Posted on March 12, 2016

Building an Arch Linux Vagrant Base Box

I've been using Vagrant at my company for over 6 months now. It was the first time I was exposed to the idea of using virtual machines as a basis for development and I've been enjoying it a lot.

I have been missing one thing though – a really minimal Arch Linux Vagrant base box.

There's a few of them available in Atlas but they most don't have any documentation and the one that's most minimal and most documented, still uses `base-devel` which I wanted to try not to need. Besides this is a good exercise and a good starting point to create ArchLinux-based AMIs so it's worth the effort.

So here we go. The steps we'll go through will be:

1. Start Virtualbox locally with a super minimum ArchLinux installation;
2. Use Vagrant to create a box out of that virtual machine;
3. Upload the Vagrant base box to Atlas.

This assumes then that you have locally installed vagrant and virtualbox. Then:

Get the install media

- Download the installation media for Archlinux;
- Verify it with `md5sum`, `sha1sum` or `gpg`.

Setup a new virtual machine

- Start VirtualBox and create a new virtual machine. You'll have a wizard to go through during which you should setup the following values:
- *Name* – `vagrant-archlinux-64` or something else that seems nice to you;
- *Type* – `Linux`;
- *Version* – Should be automatically filled in;
- *Memory size* – I used 1 gig;
- *Virtual Disk* – Should be of type `VMDK`, dynamically allocated. The size is up to you, I chose 32 gigs;
- After setting it up, you still have to modify some of its settings before you can start the machine:
- *Disable audio*;
- *Disable usb*;
- *Network* – Ensure the Network Adapter 1 is set to `NAT`;
- *Still on Network* – Add a port forwarding scheme for SSH:
    - *Name* – `SSH`;

- *Protocol* – `TCP`;
- *Host IP* – empty;
- *Host Port* – `2222`;
- *Guest IP* – empty;
- *Guest Port* – `22`;

Boot the VM and install Arch Linux

When you start the VM you'll get an option to mount the Arch ISO as the removable media. Do so to boot from it and start the installation process. You'll be greeted with a prompt so let's start `gdisk` and do the partitioning:

```
gdisk /dev/sda
```

There will be no partitions available so we'll create 3 new ones:

- A BIOS boot partition with 1 mb and type `EF02`. This will be `/dev/sda1`;
- A swap partition with 16 gigs (half of the laptop's RAM) with type `8200`. This will be `/dev/sda2`;
- And finally the root partition which will have type `8300` and use the rest of the free space. This will be `/dev/sda3`.

Write the partition table and now let's format those partitions:

```
mkfs.ext4 /dev/sda3
mkswap /dev/sda2
```

And now we mount them:

```
mount /dev/sda3 /mnt
swapon /dev/sda2
```

We'll now install the base system and generate the file system table:

```
pacstrap /mnt base
genfstab -p -L /mnt >> /mnt/etc/fstab
```

We now have a basic Linux system installed. Let's change root and start configuring it!

```
arch-chroot /mnt
bash
echo arch > /etc/hostname
```

For the locale, we `vi /etc/locale.gen` and uncommented whatever locale we want to use and then run:

```
locale-gen
locale > /etc/locale.conf
```

Let's also enable the DHCP service and create the initramfs.

```
systemctl enable dhcpcd.service
mkinitcpio -p linux
```

Almost there. We now install `grub` and configure it:

```
pacman -S grub
grub-install --recheck /dev/sda
grub-mkconfig -o /boot/grub/grub.cfg
```

And finally we prepare our root user. The root password should be `vagrant`. After that we exit bash and the chroot shell, unmount the system and reboot:

```
passwd
exit
exit
umount /mnt
swapoff -a
systemctl reboot
```

Finishing setting up

Our VM is now usable, should we want to just play with Arch Linux in a virtual machine. Since our purpose is to prepare the VM to be used by Vagrant, we still need to setup a few things.

First ensure that you remove the installation medium from VirtualBox (one of the menus) so that you don't keep booting into it.

As I mentioned in the beggining, I really didn't want to have to install anything at all for my Arch Linux base box. I wanted it to be as clean and small and simple as possible. Still, to prepare it to be used as a Vagrant base box, there's a few things we'll need to install so let's start with that now:

```
pacman -S sudo openssh virtualbox-guest-utils-nox linux-header
```

Before creating the `vagrant` user, let's enable the `NOPASS` version of the `wheel` group and setup a few more things in `/etc/passwd`. Do `visudo` and uncomment the `wheel` group line which has `NOPASSWD`. Next ensure the following settings are present:

```
Defaults:vagrant !requiretty
Defaults env_keep = "SSH_AUTH_SOCK"
```

Save and exit and now we create our `vagrant` user and set its password to `vagrant`:

```
useradd --shell /usr/bin/bash --create-home -m -G wheel vagran
passwd vagrant
```

The `vagrant` user should be able to `sudo` without being asked for a password. You can try this out:

```
su vagrant
sudo pwd
```

The last command should work without asking you for password.

Next up is configuring `SSH`. Let's start by editing `/etc/ssh/sshd_config` and ensuring that the following settings are present:

```
Port 22
PubKeyAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys
PermitEmptyPasswords no
PasswordAuthentication no
```

Now we get Vagrant's insecure key and add it to our box:

```
mkdir -p /home/vagrant/.ssh
chmod 0700 /home/vagrant/.ssh
curl -L https://raw.github.com/mitchellh/vagrant/master/keys/v
chmod 0600 /home/vagrant/.ssh/authorized_keys
chown -R vagrant /home/vagrant/.ssh
```

And finally let's make sure that SSH is enabled to start at boot time:

```
systemctl enable sshd.socket
```

Last thing we need to do is ensure we have the Virtualbox Guest tools installed, so that we can use a shared folder between the guest and the host. The dependency is already installed but we have to enable one extra kernel module.

Just `vi /etc/modules-load.d/vboxsf.conf` and add a line with `vboxsf`.

That should be it. Do `systemctl poweroff` and let's now try to create a base box from this virtual machine.

Package the VM, try it and make it public

So now, on your host system, get to an empty directory somewhere and let's package the VM we just created:

```
vagrant package --base vagrant-archlinux-64
vagrant box add ArchLinux64 package.box
vagrant init ArchLinux64
vagrant up
```

Things should go ok enough that at the end you can `vagrant ssh` into your machine and have a running ArchLinux system running. It is the absolute minimum I could come up with and a great starting base for new boxes.

So the final step is just to create a new account in Atlas and upload your new box. The web interface to do so is super simple but you can also use `cURL` to upload it.

After having uploaded it, you can now refer it with `config.vm.box = "your_username/your_base_box"` in any `Vagratfile`.

That's it. :-)

Some Sources

- The ArchLinux Wiki, which no matter what I do or search for it always helps me immensely;
- The Vagrant documentation, namely on creating base boxes and on creating virtualbox boxes;
- An old article by Ryan Skolebnick which did what I did here but for Redhat and Debian.
