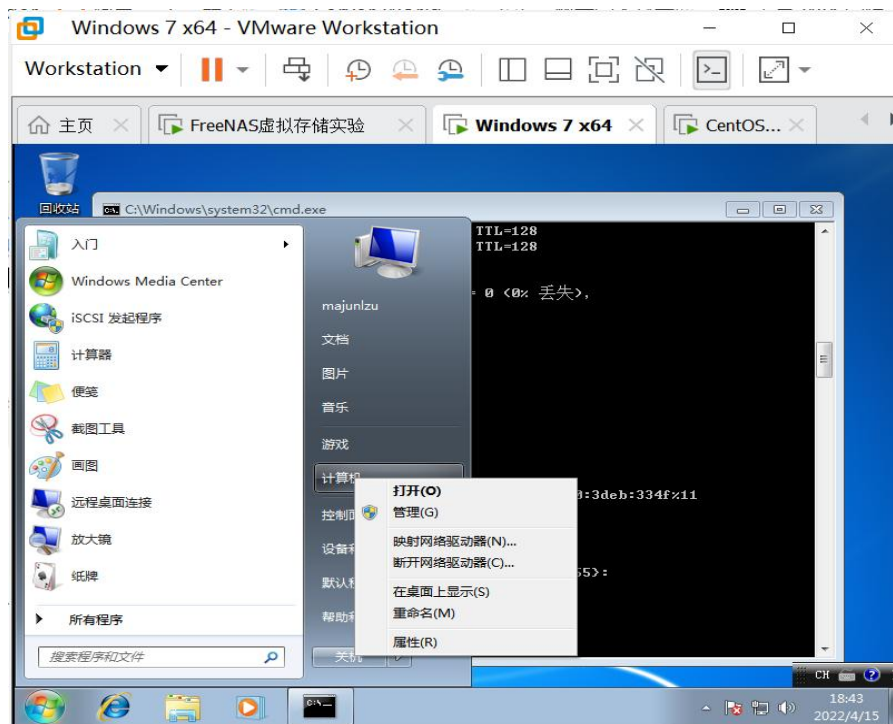


图 3-86 右键菜单打开“管理”

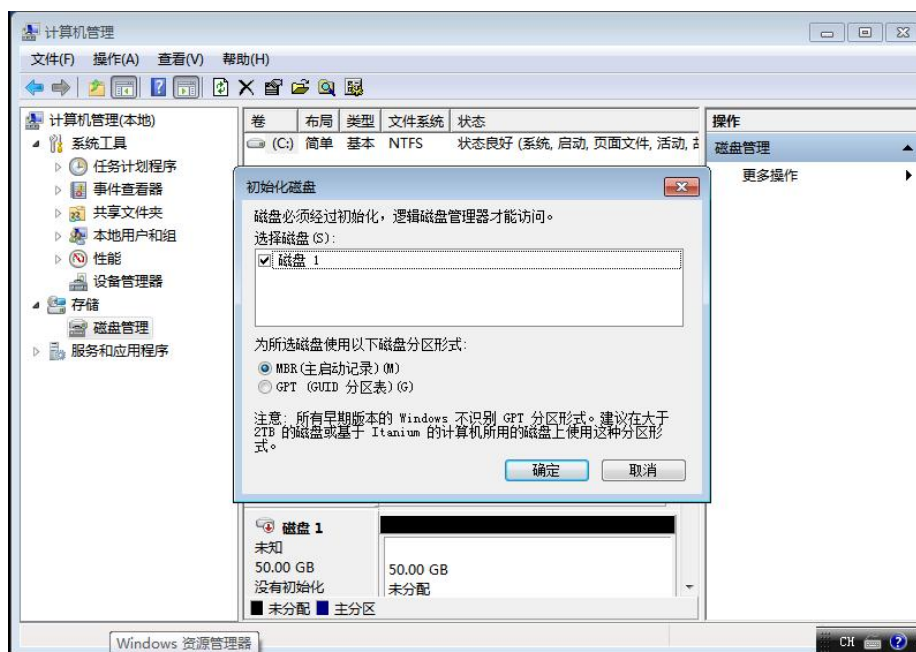


图 3-87 Windows 中发现新磁盘

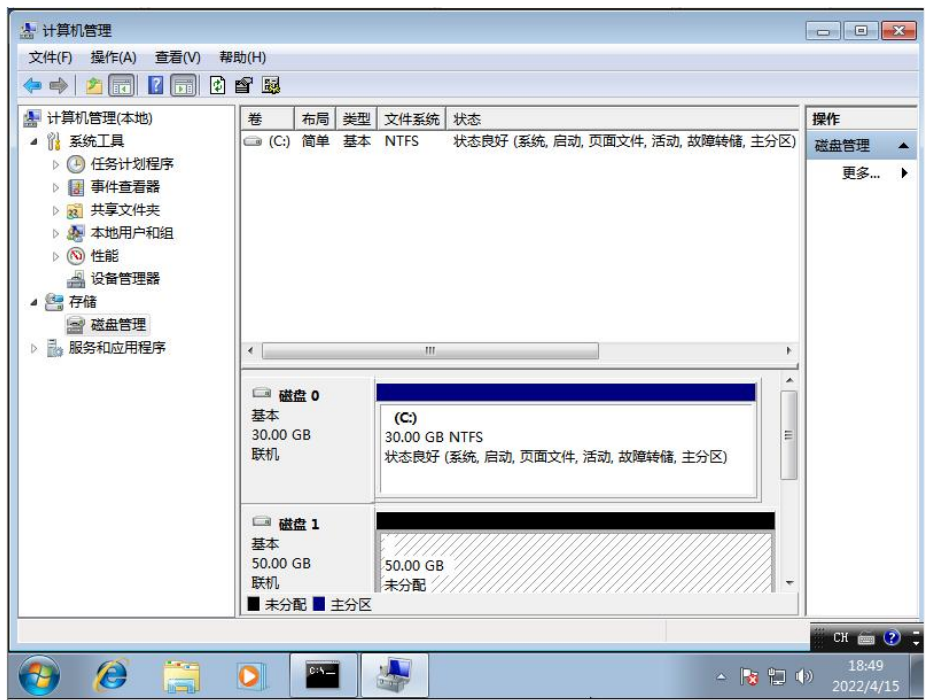


图 3-88 Windows 中多了一块 50G 的磁盘

21.后面的操作就和本地磁盘的使用是相同的，可以分区，然后格式化，指定盘符，就可以正常使用了，如图 3-89、图 3-90 所示。

22.打开新的逻辑盘，编辑文件或拷贝文件，可以正常使用了，如图 3-91 所示，给 windows 添加磁盘成功。

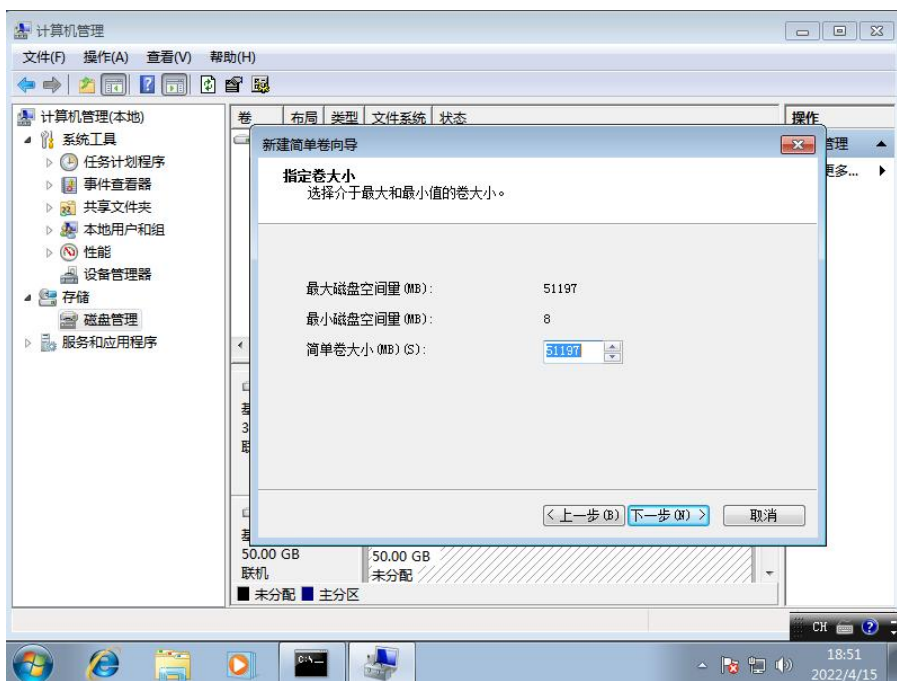


图 3-89 Windows 中新建简单卷

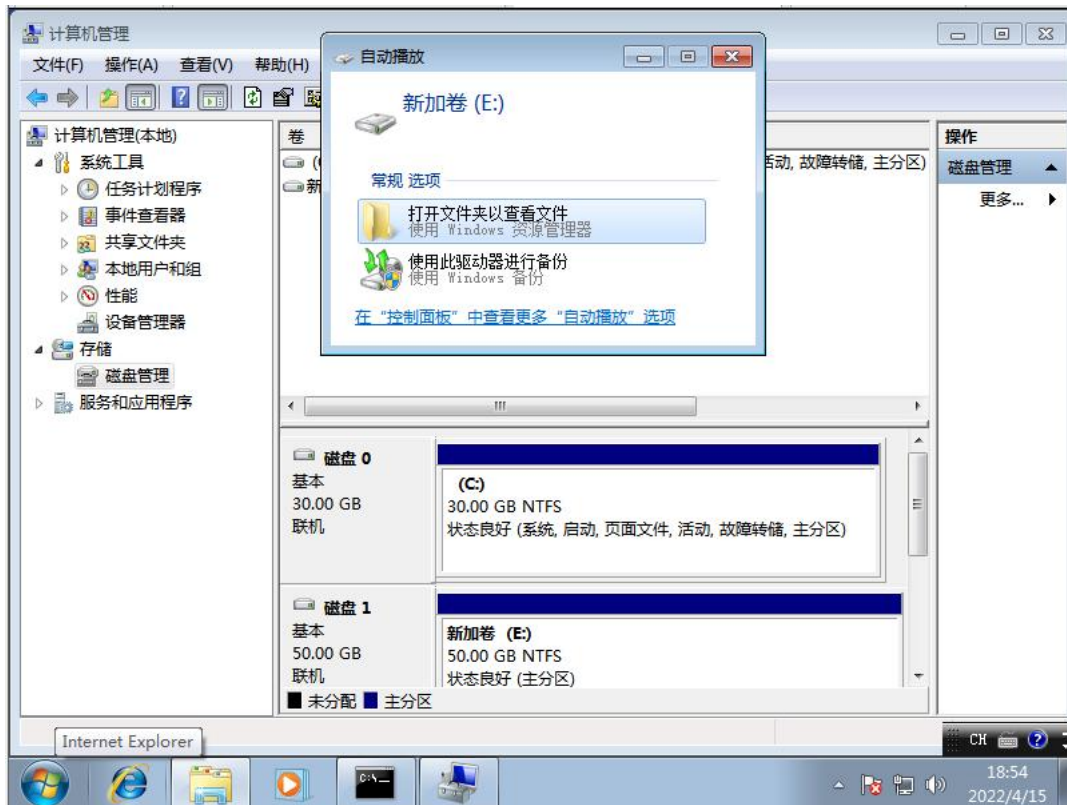


图 3-90 格式化成功

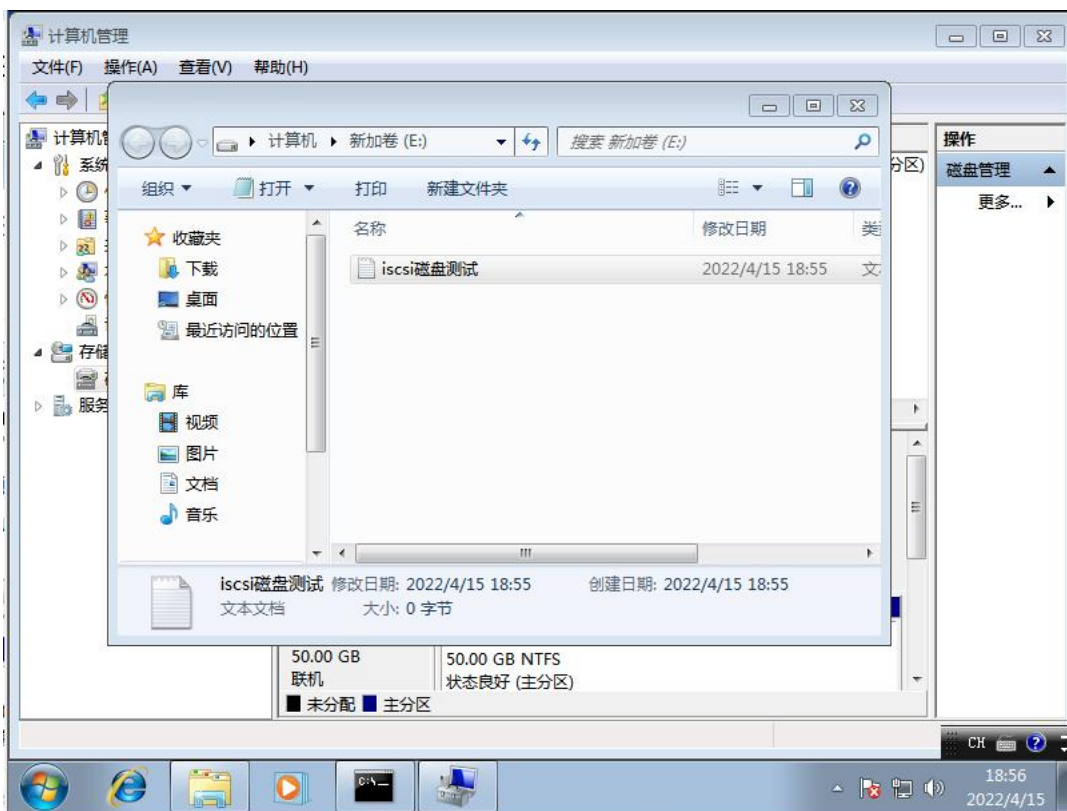


图 3-91 可以正常使用了

23.下面完成在 linux 下添加虚拟存储设备，首先启动 Centos7 虚拟机，输入用户和密

码登录，然后打开 terminal，切换到 root 用户，如图 3-92 所示。

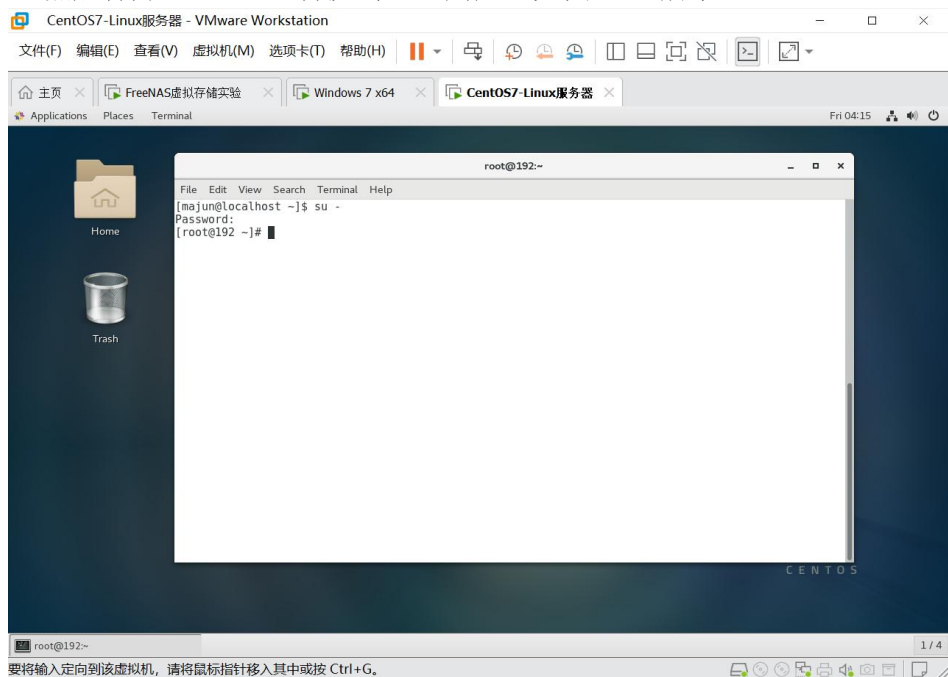


图 3-92 登录 centos 系统

24.安装 iscsi 启动器，并启动 iscsi 服务，如图 3-93 所示。

#安装 iscsi 启动器 / 启动 iscsi 启动服务

```
[root@centos7-shi-001 tom]# yum install iscsi-initiator-utils -y [root@centos7-shi-001 tom]#  
systemctl start iscsid
```

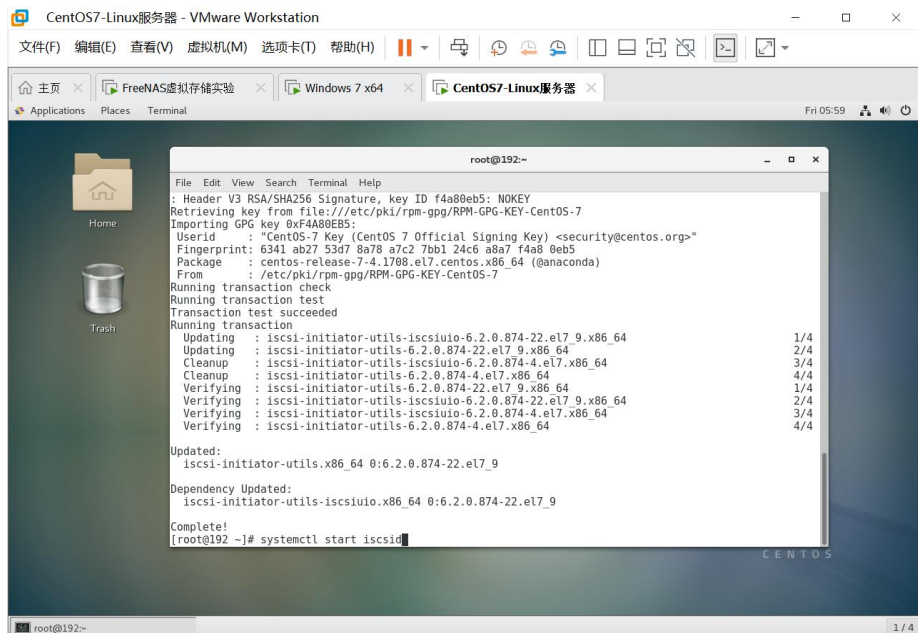


图 3-93 安装 iscsi 启动器

25.使用 iscsiadm 扫描指定 NAS 服务器上的 iSCSI 磁盘，没有扫描到，原因是我们没有为 Linux 添加 LUN，首先要查看 Linux 服务器上识别的 iSCSI 初始化(initial 识别码)，

如图 3-94 所示。

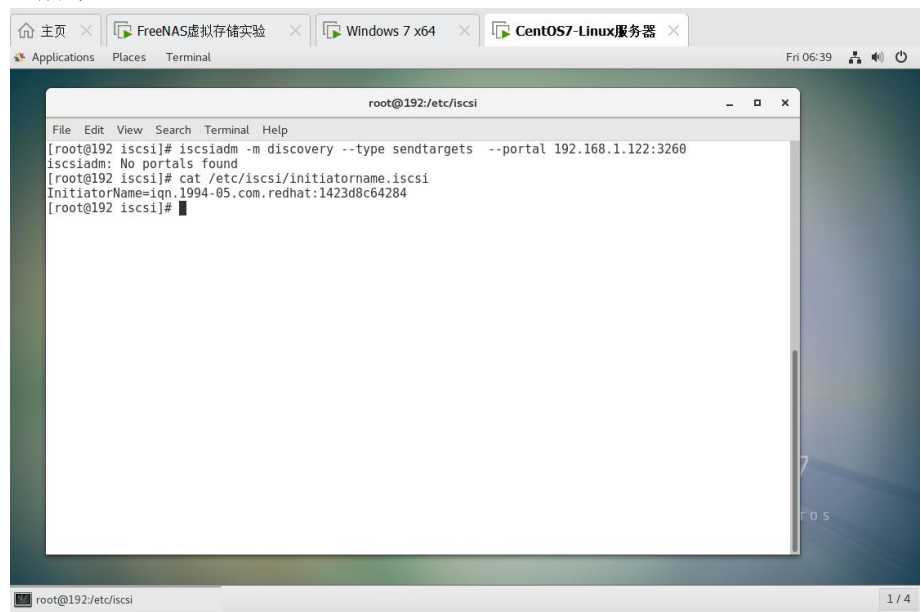


图 3-94 查看 iqn 识别码

26.在 FreeNAS 中为 Linux 添加一块 LUN，添加流程参考前面为 windows 添加 LUN 的过程，此处省略步骤说明，添加过程截图如图 3-95-图 3-100 所示。

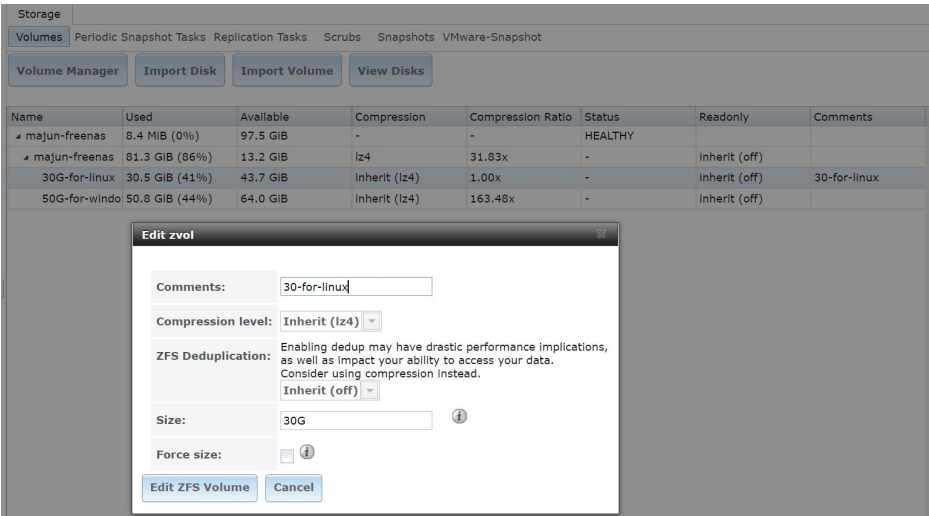


图 3-95 FreeNAS 中添加物理卷

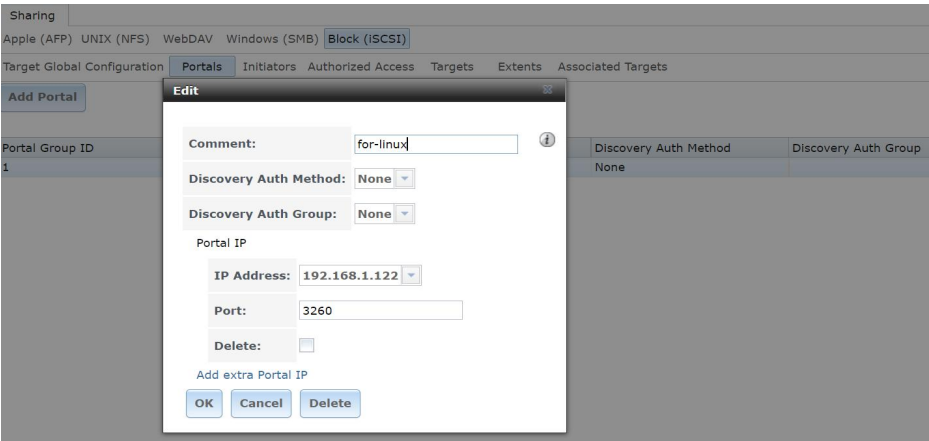


图 3-96 添加 portals

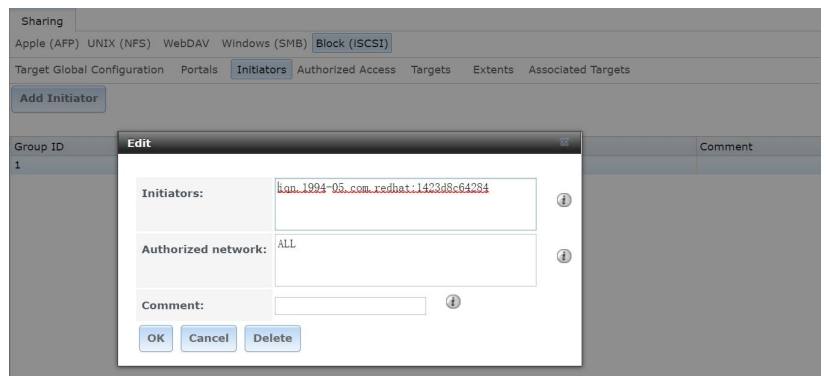


图 3-97 添加 initiator

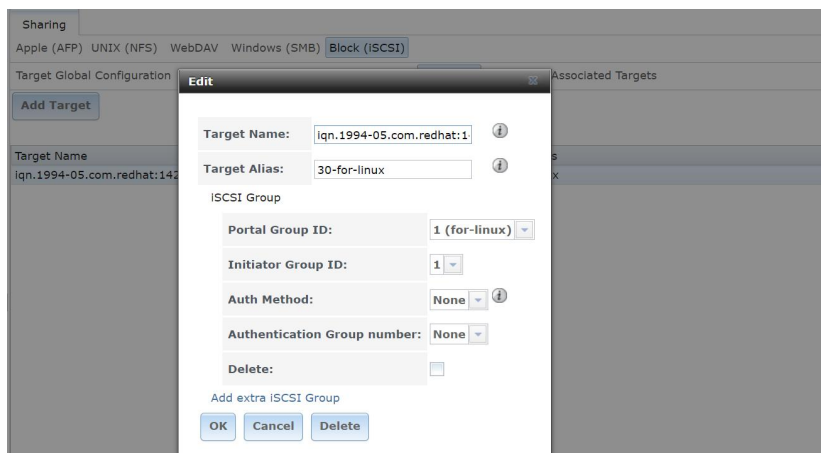


图 3-98 添加 target

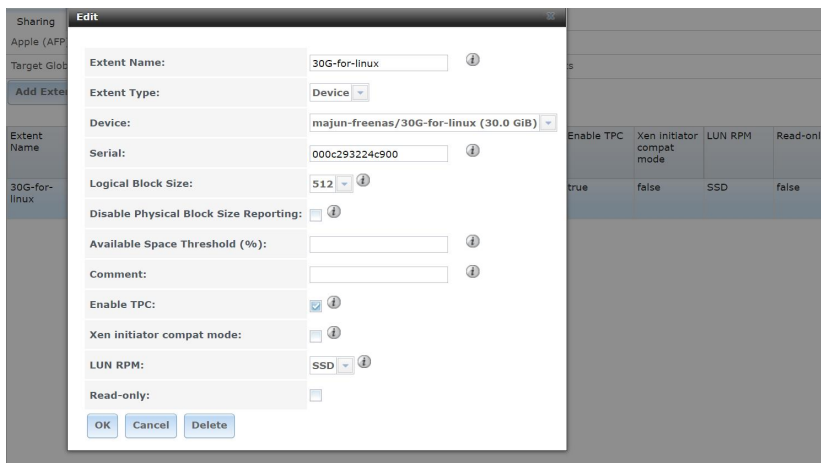


图 3-99 创建 extent name 并关联设备

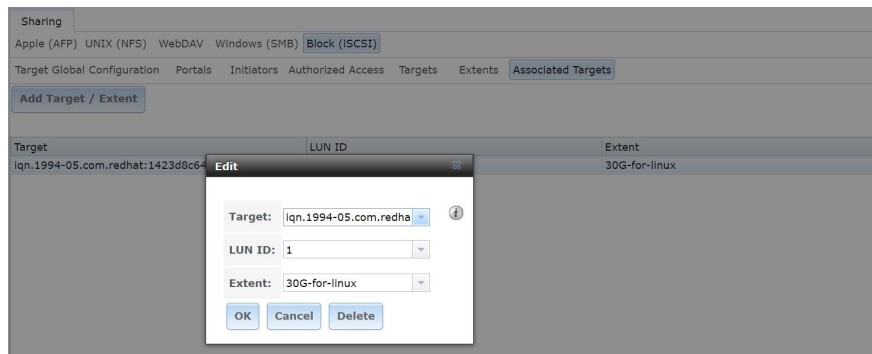


图 3-100 创建 LUN

27.重新使用 iscsi 扫描，发现一块 LUN 磁盘，如图 3-101 所示。

#重新 iscsi 扫描

```
iscsiadm -m discovery --type sendtargets --portal  
192.168.199.152:3260
```

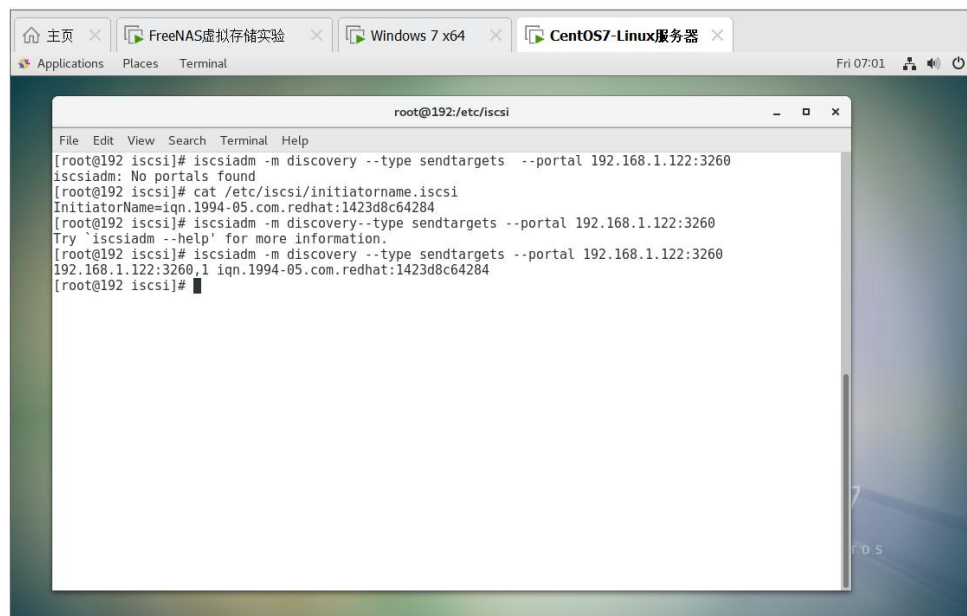


图 3-101 iscsi 扫描发现磁盘

28.连接该远程硬盘，如图 3-102 所示，连接成功。

#连接 iscsi 远程磁盘

```
iscsiadm -m node -T iqn.2005-10.org.freenas.cttl-shi:linux-server  
-p 192.168.199.152:3260 -l
```

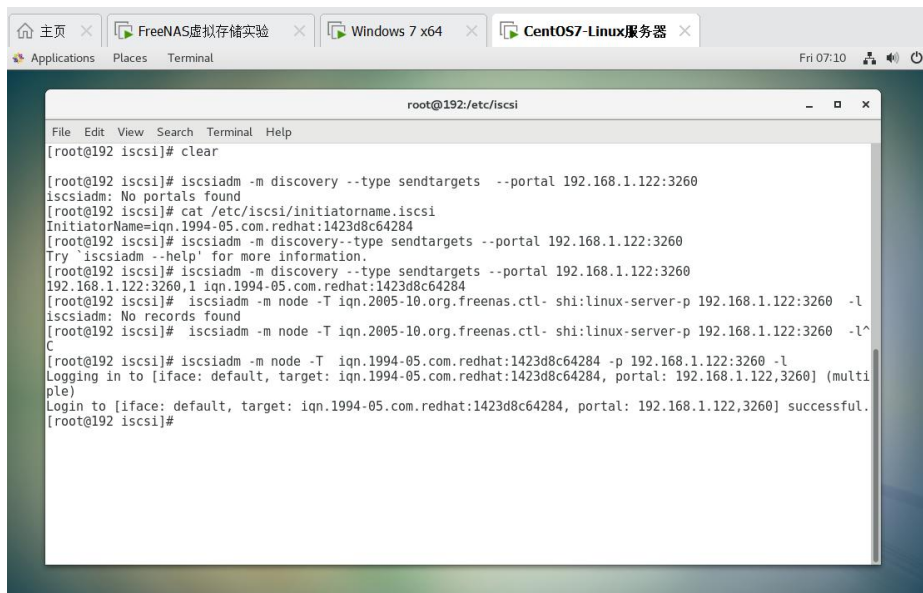


图 3-102 连接远程磁盘成功

29.在 linux 中查看磁盘信息，如图 3-103 所示。

lsblk

fdisk -l

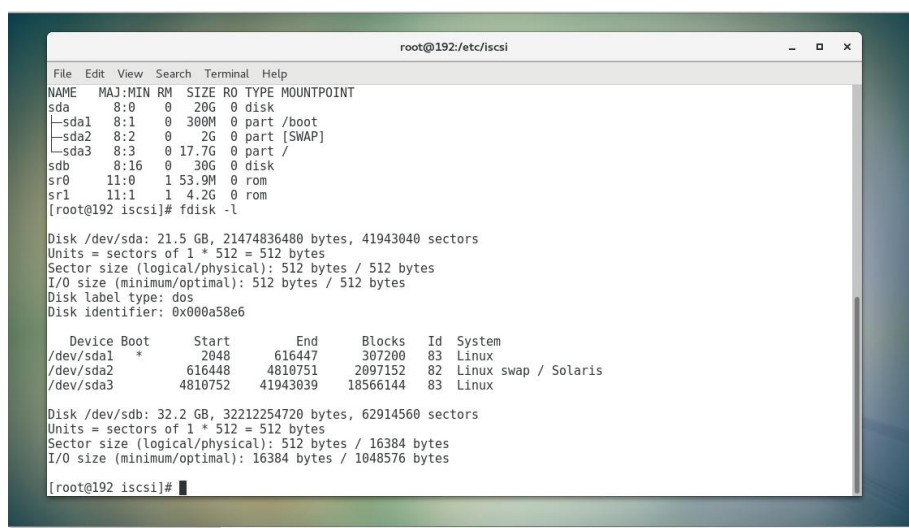


图 3-103 Linux 中多了一块 30G 的磁盘

30.在 linux 下对磁盘进行分区，并完成格式化操作，如图 3-104，图 3-105 所示。

fdisk /dev/sdb

mkfs.ext4 /dev/sdb

31.完成挂载目录，如图 3-106 和 3-107 所示

mkfs.ext4 /dev/sdb1

mkdir /freenas_disk

mount /dev/sdb1 /freenas_disk

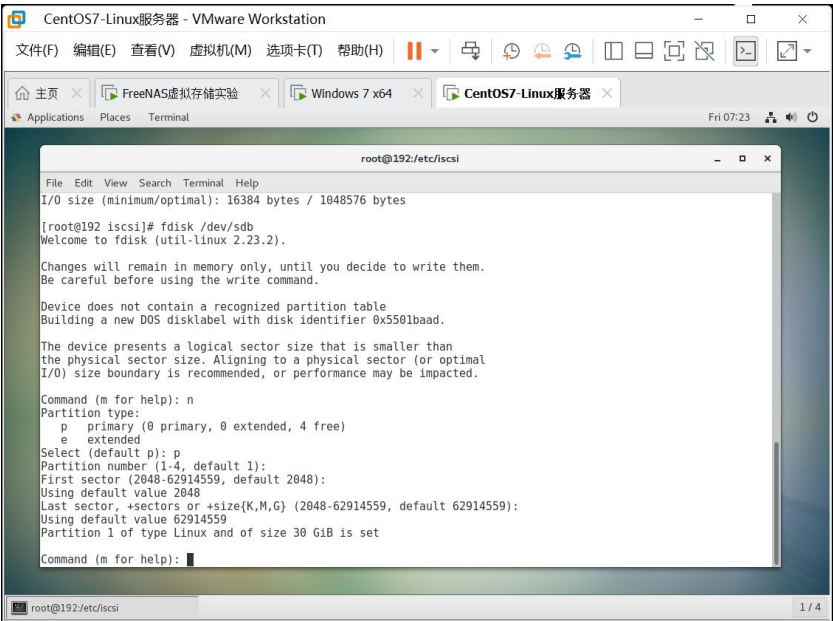


图 3-104 分区操作

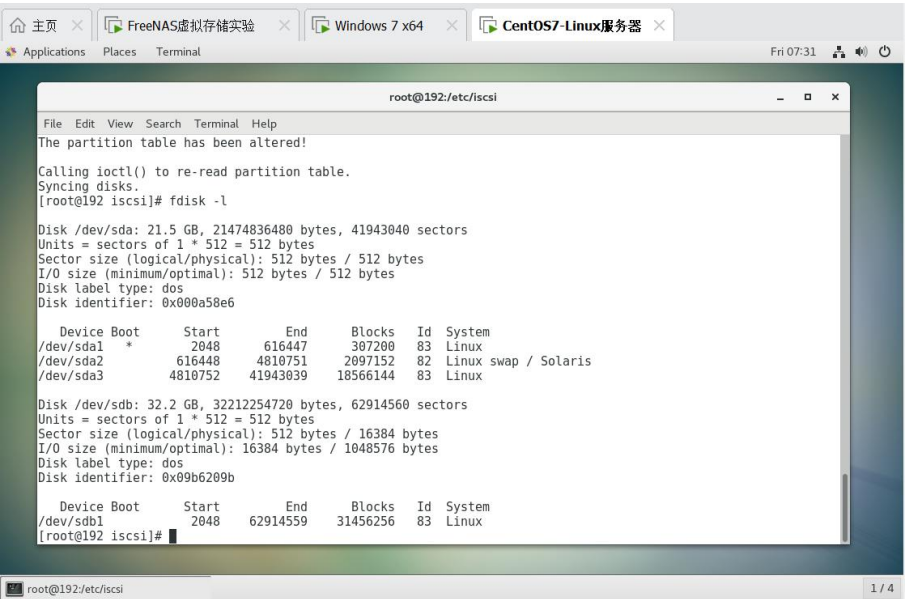


图 3-105 查看分区后信息

```
root@192:/etc/iscsi
File Edit View Search Terminal Help

Device Boot      Start         End      Blocks   Id  System
/dev/sdb1        2048        62914559    31456256    83   Linux

[root@192 iscsi]# mkfs.ext4 /dev/sdb1
mke2fs 1.42.9 (28-Dec-2013)
Discarding device blocks: done
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=4 blocks, Stripe width=256 blocks
1966080 inodes, 7864064 blocks
393203 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2155872256
240 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

[root@192 iscsi]#
```

图 3-106 格式化

```
[root@192 iscsi]# mkdir /freenas disk
[root@192 iscsi]# mount /dev/sdb1 /freenas_disk
[root@192 iscsi]# lsblk -f

NAME        FSTYPE LABEL UUID                                MOUNTPOINT
sda
├─sda1 xfs          dfbe25a7-8b00-4ac9-ae16-14ce894e4da3 /boot
├─sda2 swap        4926be15-272e-453a-bcc7-1221a2e63037 [SWAP]
├─sda3 xfs          e67ec460-8bd6-49e6-a4ba-106d607d6070 /
sdb
└─sdb1 ext4        e991f775-a2fb-488e-928e-d236b88a14bf /freenas_disk
sr0
sr1
[root@192 iscsi]#
```

图 3-107 查看格式化后文件系统信息

32.在 Linux 测试创建文件并编辑使用，可以看到正常使用了，如图 3-108 所示，最后的逻辑拓扑图如图 3-109 所示，本阶段使用完成。

```
CentOS7-Linux服务器 - VMware Workstation
文件(F) 编辑(E) 查看(V) 虚拟机(M) 选项卡(T) 帮助(H)
FreeNAS虚拟存储实验 Windows 7 x64 CentOS7-Linux服务器
Applications Places Terminal Fri 07:45

root@192:/freenas_disk
File Edit View Search Terminal Help

├─sda2 swap        4926be15-272e-453a-bcc7-1221a2e63037 [SWAP]
├─sda3 xfs          e67ec460-8bd6-49e6-a4ba-106d607d6070 /
sdb
└─sdb1 ext4        e991f775-a2fb-488e-928e-d236b88a14bf /freenas_disk
sr0
sr1
[root@192 iscsi]# mkdir /freenas disk
[root@192 iscsi]# mount /dev/sdb1 /freenas_disk
[root@192 iscsi]# lsblk -f
NAME        FSTYPE LABEL UUID                                MOUNTPOINT
sda
├─sda1 xfs          dfbe25a7-8b00-4ac9-ae16-14ce894e4da3 /boot
├─sda2 swap        4926be15-272e-453a-bcc7-1221a2e63037 [SWAP]
├─sda3 xfs          e67ec460-8bd6-49e6-a4ba-106d607d6070 /
sdb
└─sdb1 ext4        e991f775-a2fb-488e-928e-d236b88a14bf /freenas_disk
sr0
sr1
[root@192 iscsi]# cd /freenas_disk
[root@192 freenas_disk]# ls
lost+found
[root@192 freenas_disk]# iscsiadm -m session
tcp: [1] 192.168.1.122:3260,1 iqn.1994-05.com.redhat:1423d8c64284 (non-flash)
[root@192 freenas_disk]# touch freenas.txt
[root@192 freenas_disk]# vim freenas.txt
freenas.txt
[root@192 freenas_disk]# ls
freenas.txt lost+found
[root@192 freenas_disk]#
```

图 3-108 Linux 中可以正常使用 freenas 磁盘

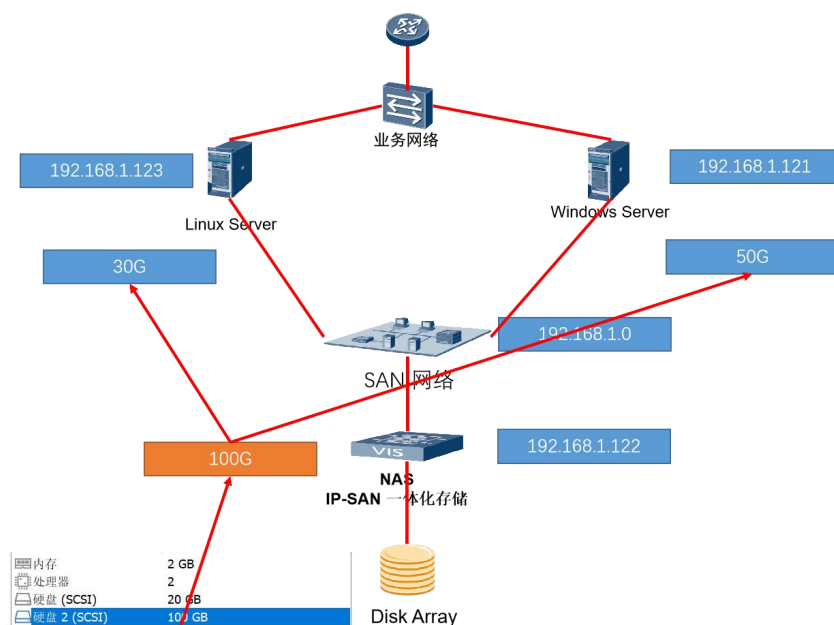


图 3-109 最后逻辑拓扑图

六、FreeNAS 共享存储

前面的实验提供的是块存储，即为服务器提供的 LUN 块设备，然后在服务器中进行分区和格式化，由服务器决定使用什么文件系统，这种存储方式一般是单机存储方式，不能共享使用，若多机使用，会发生内容覆盖现象。下面的实验演示共享存储，即在 FreeNAS 中先创建好文件系统，然后通过网络协议提供给服务器，服务器不能决定使用什么文件系统，因为 FreeNAS 已经做好了，服务器只是将其作为一个共享的存储设备来使用，我们使用 SMB 文件路径提供给 windows 和 linux 服务器共享使用。

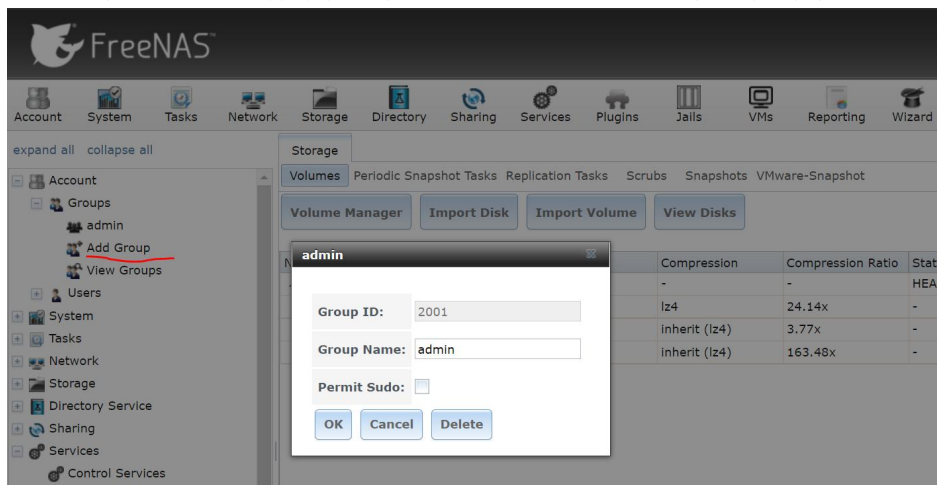


图 3-110 创建用户组

(一) CIFS 共享存储实验

CIFS (Common Internet File System) 是公共的或开放的 SMB 协议版本，并由 Microsoft 使用。它使程序可以访问远程 Internet 计算机上的文件并要求此计算机提供服务。CIFS 使用客户/服务器模式。客户程序请求远在服务器上的服务器程序为它提供服务，服务器获得请求并返回响应。相关知识请参考：

<https://baike.sogou.com/v8484569.htm?fromTitle=cifs&ch=frombaikevr。>

1. 首先在 freeNAS 中创建用户组 admin，如图 3-110 所示。
2. 创建使用用户，需要选择共享目录和记录密码（此处为 admin1234），并加入前面创建好的组，如图 4-111 所示。

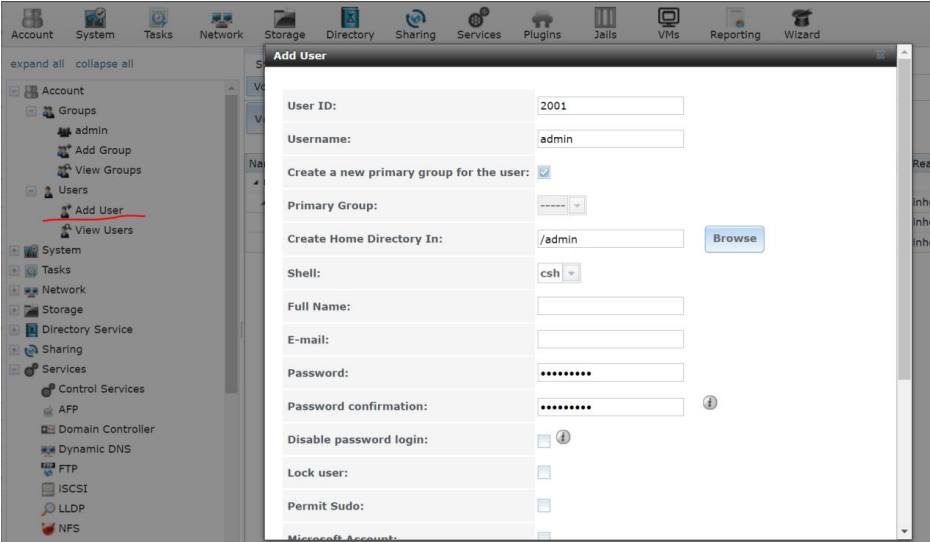


图 3-111 创建用户

3. 创建数据集，即提供共享文件路径，如图 3-112 所示。
4. 管理使用用户和用户组，可以修改访问权限，如图 3-113 所示。
5. 打开相应的服务，Windows 中网络存储使用的是 SMB（服务消息块协议）协议提供共享服务，如图 3-114 所示。
6. 在 Sharing 标签下，选择 Windows（SMB）标签，单击“Add Windows（SMB）share”按钮，如图 3-115-图 3-117 所示，选择共享路径完成添加。

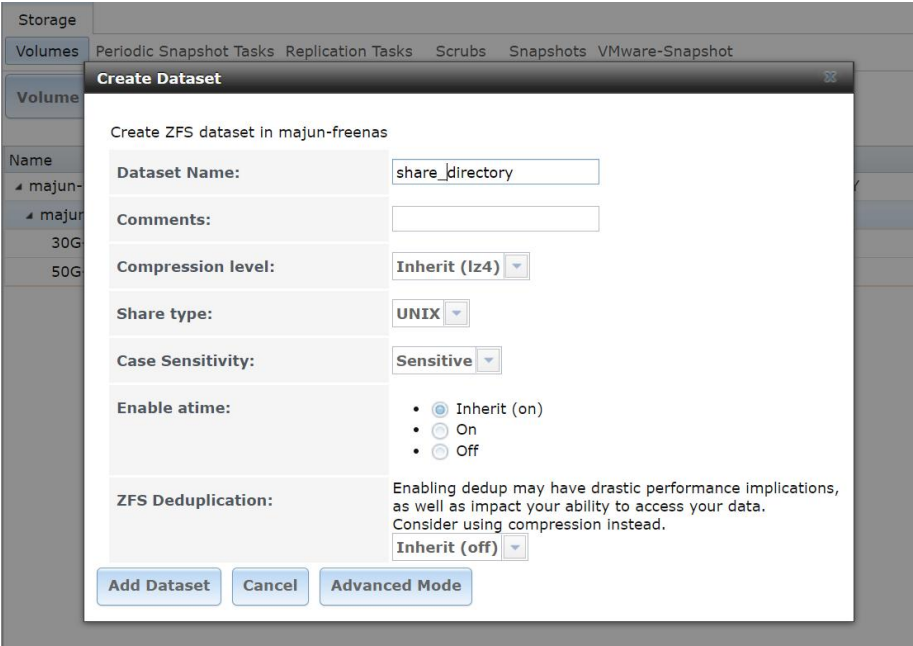


图 3-112 创建数据集（即文件共享路径）

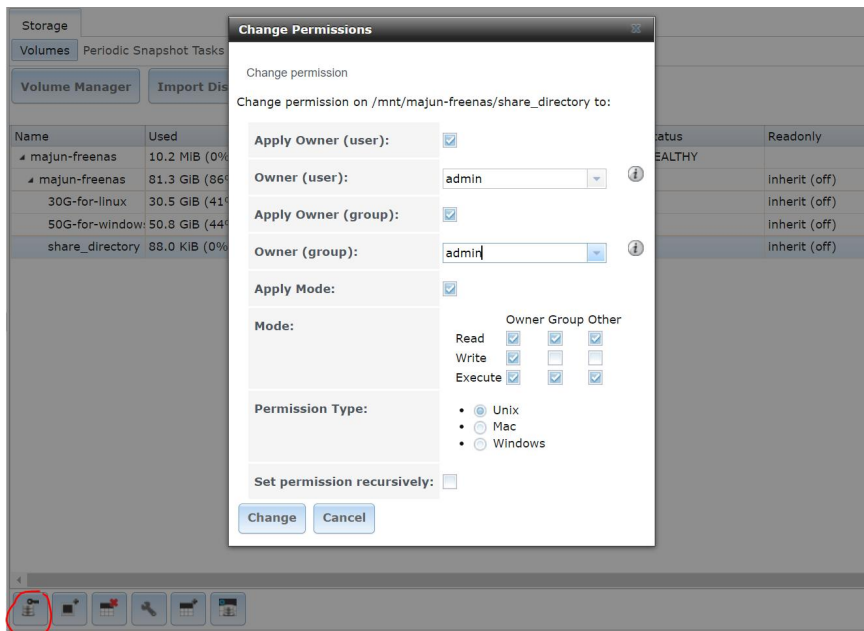


图 3-113 管理用户和用户组

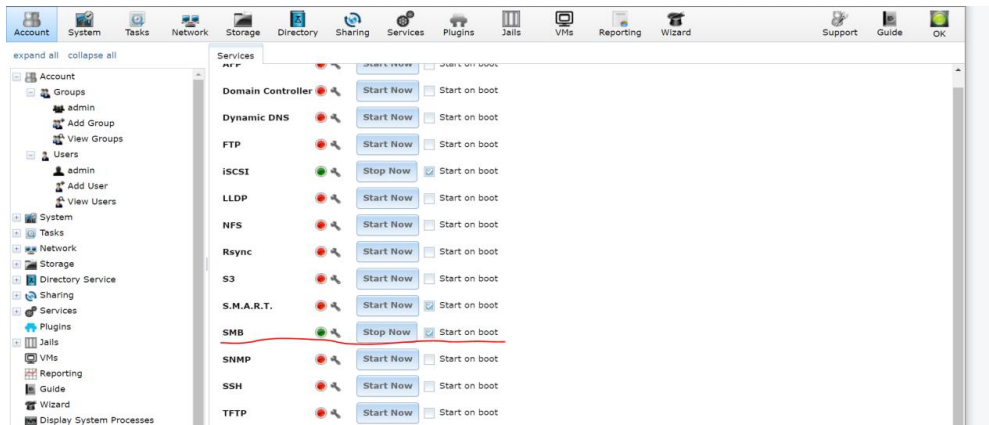


图 3-114 打开 SMB 服务

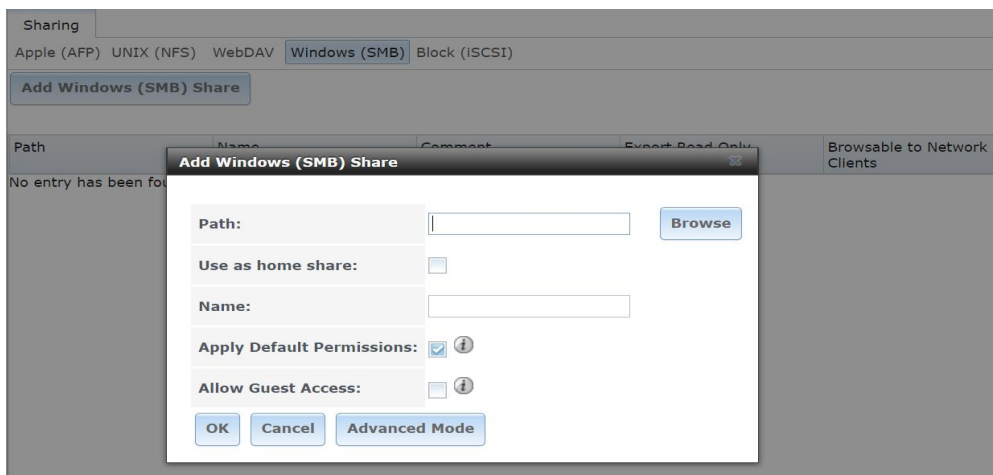


图 3-115 添加 windows (SMB) 共享

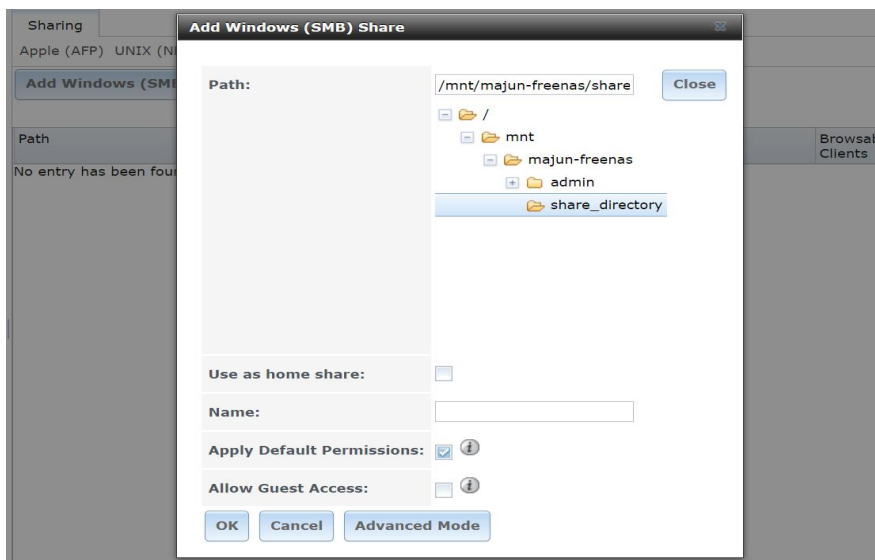


图 3-116 选择共享路径

Sharing				
Apple (AFP) UNIX (NFS) WebDAV Windows (SMB) Block (iSCSI)				
Add Windows (SMB) Share				
Path	Name	Comment	Export Read Only	Browsable to Network Clients
/mnt/majun-freenas/share_directory	share_directory		false	true

图 3-117 添加共享路径成功

7. 打开 windows7 虚拟机，Win+R，打开运行窗口，输入 IP 地址+共享路径，远程连接 freeNAS 提供的存储目录，如图 3-118-图 3-120 所示。

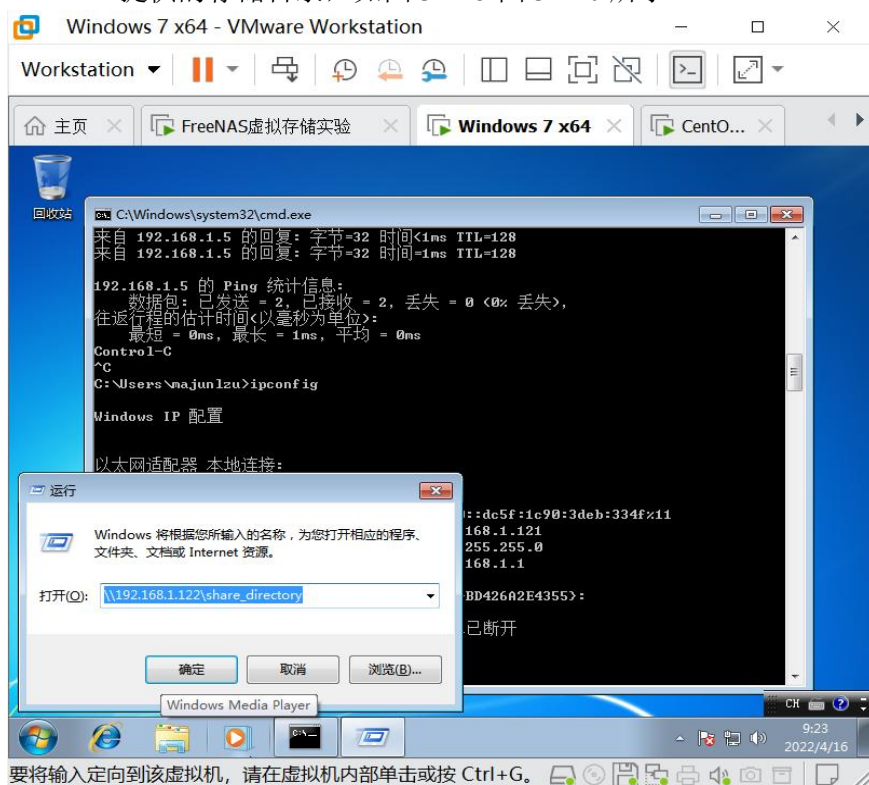


图 3-118 远程连接

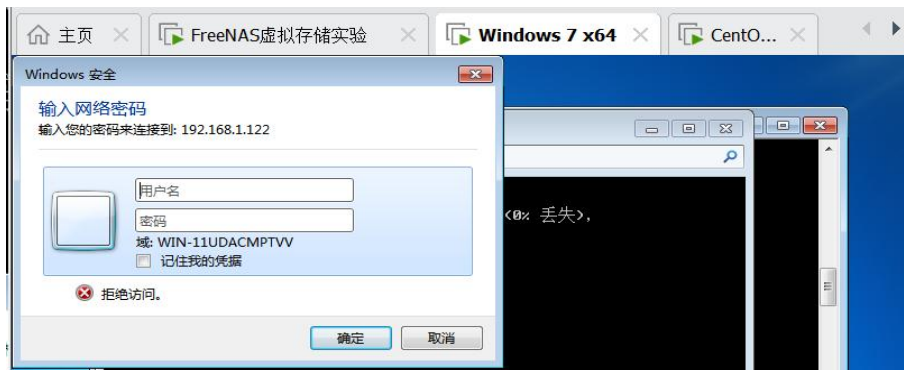


图 3-119 输入用户名和密码

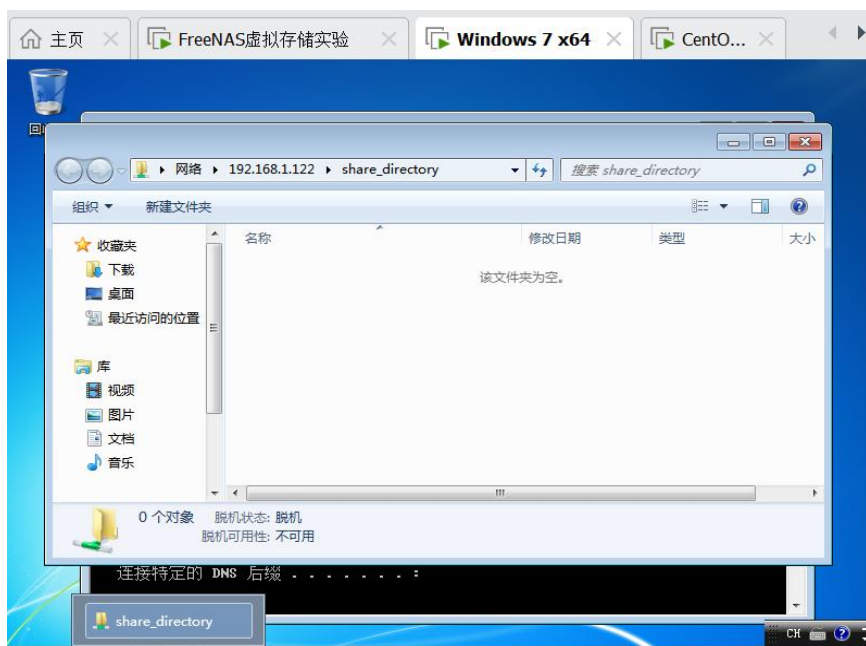


图 3-120 Windows 中连接成功

8. 在 windows 中建立磁盘映射, 如图 3-121 所示。

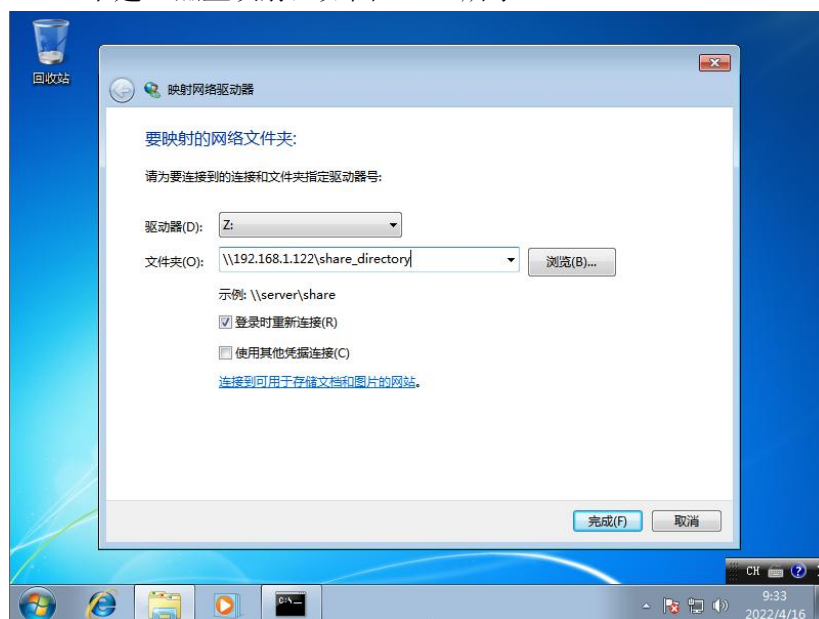


图 3-121 Windows 中建立磁盘映射

(二) NFS 共享存储实验

网络文件系统 Network File System(NFS)，是由 SUN 公司研制的 UNIX 表示层协议 (presentation layer protocol)，能使使用者访问网络上别处的文件就像在使用自己的计算机一样。NFS 是文件系统之上的一个网络抽象，允许远程客户端以与本地文件系统类似的方式通过网络进行访问。虽然 NFS 不是第一个此类系统但它已经发展并演变成 UNIX 系统中最强大最广泛使用的网络文件系统。NFS 允许在多个用户之间共享公共文件系统，并提供数据集中的优势可最小化所需的存储空间，相关资料参看：

<https://baike.sogou.com/v128848.htm?fromTitle=NFS&ch=frombaikev>

1. 在 FreeNAS 中创建用户组和用户，并记录密码信息，此处我们创建 nfs 组和用户，如图 3-122 和图 3-123 所示。
2. 在 Storage 标签下，添加 NFS 数据集，在原来的 freenas 存储池中添加如图 3-124 所示。

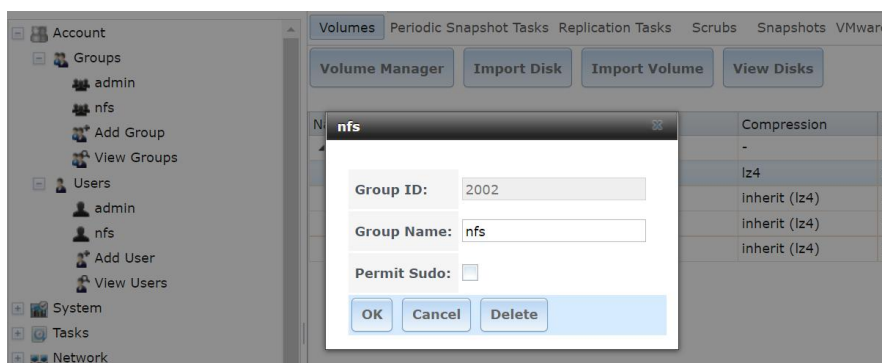


图 3-122 创建用户组 nfs

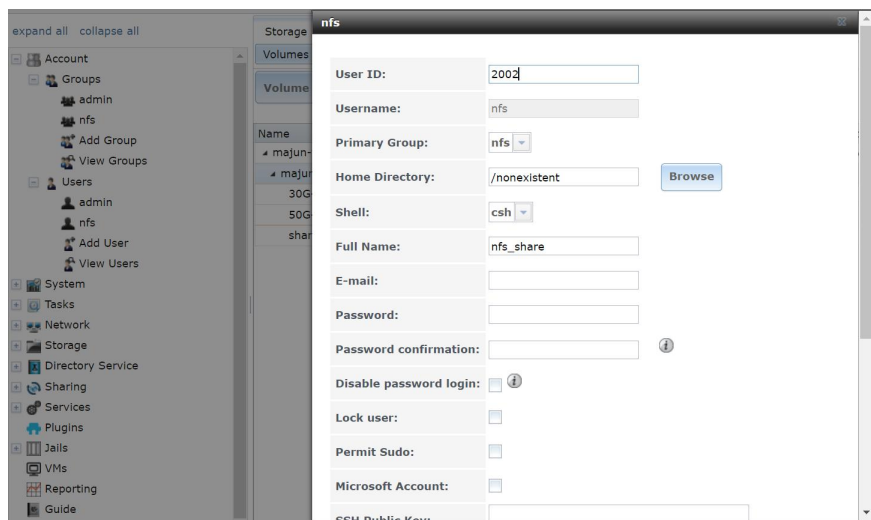


图 3-123 创建用户 nfs

3. 在 sharing 标签下，添加 Unix（NFS）共享，选择共享路径，如图 3-125-图 3-126 所示。
4. 在 services 标签下，启动 NFS 服务，如图 3-127 所示。
5. 在 Windows 中测试使用 NFS 共享服务，首先要在 windows 系统中安装和打开 NFS

客户服务，在程序和功能中打开，如图 3-128-图 3-129 所示。

6. 修改 windows 注册表，使用 win+R 键，打开运行窗口，运行 regedit 命令，打开注册表，如图 3-130 所示。

7. 在“HKEY_LOCAL_MACHINE”-“SOFTWARE”-“Microsoft”-“ClientForNFS”-“Default”下添加两个 DWORD(32-位值)：

(1)AnonymousUid 0

(2)AnonymousGid 0

然后重启计算机，如图 3-131 所示。

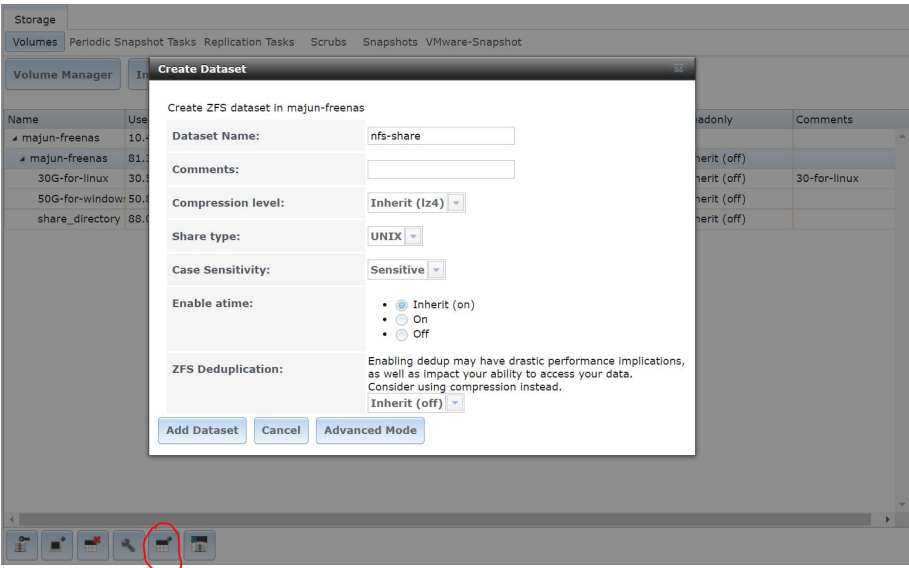


图 3-124 添加 zfs 数据集

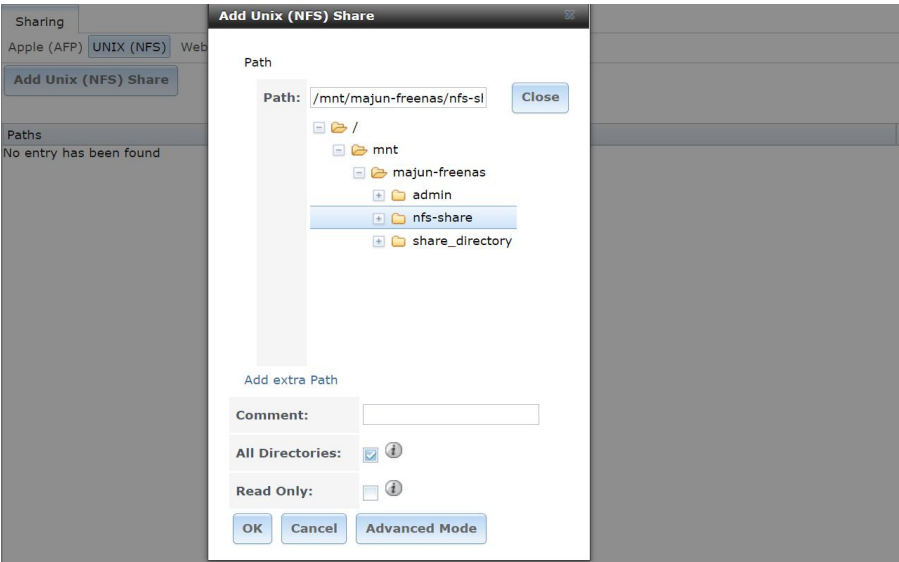


图 3-125 选择共享目录 nfs-share

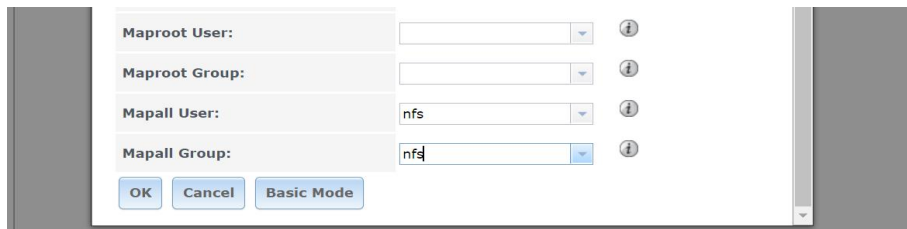


图 3-126 在 advance 模式中选择 mapall user 为 nfs

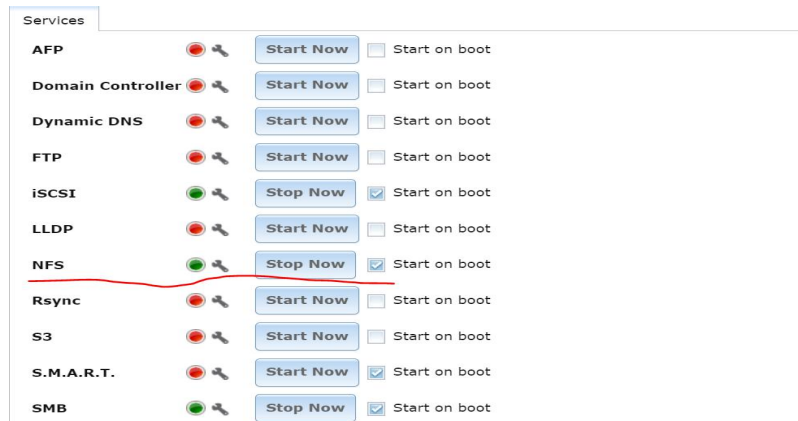


图 3-127 启动 NFS 服务

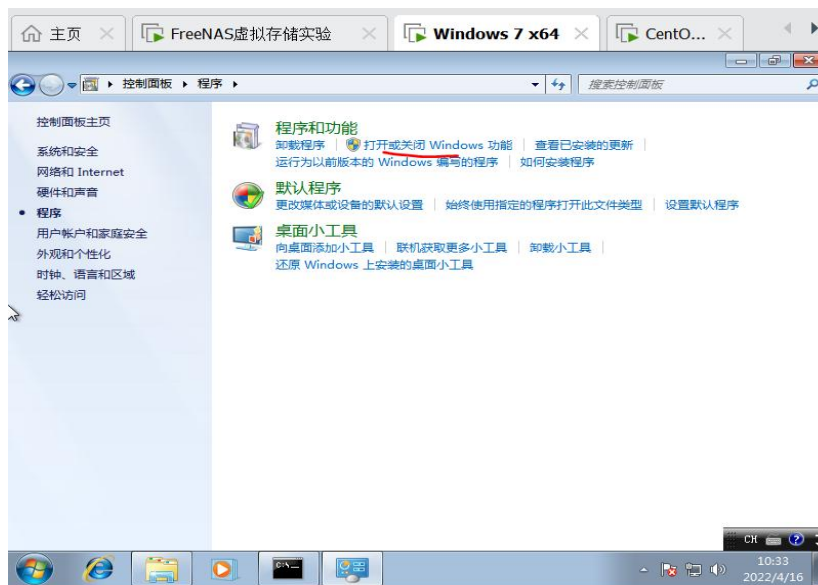


图 3-128 选择打开或关闭 Windows 功能

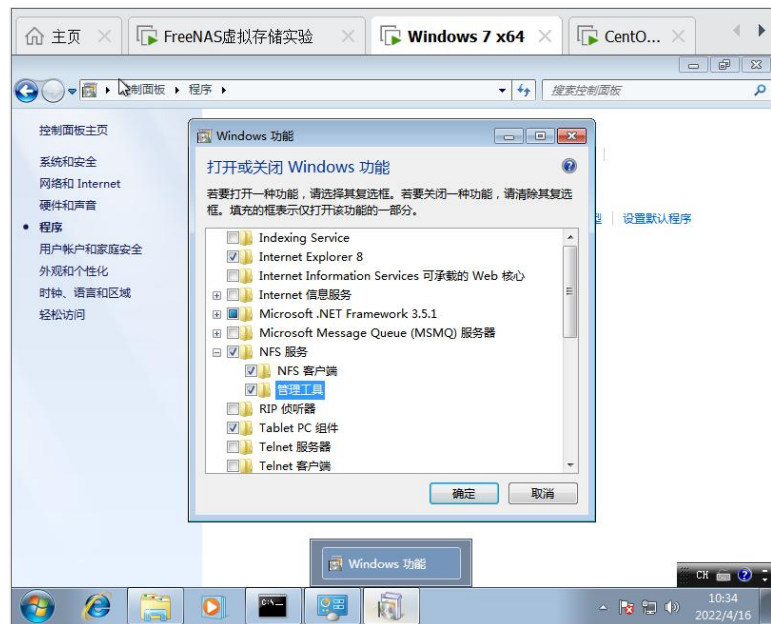


图 3-129 windows 中打开 NFS 服务

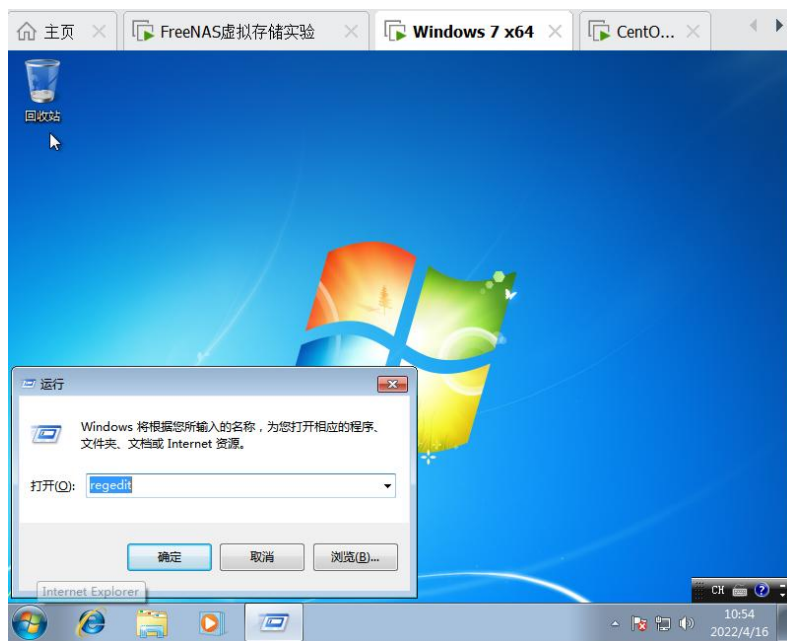


图 3-130 打开注册表

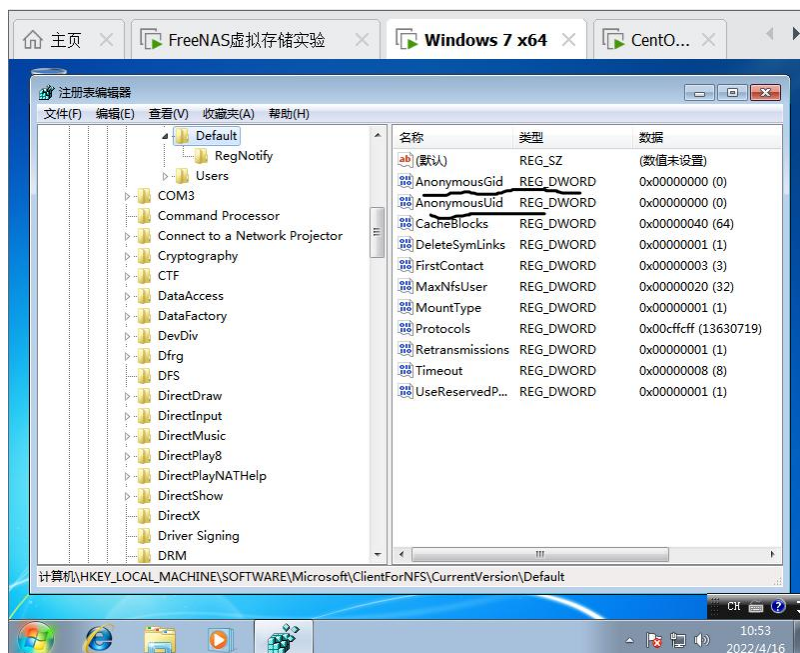


图 3-131 修改注册表添加相应的值

8. 打开一个 CMD 命令窗口，运行以下命令查看 freeNAS 服务器上提供的 NFS 共享路径，如图 3-132 所示。

```
showmount -e 192.168.1.122
```

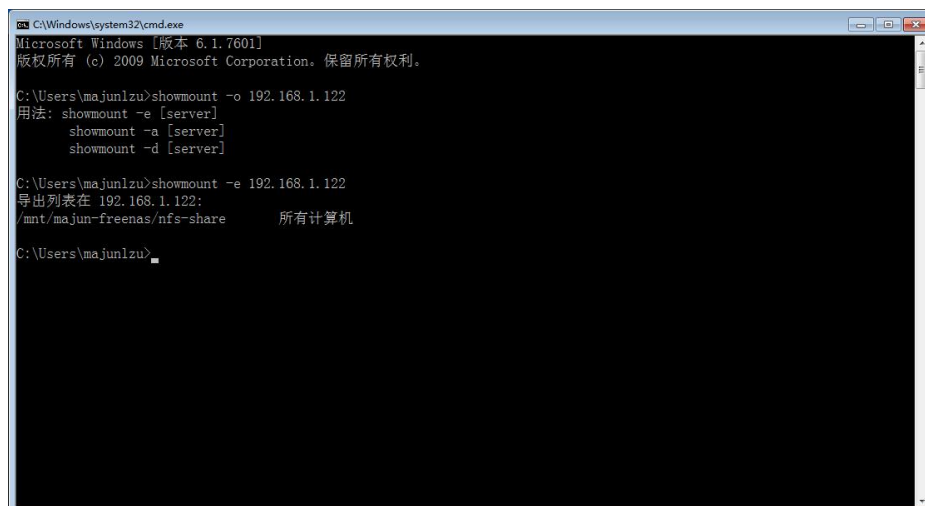


图 4-132 Windows 中查看 freeNAS 上的 NFS 共享路径

9. 使用 mount 命令进行挂载，挂载到 X 盘，如图 4-133 所示。

```
# mount -o nolock -u:用户名 -p:* 主机地址: 路径信息 盘符: \
mount -o nolock -u:nfs -p:nfs1234
192.168.1.122:/mnt/majun-freenas/nfs-share x:\
```

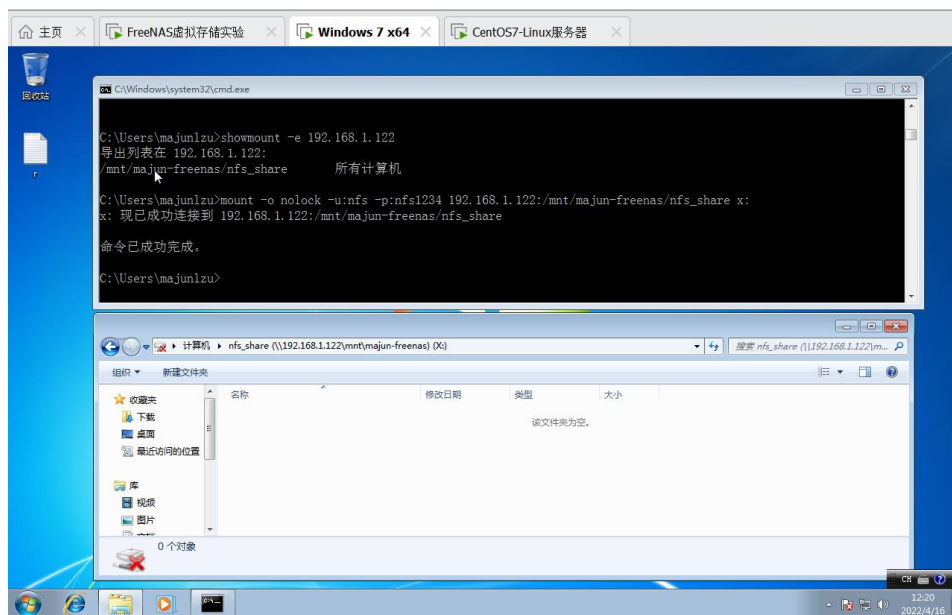


图 3-133 Windows 中挂载共享路径到 X 盘

10. 在 Linux 下测试使用 NFS 共享存储，打开 Centos7 虚拟机，在 terminal 中输入以下命令，挂载 freenas 的共享路径到指定目录，如图 3-134 所示。

```
mkdir /mnt/nfs_share
showmount -e 192.168.1.122
mount -t nfs 192.168.1.122:/mnt/majun-freenas/nfs_share /mnt/nfs_share
```

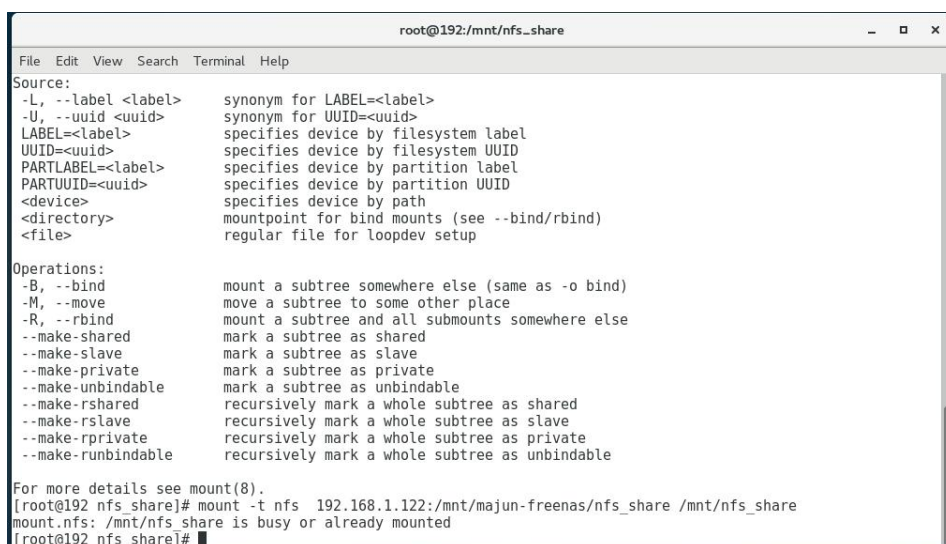


图 3-134 linux 下挂载 freenas 共享路径成功

本阶段实验完成，大家还可以测试其它协议如 ftp 等。