

# Building virtual Mac environments with VMware

Rich Trouton

Howard Hughes Medical Institute,  
Janelia Farm Research Campus

HHMI  
HOWARD HUGHES MEDICAL INSTITUTE

Before we get started, there's two things I'd like to mention. The first is that, all of the slides, speakers' notes and the demos are available for download and I'll be providing a link at the end of the talk. I tend to be one of those folks who can't keep up with the speaker and take notes at the same time, so for those folks in the same situation, no need to take notes. Everything I'm covering is going to be available for download.

The second is to please hold all questions until the end. If you've got questions, make a note of them and talk with me afterwards. With luck, I'll be able to answer most of your questions during the talk itself.



When new software appears, we all need test boxes that match our standard configuration in order to verify that the new software doesn't adversely affect anything. This usually means admins need an available test box, or they have to go find one.



The advent of good virtualization solutions means it's easier than ever before to perform testing without needing to collect and use physical test boxes. VMware Fusion and VMware ESXi have been the virtualization solutions I've been using, so let's talk about how you can use these tools to quickly build test VMs that match your needs.

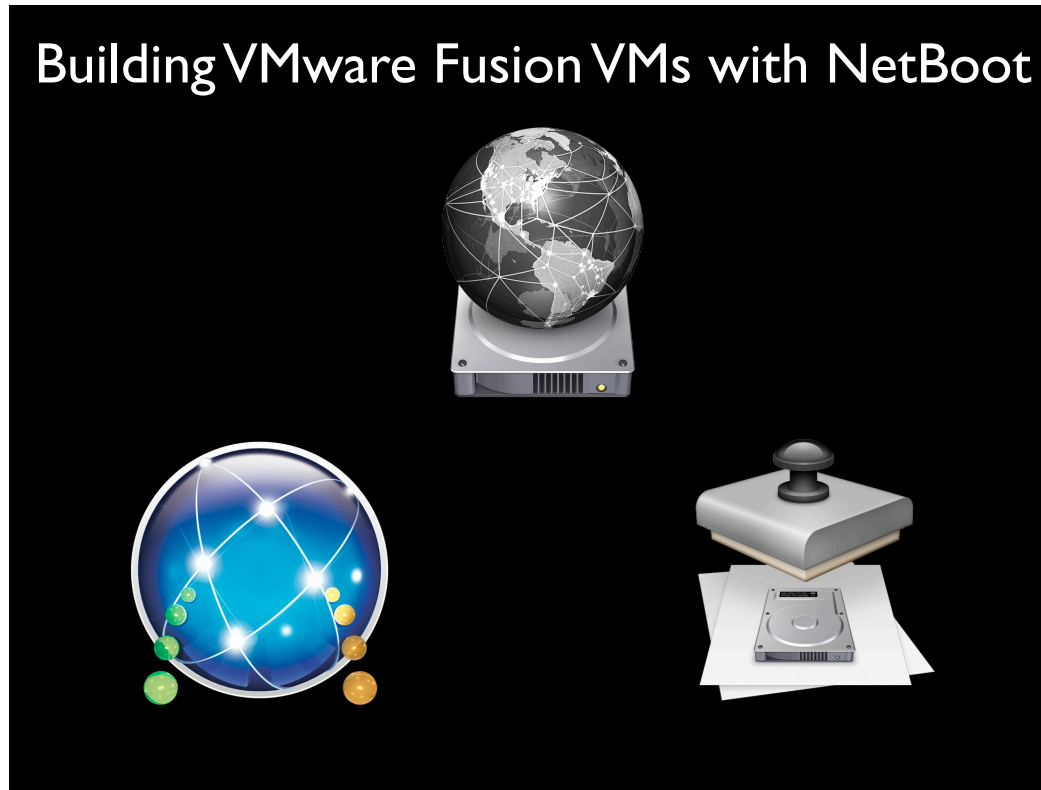
## Building VMware Fusion VMs with NetBoot



In VMWare Fusion 5, VMware added NetBoot support for virtual machines running Mac OS X. This proved to be an enormous boon to Mac admins who used NetBoot to help set up their machines because they could now build VMs like they built their users' Macs.

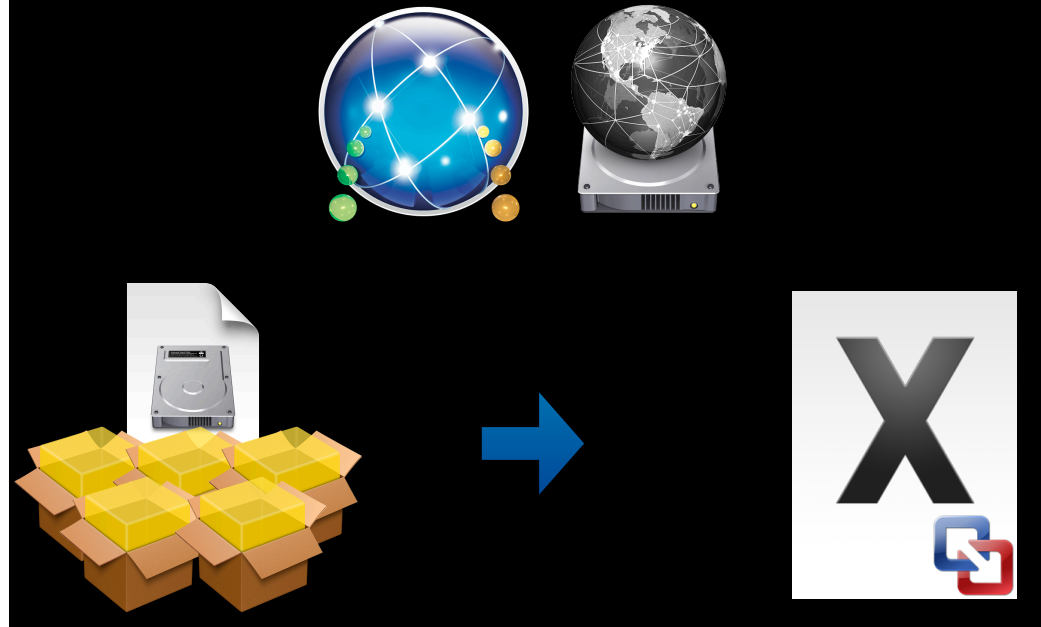


## Building VMware Fusion VMs with NetBoot



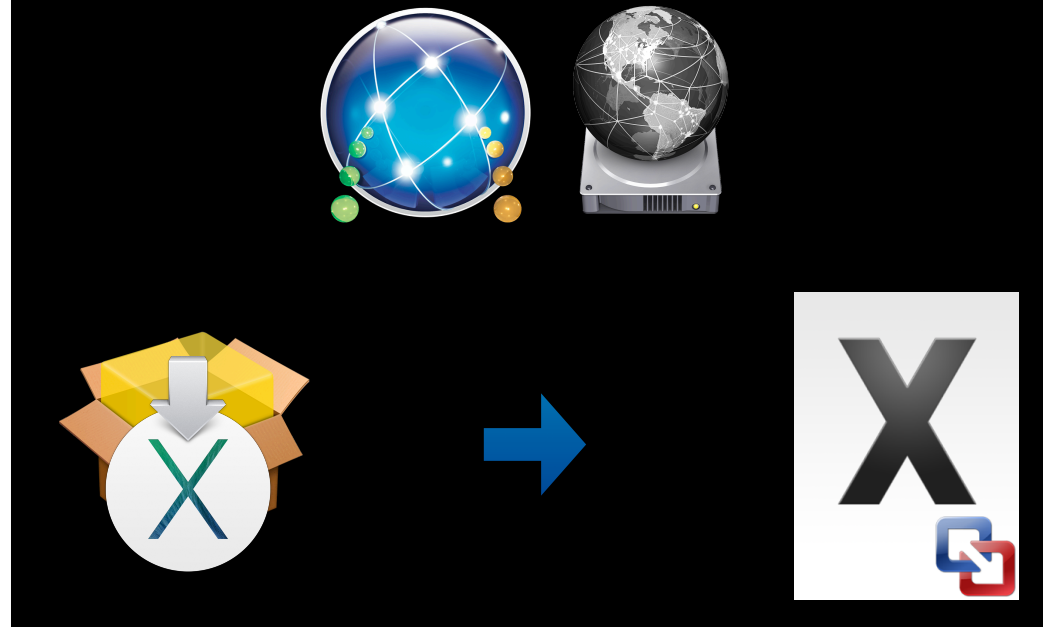
Two tools that can be used to build VMs with NetBoot are DeployStudio and Apple's System Image Utility. NetBoot sets associated with these tools allow Mac admins to boot their VM from the relevant NetBoot set and then apply whatever installs and configurations are desired.

## Building VMware Fusion VMs with DeployStudio



My preferred tool for building VMs is DeployStudio. DeployStudio offers great flexibility when building Macs and that same versatility can also be applied to VMs.

## Building VMware Fusion VMs with DeployStudio and createOSXInstallPkg



In my own shop, I'm using createOSXInstallPkg to generate OS X installers, which I then use with DeployStudio to help me build VM templates. For those unfamiliar with it, createOSXInstallPkg is a tool created by Greg Neagle at Disney. It is used to build individual installer packages that can install OS X Lion, Mountain Lion or Mavericks in the same way that you may install Microsoft Office or other applications.

The advantage of using this tool is that a number of system deployment tools for Macs can deploy the installers created by this tool, allowing OS X installations or upgrades to be performed by the system management tool already in use by a particular IT shop.

## Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg

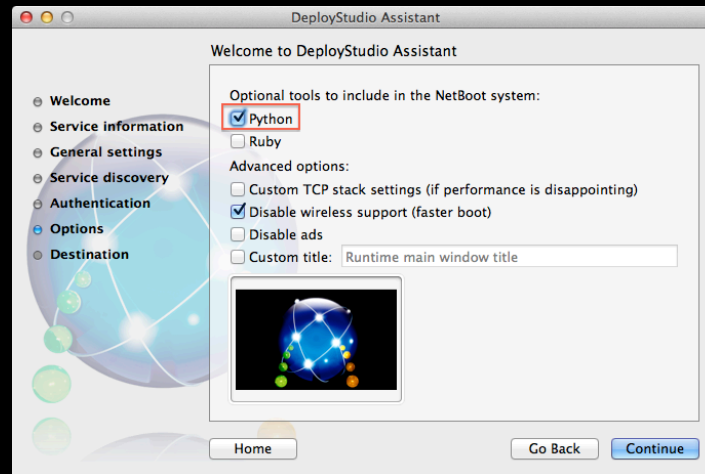


One great thing about using this tool is that createOSXinstallPkg will create an installer package that either installs a stock copy of OS X, or you can add additional packages to the stock OS X install.

There's a couple of guidelines to keep in mind here though. The first is that there is about 350 megabytes of free space available in the OS X installer. This is sufficient space for configuration or bootstrapping packages, but it's not a good idea to add Microsoft Office or similar large installers.

The second is that the limitations of the OS install environment mean that there are a number of installers that won't install correctly. In particular, packages that use preinstallation or postinstallation scripts may fail to run properly when those packages are run as part of the OS installation process. To help work around this limitation, I've developed a solution which I'll be discussing in more detail later in the talk.

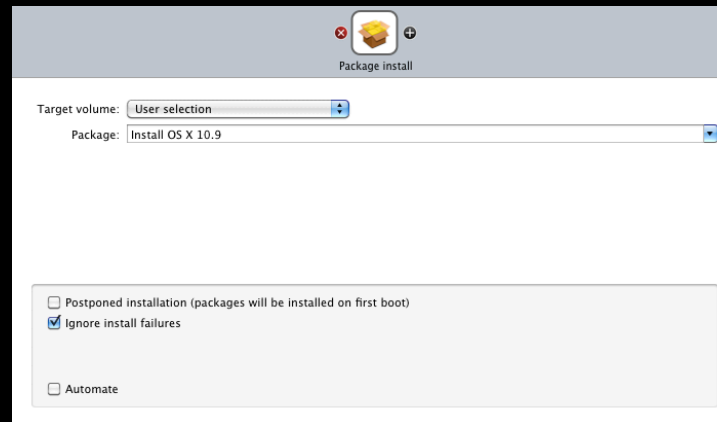
# Building VMware Fusion VMs with DeployStudio and createOSXInstallPkg



DeployStudio can deploy an OS installer built by createOSXInstallPkg, but you'll need to make sure you've added Python support. To add this support, you will need to create your DeployStudio boot set with Python selected as a tool to include in the bootable system.

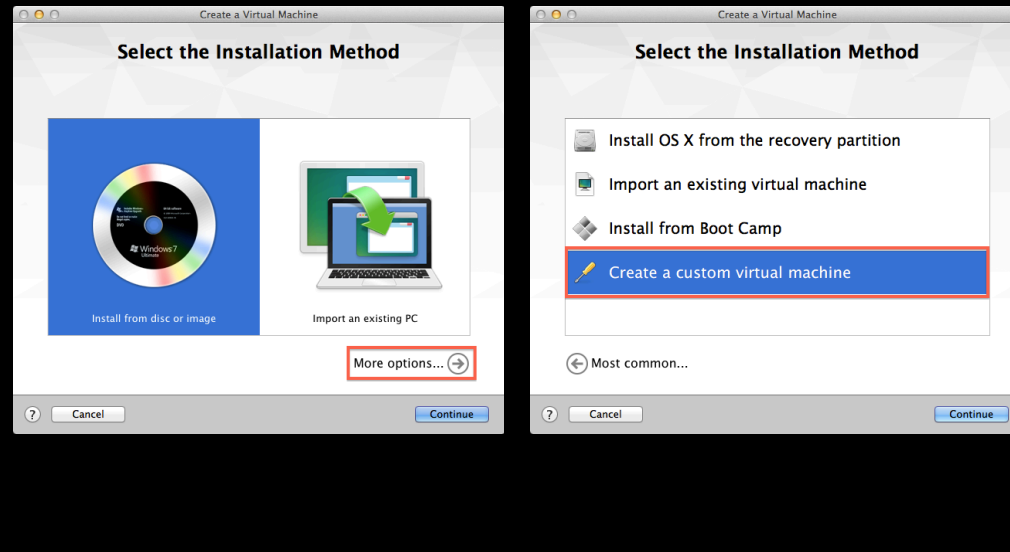
As shown on the screen, adding Python is a checkbox option in the DeployStudio Assistant application that's used to create DeployStudio boot sets.

# Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg



To deploy the OS installer in a new VM, set up a new workflow in DeployStudio to install the package as a non-postponed install. This is important because it will set up a new VM's empty boot drive with the needed boot support to install OS X.

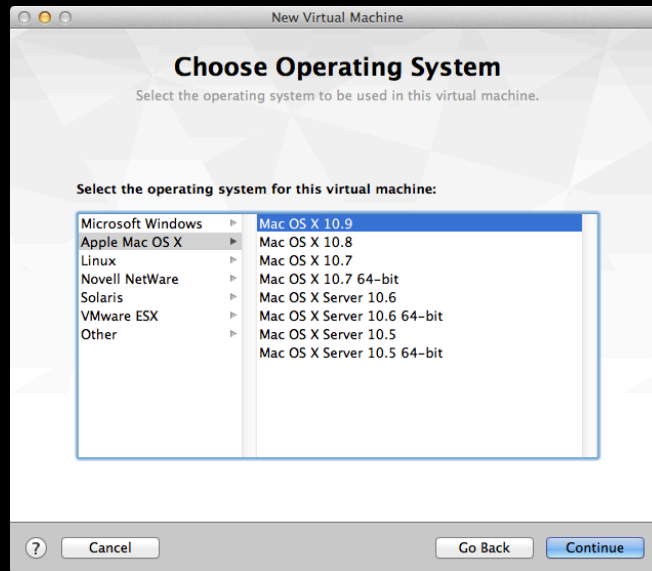
# Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg



Once the necessary workflow has been set up in DeployStudio, you can set up a new VM. In VMware Fusion, my normal method is to create a customized VM. In the “Create a Virtual Machine” window, you can access this by selecting “More Options”, then selecting “Create a custom virtual machine”.



# Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg



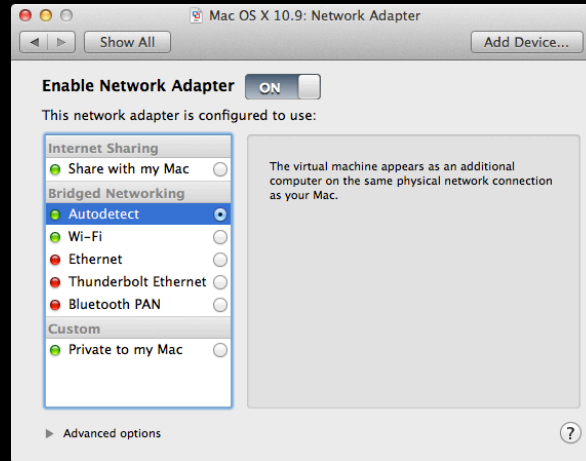
In the “Choose Operating System” window, set OS as appropriate.

# Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg



In the Finish window, select Customize Settings. This will allow you to change the VM's settings ahead of its first boot.

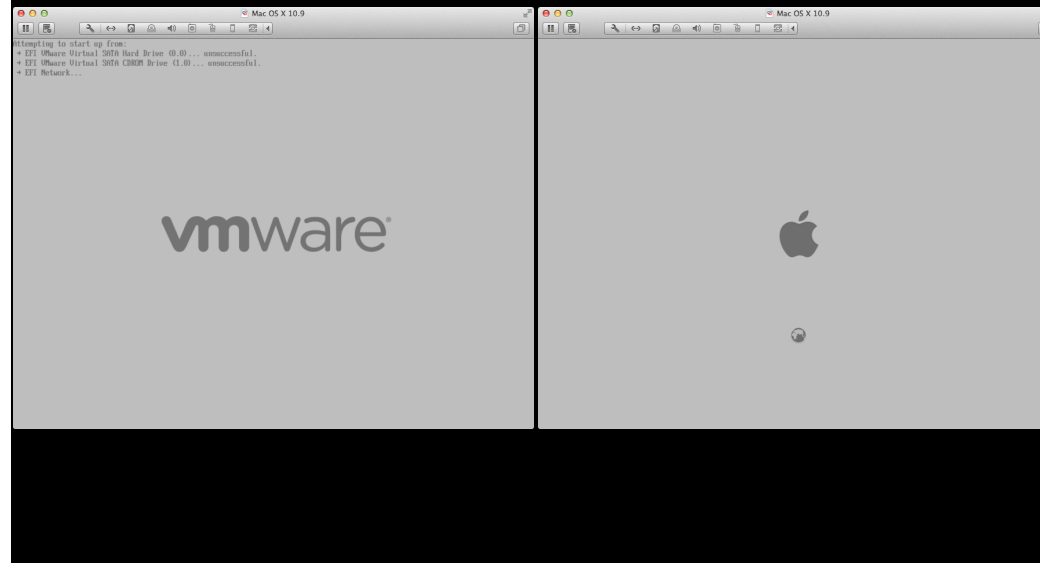
# Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg



In the Network Adapter settings, select Autodetect under Bridged Networking. This will allow the VM to correctly boot from the NetBoot set. You may also want to adjust the VM's available RAM and other settings at this point, but that's up to you.

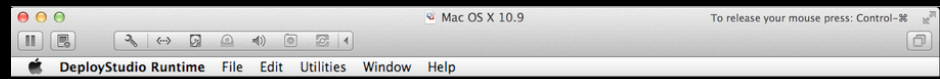
The VM is now ready for the next steps. In this example, the VM is configured to use Mavericks as its OS, but has a formatted and completely empty boot drive.

# Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg



If the DeployStudio boot set is set as the default boot set on the NetBoot server, you can start the VM and then do nothing.


The VM should boot to DeployStudio automatically after failing to boot from either the VM's hard drive or optical drive. Alternatively, you can also hold down the N key on your keyboard to boot the VM from the default NetBoot set.



DeployStudio Runtime v1.6.3 (VM/QAJ9g0Lfd)

◀ ▶ ▶▶

Please enter your login name and password to login.



Server address:

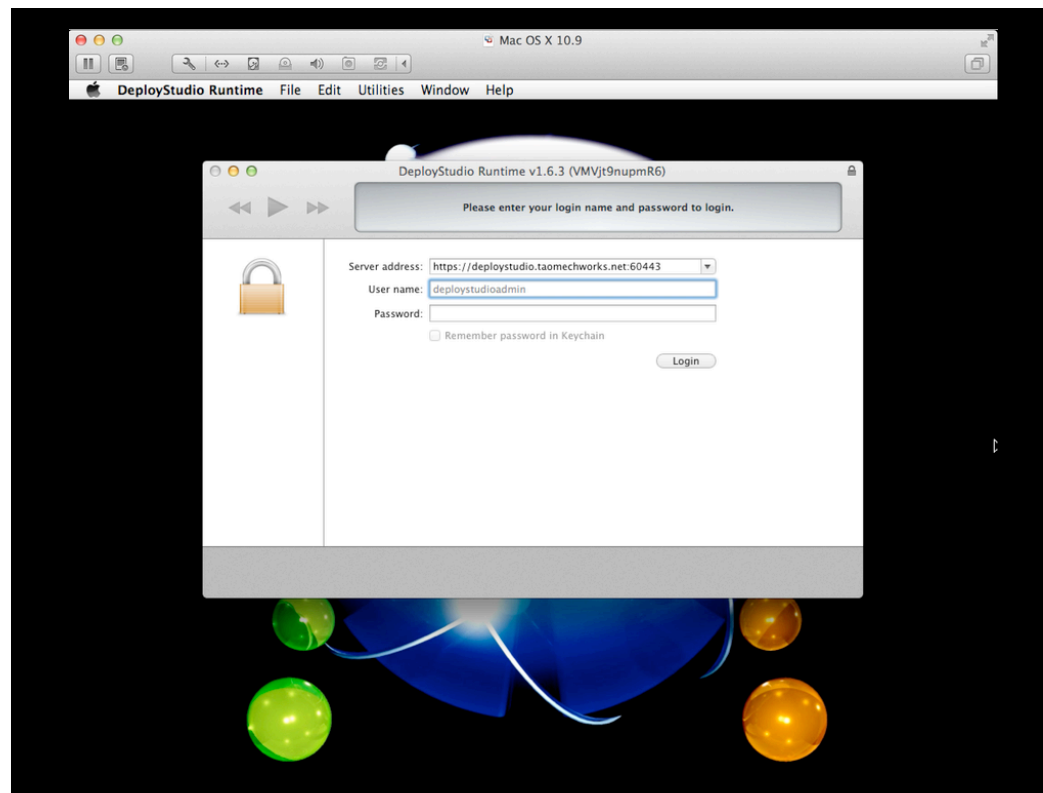
User name:

Password:

☐ Remember password in Keychain

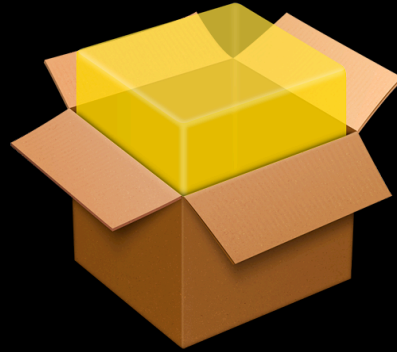
Login

DeployStudio Server lookup...



The example just described will set up a VM with only Mavericks installed, but you can customize further. In my own shop, I normally reboot back to DeployStudio and run additional workflows on the VM.

## Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg



First Boot Package Install.pkg

<http://git.io/DMuw8g>

As mentioned previously, the limitations of the OS install environment mean some packages won't install correctly. In particular, packages that use preinstall or postinstall scripts as part of their normal installation process may fail to run properly in the OS install environment.

To help work around this limitation, I've developed First Boot Package Install.pkg, an installer package that enables other packages to be installed during the initial boot of either a Mac or a VM. This solves the issue because the installers are no longer running in the OS X install environment and can run any associated preinstall or postinstall scripts.

All First Boot Package Install components and scripts are available from GitHub using the link on the screen.



## Building VMware Fusion VMs with DeployStudio and createOSXinstallPkg



One potential use of First Boot Package Install would be to allow you to add a systems management agent like Casper, Puppet, Absolute Manage or others to the OS installer. Once the agent reported in, the systems management tool could have its agent install additional software and scripts to configure the VM.

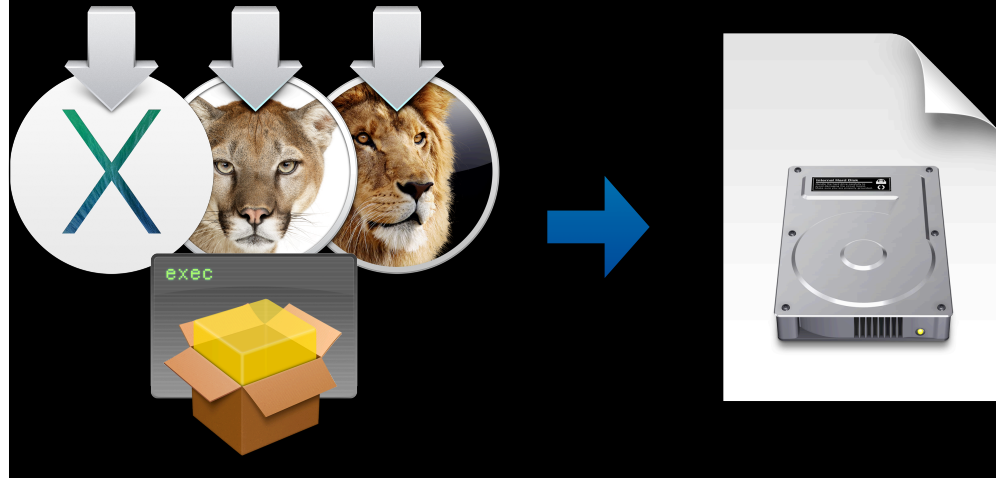
One management tool that would not require First Boot Package Install is Munki, which was also developed by Greg Neagle. Munki's tools can be added directly to a createOSXInstallPkg-built OS X installer.

## Building VMware Fusion VMs without NetBoot



As described previously, my preferred way to create VMs is by leveraging NetBoot and DeployStudio, but not all environments have access to NetBoot or DeployStudio.

## Building VMware Fusion VMs without NetBoot



For those environments, there's scripted ways to create customized Lion, Mountain Lion or Mavericks installer disk images for use with VMware Fusion. This allows the creation of OS X VMs that can configure themselves in an automated fashion without needing access to either NetBoot or server resources.

## Building VMware Fusion VMs using Custom OS X Installer Disk Images



Tim Sutton

<https://github.com/timsutton>

Tim Sutton from Concordia University in Montreal, Canada was the first person I know of who applied this to OS X VMs, as part of his work with Vagrant and Packer.

## Building VMware Fusion VMs using Custom OS X Installer Disk Images



VAGRANT



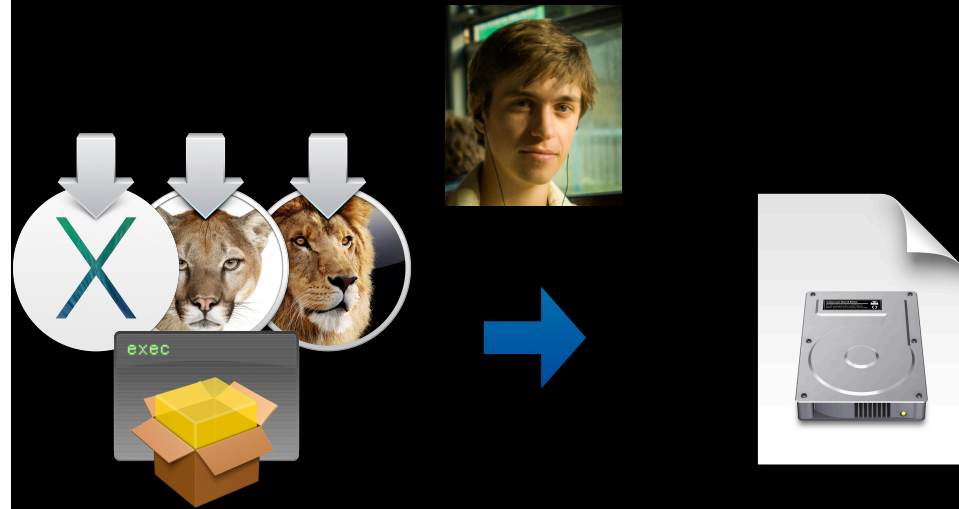
PACKER

<http://www.vagrantup.com>

<http://www.packer.io>

If you haven't previously heard of it, Vagrant is a tool for building virtual machines in a consistent, repeatable way using automated workflows. It's popular with development teams because it allows everyone working on a project to spin up identical virtual machines that contains their shared development environment. In turn, Packer is a complementary open-source tool that is used to build OS images for Vagrant to use when building new VMs.

## Building VMware Fusion VMs using Custom OS X Installer Disk Images



<https://github.com/timsutton/osx-vm-templates>

As part of his work with Packer and VeeWee, another open-source tool for building OS images for Vagrant, Tim developed a script that would convert an OS X installer into a ISO disk image that either VeeWee or Packer could use.

# Building VMware Fusion VMs using Custom OS X Installer Disk Images

## create\_os\_x\_vm\_install\_dmg.sh

```
troutant-ln2:~ troutant$ sudo /path/to/create_vmware_osx_install_dmg.sh "/Applications/Install OS X Mountain Lion.app" /path/to/output_directory
-- Attaching input OS X installer image with shadow file..
expected CRC32 5AF6E80F2
/dev/disk2 Apple.partition_scheme
/dev/disk2s1 Apple.partition_map
/dev/disk2s2 Apple.HFS /private/tmp/vmware-osx-esd.9g08
-- Adding automated components..
-- Unmounting..
"disk2" unmounted.
"disk2" ejected.
-- Converting to final output file..
Preparing imaging engine..
Reading Driver Descriptor Map (DDM : 0)..
(CRC32 517E8046: Driver Descriptor Map (DDM : 0))
Reading (Apple.Free : 1)..
(CRC32 500000000: (Apple.Free : 1))
Reading Apple (Apple.partition_map : 2)..
(CRC32 5E59909CE: Apple (Apple.partition_map : 2))
Reading disk image (Apple.HFS : 3)..
.....
(CRC32 564936FA0: disk image (Apple.HFS : 3))
Reading (Apple.Free : 4)..
.....
(CRC32 500000000: (Apple.Free : 4))
Adding resources..
.....
Elapsed Time: 38.142s
File size: 4422564980 bytes, Checksum: CRC32 58DA8D404
Sectors processed: 9463604, 8733366 compressed
Speed: 111.0Mbytes/sec
Savings: 8.7%
created: /path/to/output_directory/OSX_InstallESD_10.8.4.12ESS.dmg
-- Fixing permissions..
-- Checksumming output image..
-- MD5: b91a766bde59f76921ac1192e5e5e0
-- Done. Built image is located at /path/to/output_directory/OSX_InstallESD_10.8.4.12ESS.dmg. Add this iso and its checksum to your template.
troutant-ln2:~ troutant$
```

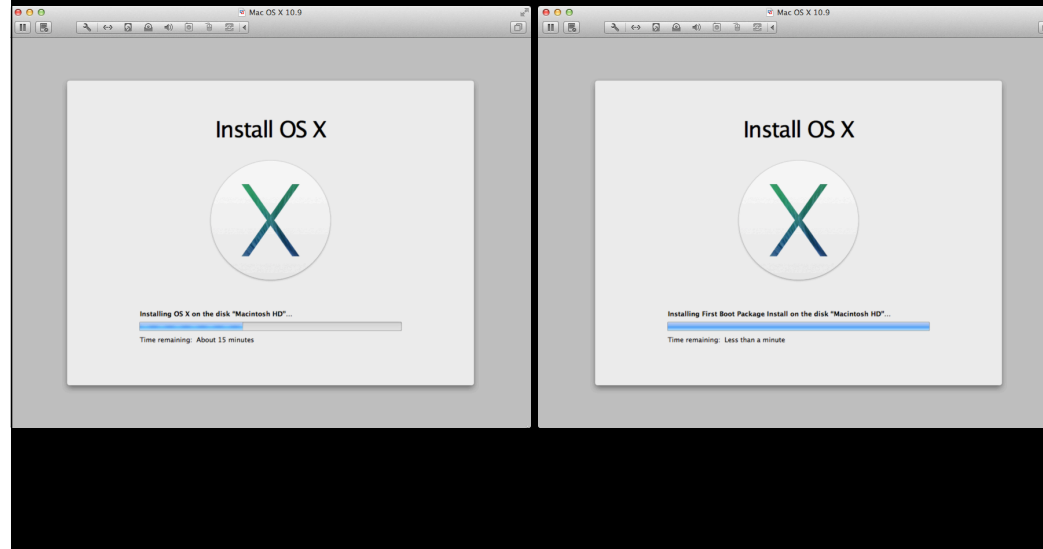
<http://git.io/2L4uag>

I was able to build on Tim's work to develop a script which creates a customized OS X installer that can be used in VMware Fusion without the need for either Vagrant or Packer. My method uses First Boot Package Install to provide customization for the OS X install.

As long as everything is configured according to the directions, users of this script will be able to produce a VMware-ready installer disk image that will install OS X and First Boot Package Install into a new VM.

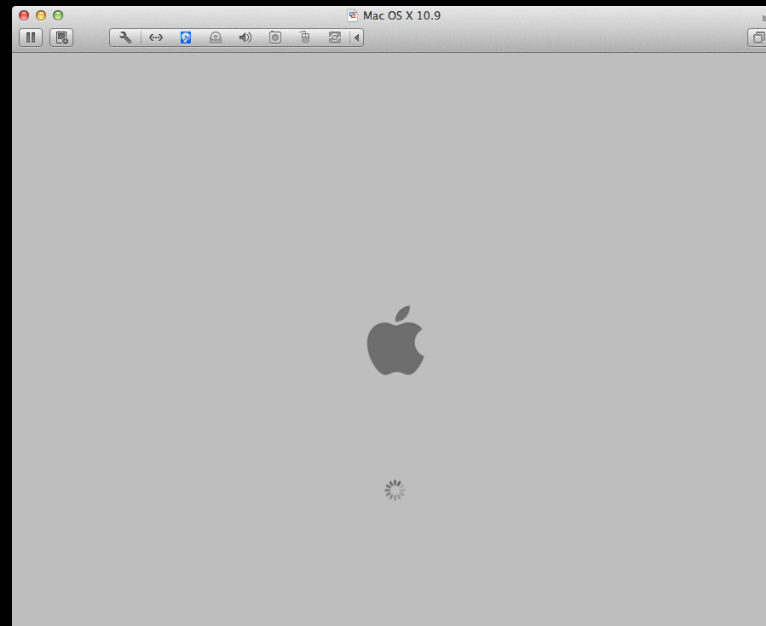


# Building VMware Fusion VMs using Custom OS X Installer Disk Images



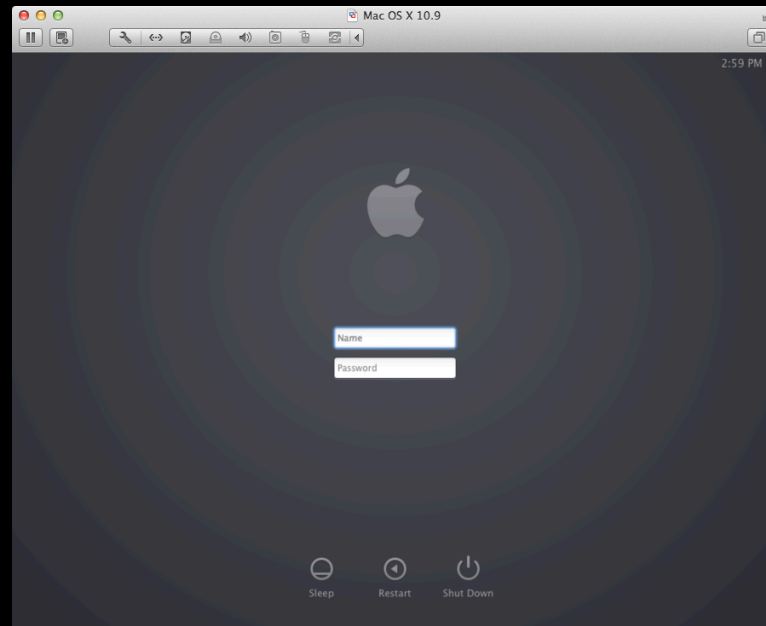
When the VM boots, OS X and First Boot Package Install will automatically install on the VM's boot drive. Once the installation completes, the VM will then reboot.

## Building VMware Fusion VMs using Custom OS X Installer Disk Images



On reboot, the login window will be stopped from loading while the packages included with First Boot Package Install.pkg are installed onto the VM. Once the packages finish installing, the VM will automatically reboot again.

# Building VMware Fusion VMs using Custom OS X Installer Disk Images



After the second reboot, the VM will be set up with the desired applications and settings.

## Building VMs with Custom OS X Installer Disk Images and System Management Tools



I'd previously mentioned this in connection with `createOSXInstallPkg`, but you can also use your existing systems management tools with a custom OS X install disk image to help you build and configure your virtual machines. In this scenario, you can build a simple installation process for your VMs that installs just the OS and your management tool's agent.

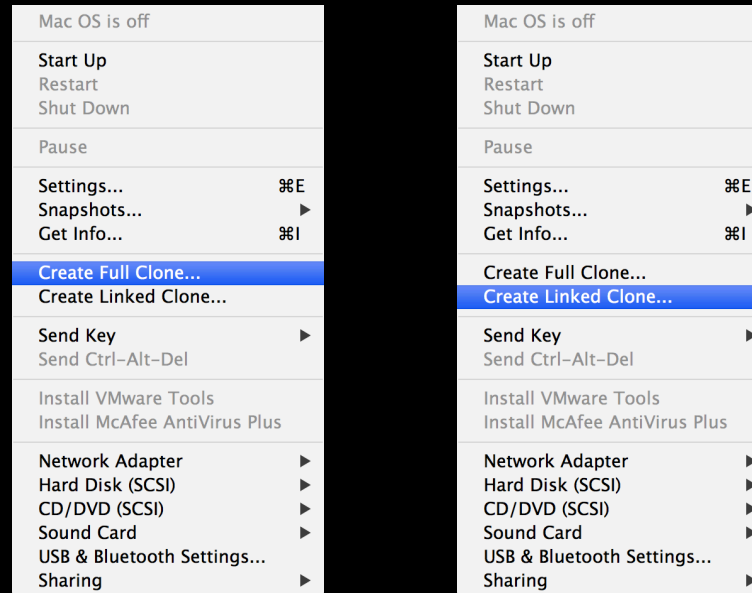
Once the agent on the VM phones home to the management service, the systems management tool can have its agent download and install additional software and scripts to configure the VM. Properly configured, this approach would allow VMs to be built with either no or very little effort on your part.

# Working with VMware Fusion VMs



Once I've got virtual machines built in VMware Fusion, I prefer to use those as templates, or parent VMs. That way, I can use what I've built as a source for other VMs to follow.

# Cloning VMware Fusion VMs



For more info on VMware VM cloning, see <http://tinyurl.com/m6a9l4g>

VMware Fusion 6 added some functionality to facilitate this way of working, by bringing the ability to clone VMs from VMware Workstation on Windows and adding it to VMware Fusion on the Mac. There's two ways that you can clone a VM in VMware Fusion 6. The first is by making a full clone and the second is by making a linked clone.

What's the difference between the two ways of cloning?

A full clone is an independent copy of a virtual machine. Ongoing operation of a full clone is entirely separate from the parent virtual machine and either the clone or the parent VM can be deleted without affecting the other.

A linked clone is made from a snapshot of the parent VM. A snapshot preserves the state and data of a virtual machine at a specific point in time. The state includes what's stored in memory, running applications and anything else the VM was doing at that time.

Using a snapshot saves space because the linked clone can reference all files available on the parent VM as of the time of the linked clone's creation, but it doesn't actually store a complete copy of the files. Because a linked clone is a snapshot of another VM and not a complete copy, a linked clone must have continued access to the parent VM that the snapshot was made from. Without access to the parent VM, a linked clone stops working.

That said, changes to the parent VM do not affect the linked clone, and changes to the disk of the linked clone do not affect the parent.

## VMware Fusion Cloning Rules of Thumb

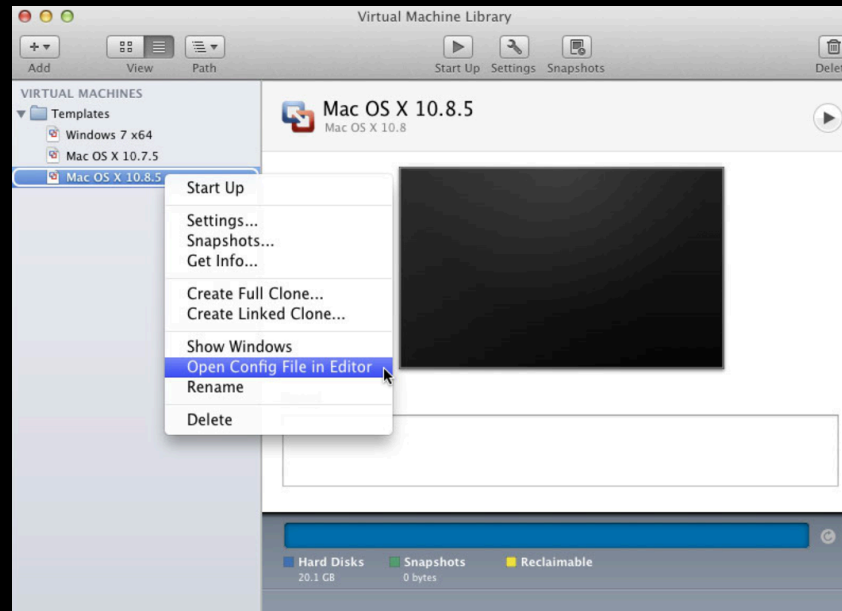
- Linked clone
  - Good for short-term, “test it and toss it” use cases
  - Saves disk space
- Full clone
  - Good for long-term use
  - Complete copy of parent VM

When I’m using clones in Fusion, I have the following rules of thumb. I do a lot of testing where I’m using a particular clone once or twice for a specific purpose, then tossing it. In this case where the test VM will only be around a short time, a linked clone makes a lot of sense because it saves on space and I don’t have to worry about keeping it for the long term.

If I’m planning for a particular clone to stay around for a while, I make a full clone. This way, I don’t have to worry about keeping track of the clone’s parent VM because the full clone is a fully self-contained copy of the parent VM.

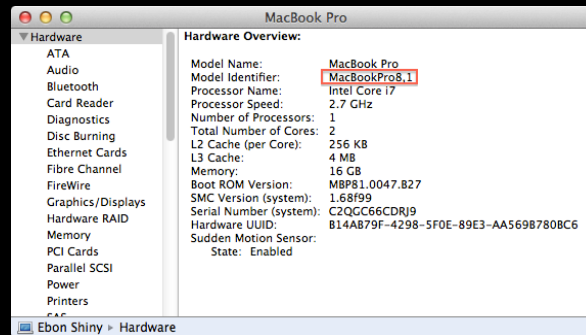


# Emulating specific Apple models in VMware Fusion



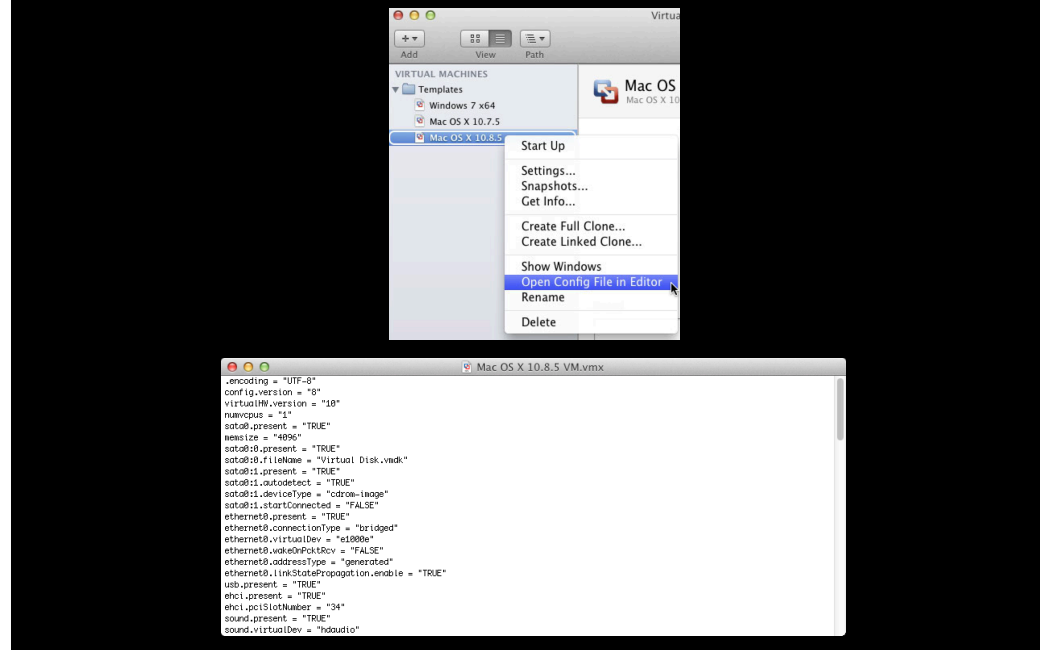
Once you have a VM built, you may want to edit it to emulate a specific Mac model. One reason for doing this would be to test model-specific updates from Apple's Software Update.

# Emulating specific Apple models in VMware Fusion



The first step is to locate the model identifier of the Mac you want to emulate. One way to do this is by checking in System Profiler on an appropriate machine. In the case of the machine on the screen, we're using the model identifier for a 2011 MacBook Pro.

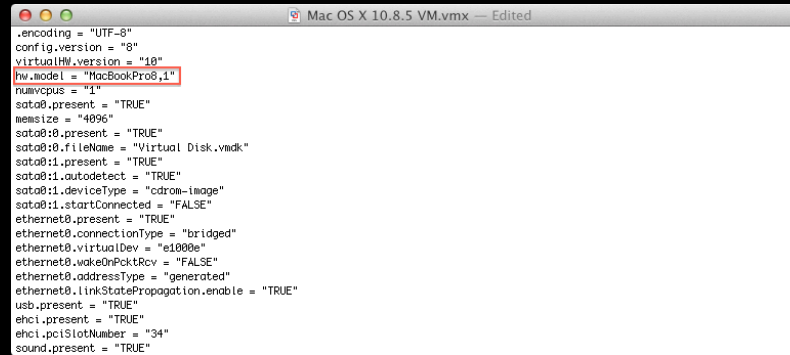
# Emulating specific Apple models in VMware Fusion



To set your VM to report itself as a specific Mac model, you would need to add hardware model settings to your VM's configuration settings. To do this, select the VM you want and make sure it's not running. Next, hold down the Option key on your keyboard and right-click the virtual machine.

Next, select Open Config File in Editor. This will make the VM's configuration available for editing.

# Emulating specific Apple models in VMware Fusion

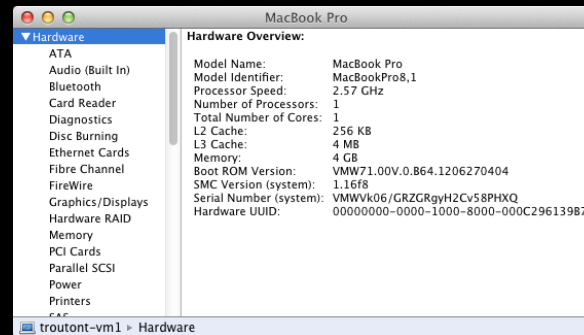


```
.encoding = "UTF-8"
config.version = "8"
virtualHW.version = "10"
hw.model = "MacBookPro8,1"
nuevopus = "1"
sata0.present = "TRUE"
memsize = "4096"
sata0:0.present = "TRUE"
sata0:0.fileName = "Virtual Disk.vmdk"
sata0:1.present = "TRUE"
sata0:1.autodetect = "TRUE"
sata0:1.deviceType = "cdrom-image"
sata0:1.startConnected = "FALSE"
ethernet0.present = "TRUE"
ethernet0.connectionType = "bridged"
ethernet0.virtualDev = "e1000e"
ethernet0.wakeOnPcktRcv = "FALSE"
ethernet0.addressType = "generated"
ethernet0.linkStatePropagation.enable = "TRUE"
usb.present = "TRUE"
ehci.present = "TRUE"
ehci.pciSlotNumber = "34"
sound.present = "TRUE"
```

Add the following line to the VM configuration:  
**hw.model = "model\_here"**

In the configuration editor, you would add a line like that shown on the screen, substituting the actual model where I've got "model\_here". Once your edits are finished, save your changes.

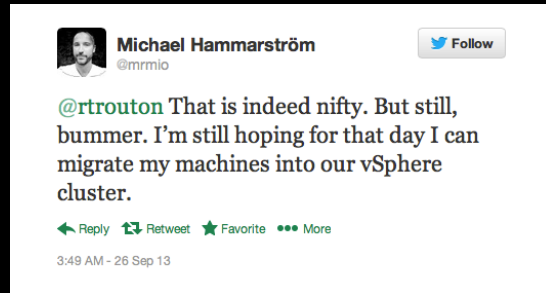
# Emulating specific Apple models in VMware Fusion



The next time you launch the VM, it should identify itself as being the specified Mac model. In the case of our example, the VM should identify itself as a MacBook Pro.



At this point, I'm going to start talking about OS X VMs and VMware's ESXi server. Before I do though, I want to say some things to hopefully save some questions later.



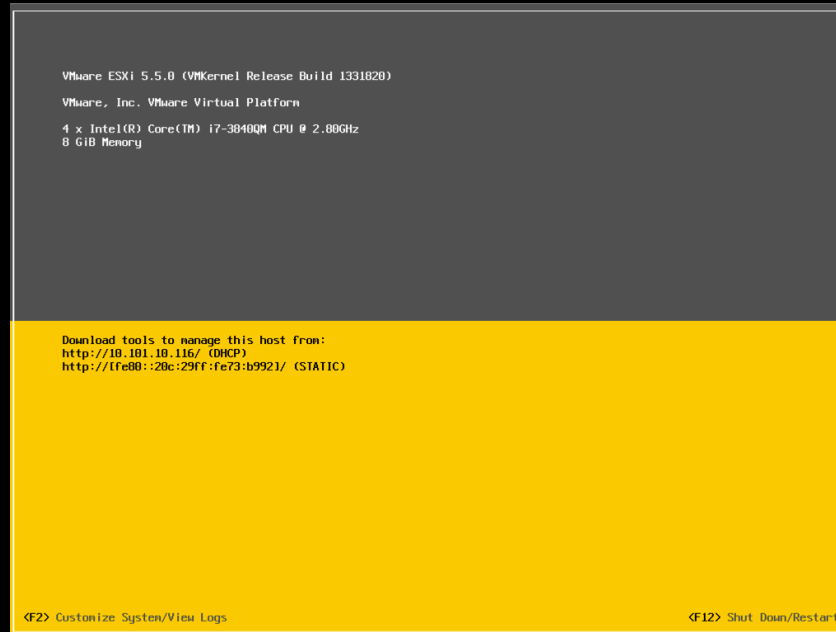
Anytime I mention Apple, VMware and ESXi in proximity to each other, I almost always have a conversation that goes like this. Someone asks if VMware now supports running OS X on non-Apple hardware. I tell him that no, OS X VMs are supported only on Apple hardware. The person usually responds that it would be nice if they could run OS X on their ESXi or vSphere setup like they can with other operating systems.



I fully agree with that opinion, it would be nice. Running OS X on non-Apple hardware is not a technical problem. It is a license issue. The people who can address this issue are at the location shown on the screen.

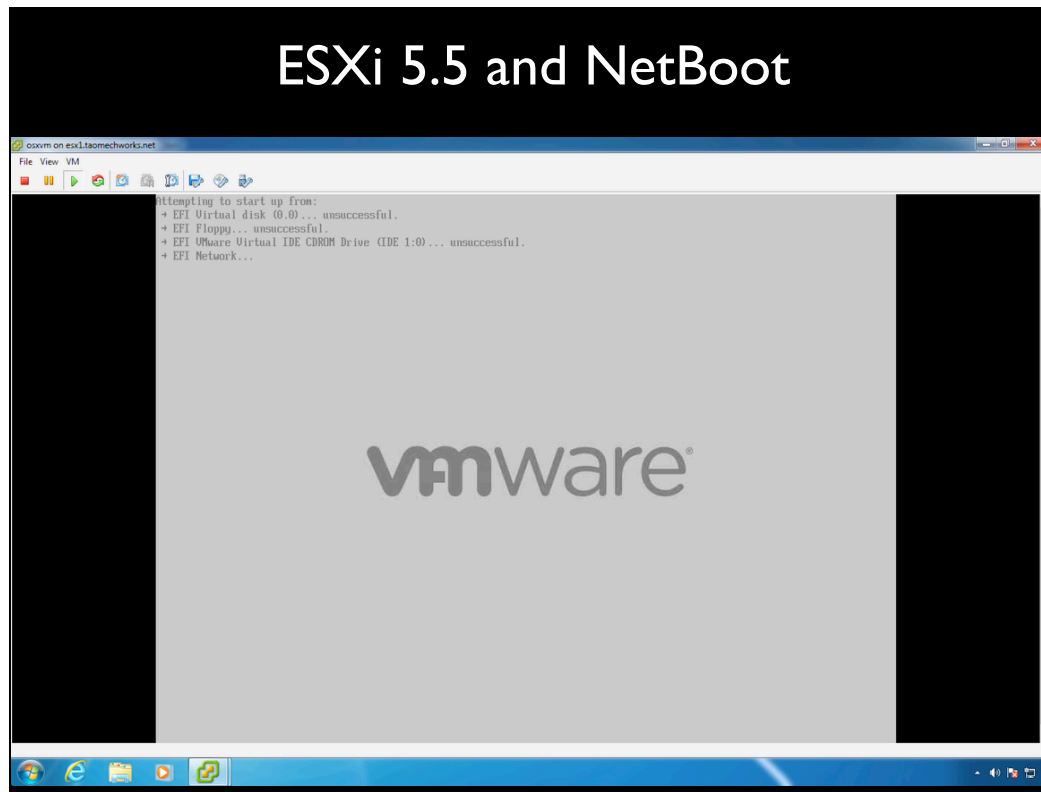


# Building OS X VMs in VMware ESXi



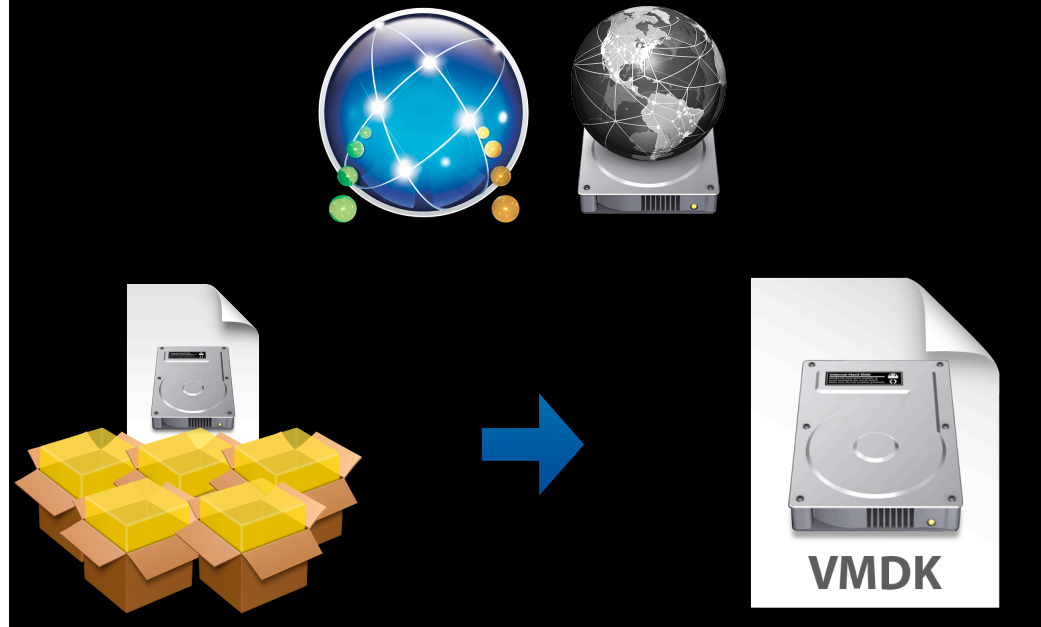
Now that it's understood that I'm only talking about ESXi running on Apple hardware, let's talk about hosting OS X VMs on ESXi. In particular VMware brought some support over to ESXi 5.5 which completely changed how I built VMs on ESXi.

# ESXi 5.5 and NetBoot



What changed things was that VMware has now added NetBoot support for ESXi-hosted OS X VMs. In fact, you can stand up a NetBoot server in one ESXi OS X VM and use it to boot other ESXi-hosted VMs.

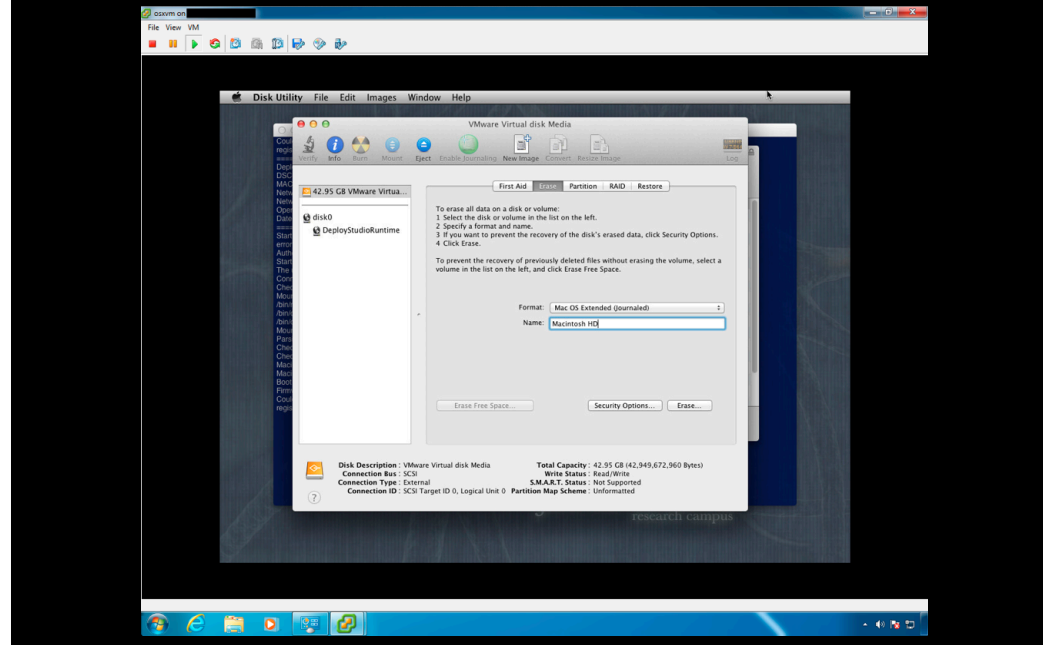
## Building VMware ESXi 5.5 OS X VMs with DeployStudio



Needless to say, this greatly simplified my build process because I can now leverage the same NetBoot deployment tools that I'm using with VMware Fusion.

Support for NetBoot in ESXi 5.5 has all but eliminated any need for me to build OS X VMs in Fusion first and then transfer them to an ESXi server.

# ESXi 5.5 OS X VM boot drives need to be formatted before use



One difference between building OS X VMs in Fusion and building them in ESXi is that Fusion will format the VM's boot drive as part of the creation of the VM. ESXi-hosted OS X VMs need to have their drives formatted. Unfortunately, DeployStudio's disk partitioning tools are not able to correctly detect the unformatted drive on an ESXi-hosted OS X VM like they can on a Mac, so it's difficult to include an automated format of the drive as part of the VM build process.

Fortunately, this issue can be solved while still booted into DeployStudio. One of the tools available on the DeployStudio boot set is Apple's Disk Utility, which can be used to format the drive before running a DeployStudio workflow on it.

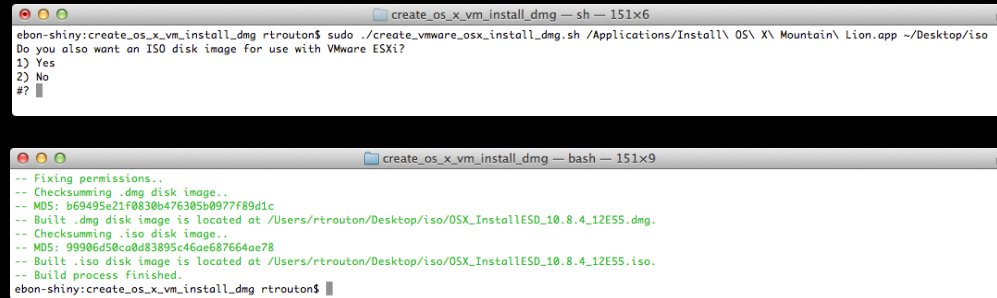
## Building VMware ESXi 5.5 OS X VMs without NetBoot



Even with the ability to host a NetBoot server on ESXi, there are going to be environments where you can't use NetBoot. You can solve this issue also, using a tool we'd looked at previously for VMware Fusion.

# Building VMware ESXi OS X VMs using custom OS X installer disk images

## create\_os\_x\_vm\_install\_dmg.sh



```
ebon-shiny:create_os_x_vm_install_dmg rtrouton$ sudo ./create_vmware_osx_install_dmg.sh /Applications/Install\ OS\ X\ Mountain\ Lion.app ~/Desktop/iso
Do you also want an ISO disk image for use with VMware ESXi?
1) Yes
2) No
#? █

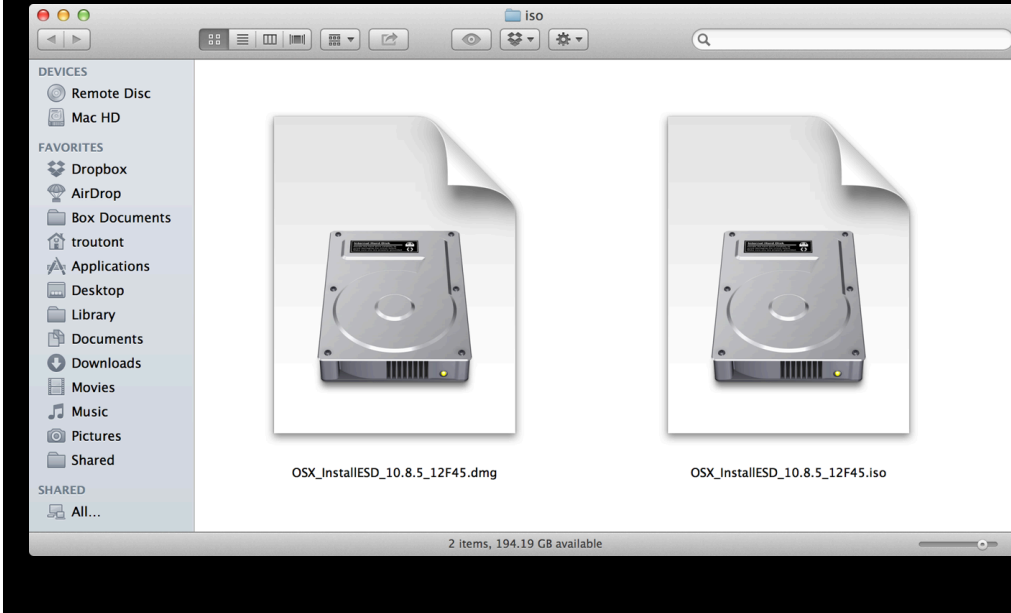
ebon-shiny:create_os_x_vm_install_dmg rtrouton$ █

-- Fixing permissions..
-- Checksumming .dmg disk image..
-- MDS: b69495e21f0830b476305b0977f89d1c
-- Built .dmg disk image is located at /Users/rtrouton/Desktop/iso/OSX_InstallESD_10.8.4_12E55.dmg.
-- Checksumming .iso disk image..
-- MDS: 99906d50ca0d83895c46ae687664ae78
-- Built .iso disk image is located at /Users/rtrouton/Desktop/iso/OSX_InstallESD_10.8.4_12E55.iso.
-- Build process finished.
ebon-shiny:create_os_x_vm_install_dmg rtrouton$ █
```

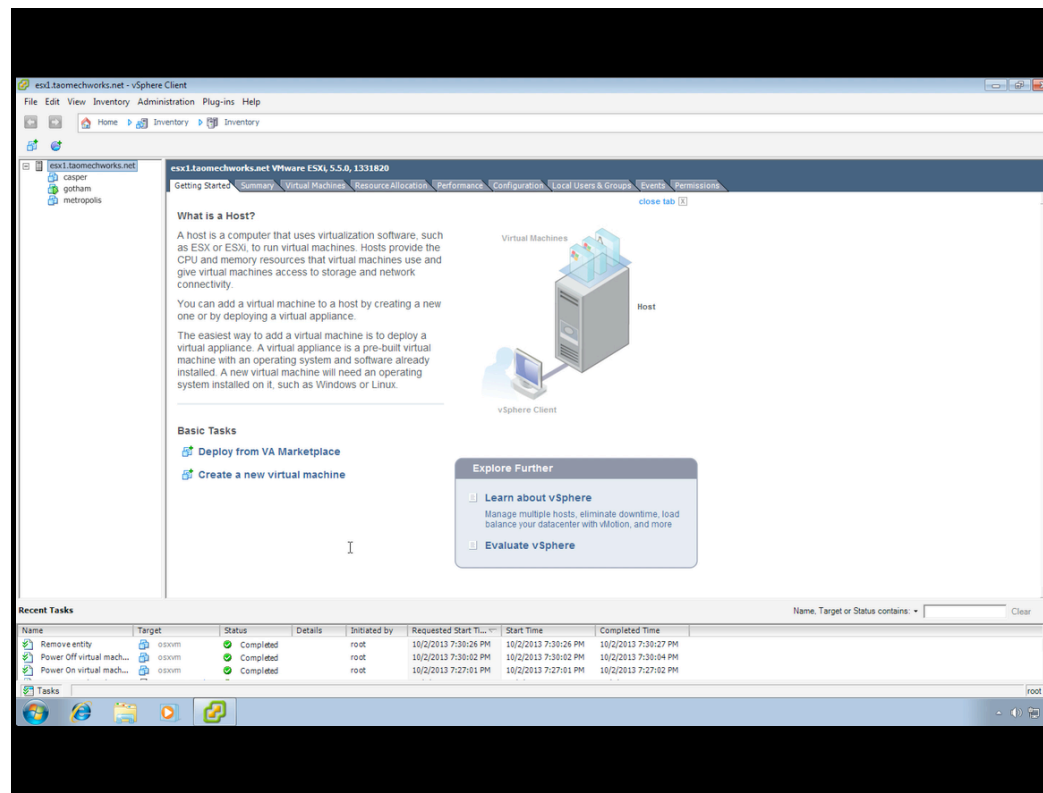
<http://git.io/2L4uag>

One of the functions built into the custom OS X installer build script we discussed earlier is the option of creating an ISO disk image for use with ESXi.

# Building VMware ESXi OS X VMs using custom OS X installer disk images



Selecting the option in the script to create an ISO for ESXi will create two disk images, one for VMware Fusion and the other being an ISO for ESXi. You can use the ISO as an installer disk for an ESXi VM, much like you can use an OS X disk image as an installer disk for VMware Fusion VMs.



Let's take a look at what the process of building an ESXi-hosted VM with a custom OS X installer looks like. In this instance, we're building a VM using Mountain Lion.



# What can't be tested in VMware VMs?

- Processes that request an Apple-registered hardware serial number
- Wi-Fi
- EFI functionality\*
- Software that requires external hardware\*

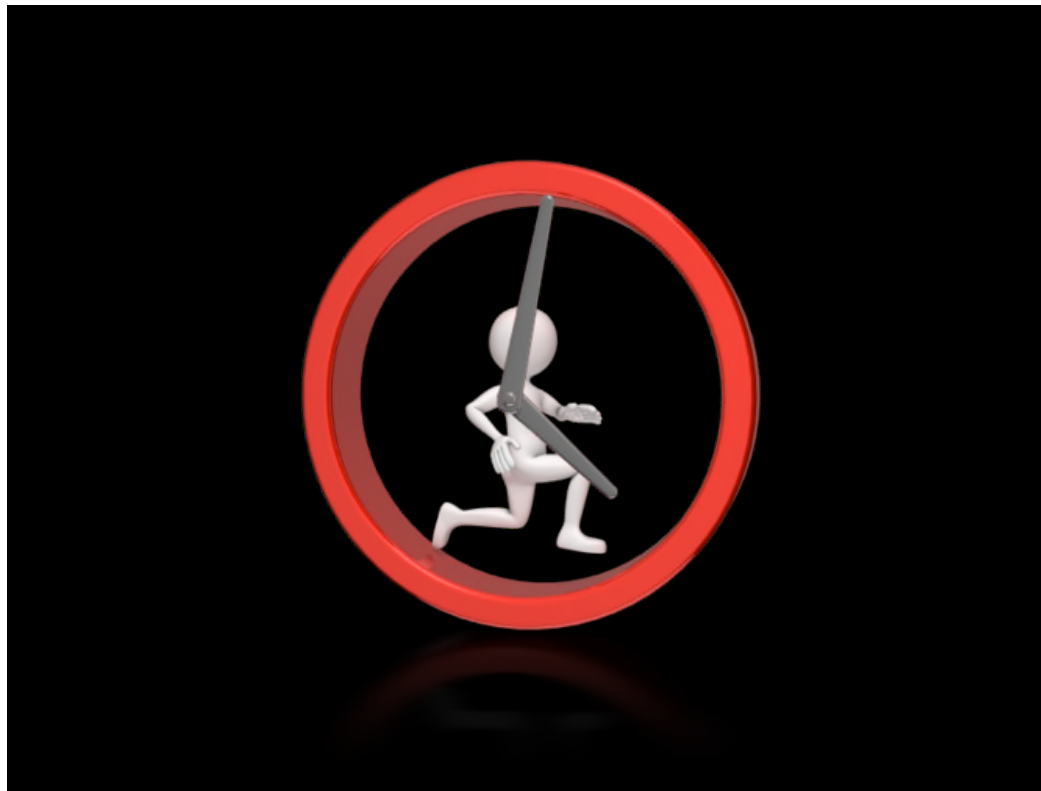
\*Exceptions may apply if VMware supports it

VMs do have some limitations because they are software constructs and not actual hardware. Here's what I've found can't be tested in a virtualized environment.

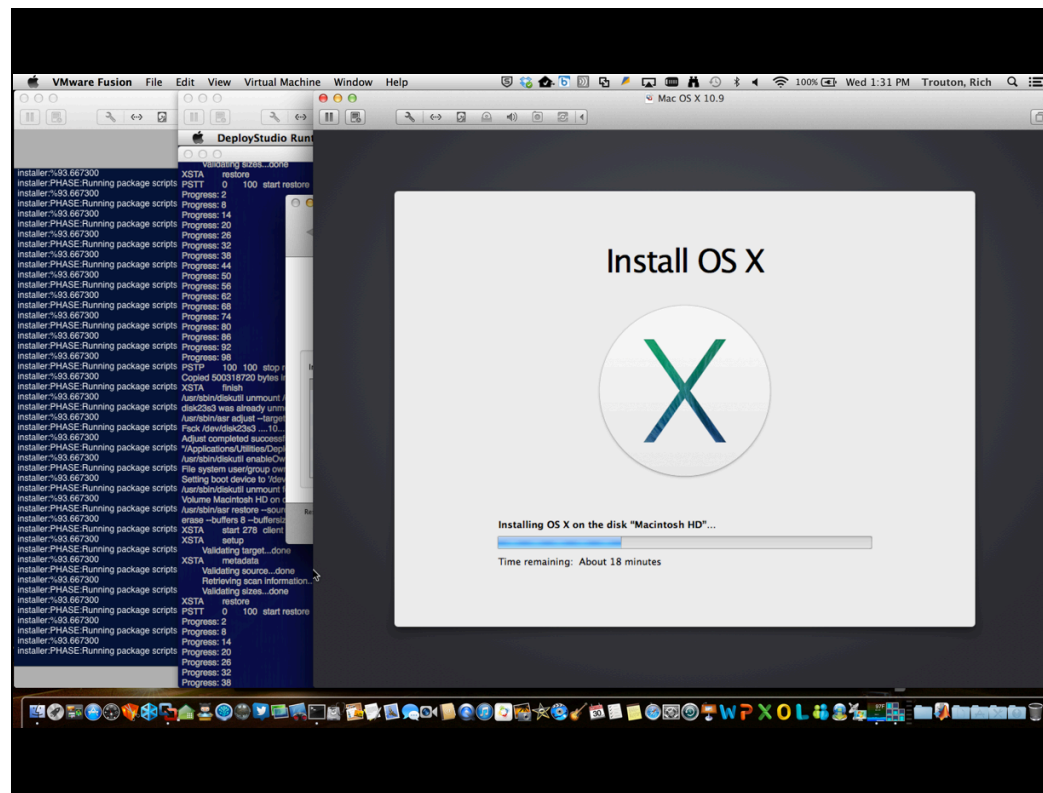
Anything involving having a Apple-registered hardware serial number / sending hardware serial number back to Apple - This includes iCloud services like Find My Mac and Messages. It also applies to getting hardware-specific OS installers via Recovery HD.

Most things involving EFI - Functions like Apple Internet Recovery or holding down the Option key to get a list of bootable volumes will not work. However, some things involving EFI work specifically because VMware made them work. For example, both NetBoot and FileVault 2 work fine in a VMware VM.

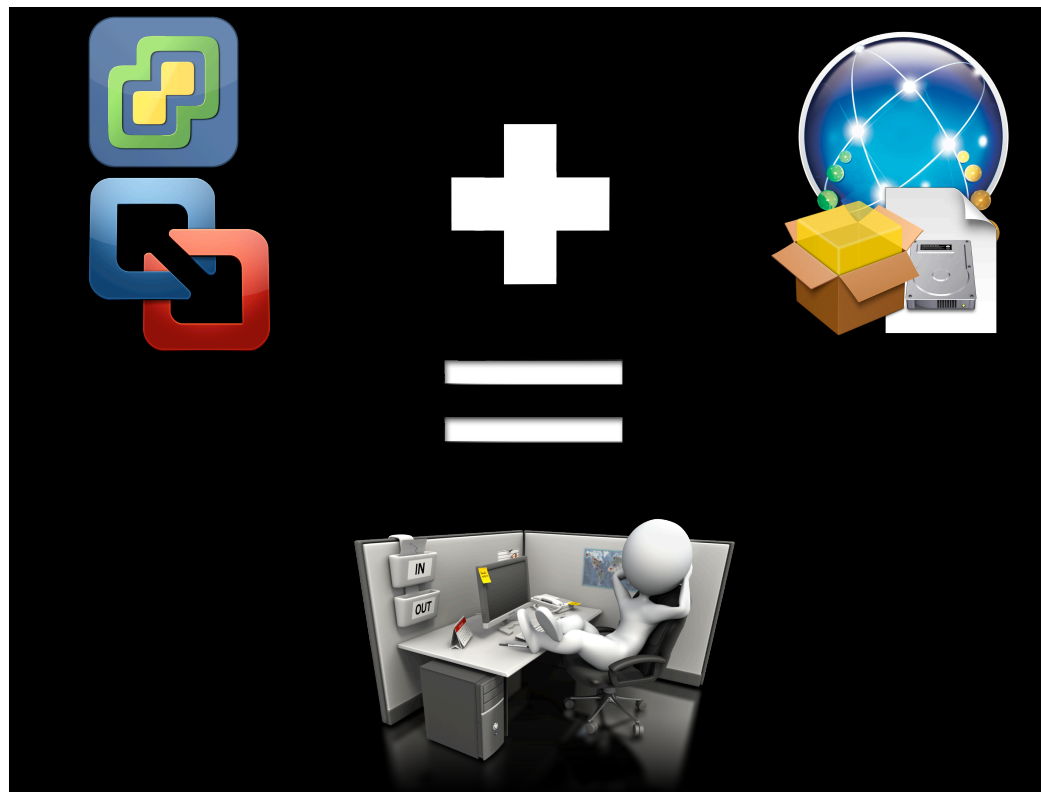
Wireless connections - Your VM doesn't have a WiFi card, though it may talk to your network via your Mac's WiFi connection. You can test in a VM to make sure that your WiFi settings apply, you can't test to verify that they work.



As Mac admins, we're always against the clock for testing. In my own environment, I've become reliant on virtual machines to speed up my development and testing cycle while reducing the physical footprint of my test environment. Instead of needing multiple machines to test changes to my deployment workflows, my testing now takes place almost exclusively on a quad-core Retina MacBook Pro with 16 GBs of RAM.



Likewise, having NetBoot and DeployStudio available to build test VMs means that I can be testing multiple workflows simultaneously on the same laptop. If a particular build hits a problem, I can discard that VM, fix the problem, then quickly create a new VM with the updated build.



To sum up, virtual machines plus automated build processes equals more time for you.

Virtual machines running OS X can be resource hogs, as you will need to assign at least one processor as well as 3 - 4GBs of RAM to have them run at usable speeds. That said, the time and resource savings realized by using virtual machines instead of actual hardware should help make the case for investing in one or two speedy Macs running virtual machines instead of a multitude of actual test machines.

# Useful Links

- Setting up ESXi 5.1 on a 2011 Mac Mini Server: <http://derflounder.wordpress.com/2013/01/21/setting-up-esxi-5-1-on-a-2011-mac-mini-server/>
- First Boot Package Install.pkg: <http://derflounder.wordpress.com/2013/05/13/first-boot-package-install-pkg/>
- Emulating specific Apple models in VMWare Fusion VMs: <http://derflounder.wordpress.com/2013/02/19/emulating-specific-apple-models-in-vmware-fusion-vms/>
- Updating my ESXi server from 5.1 to 5.5: <http://derflounder.wordpress.com/2013/09/26/updating-my-esxi-server-from-5-1-to-5-5/>

# Useful Links

- Apple Mac Pro 5,1 now on VMware HCL for ESXi 5.5: <http://www.virtuallyghetto.com/2013/10/apple-mac-pro-51-now-on-vmware-hcl-for.html>
- Running ESXi 5.5 on Apple Mac Mini + Thunderbolt Ethernet Adapter Caveat: <http://www.virtuallyghetto.com/2013/09/running-esxi-55-on-apple-mac-mini.html>
- Running ESXi 5.0 & 5.1 on 2012 Mac Mini 6,2: <http://www.virtuallyghetto.com/2012/12/running-esxi-50-51-on-2012-mac-mini-62.html>
- How to change hardware serial number for Mac OS X Guest: <http://www.virtuallyghetto.com/2013/10/how-to-change-hardware-serial-number.html>

# Useful Links

- Creating customized OS X installer disk images for VMware Fusion: <http://derflounder.wordpress.com/2013/08/02/creating-customized-os-x-installer-disk-images-for-vmware-fusion/>
- VMware custom OS X installer script adds support for VMware ESXi: <http://derflounder.wordpress.com/2013/08/10/vmware-custom-os-x-installer-script-adds-support-for-vmware-esxi/>
- create\_vmware\_osx\_install\_dmg script updated with Mavericks support: [http://derflounder.wordpress.com/2013/10/23/create\\_vmware\\_osx\\_install\\_dmg-script-updated-with-mavericks-support/](http://derflounder.wordpress.com/2013/10/23/create_vmware_osx_install_dmg-script-updated-with-mavericks-support/)

# Useful Links

- Automatically Partition Virtual Hard Drives when Netbooting OS X Images from ESXi 5.5: <http://cobbservations.wordpress.com/2013/10/15/automatically-partition-virtual-hard-drives-when-netbooting-os-x-images-from-esxi-5-5/>
- Building OS X Test Environments in VMWare Fusion w/ System Image Utility: <http://cobbservations.wordpress.com/2013/06/11/92/>
- createOSXinstallPkg page on Munki wiki: <http://code.google.com/p/munki/wiki/InstallingOSX>



# Downloads

PDF available from the following link:

<http://tinyurl.com/mt2013vmPDF>

Keynote slides available from the  
following link:

<http://tinyurl.com/mt2013vmkeynote>