

# Arch Linux VM Setup

## Base VM Needs.

Select guest OS type: Linux and Linux 4.x from the Vsphere dropdowns.

Make sure the install ISO is mounted, and boot the VM.

Once the live terminal is up, determine if the guest VM is efi or not:

```
ls /sys/firmware/efi/efivars
```

If it's an EFI partition, you will see something like this:

```
Arch - VMware Workstation 14 Player (Non-commercial use only)
File Virtual Machine Help

Arch Linux 4.14.15-1-ARCH (tty1)
archiso login: root (automatic login)
root@archiso ~ # ls /sys/firmware/efi/efivars
Boot0000-8be4df61-93ca-11d2-aa0d-00e098032b8c
Boot0001-8be4df61-93ca-11d2-aa0d-00e098032b8c
Boot0002-8be4df61-93ca-11d2-aa0d-00e098032b8c
Boot0003-8be4df61-93ca-11d2-aa0d-00e098032b8c
BootCurrent-8be4df61-93ca-11d2-aa0d-00e098032b8c
BootOptionSupport-8be4df61-93ca-11d2-aa0d-00e098032b8c
BootOrder-8be4df61-93ca-11d2-aa0d-00e098032b8c
ConIn-8be4df61-93ca-11d2-aa0d-00e098032b8c
ConInDev-8be4df61-93ca-11d2-aa0d-00e098032b8c
ConOut-8be4df61-93ca-11d2-aa0d-00e098032b8c
ConOutDev-8be4df61-93ca-11d2-aa0d-00e098032b8c
ConsoleOutMode-793d9786-44dc-4709-b57f-85b8e8fdbfd2
db-d719b2cb-3d3a-4596-a3bc-dad00e67656f
dbDefault-8be4df61-93ca-11d2-aa0d-00e098032b8c
dbx-d719b2cb-3d3a-4596-a3bc-dad00e67656f
dbxDefault-8be4df61-93ca-11d2-aa0d-00e098032b8c
KEK-8be4df61-93ca-11d2-aa0d-00e098032b8c
KEKDefault-8be4df61-93ca-11d2-aa0d-00e098032b8c
Lang-8be4df61-93ca-11d2-aa0d-00e098032b8c
LangCodes-8be4df61-93ca-11d2-aa0d-00e098032b8c
LoaderEntrySelected-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
LoaderFirmwareInfo-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
LoaderFirmwareType-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
LoaderImageIdentifier-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
LoaderInfo-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
LoaderTimeExecUsec-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
LoaderTimeInitUsec-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
LoaderTimeMenuUsec-4a67b082-0a4c-41cf-b6c7-440b29bb8c4f
MemoryOverwriteRequestControl-e20939be-32d4-41be-a150-897f85d49829
MemoryOverwriteRequestControlLock-bb983ccf-151d-40e1-a07b-4a17be168292
MemoryTypeInfo-4c19049f-4137-4dd3-9c10-8b97a83ffdfa
MTC-eb704011-1402-11d3-8e77-00a0c969723b
OsIndicationsSupported-8be4df61-93ca-11d2-aa0d-00e098032b8c
PK-8be4df61-93ca-11d2-aa0d-00e098032b8c
PKDefault-8be4df61-93ca-11d2-aa0d-00e098032b8c
PlatformLang-8be4df61-93ca-11d2-aa0d-00e098032b8c
PlatformLangCodes-8be4df61-93ca-11d2-aa0d-00e098032b8c
SbConfigState-793d9786-44dc-4709-b57f-85b8e8fdbfd2
SecureBoot-8be4df61-93ca-11d2-aa0d-00e098032b8c
SetupMode-8be4df61-93ca-11d2-aa0d-00e098032b8c
SignatureSupport-8be4df61-93ca-11d2-aa0d-00e098032b8c
root@archiso ~ #

To release input, press Ctrl+Alt
```

Make sure your live terminal can reach the internet:

```
ping -c www.google.com
```

Update the datetime:

```
timedatectl set-ntp true
```

## Partitioning your hard disk.

Use this command to see what devices are available:

```
lsblk
```

```
Arch - VMware Workstation 14 Player (Non-commercial use only)
File Virtual Machine Help
root@archiso ~ # lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
loop0 7:0 0 432M 1 loop /run/archiso/sfs/airootfs
sda 8:0 0 8G 0 disk
sr0 11:0 1 539M 0 rom /run/archiso/bootmnt
root@archiso ~ # _
```

To release input, press Ctrl+Alt

Start the partition editor with the hard disk of where you want to install the OS:

```
cfdisk /dev/sda
```

The partition utility UI will show an empty list of partitions.

We will create a boot partition, swap partition, and root partition.

To do so, select `gpt` from the list (if asked) and then partition the hard drive. I like to have 4 separate partitions for an install:

1. Boot partition, with 512MB
2. Swap partition, which can vary. An old rule of thumb is to use 2x the amount of RAM you have, so I chose 1GB
3. Root partition, which can also vary. I chose 3GB, but it is up to you. Depending on what you plan on using the vm for, you might want to go to as much as 1/3 of the disk space that you're using.
4. Home partition, which is just the leftover space.

Make sure that you select the proper filesystem types: Your boot partition should be Microsoft basic data, your swap should be Linux swap, and your home and root partitions should be Linux filesystem. You should have something like this when you're done:

```

Disk: /dev/sda
Size: 8 GiB, 8589934592 bytes, 16777216 sectors
Label: gpt, identifier: 3E83AFB5-48B8-49D4-ABE8-355AA82596A0

Device          Start      End          Sectors      Size Type
>> /dev/sda1    2048      1050623     1048576     512M Microsoft basic data
/dev/sda2      1050624   2097152     2097152      1G Linux swap
/dev/sda3      3147776   16775167    13627392    6.5G Linux filesystem

Partition UUID: A1F963FD-29D3-4DDC-A27B-8090B33874D5
Partition type: Microsoft basic data (EBD0A0A2-B9E5-4433-87C0-68B6B72699C7)
Filesystem UUID: E3A0-0DCB
Filesystem: ufat
Mountpoint: /boot (mounted)

[ Delete ] [ Resize ] [ Quit ] [ Type ] [ Help ] [ Write ] [ Dump ]

Device is currently in use, repartitioning is probably a bad idea.
Quit program without writing changes

```

Don't forget to hit Write before you quit.

Now, type the following to see your filesystem layout:

```
$ lsblk
```

```

[glwhite@archlinux ~]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda         8:0    0   8G  0 disk
├─sda1      8:1    0  512M 0 part /boot
├─sda2      8:2    0    1G  0 part [SWAP]
└─sda3      8:3    0   6.5G 0 part /
sr0        11:0    1 1024M 0 rom
[glwhite@archlinux ~]#

```

## Populate the Partitions.

### Mounting the filesystem Let's mount the filesystem next. Type out everything below, and we'll meet on the other side and see how things went.

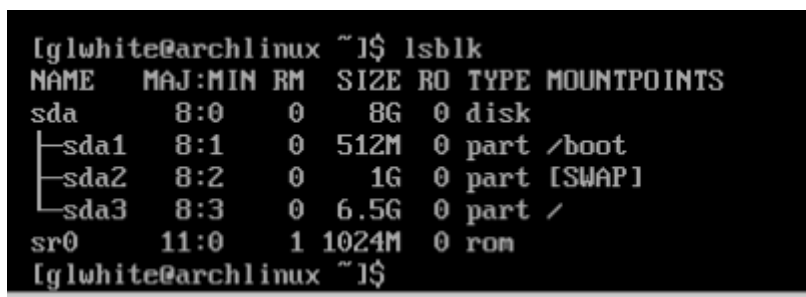
```
$ mkfs.vfat -F32 /dev/sda1
$ mkswap /dev/sda2
$ mkfs.ext4 /dev/sda3
$ swapon /dev/sda2
```

We just set the file system to fat32 for the boot partition, swap on the swap partition, and ext4 for the root partition. After that, we told the computer to use the swap partition as swap.

Now we can actually mount them:

```
$ mount /dev/sda3 /mnt
$ mkdir /mnt/boot
$ mkdir /mnt/home
$ mount /dev/sda1 /mnt/boot
```

If everything went according to plan, you have the skeleton of your Arch install all set up. Now we can get into the actual nuts and bolts of the installation, and the hard part is over with. If we run our trusty `lsblk`, we should see:



```
lglwhite@archlinux ~]# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
sda   8:0    0   8G  0 disk
├─sda1 8:1    0 512M 0 part /boot
├─sda2 8:2    0    1G  0 part [SWAP]
└─sda3 8:3    0 6.5G  0 part /
sr0   11:0   1 1024M 0 rom
```

## Install the OS.

First, sync the repository:

```
pacman -Syy
```

Start the install with the baseline files and packages:

```
pacstrap -i /mnt base base-devel linux linux-firmware vim nano
```

The above will take a bit to install files and packages. It will install vim and nano for editing.

## Configure the OS.

Generate an fstab file that matches our current layout:

```
genfstab -U /mnt >> /mnt/etc/fstab
```

```
GNU nano 7.2 /etc/fstab
# Static information about the filesystems.
# See fstab(5) for details.
# <file system> <dir> <type> <options> <dump> <pass>
# UUID=1b057030-0f18-4ca9-994d-fe2ed43dec38
/dev/sda3 / ext4 rw,relatime 0 1
# UUID=E3A0-0DCB
/dev/sda1 /boot ufat rw,relatime,fnask=0022,dnask=0022,codepage=437,icharset=ascii,shortname=mixed,utf8,errors=remount-ro 0 1
# UUID=48de4b3a-dc1d-4ce9-a107-187beb1607ea
/dev/sda2 none swap defaults 0 0
```

These next steps, we will use chroot to configure the OS before running it.  
Step into the mounted OS, so we can set it up:

```
$ arch-chroot /mnt
```

Setup timezone and the system clock:

```
$ ln -sf /usr/share/zoneinfo/America/New_York /etc/localtime
$ hwclock --systohc.
```

Setup our locale, by uncommenting one, here:

```
$ nano /etc/locale.gen
```

Uncomment one, like this, and save the file:

```
GNU nano 7.2 /etc/locale.gen
#de_LU.UTF-8 UTF-8
#de_LU ISO-8859-1
#de_LU@euro ISO-8859-15
#doi_IN UTF-8
#dsb_DE UTF-8
#dv_MV UTF-8
#dz_BT UTF-8
#el_GR.UTF-8 UTF-8
#el_GR ISO-8859-7
#el_GR@euro ISO-8859-7
#el_CY.UTF-8 UTF-8
#el_CY ISO-8859-7
#en_AG UTF-8
#en_AU.UTF-8 UTF-8
#en_AU ISO-8859-1
#en_BW.UTF-8 UTF-8
#en_BW ISO-8859-1
#en_CA.UTF-8 UTF-8
#en_CA ISO-8859-1
#en_DK.UTF-8 UTF-8
#en_DK ISO-8859-1
#en_GB.UTF-8 UTF-8
#en_GB ISO-8859-1
#en_HK.UTF-8 UTF-8
#en_HK ISO-8859-1
#en_IE.UTF-8 UTF-8
#en_IE ISO-8859-1
#en_IE@euro ISO-8859-15
#en_IL UTF-8
#en_IN UTF-8
#en_NG UTF-8
#en_NZ.UTF-8 UTF-8
#en_NZ ISO-8859-1
#en_PH.UTF-8 UTF-8
#en_PH ISO-8859-1
#en_SC.UTF-8 UTF-8
#en_SG.UTF-8 UTF-8
#en_SG ISO-8859-1
#en_US.UTF-8 UTF-8
#en_US ISO-8859-1
#en_ZA.UTF-8 UTF-8
#en_ZA ISO-8859-1
#en_ZM UTF-8
#en_ZW.UTF-8 UTF-8
#en_ZW ISO-8859-1
#eo UTF-8

[ Read 513 lines ]
^G Help      ^O Write Out  ^W Where Is   ^X Cut        ^T Execute    ^C Location
^X Exit      ^R Read File  ^_ Replace    ^U Paste      ^J Justify    ^_ Go To Line
```

Accept the change, with:

```
locale-gen
```

Set the language variable in /etc/locale.conf:

```
$ echo "LANG=en_US.UTF-8" >> /etc/locale.conf
```

Specify a hostname:

```
echo "somehosename we choose" >> /etc/hostname
```

Update our hosts file:

```
nano /etc/hosts
```

```
GNU nano 7.2
# Static table lookup for hostnames.
# See hosts(5) for details.

127.0.0.1    localhost
127.0.1.1    jumpvm03.localdomain  jumpvm03
```

Make sure to add a localhost and loopback entries, like above.

Add the sudo package, so it can be used later:

```
pacman -S sudo
```

Change the root password with:

```
passwd
```

Add a secondary user:

```
useradd -m secondaryuser
passwd secondaryuser
```

Give the second user some privileges:

```
usermod -aG wheel,audio,video,storage secondaryuser
```

## Install Grub Bootloader.

Make sure you are still inside arch-chroot.

Install packages:

```
pacman -S grub efibootmgr
```

Create the efi folder:

```
mkdir /boot/efi
```

Mount the boot partition into the efi folder:

```
mount /dev/sda1 /boot/efi
```

Install grub:

```
grub-install --target=x86_64-efi --bootloader-id=GRUB --efi-directory=/boot/efi
```

Set grub config:

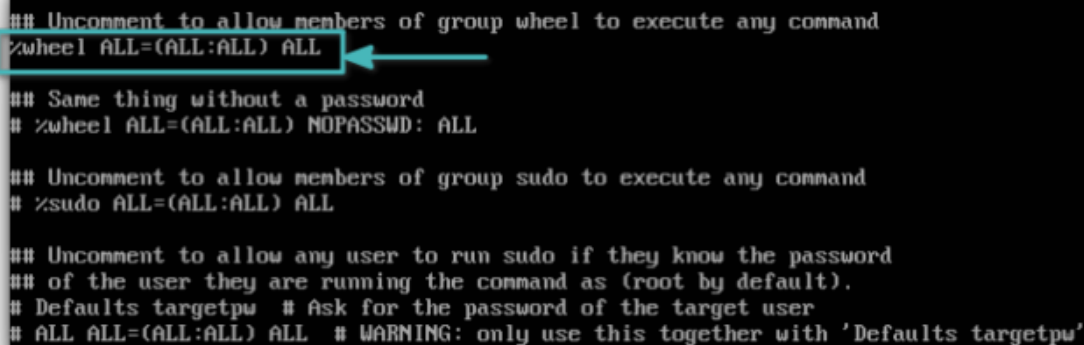
```
grub-mkconfig -o /boot/grub/grub.cfg
```

## Setup Visudo

Tell visudo to use nano:

```
EDITOR=nano visudo
```

Uncomment the wheel group, so it can do anything:



```
## Uncomment to allow members of group wheel to execute any command
%wheel ALL=(ALL:ALL) ALL
## Same thing without a password
# %wheel ALL=(ALL:ALL) NOPASSWD: ALL

## Uncomment to allow members of group sudo to execute any command
# %sudo ALL=(ALL:ALL) ALL

## Uncomment to allow any user to run sudo if they know the password
## of the user they are running the command as (root by default).
# Defaults targetpw # Ask for the password of the target user
# ALL ALL=(ALL:ALL) ALL # WARNING: only use this together with 'Defaults targetpw'
```

Save change, and close it.

## Install a Desktop Environment

The first step is to install the X environment. Type the below command to install the [Xorg as display server](#) along with the network manager. You can refer to the [official documentation](#) for Wayland.

```
pacman -S xorg networkmanager
```

Now, you can install GNOME desktop environment on Arch Linux using:

```
pacman -S gnome
```

Enable the desktop and network manager:

```
systemctl enable gdm.service  
systemctl enable NetworkManager.service
```

## Reboot.

Exit Chroot, with this:

```
exit
```

Unmount the os, so we can restart:

```
umount -R /mnt
```

Restart the machine:

```
sudo reboot
```

## Identify Network Adapter.

Use this to find the name of our network adapter:

```
ip link show
```

Static IP Address

Create a file, here:

```
/etc/systemd/network/20-wired.network
```

Populate it:

```
[Match]  
Name=ens32  
  
[Network]  
Address=10.0.1.2/24  
Gateway=10.0.1.1  
DNS=1.1.1.1
```

```
DNS=1.0.0.1
```

# Enable Network Setting Services.

This first one makes the network settings permanent.

The second one grants name resolution.

```
systemctl enable systemd-networkd.service
systemctl enable systemd-resolved.service
```

# Add OpenSSH.

Install openssh:

```
sudo pacman -S openssh
```

Start the service:

```
sudo systemctl start sshd
```

Enable the service:

```
sudo systemctl enable sshd
```

Add a firewall rule:

```
sudo ufw allow 22/tcp
```

# SSH Problems.

If your ssh server is not accepting connections, ping it from another box.

If ping works, check that the ssh service is listening, with this:

```
sudo nmap -p 22 127.0.0.1
```

---

Revision #4

Created 21 February 2025 08:13:29 by glwhite

Updated 21 February 2025 08:23:51 by glwhite