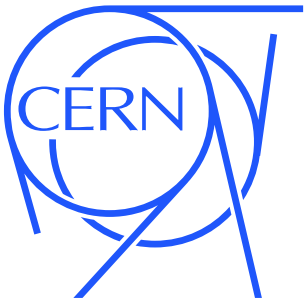

CernVM Release Testing Walkthrough

CernVM Release Testing - Developer Manual

GNU USER

Technical Report
Version: 1.5 (Draft) June 2011

Abstract

The CernVM Release Testing project is a testing infrastructure for CernVM images, the usecase for the project is to provide an automated testing environment, which will install and configure CernVM images, run the set of tests and report the results on a web interface.

Contents

1 Overview

CernVM currently supports images for VirtualBox, VMware, Xen, KVM and Microsoft Hyper-V hypervisors, each new release of a CernVM image needs to be thoroughly tested on each supported platform and hypervisor. The CernVM Release Testing project is designed to meet this requirement by providing an automated testing environment for CernVM images, which will install and configure CernVM images, run the set of tests and report the results on a web interface.

The intent of this document is to provided a step-by-step guide on setting up an entire CernVM Release Testing infrastructure, including instructions on how to set up and configure test clients, the main server running the web interface and database, as well as writing and executing tests. If you are new to release testing and want a document to guide you through the entire process of setting up a working CernVM Release Testing infrastructure, then this guide for you.

All the code needed to setup the entire Release Testing infrastructure for CernVM image testing, is located at the CernVM Release Testing Google Code project page[1] including this document and all other documentation.

While this document is not intended to be a replacement for the reference manual, the following is a brief description of the Release Testing infrastructure including an introduction to the core component, AMD Tapper [2]. Figure 1.1 consists of a diagram outlining the Tapper Architecture, which consists of test clients and a server, the server is what controls the test clients, gathers results, and then displays the results through a web interface.

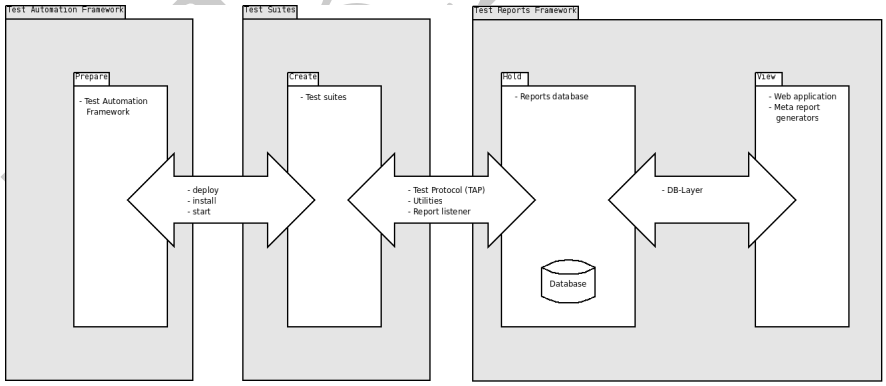

Figure 1.1: Overview of the Tapper architecture

Release Testing component name	Description

Table 1.1: List of Release Testing and AMD Tapper components

The components of the Release Testing Framework are listed in Table 1.1.

2 CernVM Release Testing Test Client Platform Setup

2.1 Introduction

The intent of this document is to provided a step-by-step guide on setting up an entire CernVM Release Testing infrastructure, including instructions on how to set up and configure test clients, the main server running the web interface and database, as well as writing and executing tests. If you are new to release testing and want a document to guide you through the entire process of setting up a working CernVM Release Testing infrastructure, then this guide for you.

This section provides complete step by step instructions on how to setup and configure the test clients which are part of a basic working Release Testing environment by outlining the procedure for setting up test clients on numerous platforms, hence why this is called a *walkthrough* document. This guide is intended for users familiar enough with computers and desktop environments to enter basic commands in a terminal and install various operating systems.

As this guide is directed towards users who are new to CernVM Release Testing and Tapper and are interested in quickly getting a CernVM testing infrastructure quickly set up, many assumptions regarding the requirements necessary are made.As is the case, these instructions are provided for a generalized audience based on our own experience and the requirements that we feel most users will have, *so feel free to deviate from the instructions.*

2.2 Windows 7 Test Client Setup

2.2.1 Configuring the system

1. Now that the Windows 7 installation has completed the first step in configuring the system is to remove unnecessary prompts, which is necessary for setting up a test client. Begin by going to the control panel from `Windows Logo -> Control Panel` and set the "View by:" option to "Small icons" as this will make it easier to find configuration options.

2. Next, begin by disabling User Account Control (UAC)[1] from the control panel select "User Accounts". Then select the option "Change user account control settings" and slide the bar down to the bottom, this turns UAC off, then click "OK" to apply the changes.

3. Next, disable automatic updates to prevent updates conflicting with testing and to provide control over when to apply Windows 7 updates, go to the control panel and select "Windows Update", then select the option "Change settings" and then from the drop down select the option "Never check updates" then click "OK" to apply the changes.

4. Next, disable Windows Defender, which will conflict with testing and cause other issues, begin by going to the control panel and selecting "Windows Defender", then click "Tools", *if for some reason you cannot access the "Tools" option for Windows Defender then ignore this step and continue.* Otherwise, click "Options", under "Administrator options", select or clear the "Use Windows Defender" check box and click "Save" to apply the changes. You may be prompted for an administrator password or confirmation, simply type the password or provide confirmation.

5. Next, disable the Windows Firewall, which will conflict with testing and cause issues such as blocking virtual machine connections and testing services[2]. Begin by navigating to the control panel and select "Windows Firewall", then click the option "Turn Windows Firewall on or off" and for each available firewall configuration, ensure that the option "Notify me when Windows Firewall blocks a new program" is unchecked and that the option "Turn off Windows Firewall (not recommended)" is selected, then click "OK" to apply the changes.

6. Next, disable Action Center prompts which conflict with testing, begin by going to the control panel and selecting "Action Center" then select the option "Change Action Center settings". This will bring up the option to turn messages on or off, **uncheck all of the "Security messages" and "Maintencance messages"**

[1] UAC is the annoying prompt which asks you to authenticate certain actions and must be removed for services such as VNC, SSH, and running tests to work

[2] This may seem like an unwise security decision, but this guide assumes that the test client has been setup in a work environment, which should already have an industry grade firewall that is far superior to Windows Firewall

such as "Windows Update" and "Virus Backup", then click "OK" to apply the changes.

7. Next, disable error reporting and customer experience improvement messages by again going to the control panel and selecting "Action Center" then select the option "Change Action Center settings". This will bring up the previous menu from the last previous step, this time under the section "Related settings" do the following.
   a. Click "Customer Experience Improvement Program settings" and ensure that it is set to "No, I don't want to participate in the program" and then click "Save Changes", you may be prompted for an administrator password or confirmation, simply type the password or provide confirmation.
   b. Next, click "Problem reporting settings" and ensure that it is set to "Never check for solutions (not recommended)" and then click "Change report settings for all users" and ensure that it is set to "Never check for solutions (not recommended)" as well and click "OK". You may be prompted for an administrator password or confirmation, simply type the password or provide confirmation.
8. Now that Windows 7 has been configured to remove unnecessary prompts, the next steps involve performance enhancements and installing and configuring necessary software, begin by going to the control panel and select "Power Options" then select the option "High Performance". Next click the option "Change plan settings" and ensure that the option "Put the computer to sleep" is set to 'Never" then click "Change advanced power settings" and ensure that the following options are set.
   - Disable require password on wakeup
   - Set hard disk to never turn off
   - Ensure that the options "Enable wakeup timers" is enabled
   - Ensure that the option "hibernate" is enabled
   - Ensure that computer never hibernates
   - Allow hybrid sleep
9. Next, disable search indexing[3], begin my clicking `Start -> Computer` and then right click on the "Local Disk (C:)" drive and select "Properties" and then uncheck the option "Allow files on this drive to have contents indexed..." and click "Apply" to apply the changes, it may take several minutes for the indexes to be removed.
10. Next, from the control panel select "Administrative Tools" and click "Services" to launch the Windows services administration tool, then disable the following

[3] Which does have an impact on disk I/O and is important to virtualized testing where you have several virtual machines running using the same physical disk

unnecessary services by double clicking on each entry, setting the "Startup type:" to "Disabled" and clicking "Apply".

- BitLocker Drive Encryption Service
- Bluetooth Support Service
- Remote Registry
- Tablet PC Input Service
- Windows Biometric Services
- Windows Defender
- All Windows Media services
- Windows Search

11. Next, since this is a test client and Windows Aero and other Windows aesthetic effects are not needed, disable graphics effects and enhancements. Begin by going to the control panel and select "Performance Information and Tools" and select the option "Adjust visual effects", then select the option "Adjust for best performance" and click "Apply" to apply the changes.

12. Next, since some of the necessary drivers for graphics support and other devices may have not been installed automatically begin by updating the system, from the control panel select "Windows Update", then click the button "Check for updates" to check for the latest updates, including driver updates. Select from both the regular and recommended updates available any drivers listed as they are specific to the system, also if you wish to install updates for the system now, select the updates from the list **EXCEPT for update KB971033 which is known to cause issues even with genuine copies of Windows**. Then click "Apply" to apply the updates and restart the system after the updates have completed, if you wish to install the latest Windows 7 Service Pack you will have to repeat the update procedure after the system restarts.

13. Now that the necessary drivers for the system have been installed, and perhaps the updates as well, the next step is to disable Windows 7 from automatically selecting the drivers for the system, which can conflict with the drivers installed by the virtualization hypervisors. Begin by going to the control panel and select "System" then select the option "Advanced system settings" and in then click the "Hardware" tab. Now, click the "Device Installation Settings" button and set only the following two options "No, let me choose what to do" and "Never install driver software from Windows Update" then click "Save Changes". You may be prompted for an administrator password or confirmation, simply type the password or provide confirmation.

14. Next, since not all versions of Windows 7 include remote desktop support, TightVNC Server will be used instead, begin by navigating to control panel and select "System" then the option "Remote settings" ensure that the option

"Allow Remote Assistance connections to this computer" is disabled and that the option "Don't allow connections to this computer" then click "Apply".

15. Now, download the latest version of TightVNC for Windows from the following location:http://www.tightvnc.com/download.php and ensure that you download the "Self-installing package". Next, execute the installer and simply click "Next" and agree to the license agreement, then at the "Choose Components" stage of the installation select only "TightVNC Server" from the list and click "Next" until you get to the "Select Additional Tasks" stage of the installer and ensure that the option "Set passwords for the service before finishing the installation" is disabled.

16. Finally, configure Windows 7 to login automatically on start, begin by click the Windows logo and typing "run" and then press enter in the search box, next in the "Run" dialog box, type in **control userpasswords2** and press enter. This will display the "User Accounts" window, uncheck the option "Users must enter a user name and password to use this computer" and click "OK". You will then be prompted to enter the current password and confirm it, after doing so, you will no longer be prompted to enter your password to login on the system.

17. Now that the system has finally been configured for testing[4] reboot the system and ensure that the following work.
    - It automatically boots up into the full desktop environment without having to login
    - You have VNC access to the machine and can control the system using VNC

[4] Hey, it's Windows, you didn't expect this to be a cakewalk did you?

2.3 Red Hat Based Test Client Setup

2.3.1 Configuring the system

1. After the system has booted remove the follow unnecessary startup applications by selecting from the menu `System -> Preferences -> Startup Applications`
   - bluetooth
   - evolution alarm
   - Gnome Login Sound
   - PackageKit Update Applet
   - print queue
   - screensaver
   - visual assistance/aid
   - volume control
   - any others you think are unnecessary based on your own discretion
2. Next enable and configure the remote desktop from the menu `System -> Preferences -> Remote Deskto` and ensure that the following options are configured
   - Enable the option "Allow others to view your desktop"
   - Enable the option "Allow other users to control your desktop"
   - Disable the option "You must confirm access to this machine"
3. Next enable SSH access to the machine, in order for SSH and VNC access to work the firewall will have to be disabled
   a. First disable the firewall from the menu `System -> Adminisration -> Firewall` and click the "Disable" button and then click "Apply" to apply the changes! **This is a quick solution for now because it's too much work to configure the firewall for VNC, SSH, Apache, MySQL, PHPMyAdmin, MCP, and all the other network daemons and should not be a problem if this is just being accessed internally**.
   b. Now that the firewall is disabled, configure sshd, the ssh daemon, to run on startup

      ```
      $ su -c "chkconfig --level 345 sshd on"
      ```

4. Next, configure the system to login automatically at boot
   a. Edit the login screen configuration file for gdm using the following command

      ```
      $ su -c "gedit /etc/gdm/custom.conf"
      ```

   b. Then in the custom.conf file, put the following under the heading [daemon], which will automatically log the system in as the user you created, *make sure you replace the user cernvm with the user that you created.*

Listing 2.1: Configure Automatic Login

```
AutomaticLoginEnable=true
AutomaticLogin=cernvm
TimedLoginEnable=true
TimedLogin=cernvm
TimedLoginDelay=0
```

5. Next, configure the screen saver from the menu `System -> Preferences -> Screensaver` and ensure that the following options are configured
   - Disable the option "Lock screen when screensaver active"
6. Now, reboot the machine, and ensure that the following work
   - It automatically boots up into the full desktop environment without having to login
   - You have access to the machine using SSH and can login on the root account
   - You have VNC access to the machine and can control the system using VNC
7. Finally, update the system from the menu `System -> Administration -> Software Update` and after it has completed the updates reboot the system

2.3.2 Installing libvirt and virsh

1. The virtualization API libvirt and the command line tool virsh [3] are the essential components required for setting up a test client and must be installed and properly configured before any testing can begin. Ensure that you follow the proceeding directions carefully and validate that virsh is working properly before proceeding to install and configure the various hypervisors.

2. First, begin by reviewing the release news listed on the libvirt website, http://libvirt.org/news.htm and read through the release notes for the latest version released to make sure that there are no regressions or deprecated support for the platforms you wish to support. If you intend to set up an entire infrastructure and support all of the CernVM virtualization platforms, which would include *Xen, KVM, VirtualBox, and VMware*, then you must download a version later than *0.8.7* as there was no support for VMware prior to that release.

3. Next, download the latest release that is a **src.rpm** file from the libvirt release server, http://libvirt.org/sources/ based on the latest release which does not have any regressions or deprecations for the virtualization platforms you wish to support [5]. As of this date, the latest release of libvirt is version 0.9.2, this is the release that will be used for the following instructions and examples.

4. Next, install the following dependencies which are required to generate the libvirt rpm files from the src.rpm file that was downloaded, *from now on execute all commands as root.*

Listing 2.2: Install src.rpm Dependencies

```
# Change to root account, enter password if prompted
$ su

# Install dependencies for using a src.rpm
$ yum install rpm rpm-devel rpm-libs rpmdevtools rpm-python \
rpm-build rpmrebuild
```

5. Next, install the following dependencies which are required to install libvirt.

Listing 2.3: Install libvirt Dependencies

```
# Change to root account, enter password if prompted
$ su

# Install dependencies for libvirt
$ yum install xhtml1-dtds readline-devel ncurses-devel gettext augeas \
libpciaccess-devel yajl-devel libpcap-devel avahi-devel radvd \
cyrus-sasl-devel parted-devel libcap-ng-devel libssh2-devel \
```

[5] This shouldn't be an issue but just in case there is a newer version in which Xen support is deprecated, then you would need to use the last release which has Xen support

```
audit-libs-devel systemtap-sdt-devel gnutls-utils gnutls-devel \
python-devel xen-devel libudev-devel libnl-devel device-mapper-devel \
numactl-devel netcf-devel libcurl-devel libcgroup
```

6. Next, create the libvirt RPM installation files using the following command, replace the src.rpm file shown in the example with the file you downloaded previously.

Listing 2.4: Create libvirt RPM Files

```
# Change to root account, enter password if prompted
$ su

$ rpmbuild --rebuild libvirt-0.9.2-1.fc14.src.rpm
```

7. Finally, install the libvirt RPM files by navigating to the `/root/rpmbuild/RPMS` folder and then changing to the directory for your computer architecture *such as x86_64*. Then install the files in the same order as shown in the example, replacing the version of src.rpm file shown in the example with the version of the files in your directory, most importantly install the files in the following order: *libvirt-client, libvirt-devel, libvirt-python, libvirt.* Finally, if a package does not install or complains about conflicts use the *–force* argument to force the installation.

Listing 2.5: Install libvirt

```
# Change to root account, enter password if prompted
$ su

# Change to location of RPM files
$ cd /root/rpmbuild/RPMS/x86_64/

# Install the files in the following order, if install
# fails or has conflicts use   rpm -iv --force
$ rpm -iv libvirt-client-0.9.2-1.fc13.x86_64.rpm
$ rpm -iv libvirt-devel-0.9.2-1.fc13.x86_64.rpm
$ rpm -iv libvirt-python-0.9.2-1.fc13.x86_64.rpm
$ rpm -iv libvirt-0.9.2-1.fc13.x86_64.rpm
```

8. Finally, start the service libvirtd, and ensure that virsh installed correctly and is running by connecting to the test hypervisor and ensuring that the test virtual machine, named “test” is running.

Listing 2.6: Verify virsh was Installed Properly

```
# Change to root account, enter password if prompted
$ su
```

```
# Verify virsh is working, test should be running
$ service libvirtd start
$ virsh -c test:///default list --all
```

2.3.3 Installing and configuring KVM

1. The first step is to install the KVM hypervisor, start by installing KVM and the other additional packages such as virt-manager, which is a graphical management tool and virt-install, which is a command line interface (CLI) virtual machine creation/installation/configuration tool using the following commands with root privileges. If you receive a message that a package is already installed then simply continue.

Listing 2.7: Installing KVM and Other Related Programs

```
$ yum install virt-manager qemu-kvm python-virtinst virt-viewer
```

2. Next, verify that KVM has been installed properly and that virsh can connect to the KVM hypervisor using the following command, if you are able to connect to the virsh console without any errors then virsh is able to connect to the KVM hypervisor.

Listing 2.8: Verify that virsh can Access KVM

```
$ su
$ virsh -c qemu:///session
```

3. Now, proceed to download and extract the desired KVM virtual machine image onto the system from the CernVM download portal, urlhttp://cernvm.cern.ch/portal/downloads it is recommended that you download the KVM Basic image for your architecture. For this guide the basic image will be downloaded as it is the most practical image for the majority of users.

Listing 2.9: Download and Extract CernVM KVM Basic Image

```
$ wget http://rbuilder.cern.ch/downloadImage?fileId=1719
$ gunzip cernvm*.gz
```

4. Now, to ensure that KVM is properly configured and installed, follow this guide provided on the CernVM website http://cernvm.cern.ch/portal/kvm **except, instead use the following virtual machine definition file to create the virtual machine**. *You will need to change the following XML tags in the configuration file accordingly*
   - `<uuid>` by generating a uuid using the uuid command
   - `<source file>` according to wherever you extracted the cernvm kvm image
   - `<mac address>` not a necessity, but change it to something slightly different

Listing 2.10: Create CernVM KVM Definition File

```
# Install uuid tool and generate uuid
$ su -c "yum install uuid"
$ uuid
```

```
# Create the cernvm.xml definition file and set the XML tags accordingly
$ gedit cernvm.xml

<domain type='kvm'>
  <name>cernvm</name>
  <uuid>b32147e7-9b89-dda9-b15d-53ba5f54f590</uuid>
  <memory>524288</memory>
  <currentMemory>524288</currentMemory>
  <vcpu>1</vcpu>
  <os>
    <type arch='x86_64' machine='pc-0.12'>hvm</type>
    <boot dev='hd'/>
  </os>
  <features>
    <acpi/>
    <apic/>
    <pae/>
  </features>
  <clock offset='utc'/>
  <on_reboot>restart</on_reboot>
  <on_crash>restart</on_crash>
  <devices>
    <emulator>/usr/bin/qemu-kvm</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='raw'/>
      <source file='/home/cernvm/image/cernvm-2.3.0-x86_64.hdd'/>
      <target dev='hda' bus='ide'/>
      <address type='drive' controller='0' bus='0' unit='0'/>
    </disk>
    <controller type='ide' index='0'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x01'
      function='0x1'/>
    </controller>
    <interface type='network'>
      <mac address='52:54:00:ca:d5:d3'/>
      <source network='default'/>
      <target dev='vnet0'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x03'
      function='0x0'/>
    </interface>
    <serial type='pty'>
      <target port='0'/>
    </serial>
    <console type='pty'>
```

```
    <target type='serial' port='0'/>
  </console>
  <input type='mouse' bus='ps2'/>
  <graphics type='vnc' port='-1' autoport='yes'/>
  <video>
    <model type='cirrus' vram='9216' heads='1'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02'
    function='0x0'/>
  </video>
  <memballoon model='virtio'>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x04'
    function='0x0'/>
  </memballoon>
 </devices>
</domain>
```

5. Finally, configure the the virtual machine network to automatically start when the system boots, then create the virual machine and ensure that it can be started and stopped using virsh, *all virsh commands require root privileges, so it's easiest to simply run as as root.*

Listing 2.11: Configure the Virtual Machine and Verify it Works

```
$ su
$ virsh -c qemu:///system
$ net-autostart default
$ define /home/cernvm/image/cernvm.xml

# Verify the virtual machine was added, should be in list
$ list --all

# Verify the virtual machine can be turned on/off
$ start cernvm
# Wait about 2 minutes for the system to boot
$ shutdown cernvm
```

2.3.4 Installing and configuring VirtualBox

1. First, begin by downloading the latest version of VirtualBox from the VirtualBox download page, http://www.virtualbox.org/wiki/Downloads ensure that you select the appropriate Red Hat based distribution, version and architecture for your system. The following instructions for this section of the guide uses VirtualBox 4.0.10 for Fedora 13, AMD64.

2. Next, after downloading the latest version of VirtualBox for your distribution install VirtualBox as the root account using the following command.

```
# Enter the root password when prompted
$ su
$ rpm -iv VirtualBox-*.rpm
```

3. Next, in order to use VirtualBox and have full access to the drivers needed for USB support, ensure that the root account belongs to the group "vboxusers". Begin by navigating to `System -> Users and Groups` and then from the "User Manager" window click the "Groups" tab, under the column "Group Name" for the group "vboxusers" ensure that root is one of the group members. If the root account is not a group member of "vboxusers" highlight the "vboxusers" entry and click the "Properties" button, then enable the root account from the list of users and click "OK" to apply the changes.

4. Due to an issue with VirtualBox[6], in order for it to work with virsh the virtual machine(s) must be created and configured as the root account, otherwise when you try to connect or start a VirtualBox virtual machine with virsh you will get an "unknown error", which is obviously very vague and difficult to resolve. **Therefore ALWAYS start VirtualBox as the root account using the following procedure**.

Listing 2.12: Always Start VirtualBox as Root

```
# Switch to the root account, enter root password
$ su

# Start VirtualBox as root
$ virtualbox
```

5. Next, to verify that VirtualBox has been installed properly and that virsh can connect to the VirtualBox hypervisor, verify that the VirtualBox module, *vboxdrv* has been loaded and that you are able to connect to the virsh console without any errors.

[6] The issues is that VirtualBox looks for virtual machine configuration files (*.vbox) in the "VirtualBox VMs" folder of the user that launched VirtualBox. The issue is worsened by the fact that there can only be one "VirtualBox VMs" folder which causes conflicts with multiple users.

Listing 2.13: Verify that virsh can Access VirtualBox

```
$ su

# Verify that the vboxdrv module is loaded
$ lsmod | grep -i vboxdrv

# Verify that virsh can connect to virtualbox
$ virsh -c vbox:///session
```

6. Now, proceed to download and extract the desired VirtualBox virtual machine image onto the system from the CernVM download portal, urlhttp://cernvm.cern.ch/portal/downloads it is recommended that you download the VirtualBox Desktop image for your architecture. For this guide the Desktop image will be downloaded as it is the most practical image for the majority of users.

   Listing 2.14: Download and Extract CernVM VirtualBox Desktop Image

   ```
   $ wget http://rbuilder.cern.ch/downloadImage?fileId=1711
   $ gunzip cernvm*.gz
   ```

7. Now, to ensure that VirtualBox is properly configured and installed, follow this guide provided on the CernVM website http://cernvm.cern.ch/portal/vbinstallation which provides step by step instructions.**Again, ensure that you ALWAYS start VirtualBox as the root account**.

8. Furthermore the following VirtualBox options must also be applied, as they are specific to configuring a CernVM test client.

   a. Navigate to `Settings -> System -> Motherboard` for boot order list, floppy disks are redundant nowadays so disable floppy from the boot order list. Then, disable the option "Enable absolute pointing device", which is for tablets.

   b. Next, go to the option `System -> Processor`, and ensure that the option "Enable PAE/NX" is disabled as it best left disabled for CernVM images [7].

   c. Now, go to the next option `System -> Acceleration`, if your system supports VT-x or AMD-V it is **HIGHLY** recommended that you enable these options "Enable Nested Paging" and "Enable VT-x/AMD-V" for performance gains. To verify that your system supports this execute the following command, the output should not be empty.

   Listing 2.15: Download and Extract CernVM VirtualBox Desktop Image

   ```
   egrep '(vmx|svm)' --color=always /proc/cpuinfo
   ```

[7] Physical Address Extension (PAE) is useful if you are running a 32-bit processor as this enables a 32-bit operating system to access to more than 4GB of memory

d. Next go the setting for "Audio", audio is mostly redundant and unnecessary for CernVM , so disable the audio unless you explicitly require audio support.

e. Finally, since virsh only supports console access through a serial port for LXC, Xen, QEMU/KVM, and UML, go to the setting "Serial Ports" and simply verify that all of the serial ports as disabled for VirtualBox.

9. Now that the virtual machine has been created and configured verify that it is able to boot completely without crashing, *you will be presented with a login screen when it has booted completely.* Then shutdown the virtual machine by clicking "Actions" from within the virtual machine and selecting 'Shutdown", after the virtual machine has shutdown close VirtualBox and then connect to the VirtualBox hypervisor and determine that you can view, start, and stop virtual machine.

Listing 2.16: Verify VirtualBox Works with Virsh

```
$ su
$ virsh --connect vbox:///session

# Verify the virtualbox virtual machine is accessible
# Name of the virtual machine created should be listed
$ list --all

# Verify the virtual machine can be turned on/off
$ start <name of virtual machine>
# Wait about 2 minutes for the system to boot
$ shutdown <name of virtual machine>
```

10. Finally, configure the the virtual machine network to automatically start when the system boots and then create an XML definition file of the virtual machine, which will be used later by the test scripts. In the following example save the XML definition file with the same name as the virtual machine that was created, such as *cernvm-vbox-2.3.0.xml* so that it is easy to differentiate between multiple virtual machines for different hypervisors.

Listing 2.17: Create XML Definition File and Configure Network

```
$ su

# Create virtual machine XML definition file
$ virsh --connect vbox:///session dumpxml <name of virtual machine> \
> <name of virtual machine>.xml

# Configure virsh network for VirtualBox
$ virsh --connect vbox:///session
$ net-start vboxnet0
```

```
$ net-autostart vboxnet0
```

2.3.5 Configuring the CernVM Image

1. The next steps involve configuring the CernVM image to integrate with virsh as well as the test suite, first start the virtual machine, execute the command to get the IP address of the CernVM image. Then follow this guide provided on the CernVM website on how to configure the CernVM image and create a new user using the web interface http://cernvm.cern.ch/portal/cvmconfiguration and reboot the system, *all virsh commands require root privileges.*

```
$ virsh start cernvm
# Wait about 2 minutes for the system to boot
# Then get IP Address using the following command
$ arp -an
```

2. Now that the system has booted, login in to the system using SSH for the user you created using the CernVM web interface

```
$ ssh <user you created>@cernvm-image-ipaddress
```

3. Now that you are logged into the system through SSH, execute the following instructions to enable virsh console access.

Listing 2.18: Enable Virsh Console Access

```
# Type the following command, and enter a root password you won't forget
$ sudo passwd root
# Change to the root account and enable console for root so that you can
# login using virsh console
$ su

# Enable root login on tty
$ echo    ttyS0    >> /etc/securetty

# Then add console=ttyS0 to the kernel parameter line in /etc/grub.conf
$ vi /etc/grub.conf

# Add getty to /etc/inittab file after all the other "tty" lines,
# add the following line to /etc/inittab
s0:2345:respawn:/sbin/agetty -L 38400 ttyS0 vt100
```

4. Now, reboot and login to the system using virsh console and then simply type "root" and enter the password to login as the root account, *there will most likely not be any console display from the virtual machine until you press enter after entering the password*, then enable root login through SSH using the following commands. **To disconnect from the virsh console and return to the host machine console, press CTRL +] which is (^])**

Listing 2.19: Enable SSH Root Login

```
# Reboot the system and the console should now work
# Wait for it to completely shut off and turn on
$ virsh shutdown cernvm
$ virsh start cernvm

# Login as root using password you set with the passwd command
$ virsh console cernvm

# edit the file /etc/ssh/sshd_config and uncomment the line
# PermitRootLogin yes   in order to enable root login
$ vi /etc/ssh/sshd_config
```

5. Next from the host machine *ie. the machine you're currently using* enable automatic login as root through ssh on the KVM guest, first ensure that the guest machine has been started.

Listing 2.20: Enable Automatic SSH Root Login

```
# Restart the virtual machine wait for
# it to completely shut off and turn on
$ virsh shutdown cernvm
$ virsh start cernvm

# Generate a public key, when prompted press enter for everything
$ ssh-keygen -t rsa

# Get the ip address of the running cernvm guest, as done previously
$ arp -an

# Next, run the following command to setup automatic login for ssh
# without having to type the password. When prompted for the password
# enter the password you just set previously
$ ssh-copy-id -i ~/.ssh/id_rsa.pub root@cernvm-image-ipaddress

# Disconnect, and try and login to the machine using ssh and ensure
# that you can login as root without having to type the password
$ ssh root@cernvm-image-ipaddress
```

6. If everything so far has worked, then the test client and CernVM image have been installed and configured properly, if you have any outstanding issues solve them before proceeding further, or go to the section "Server Platform Setup" 3 as the Tapper server does not require virtual machine creation and configuration.

2.3.6 Setting up the Tapper Test Suite

1. **Before proceeding any further ensure that you have all other test clients set up this far, and then proceed to follow the instructions for setting up and configuring the Tapper server in the section "Server Platform Setup"** 3.

2. Now that the Tapper server has been installed and configured and the Tapper web interface and database have proven to be working, the next step is to verify that the test client can actually send a report to the Tapper server in the form of a TAP file. After sending the TAP report to the server, ensure that the test client is working by viewing the tapper reports in your browser at the following url: http://localhost/tapper/reports. You should now see a report from the test client, there should be a report from a system named whatever the "Tapper-Machine-Name" in demo_report.tap was set as. *For the example demo_report.tap provided below it would be cernvm-rhtestclient.* [8].

Listing 2.21: Send a Basic Report to the Tapper Server

```
# Save the following in a file named demo_report.tap
$ vi demo_report.tap

        1..2
        # Tapper-Suite-Name: Tapper-Deployment
        # Tapper-Suite-Version: 1.001
        # Tapper-Machine-Name: cernvm-rhtestclient
        ok - Hello test world
        ok - Just another description

# Send the report to the tapper server using netcat
$ cat demo_report.tap | nc -w10 cernvm-server 7357
```

3. Next, download a copy of tapper-autoreport and the CernVM Test Cases from the Google Code svn repository [1] and install the tapper-autoreport dependencies.

Listing 2.22: Install Tapper AutoReport and Dependencies

```
# Install subversion, required to checkout auto-tapper
$ yum install subversion

# Checkout a copy of auto-tapper and cernvm testcases
$ svn checkout http://cernvm-release-testing.googlecode.com/svn/\
trunk/tapper/tapper-autoreport/ cernvm-release-testing-read-only
```

[8] This is why a consistent hostname convention was emphasized earlier, as reports are often sorted and organized based on hostnames

```
# Install the missing dependencies
$ yum install perl-Module-CoreList
$ yum install perl-CPAN

# Install the required perl modules
$ cpan
$ install prove
$ install XML::XPath
```

4. Now that that tapper-autoreport has been installed, configure the following variables in the script "cernvm-tests.sh" according to your Tapper infrastructure setup.
   - OSNAME - The operating system of the test client, such as "Red Hat 5"
   - VMNAME - The domain name of the virtual machine used earlier with virsh
   - VM_XML_DEFINITION - The location and name of cernvm virtual machine definition XML file used to create virtual machine
   - HOSTNAME - The hostname of the test client
   - GUESTIP - This is the IP address of the CernVM image
   - TAPPER_REPORT_SERVER - The hostname of the Tapper Report Server
5. Finally, now that tapper-autoreport has been installed and configured on the test client and the test client and Tapper Server have proven to be working, the next step is to verify that tapper-autoreport works correctly and can actually send a report to the Tapper server in the form of a TAP file. After the CernVM Test Cases script, "cernvm-tests.sh" has completed and sent a TAP report to the server, ensure that the test client is working by viewing the tapper reports in your browser at the following url: http://localhost/tapper/reports. You should now see new report from the test client, there should be a report from a system with the same hostname[9].

Listing 2.23: Run Tapper -AutoReport for CernVM Test Cases

```
# Simply execute the script and wait for it to finish
./cernvm-tests.sh
```

[9] This is why a consistent hostname convention was emphasized earlier, as reports are often sorted and organized based on hostnames

2.4 Debian Based Test Client Setup

2.4.1 Installing the system

1. Install the system as you would for any other Linux distribution, except pay attention to the following instructions on configuring Debian to use ext4 (if available) instead of ext3 as the performance gains are noticeable.

2. Now, when prompted by the installer to configure the partitioning layout, if there are other operating systems installed on the system select the "Guided - use entire disk" option, and if available select the option "Use Remaining Free Space". Otherwise, if there are no other operating system installed on the hard drive select the "Manual" option, *beware that doing so will risk erasing everything on the hard drive if you create a new partition table*. Using the manual option, create two primary partitions, with the first taking up the size of the hard drive minus twice the size of the amount of RAM installed, and the second primary partition as a SWAP file using the remaining free space. The following is an example of what the partiton layout would look like for a 40.0 GB hard drive with 2GB of ram.

Listing 2.24: Manual Partition Layout Example

```
#1      PRIMARY  36.0  GB      B  f      EXT4    /
#2      PRIMARY  4.0   GB         f      SWAP    SWAP
```

3. Finally, the last important installation setting, when prompted to choose software to install, select the following
   - Graphical desktop environment
   - SSH Server
   - Standard system utilities

2.4.2 Configuring the system

1. After the system has booted remove the follow unnecessary startup applications by selecting from the menu
   `System -> Preferences -> Startup Applications`
   - bluetooth
   - evolution alarm
   - Gnome Login Sound
   - print queue
   - screensaver
   - update notifier
   - visual assistance/aid
   - volume control
   - any others you think are unnecessary based on your own discretion
2. Remove the follow unnecessary services by selecting from the menu
   `System -> Administration -> Services`
   - alsa utils
   - bluetooth
   - CUPS
   - exim4
   - any others you think are unnecessary based on your own discretion
3. Next enable and configure remote desktop from the menu
   `System -> Preferences -> Remote Desktop` and ensure that the following options are configured
   - Enable the option "Allow others to view your desktop"
   - Enable the option "Allow other users to control your desktop"
   - Disable the option "You must confirm access to this machine"
4. Next configure the system to login automatically at boot from the menu select `System -> Administration -> Login Screen` and then set it to login to the user account you created previously (such as cernvm) automatically.
5. Next, remove cd-rom support from sources.list, which is used by Debian for updates [10], execute the following command with root privileges and comment out any lines that start with "deb cdrom" by using a #

[10] And is a nuisance for any new user as it forces you to find the CD and put it in the computer for the update to continue

Listing 2.25: Removing CD-ROM Requirement for Updates

```
$ su -c "gedit /etc/apt/sources.list"
```

6. Again, continue to edit /etc/apt/sources.list still with root privileges and ensure that each line ends with "main contrib non-free", then save the file and do the following command with root privileges.

Listing 2.26: Updating the System

```
$ su -c "apt-get update"
```

7. Next, configure the screen saver from the menu
`System -> Preferences -> Screensaver` and ensure that the following options are configured
   - Disable the option "Lock screen when screensaver active"
8. The following instructions involve enabling headless support so that you can remote desktop to the machine without having a monitor connected to the computer
   a. Edit the xorg.conf file and put the following in it

Listing 2.27: Configuring Xorg for Headless Support

```
$ su -c "gedit /etc/X11/xorg.conf"

Section "Device"
Identifier "VNC Device"
Driver "vesa"
EndSection

Section "Screen"
Identifier "VNC Screen"
Device "VNC Device"
Monitor "VNC Monitor"
SubSection "Display"
Modes "1280x1024"
EndSubSection
EndSection

Section "Monitor"
Identifier "VNC Monitor"
HorizSync 30-70
VertRefresh 50-75
EndSection
```

   b. Then edit grub and set the option "nomodeset", and proceed to update grub and reboot

Listing 2.28: Configuring Grub for Headless Support

```
$ su -c "gedit /etc/default/grub"

GRUB_CMDLINE_LINUX="nomodeset"

$ su -c "update-grub"
```

9. Now, reboot the machine, and ensure that the following work
   - It automatically boots up into the full desktop environment without having to login
   - You have access to the machine using SSH and can login on the root account
   - You have VNC access to the machine and can control the system using VNC
10. Finally, update the system from the menu `System -> Administration -> Update Manager` and after it has completed the updates reboot the system

2.4.3 Installing and configuring the hypervisors

1. The first step is to install the hypervisor, start by installing kvm and virsh using the following command with root privileges

Listing 2.29: Installing KVM and Virsh Support

```
$ su -c "apt-get install qemu-kvm libvirt-bin"
```

2. Now, proceed to download and extract the desired kvm virtual machine image onto the system from the CernVM download portal[11], for this guide the basic image will be downloaded as it is the most practical image for the majority of users.

Listing 2.30: Download and Extract CernVM KVM Basic Image

```
$ wget http://rbuilder.cern.ch/downloadImage?fileId=1719
$ gunzip cernvm*.gz
```

3. Now, to ensure that KVM is properly configured and installed, follow this guide provided on the CernVM website http://cernvm.cern.ch/portal/kvm **except, instead use following virtual machine definition file to create the virtual machine**. *You will need to change the following XML tags in the configuration file accordingly.*

   - `<uuid>` by generating a uuid using the uuid command
   - `<source file>` according to wherever you extracted the cernvm kvm image
   - `<mac address>` not a necessity, but change it to something slightly different

Listing 2.31: Create CernVM KVM Definition File

```
# Install uuid tool and generate uuid
$ su -c "apt-get install uuid"
$ uuid

# Create the cernvm.xml definition file and set the XML tags accordingly
$ gedit cernvm.xml

<domain type='kvm'>
  <name>cernvm</name>
  <uuid>b32147e7-9b89-dda9-b15d-53ba5f54f590</uuid>
  <memory>524288</memory>
  <currentMemory>524288</currentMemory>
```

[11] The images can be obtained here: http://cernvm.cern.ch/portal/downloads it is recommended that you download the KVM Basic image for your architecture

```
<vcpu>1</vcpu>
<os>
  <type arch='x86_64' machine='pc-0.12'>hvm</type>
  <boot dev='hd'/>
</os>
<features>
  <acpi/>
  <apic/>
  <pae/>
</features>
<clock offset='utc'/>
<on_reboot>restart</on_reboot>
<on_crash>restart</on_crash>
<devices>
  <emulator>/usr/bin/kvm</emulator>
  <disk type='file' device='disk'>
    <driver name='qemu' type='raw'/>
    <source file='/home/cernvm/image/cernvm-2.3.0-x86_64.hdd'/>
    <target dev='hda' bus='ide'/>
    <address type='drive' controller='0' bus='0' unit='0'/>
  </disk>
  <controller type='ide' index='0'>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x01'
    function='0x1'/>
  </controller>
  <interface type='network'>
    <mac address='52:54:00:ca:d5:d3'/>
    <source network='default'/>
    <target dev='vnet0'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x03'
    function='0x0'/>
  </interface>
  <serial type='pty'>
    <target port='0'/>
  </serial>
  <console type='pty'>
    <target type='serial' port='0'/>
  </console>
  <input type='mouse' bus='ps2'/>
  <graphics type='vnc' port='-1' autoport='yes'/>
  <video>
    <model type='cirrus' vram='9216' heads='1'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02'
    function='0x0'/>
  </video>
```

```
    <memballoon model='virtio'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x04'
      function='0x0'/>
    </memballoon>
  </devices>
</domain>
```

4. Finally, configure the the virtual machine network to automatically start when the system boots, then create the virual machine and ensure that it can be started and stopped using virsh, *all virsh commands require root privileges, so it easiest to simply run as as root.*

Listing 2.32: Configure the Virtual Machine and Verify it Works

```
$ su
$ virsh net-autostart default
$ virsh define /home/cernvm/image/cernvm.xml

# Verify the virtual machine was added, should be in list
$ virsh list --all

# Verify the virtual machine can be turned on/off
$ virsh start cernvm
# Wait about 2 minutes for the system to boot
$ virsh shutdown cernvm
```

2.4.4 Configuring the CernVM Image

1. The next steps involve configuring the CernVM image to integrate with virsh as well as the test suite, first start the virtual machine, execute the command to get the IP address of the CernVM image. Then follow this guide provided on the CernVM website on how to configure the CernVM image and create a new user using the web interface http://cernvm.cern.ch/portal/cvmconfiguration and reboot the system, *all virsh commands require root privileges.*

```
$ virsh start cernvm
# Wait about 2 minutes for the system to boot
# Then get IP Address using the following command
$ arp -an
```

2. Now that the system has booted, login in to the system using SSH for the user you created using the CernVM web interface

```
$ ssh <user you created>@cernvm-image-ipaddress
```

3. Now that you are logged into the system through SSH, execute the following instructions to enable virsh console access.

Listing 2.33: Enable Virsh Console Access

```
# Type the following command, and enter a root password you won't forget
$ sudo passwd root
# Change to the root account and enable console for root so that you can
# login using virsh console
$ su

# Enable root login on tty
$ echo    ttyS0    >> /etc/securetty

# Then add console=ttyS0 to the kernel parameter line in /etc/grub.conf
$ vi /etc/grub.conf

# Add getty to /etc/inittab file after all the other "tty" lines,
# add the following line to /etc/inittab
s0:2345:respawn:/sbin/agetty -L 38400 ttyS0 vt100
```

4. Now, reboot and login to the system using virsh console and then simply type "root" and enter the password to login as the root account, *there will most likely not be any console display from the virtual machine until you press enter after entering the password*, then enable root login through SSH using the following commands. **To disconnect from the virsh console and return to the host machine console, press CTRL +] which is (^])**

Listing 2.34: Enable SSH Root Login

```
# Reboot the system and the console should now work
# Wait for it to completely shut off and turn on
$ virsh shutdown cernvm
$ virsh start cernvm

# Login as root using password you set with the passwd command
$ virsh console cernvm

# edit the file /etc/ssh/sshd_config and uncomment the line
# PermitRootLogin yes   in order to enable root login
$ vi /etc/ssh/sshd_config
```

5. Next from the host machine *ie. the machine you're currently using* enable automatic login as root through ssh on the KVM guest, first ensure that the guest machine has been started.

Listing 2.35: Enable Automatic SSH Root Login

```
# Restart the virtual machine wait for
# it to completely shut off and turn on
$ virsh shutdown cernvm
$ virsh start cernvm

# Generate a public key, when prompted press enter for everything
$ ssh-keygen -t rsa

# Get the ip address of the running cernvm guest, as done previously
$ arp -an

# Next, run the following command to setup automatic login for ssh
# without having to type the password. When prompted for the password
# enter the root password you just set previously
$ ssh-copy-id -i ~/.ssh/id_rsa.pub root@cernvm-image-ipaddress

# Disconnect, and try and login to the machine using ssh and ensure
# that you can login as root without having to type the password
$ ssh root@cernvm-image-ipaddress
```

6. If everything so far has worked, then the test client and CernVM image have been installed and configured properly, if you have any outstanding issues solve them before proceeding further, or go to the section "Server Platform Setup" 3 as the Tapper server does not require virtual machine creation and configuration.

2.5 OS X Test Client Setup

2.5.1 Configuring the system

1. After the system has booted, the first thing to configure are the power management settings and other preferences, as this system will be running as a test client, sleep and other automatic energy saving features must be disabled. Begin by navigating to the power options,
   `Apple logo -> System Preferences -> Energy Saver` for the option "Computer sleep" slide the bar to the far right so that it is set to "Never" and ensure that the following options are all disabled.
   - Put the hard disk(s) to sleep when possible
   - Wake for Ethernet network access
   - Allow power button to put the computer to sleep
2. Next, set a hostname for the system from the menu
   `Apple logo -> System Preferences -> Sharing` beside the "Computer Name:" option at the top, click the "Edit..." button. Then enter a relevant hostname for the system based on the hardware or operating system it is running; the hostname should be relevant and unique to better identify the system. A good naming convention should refer to the hardware or operating system and call it a host to differentiate from the virtual machine that will be running as a guest, for example a hostname such as *cernvm-osx-host* could be used, **whatever convention you use make sure it is consistent**.
3. Next, enable SSH access to the system by navigating to
   `Apple logo -> System Preferences -> Sharing` and from the list of services that can be shared, enable "Remote Login", which is SSH.
4. Now, to enable VNC access to the system, select from the same list of services that can be shared, "Remote Management" and for local access options window that appears, enable all of the options listed such as "Observe" and "Change settings". Then to enable VNC compatibilty so that the OS X system can be accessed by other non-Apple computers, click the "Computer Settings..." button and enable the following options and set a password for the "VNC viewers..." option.
   - Show Remote Management status in menu bar
   - Anyone may request permission to control screen
   - VNC viewers may control screen with password
5. Now, to ensure that your user logs in automatically, navigate to
   `Apple logo -> System Preferences -> Accounts` and click "Login Options", you may have to click on the lock icon and enter your password in order to make changes to the login options. Then for the option "Automatic login:" select your user from the list of accounts to enable automatic login.

6. Finally, to ensure that the settings were configured properly, reboot the machine and ensure that the following work.
   - It automatically boots up into the full desktop environment without having to login
   - You have access to the machine using SSH and can login
   - You have VNC access to the machine and can control the system using VNC

2.5.2 Installing libvirt and virsh

1. The virtualization API libvirt and the command line tool virsh [3] are the essential components required for setting up a test client and must be installed and properly configured before any testing can begin. Ensure that you follow the proceeding directions carefully and validate that virsh is working properly before proceeding to install and configure the various hypervisors.

2. First, to build libvirt from source Xcode must be installed, http://developer.apple.com/xcode/, Xcode 4 requires either a paid developer membership, or must be purchased from the App Store; but, Xcode 3 is freely available, **this guide uses Xcode 3 to build libvirt from source**.

3. Next, review the release news listed on the libvirt website, http://libvirt.org/news.htm and read through the release notes for the latest version released to make sure that there are no regressions or deprecated support for the platforms you wish to support. If you intend to set up a test client which supports VMware then you must download a version later than *0.8.7* as there was no support for VMware prior to that release.

4. Next, download the latest source release that is a **tar.gz** file from the libvirt release server, http://libvirt.org/sources/ based on the latest release which does not have any regressions or deprecations for the virtualization platforms you wish to support. As of this date, the latest release of libvirt is version 0.9.2, this is the release that will be used for the following instructions and examples.

5. Next, download and install the following dependencies which are required by libvirt, some of the newer releases are broken on OS X so the follow versions must be used. Before issuing make, configure each with *–prefix=/usr/local* and for libgcrypt also add *–disable-asm*.

   - libgpgerror-1.7 - ftp://ftp.gnupg.org/gcrypt/libgpg-error/libgpg-error-1.7.tar.gz
   - libgcrypt-1.4.5 - ftp://ftp.gnupg.org/gcrypt/libgcrypt/libgcrypt-1.4.5.tar.gz
   - gnutls-2.8.5 - ftp://ftp.gnu.org/pub/gnu/gnutls/gnutls-2.8.5.tar.bz2

6. Next, configure and install libvirt using the following commands, which builds libvirt with support for VMware and VirtualBox.

Listing 2.36: Configure and Install libvirt

```
# Configure libvirt with VMware/VirtualBox support
./configure --without-xen --without-sasl --without-avahi \
--without-polkit --without-qemu --without-lxc --without-openvz \
--without-remote --with-libvirtd --without-uml --with-vmware --with-vbox
```

```
# Build/install libvirt
sudo make
sudo make install
```

7. Finally, ensure that virsh installed correctly and is running by connecting to the test hypervisor and ensuring that the test virtual machine, named "test" is running.

Listing 2.37: Verify virsh was Installed Properly

```
# Verify virsh is working, test should be running
$ virsh -c test:///default list --all
```

2.5.3 Installing and configuring VirtualBox

1. First, begin by downloading and installing the latest version of VirtualBox for OS X from the VirtualBox download page, http://www.virtualbox.org/wiki/Downloads ensure that you select the appropriate architecture for your system. The following instructions for this section of the guide uses VirtualBox 4.0.10 for AMD64.

2. Next, to verify that VirtualBox has been installed properly and that virsh can connect to the VirtualBox hypervisor, verify that the following VirtualBox kernel extensions are loaded and that you are able to connect to the virsh console without any errors.

   - org.virtualbox.kext.VBoxDrv
   - org.virtualbox.kext.VBoxUSB
   - org.virtualbox.kext.VBoxNetFlt
   - org.virtualbox.kext.VBoxNetAdp

   Listing 2.38: Verify that virsh can Access VirtualBox

```
# Verify that the kernel extentsions are loaded
$ kextstat | grep -i virtualbox

# Verify that virsh can connect to virtualbox
$ virsh -c vbox:///session
```

3. Now, proceed to download and extract the desired VirtualBox virtual machine image onto the system from the CernVM download portal, http://cernvm.cern.ch/portal/downloads it is recommended that you download the VirtualBox Desktop image for your architecture. For this guide the Desktop image will be downloaded as it is the most practical image for the majority of users.

4. Now that the virtual machine has been created and configured verify that it is able to boot completely without crashing, *you will be presented with a login screen when it has booted completely.* Then shutdown the virtual machine by clicking "Actions" from within the virtual machine and selecting 'Shutdown", after the virtual machine has shutdown close VirtualBox and then connect to the VirtualBox hypervisor and determine that you can view, start, and stop virtual machine.

   Listing 2.39: Verify VirtualBox Works with Virsh

```
$ virsh --connect vbox:///session

# Verify the virtualbox virtual machine is accessible
# Name of the virtual machine created should be listed
```

```
$ list --all

# Verify the virtual machine can be turned on/off
$ start <name of virtual machine>
# Wait about 2 minutes for the system to boot
$ shutdown <name of virtual machine>
```

5. Finally, configure the the virtual machine network to automatically start when the system boots.

Listing 2.40: Configure Network

```
# Configure virsh network for VirtualBox
$ virsh --connect vbox:///session
$ net-start vboxnet0
$ net-autostart vboxnet0
```

2.5.4 Installing and configuring VMware

1. First, begin by downloading and installing the latest version of VMware Fusion for OS X from the VMware product page, http://www.vmware.com/products/, VMware Fusion requires a license, so you will have to purchase it in order to continue.

2. Next, to verify that VMware Fusion has been installed properly, verify that the following VMware kernel extensions are loaded, currently virsh has support to connect to the VMware hypervisor, but does not support interacting with VMware through VMware Fusion.

   - com.vmware.kext.vmx86
   - com.vmware.kext.vmci
   - com.vmware.kext.vmioplug
   - com.vmware.kext.vmnet

   Listing 2.41: Verify VMware Kernel Extensions Loaded

   ```
   # Verify that the kernel extentsions are loaded
   $ kextstat | grep -i vmware
   ```

3. Now, proceed to download and extract the desired VMware virtual machine image onto the system from the CernVM download portal, urlhttp://cernvm.cern.ch/portal/downloads it is recommended that you download the VMware Desktop image for your architecture. For this guide the Desktop image will be downloaded as it is the most practical image for the majority of users.

3 CernVM Release Testing Server Platform Setup

3.1 Introduction

This section provides complete step by step instructions on how to setup and configure the Tapper server which is part of a basic working Release Testing environment by outlining the procedure for setting up the server, hence why this is called a *walkthrough* document. This guide is intended for users familiar enough with computers and desktop environments to enter basic commands in a terminal and install various operating systems.

3.2 Red Hat Based Server Setup

3.2.1 Installing the system

3.2.2 Configuring the system

1. For installing a Red Hat based server, follow the instructions outlined in the sections "Installing the system" **??** and "Configuring the system" 2.3.1 for installation and configuration instructions with the only exception being that the hostname should be something unique such as *cernvm-redhat5-server*, to indicate that it is running the Tapper server, **again keep the hostname convention consistent**.

3.2.3 Installing the Tapper Server

1. Next, execute the following commands to install necessary dependencies, *from now on all commands require root privileges.*

   Listing 3.1: Install Dependencies

   ```
   $ yum install make
   $ yum install subversion
   ```

2. Now, download the latest copy of the Tapper-Deployment, which is an installer for Tapper from the CernVM Release Testing Google Code Project page

   Listing 3.2: Download Tapper-Deployment

   ```
   $ svn checkout http://cernvm-release-testing.googlecode.com/svn/trunk/\
   installer/tapper-deployment
   ```

3. Now edit the Makefile in the Tapper-Deployment installer folder and configure variable TAPPER_SERVER which is the hostname of the machine that is currently installing the starter-kit. For now disregard the TESTMACHINE variables, you should have something similar to this in the Makefile.

   Listing 3.3: Makefile Configuration

   ```
   # initial machine names
   TAPPER_SERVER=cernvm-server
   TESTMACHINE1=johnconnor
   TESTMACHINE2=sarahconnor
   TESTMACHINE3=bullock
   ```

4. After you have configured the Makefile in the installer folder, install Tapper-Deployment by executing the following command, for any prompts during the installation leave them as default and press enter. **During the installation, you will be prompted for the mysql password, DO NOT ENTER a password here UNLESS you already have an existing MySQL installation/database with a password set for the "root" account**. Finally, if you have any errors or other issues during the installation, please contact us with a summary of the problem and send us a copy of the installation log "install.log".

3.3 Debian Based Server Setup

3.3.1 Installing the system

3.3.2 Configuring the system

1. For installing a Debian based server, follow the instructions outlined in the sections "Installing the system" 2.4.1 and "Configuring the system" 2.4.2 for installation and configuration instructions with the only exception being that the hostname should be something unique such as *cernvm-debian6-server*, to indicate that it is running the Tapper server, **again keep the hostname convention consistent**.

3.3.3 Installing the Tapper Server

1. Next, execute the following commands to install necessary dependencies, *from now on all commands require root privileges.*

Listing 3.4: Install Dependencies

```
$ apt-get update
$ apt-get install make
$ apt-get install subversion
```

2. Now, download the latest copy of the Tapper-Deployment, which is an installer for Tapper from the CernVM Release Testing Google Code Project page

Listing 3.5: Download Tapper-Deployment

```
$ svn checkout http://cernvm-release-testing.googlecode.com/svn/trunk/\
installer/tapper-deployment
```

3. Now edit the Makefile in the Tapper-Deployment installer folder and configure variable TAPPER_SERVER which is the hostname of the machine that is currently installing the starter-kit. For now disregard the TESTMACHINE variables, you should have something similar to this in the Makefile.

Listing 3.6: Makefile Configuration

```
# initial machine names
TAPPER_SERVER=cernvm-server
TESTMACHINE1=johnconnor
TESTMACHINE2=sarahconnor
TESTMACHINE3=bullock
```

4. After you have configured the Makefile in the installer folder, install Tapper-Deployment by executing the following command, for any prompts during the installation leave them as default and press enter. **During the installation, you will be prompted for the mysql password, DO NOT ENTER a password here UNLESS you already have an existing MySQL installation/database with a password set for the "root" account**. Finally, if you have any errors or other issues during the installation, please contact us with a summary of the problem and send us a copy of the installation log "install.log".

Listing 3.7: Install Tapper-Deployment

```
$ cd installer/
$ make localsetup 2>&1 | tee install.log
```

3.3.4 Setting up Tapper Web Interface and Database

1. Next you need to set a password for the root account of the mysql database[1]

   Listing 3.8: Set MySQL Root Password

   ```
   # Example: mysqladmin -u root password abc123
   $ mysqladmin -u root password <newpassword>
   ```

2. Now that the installion has completed and the security issue has been dealt with, ensure that you can access the tapper web interface and that it is working by viewing it in your browser using the url, http://localhost/tapper[2]

3. Next, install PHPMyAdmin so that it's easy to administrate and configure the Tapper databases, when prompted to select the "Web server to reconfigure automatically" select apache2 by pressing the space bar and press enter. *If you are prompted to "configure databases for phpmyadmin with dbconfig-common" select* **NO** .

   Listing 3.9: Install PHPMyAdmin

   ```
   $ apt-get update

   # When prompted for the server to reconfigure automatically select apache2
   # when prompted to configure the database with dbconfig-common select NO
   $ apt-get install phpmyadmin
   ```

4. Verify that PHPMyAdmin has been installed and configured correctly and that you can access the tapper web interface by viewing it in your browser using the url, http://localhost/phpmyadmin. Login to the PHPMyAdmin web interface using the *username root and the MySQL root password you set earlier using mysqladmin.*

5. Now, add all of the configured test machines created in the "Test Client Platform Setup" section to the Tapper database and set the test clients as active, then add the hardware specifications for each test client to the database. This example is just using a single generic test machine, you will have to repeat these commands for each test client and change the hostname *cernvm-host* and the values for mem, core,vendor, and has_ecc as needed; the vendor can be AMD or Intel.

   Listing 3.10: Adding Test Clients to Tapper Database

   ```
   # Add the hostname of the test client to database
   $ tapper-testrun newhost --name cernvm-host --active
   ```

[1] This will eventually be implemented in the makefile

[2] This can be accessed locally and remotely from other systems using the server hostname or IP address

```
# Add the hardware specifications for the test client
$ mysql testrundb -utapper -ptapper
$ insert into host_feature(host_id, entry, value) values \
((select id from host where name = 'cernvm-host'), 'mem',
4096);
$ insert into host_feature(host_id, entry, value) values \
((select id from host where name = 'cernvm-host'), 'cores',
4);
$ insert into host_feature(host_id, entry, value) values \
((select id from host where name = 'cernvm-host'), 'vendor', 'AMD');
$ insert into host_feature(host_id, entry, value) values \
((select id from host where name = 'cernvm-host'), 'has_ecc',
0);
```

6. Next, send a sample test report to the tapper server, to ensure that the web interface, MCP, database, and reports framework are all working by viewing the tapper reports in your browser at the following url, http://localhost/tapper/reports You should now see a report from whatever the "Tapper-Machine-Name" in demo_report.tap was set as. *For the example demo_report.tap provided below it would be cernvm-server.*

Listing 3.11: Send a Report from the Tapper Server to Itself

```
# Save the following in a file named demo_report.tap
$ vi demo_report.tap

        1..2
        # Tapper-Suite-Name: Tapper-Deployment
        # Tapper-Suite-Version: 1.001
        # Tapper-Machine-Name: cernvm-server
        ok - Hello test world
        ok - Just another description

# Send the report to the tapper server using netcat
$ cat demo_report.tap | netcat -q7 -w1 cernvm-server 7357
```

7. Finally, ssh login to one of the test machine that was set up earlier, *in our examples, cernvm-host* and send another sample test report to the tapper server, to ensure that the web interface, MCP, database, and reports framework are all working by viewing the tapper reports in your browser at the following url: http://localhost/tapper/reports. You should now see a report from whatever the "Tapper-Machine-Name" in demo_report.tap was set as. *For the example demo_report.tap provided below it would be cernvm-testclient.*

Listing 3.12: Send a Report to the Tapper Server from a Test Client

```
# Save the following in a file named demo_report.tap
```

```
$ vi demo_report.tap

        1..2
        # Tapper-Suite-Name: Tapper-Deployment
        # Tapper-Suite-Version: 1.001
        # Tapper-Machine-Name: cernvm-testclient
        ok - Hello test world
        ok - Just another description

# Send the report to the tapper server using netcat
$ cat demo_report.tap | netcat -q7 -w1 cernvm-server 7357
```

8. Now that it has been verified that the tapper server, including the web interface, MCP, database, and reports framework are all working; return to the sections titled "Setting up the Tapper Test Suite" for each of the test client, as there are unique instructions for each operating system.

3.4 Ubuntu 10.04 Based Server Setup

3.4.1 Configuring the system

1. After the system has booted remove the follow unnecessary startup applications by selecting from the menu
   `System -> Preferences -> Startup Applications`
   - bluetooth
   - check for new hardware drivers
   - evolution alarm
   - Gnome Login Sound
   - print queue
   - pulseaudio
   - ubuntu one
   - update notifier
   - visual assistance/aid
   - any others you think are unnecessary based on your own discretion
2. Next enable and configure remote desktop from the menu
   `System -> Preferences -> Remote Desktop` and ensure that the following options are configured
   - Enable the option "Allow others to view your desktop"
   - Enable the option "Allow other users to control your desktop"
   - Disable the option "You must confirm access to this machine"
3. Next, enable the following repositories so that Ubuntu has access to an even larger amount of software packages, from the menu select
   `System -> Administration -> Software Sources` then click the "Other Software" tab and enable the items listed. Then click "Close" and if prompted click the "Reload" button.
4. Next, configure the screen saver from the menu
   `System -> Preferences -> Screensaver` and ensure that the following options are configured
   - Disable the option "Lock screen when screensaver active"
5. Next, open the terminal and enable SSH support by installing openSSH using the following commands

```
# You will be prompted to enter your password
$ sudo apt-get update
$ sudo apt-get install openssh-server
```

6. The following instructions involve enabling headless support so that you can remote desktop to the machine without having a monitor connected to the computer.
   a. Edit the xorg.conf file and put the following in it

Listing 3.13: Configuring Xorg for Headless Support

```
$ sudo gedit /etc/X11/xorg.conf

Section "Device"
Identifier "VNC Device"
Driver "vesa"
EndSection

Section "Screen"
Identifier "VNC Screen"
Device "VNC Device"
Monitor "VNC Monitor"
SubSection "Display"
Modes "1280x1024"
EndSubSection
EndSection

Section "Monitor"
Identifier "VNC Monitor"
HorizSync 30-70
VertRefresh 50-75
EndSection
```

   b. Then edit grub and set the option "nomodeset", and proceed to update grub and reboot

Listing 3.14: Configuring Grub for Headless Support

```
$ sudo gedit /etc/default/grub

GRUB_CMDLINE_LINUX="nomodeset"

$ sudo update-grub"
```

7. Now, reboot the machine, and ensure that the following work
   - It automatically boots up into the full desktop environment without having to login
   - You have access to the machine using SSH and can login
   - You have VNC access to the machine and can control the system using VNC

8. Finally, update the system from the menu
   `System -> Administration -> Update Manager` and after it has completed the updates reboot the system

3.4.2 Installing the Tapper Server

1. Next, execute the following commands to install necessary dependencies.

   Listing 3.15: Install Dependencies

   ```
   $ sudo apt-get update
   $ sudo apt-get install make
   $ sudo apt-get install subversion
   ```

2. Now, download the latest copy of the Tapper-Deployment, which is an installer for Tapper from the CernVM Release Testing Google Code Project page

   Listing 3.16: Download Tapper-Deployment

   ```
   $ svn checkout http://cernvm-release-testing.googlecode.com/svn/trunk/\
   installer/tapper-deployment
   ```

3. Now edit the Makefile in the Tapper-Deployment installer folder and configure variable TAPPER_SERVER which is the hostname of the machine that is currently installing the starter-kit. For now disregard the TESTMACHINE variables, you should have something similar to this in the Makefile.

   Listing 3.17: Makefile Configuration

   ```
   # initial machine names
   TAPPER_SERVER=cernvm-server
   TESTMACHINE1=johnconnor
   TESTMACHINE2=sarahconnor
   TESTMACHINE3=bullock
   ```

4. After you have configured the Makefile in the installer folder, install Tapper-Deployment by executing the following command, for any prompts during the installation leave them as default and press enter. **During the installation, you will be prompted for the mysql password, DO NOT ENTER a password here UNLESS you already have an existing MySQL installation/database with a password set for the "root" account**. Finally, if you have any errors or other issues during the installation, please contact us with a summary of the problem and send us a copy of the installation log "install.log".

   Listing 3.18: Install Tapper-Deployment

   ```
   $ cd installer/
   $ sudo make localsetup 2>&1 | tee install.log
   ```

3.4.3 Setting up Tapper Web Interface and Database

1. Next you need to set a password for the root account of the mysql database[3]

   Listing 3.19: Set MySQL Root Password

   ```
   # Example: mysqladmin -u root password abc123
   $ sudo mysqladmin -u root password <newpassword>
   ```

2. Now that the installion has completed and the security issue has been dealt with, ensure that you can access the tapper web interface and that it is working by viewing it in your browser using the url, http://localhost/tapper[4]

3. Next, install PHPMyAdmin so that it's easy to administrate and configure the Tapper databases, when prompted to select the "Web server to reconfigure automatically" select apache2 by pressing the space bar and press enter. *If you are prompted to "configure databases for phpmyadmin with dbconfig-common" select* **NO** .

   Listing 3.20: Install PHPMyAdmin

   ```
   $ sudo apt-get update

   # When prompted for the server to reconfigure automatically select apache2
   # when prompted to configure the database with dbconfig-common select NO
   $ sudo apt-get install phpmyadmin
   ```

4. Verify that PHPMyAdmin has been installed and configured correctly and that you can access the tapper web interface by viewing it in your browser using the url, http://localhost/phpmyadmin. Login to the PHPMyAdmin web interface using the *username root and the MySQL root password you set earlier using mysqladmin.*

5. Now, add all of the configured test machines created in the "Test Client Platform Setup" section to the Tapper database and set the test clients as active, then add the hardware specifications for each test client to the database. This example is just using a single generic test machine, you will have to repeat these commands for each test client and change the hostname *cernvm-host* and the values for mem, core,vendor, and has_ecc as needed; the vendor can be AMD or Intel.

   Listing 3.21: Adding Test Clients to Tapper Database

   ```
   # Add the hostname of the test client to database
   $ tapper-testrun newhost --name cernvm-host --active
   ```

[3] This will eventually be implemented in the makefile

[4] This can be accessed locally and remotely from other systems using the server hostname or IP address

```
# Add the hardware specifications for the test client
$ mysql testrundb -utapper -ptapper
$ insert into host_feature(host_id, entry, value) values \
((select id from host where name = 'cernvm-host'), 'mem',
4096);
$ insert into host_feature(host_id, entry, value) values \
((select id from host where name = 'cernvm-host'), 'cores',
4);
$ insert into host_feature(host_id, entry, value) values \
((select id from host where name = 'cernvm-host'), 'vendor', 'AMD');
$ insert into host_feature(host_id, entry, value) values \
((select id from host where name = 'cernvm-host'), 'has_ecc',
0);
```

6. Next, send a sample test report to the tapper server, to ensure that the web interface, MCP, database, and reports framework are all working by viewing the tapper reports in your browser at the following url, http://localhost/tapper/reports You should now see a report from whatever the "Tapper-Machine-Name" in demo_report.tap was set as. *For the example demo_report.tap provided below it would be cernvm-server.*

Listing 3.22: Send a Report from the Tapper Server to Itself

```
# Save the following in a file named demo_report.tap
$ vi demo_report.tap
```

```
1..2
# Tapper-Suite-Name: Tapper-Deployment
# Tapper-Suite-Version: 1.001
# Tapper-Machine-Name: cernvm-server
ok - Hello test world
ok - Just another description
```

```
# Send the report to the tapper server using netcat
$ cat demo_report.tap | netcat -q7 -w1 cernvm-server 7357
```

7. Finally, ssh login to one of the test machine that was set up earlier, *in our examples, cernvm-host* and send another sample test report to the tapper server, to ensure that the web interface, MCP, database, and reports framework are all working by viewing the tapper reports in your browser at the following url: http://localhost/tapper/reports. You should now see a report from whatever the "Tapper-Machine-Name" in demo_report.tap was set as. *For the example demo_report.tap provided below it would be cernvm-testclient.*

Listing 3.23: Send a Report to the Tapper Server from a Test Client

```
# Save the following in a file named demo_report.tap
```

```
$ vi demo_report.tap

        1..2
        # Tapper-Suite-Name: Tapper-Deployment
        # Tapper-Suite-Version: 1.001
        # Tapper-Machine-Name: cernvm-testclient
        ok - Hello test world
        ok - Just another description

# Send the report to the tapper server using netcat
$ cat demo_report.tap | netcat -q7 -w1 cernvm-server 7357
```

8. Now that it has been verified that the tapper server, including the web interface, MCP, database, and reports framework are all working; return to the sections titled "Setting up the Tapper Test Suite" for each of the test client, as there are unique instructions for each operating system.

Bibliography

[1] CernVM Release Testing Google Code Project. https://code.google.com/p/cernvm-release-testing/.

[2] Advanced Micro Devices Inc. AMD Tapper. http://developer.amd.com/zones/opensource/amdtapper/pages/default.aspx/, 2011.

[3] Red Hat Inc. libvirt Virtualization Api. http://libvirt.org/, 2011.

Index