**Setting up Virtual Machine for LiveCD.**

Setting up the virtual machine allows for two options. You can start with a prebuilt virtual image or install your own release quickly with the use of Vagrant. Importing an .ovf file is much easier and requires less time. Please go to the appropriate section for instruction.

**From prebuilt .ovf file – import file for Virtualbox**

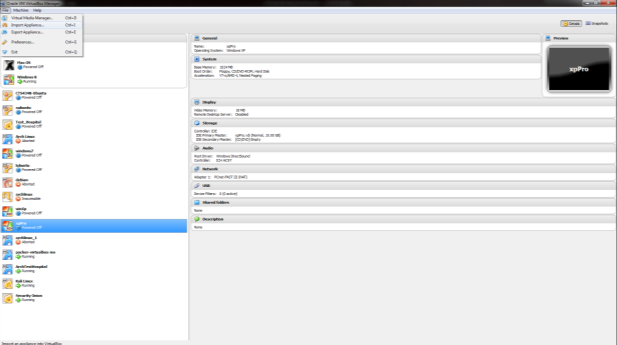
Follow the installation guide for importing from .ovf file.

Downloading the .ovf from Github will be your easiest route. If you would like to personalize settings or customize the environment, follow the **From Scratch Method with Vagrant.**

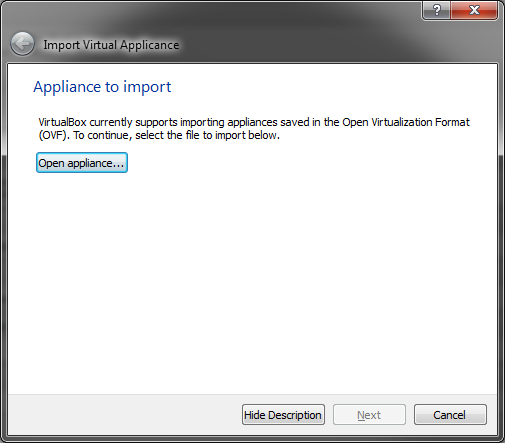
Download the .ovf file here:

Save your .ovf file to a desired location on the host computer.

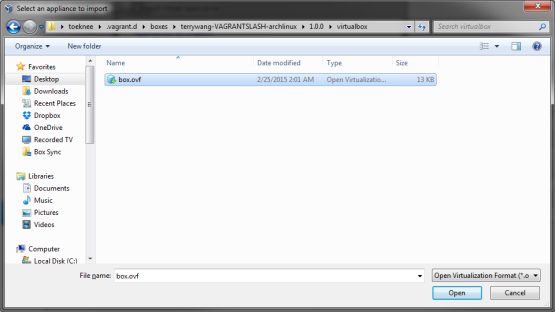
You will then import to your Virtualbox installation. Follow steps and screenshots.



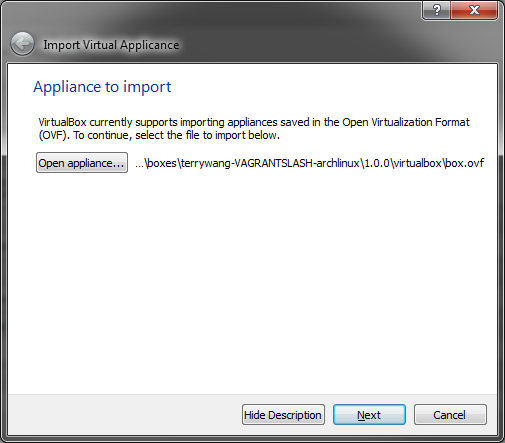
You will see the import virtual appliance wizard in Virtualbox.



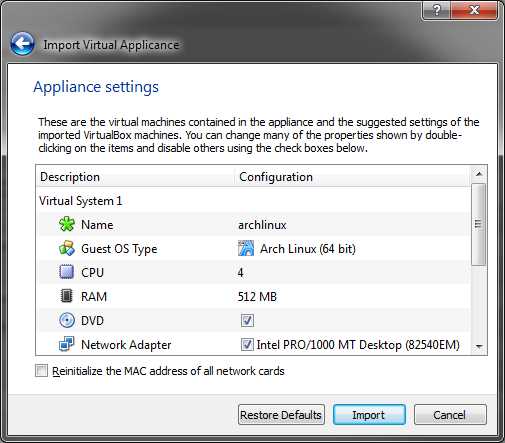
Click Open appliance and navigate to your recently downloaded .ovf image. This is the file format for importing virtual machines to Virtualbox.



Click Next.



Customize settings to the imported virtual machine. Name, memory, number of processors, networking equipment, location for storage of image, etc. I would recommend chosing a VMware image type for this image. This may help to migrate the image later.



Notes:

* Keep in mind that the build image might take up about 40GB.
* The more free space available, the better.
* Many files are copied and moved per each build. It is also a good idea to keep a few past images available as Arch Linux can sometimes have bugs in newly released packages and kernels.
* It is also a good idea to give ample memory and processing power for you build.
  + A script will run and build each image.
  + As the script runs, pacman (arch package manager) will run and download packages for build.
  + With 4 processors and 4GB of ram build time is about 20 minutes.

After importing you should login automatically to a user account. User account will have a limited desktop. Default passwords are listed below. You will have sudo user that can edit passwords and other system files. You will need to take note of the root user as some scripts will require root permissions.

Login: root

Password: vagrant

User:

Password:

**From Scratch with Vagrant**

Follow the installation guide for installing Vagrant.

<http://docs.vagrantup.com/v2/installation/index.html>

Installing Vagrant is extremely easy. Head over to the [downloads page](http://www.vagrantup.com/downloads) and get the appropriate installer or package for your platform. Then install it using standard procedures for your operating system.

The installer will automatically add vagrant to your system path so that it is available in terminals. If it is not found, please try logging out and logging back in to your system (this is particularly necessary sometimes for Windows).

If you have an old version of Vagrant 1.0.x installed via [RubyGems](http://en.wikipedia.org/wiki/RubyGems), please remove it prior to installing a newer version of Vagrant.

[**http://www.vagrantup.com/downloads**](http://www.vagrantup.com/downloads) **- Downloads page.**

**Gem Install?**

Vagrant 1.0.x had the option to be installed as a [RubyGem](http://en.wikipedia.org/wiki/RubyGems). This installation method has been removed for installers and packages only.

**After Installation**

Vagrant is installed, but you need to download an image file.

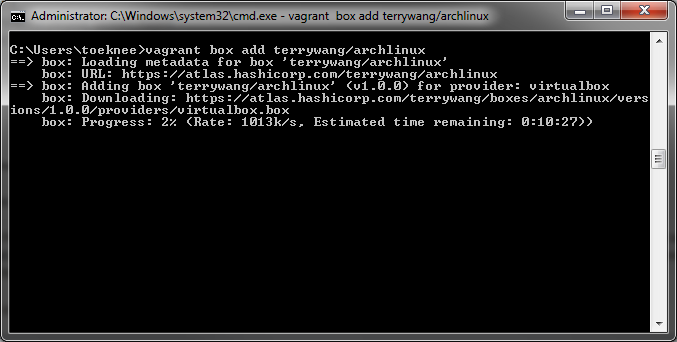
<https://atlas.hashicorp.com/boxes/search>

Vagrant up has a hashicorp public directory of virtual images. Search the directory to find distribution specific builds. All of our build tools are based on Arch Linux and the archiso packages. For the project we used the following images: <https://atlas.hashicorp.com/terrywang/boxes/archlinux>

<https://atlas.hashicorp.com/jfredett/boxes/arch-base>

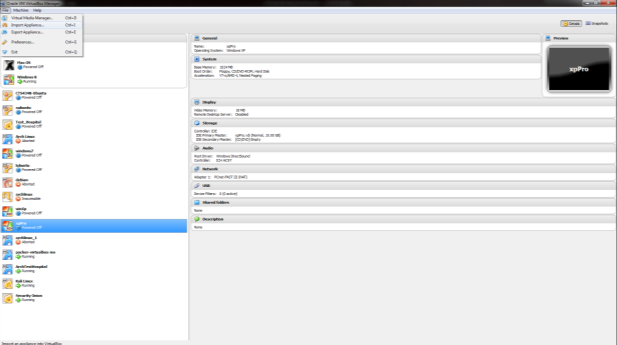
Via the shell run the command: vagrant box add terrywang/archlinux

On a Windows machine, this will download a image file to C:\Users\<Your User Name>\.vagrant.d\boxes\terrywang-VAGRANTSLASH-archlinux\1.0.0\virtualbox

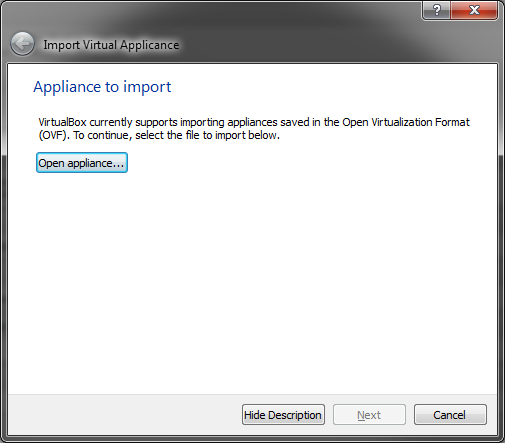


Your now ready to run Virtual box and import your image.

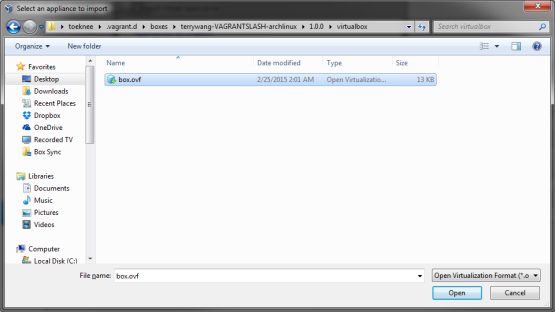
From the file menu, Click File, Import Appliance.



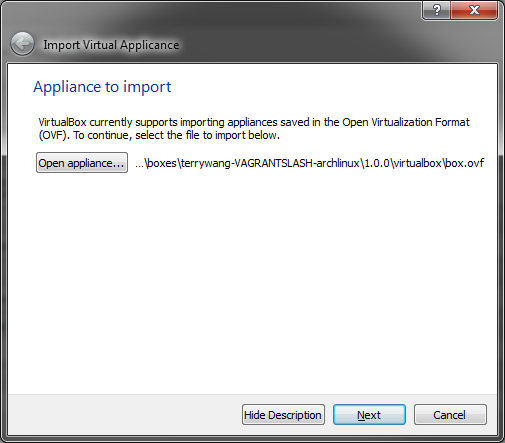
You will see the import virtual appliance wizard in Virtualbox.



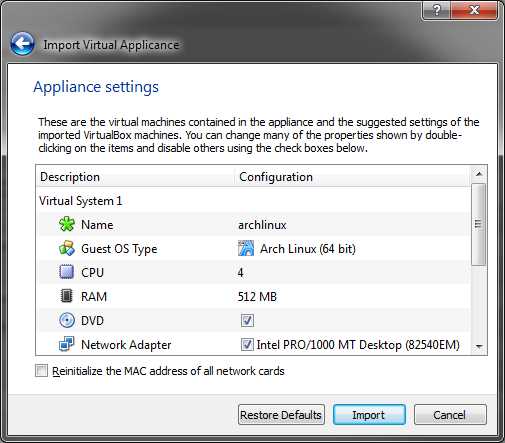
Click Open appliance and navigate to your recently downloaded .ovf image. This is the file format for importing virtual machines to Virtualbox.



Click Next.



Customize settings to the imported virtual machine. Name, memory, number of processors, networking equipment, location for storage of image, etc. I would recommend chosing a VMware image type for this image. This may help to migrate the image later.



Notes:

* Keep in mind that the build image might take up about 40GB. The more free space the better.
* Many files are copied and moved per each build. It is also a good idea to keep a few past images available as Arch Linux can sometimes have bugs in newly released packages and kernels.
* It is also a good idea to give ample memory and processing power for you build.
  + A script will run and build each image.
  + As the script runs, pacman (arch package manager) will run and download packages for build.
  + With 4 processors and 4GB of ram build time is about 20 minutes.

Start your virtual machine and add needed packages.

Default Login on vagrant Images are:

Login: root

Password: vagrant

Be sure to change root password via passwd command.

Depending on the build chosen you may need to install wget

# pacman –S wget

You can download a script with wget for easy installation of needed packages

wget https://raw.githubusercontent.com/Anthonykarn/SeniorProject/master/depends.sh

Shell script will first update packages via pacman and then install packages in .sh list. Feel free to add any packages to the .sh via your favorite editor, vi, vim, nano.

Ensure the script is executable

# chmod +x depends.sh

Run the script

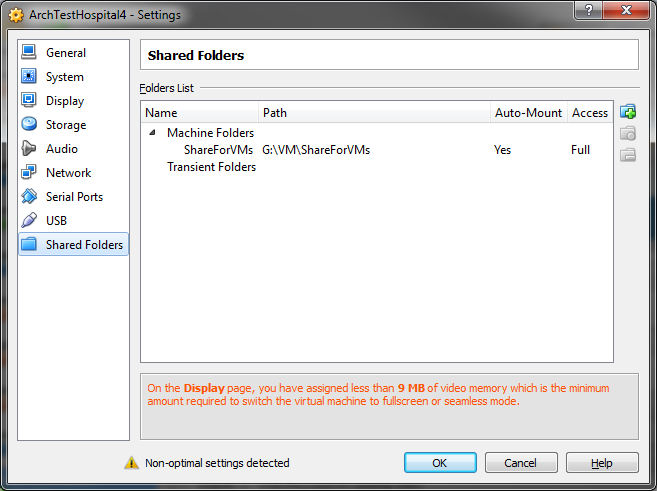
# ./depends.sh

System will update. Get a cup of coffee… System will update package databases. You may be prompted for input, if not sure, select the defaults. System will continue by installing package updates. When finished updating, the system will install needed packages for build environment.

Next you need to create a shared folder in Virtualbox and assign it to your virtual machine.

Share name: ShareForVMs

Click on the machine. Click on Settings from the file menu and window will pop for machine settings. Select Shared folder and assign the share.



Notes:

Using the virtualbox share in this manner will give you a little more control over when items commit to github, when builds occur and when the server runs. As this installation will probably be located on a server somewhere, it makes more sense in your windows shop to use a windows share that can be accessed across the domain. The windows share with build .iso files can be shared across the domain to multiple locations. Basically, this will become your own private github.

In the guest, you will need to mount the share manually.

(You don’t want to add the rule to /etc/fstab/ and copy to the build scripts, could create bot problems.)

mount -t vboxsf ShareForVMs /media/share/