IT Personnel Configuration Guide

Introduction

Building a platform that is suitable for software development and document design takes utilizing many different tools simultaneously. This is a guide for cloning and duplicating a pre-configured virtual machine (VM) that comes equipped with the necessary tools developers will use during their work. Taking advantage of the ease of use of an ISO image, a .iso file has been created that contains a copy of the configured VM's file system and disk drive. This file can then be used by the IT personnel to pre-configure multiple blank VMs at a minimized amount of time.

What is an ISO?

We leverage the compactness and disk copying capabilities of an ISO image. An ISO image, or optical disk image, is essentially an archive file that contains an identical copy of data found in a disk or file system. These .iso files can be used to distribute large programs and even operating systems. We are using the file to easily distribute a copy of our pre-configured virtual machine. Normally, the ISO is put on a USB drive or burned onto a CD or DVD. From there, the image can be mounted in a virtual optical disk drive or it can be extracted and individual files can be accessed.

The act of mounting an ISO on another machine essentially means to be able to access its context as if it was recorded on the physical medium and inserted on the optical drive. It is like having a real disk inserted into your computer. Linux even has native commands that allow the user to mount an ISO with ease.

What is on the ISO Image?

Aside from the Linux terminal, the standard sandbox VM comes with tools such as VIM code editor, and a simple Text Editor NotePad. The Linux terminal already comes with the power to execute Python scripts as well. The default web browser is Firefox.

The new features for software and document developer that were added are:

- Visual Studio Code
- GCC compiler on Linux terminal
- WordPress
- Apache OpenOffice Writer (Word, SpreadSheet, and PDF editors)
- Okular PDF Viewer
- Spotify Desktop

Using the command: `sudo dd if=/dev/sdb of=vm-configured.iso`
I was able to copy the entire contents of my VM (it's disk) with everything I had installed save on the ISO file.

Getting Started

The .iso file contains an exact copy of the Virtual Machine system and files that were pre-configured on the target machine. The downloadable file is available on https://github.com/victoriarivera18/vm-configuration-412. (This file wasn't created due to the lack of resources allocated to the VM. It couldn't handle such a large file being created and would crash. The rest of the instruction is my hypothetical installation methods.)

From here, to replicate this environment on another machine, you can use the following commands:

1. `sudo mkdir /media/my_env`
   (create a directory to mount ISO image, assuming you're in the home directory)
2. `sudo mount /path/to/vm-configured.iso /media/my_env -o loop`
   (mount command to essential extract contents in ISO to given directory)
3. `ls /media/my_env`
   (to view the contents of ISO image just mounted)

How to Scale-Up Configuration?

Using the above steps is fine when the environment only needs to be replicated 1-2 times. However, if we are talking about 100 VMs needing to be pre-configured, doing those steps manually would be inefficient. For you, the IT personnel, to be able to replicate the original VM's environment using the ISO image, there needs to be some form of automation involved.

On a high level, what would happen is the .iso file would be mounted on various VMs after creation. All that would need to happen is to boot the computer off of the USB stick that contains the ISO image. Then, attach the configuration automation to it and let the Chef (automation configuration tool for infrastructure) code configure the rest of it. When the Chef code is done, the machine is installed with the needed media, reboot and a user could log in.

The ISO image would be stored in the USB stick in this case. The automation tool Chef would be used to create the infrastructure for the VM configuration on a large scale. Assuming all the VMs have been created with the appropriate space allocated, each VM should have its own IP address. Through IT Personnel admin access and the known IP addresses of each VM, a Chef automation script can be written to loop through all IP addresses, install the ISO image to each machine (using the commands from the 'Getting Started' section), reboot, and repeat.

This idea comes from the fact that TxCr could access my machine remotely and I only gave them the IP address via email.

Pseudocode (Chef is primarily written in Ruby)

```
VM_list = list of IP address
Target_directory = /media/my_env
ISO = /path/to/vm-configured.iso

For every VM in VM_list:
    Log into VM using admin rights and IP address
    Make Target_directory
    Mount ISO into Target_directory
    Reboot VM
```

Appendix

1. In order for all files to be properly extracted and mounted, there need to be the appropriate resources allocated to each VM
    a. Slow-downs or crashes can occur if this is not the case.
2. The ISO image describe in this paper was not able to be physically created. My VM kept crashing when used for a long period of time or had multiple apps opened. This is also due to ‘noisy neighbors’ issues that caused my access to be limited/slowed.
3. IT personnel have admin rights since they can create the VMs (using TxCr as an example) and they should have the capabilities to access each VM through the Chef code automation.
4. Chef link for more information: https://www.chef.io/products/chef-infra