

HOWTO: Manual Linked Cloning in Fusion

Disclaimer: This is a personal document and is not official or endorsed by VMware. Feedback and suggestions are welcome.

Note : This is **very unsupported**. Normally you'd do this through the UI or vmrun, but Fusion does not currently support cloning.

Introduction

A large number of virtual machines usually means a large amount of space needed to store them. However, if you happen to be running similar setups on multiple virtual machines, you may be able to **save space by reducing duplication of common files between identical virtual machines** via a combination of snapshot and cloning. Note this is not deduplication - you can't smoosh two existing virtual machines together, the technique only works when creating new ones. Workstation users may recognize this as being very similar to Linked Clones.

This is a fairly advanced technique that is **only useful in a few situations** and has some drawbacks. While it's always a good idea to read things through fully before trying them, it is especially true here so you know what to expect. I assume you are comfortable getting inside .vmwarevm bundles and are familiar with the concepts in [A Beginner's Guide to VMware Fusion](#).

Difficulty Level

Advanced : You should be familiar with editing .vmx files and be experienced with creating and using virtual machines, and preferably other VMware products. Additionally, the group of people interested in doing this in the first place is expected to be power users.

Motivation and Use Case

I like sandboxing applications not only in virtual machines, but having a separate virtual machine for each task -- this way if there's some problem with program A, not only will it not spill over to the host, but it also won't affect program B (which is in a different virtual machine). Each virtual machine is clean for that application (no build up of cruft from incomplete uninstalls of unrelated programs). Sometimes application upgrades require guest OS upgrades or vice versa; if everything were on one virtual machine I would have to make sure all my programs worked in the new setup before moving. With per-application virtual machines I can upgrade piece by piece as I want. This isn't something I suspect a lot of people do, and is definitely not something a normal Fusion user would ever need, but for me it's useful.

One thing I do to get this is to keep a clean copy of my commonly-used guest OSes - to create a new virtual machine all I have to do is copy the master template. It's way better than sitting through a new install, and even Easy Install takes a while. However, creating copies takes space, and with a limited amount on my laptop, this isn't a great solution.

Another possible approach would be to have a snapshot branch per application. This would achieve the space savings by not duplicating the base install, but has the drawback that you can't run multiple branches at the same time.

Enter Linked Clones, which get you space savings while also allowing you to run multiple versions simultaneously.

Limitations

You must be running the **same guest at the same patch level across multiple virtual machines**. This *won't* help if you have one XP virtual machine, one Vista, one OS X, and so on - but it *will* help if you have 8 Ubuntu 8.10 virtual machines.

This is **not appropriate if you have a single copy of Windows or other non-free OS** (from a technical point of view it works, but we're basically creating multiple installs, which is probably against the EULA). For volume-licensed versions or free OSes, this technique is fine.

You **will not be able to shrink these guests** or expand the disk (since a snapshot is involved). I don't think this is much of a drawback as each virtual machine has a pretty specific purpose, so should not grow that much. Also, the space savings from eliminating duplication should outweigh not being able to shrink.

The cloned virtual machines will have **external references to the base disk**. If you want to move such virtual machines around, you need to copy the base disk too.

Along the same lines, if anything happens to the base disk, **all the dependent virtual machines will be unusable**. I get around this in two ways - **I keep a backup of the base disk** which I can restore, and **I don't keep data I care about in the guest** (so if anything does go wrong *and* I can't restore the base disk, my data is still safe).

To head off possible confusion, you will not be able to update the base disk (say to include a new system update) and have all the children pick up the changes, but will instead have to apply the change to each clone individually. Over time, this will cause the clones to get bigger. This is also true of snapshots - you can't update the base snapshot and have all the child snapshots pick up the change (that defeats the point of a snapshot), so this disadvantage is not unique to this technique.

Instructions

Overview

When you take a snapshot, the original virtual disk becomes read-only and a new COW disk is created. We're going to set up a virtual machine, then create new ones using that as the base disk and snapshot so they never try to modify the base disk. I will use Ubuntu 8.10 as an example.

Prepare the guest

Install the guest as you normally would, getting all updates, installing Tools, and so on (I also like to install any commonly used software or plugins). For the sake of example, I keep my virtual machines in /Users/etung/Virtual Machines/. Let's call the base virtual machine "8.10 Core".

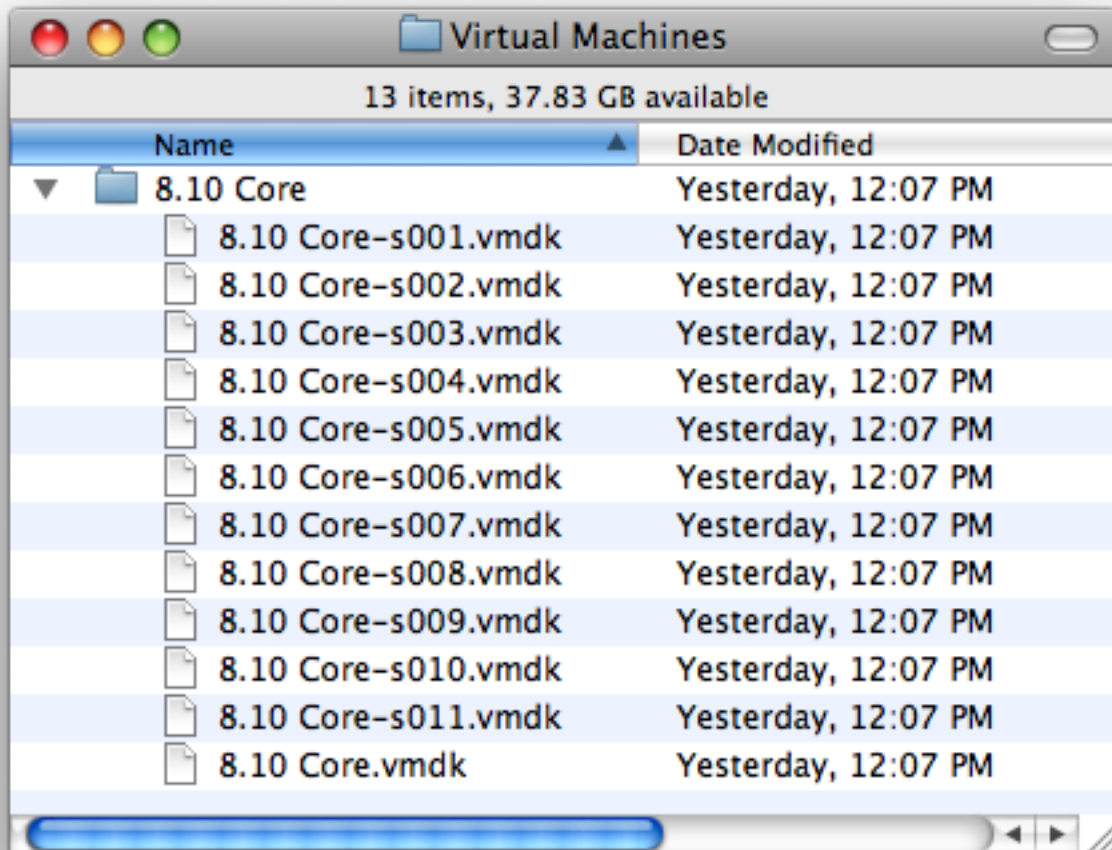
Important: if you plan to run multiple versions simultaneously, the system must be set up so that it can be easily cloned, e.g. there are no unique identifiers lurking in the system. For example, Windows has a SID and you must sysprep the guest. The specifics of doing this are beyond the scope of this document, consult the documentation for your guest OS. In our example of Ubuntu 8.10, no additional setup is needed (as far as I know).

Another good thing to do at this point is to shrink the guest.

Shut down the virtual machine and Fusion. You may wish to verify your setup by making two copies of the base virtual machine and running them simultaneously. Tell Fusion you copied them. If set up properly, both should work at the same time (e.g. access the internet). After verification, shut down the virtual machines and quit Fusion - you can delete the copies made for verification.

I highly recommend backing up the base virtual machine at this point. As noted in the Limitations section, if anything happens to the base virtual disk, all the clones will stop working (bad!). Having a backup will work around this problem.

Since we only care about the virtual disk, not the virtual machine, I would also highly recommend moving the virtual disks out of the .vmwarevm bundle and deleting the bundle -- this makes it harder to accidentally modify the base virtual disk. I created a new folder, /Users/etung/Virtual Machines/8.10 Core/, and moved the vmdk files there.



Fix paths

Open the plaintext metadata file "8.10 Core.vmdk" in your favorite text editor. This file contains paths to the actual data files, and we need to fix them up.

```
# Disk DescriptorFile
```

HOWTO: Manual Linked Cloning in Fusion

```
version=1
encoding="UTF-8"
CID=242293ca
parentCID=ffffffff
createType="twoGbMaxExtentSparse"

# Extent description
RW 4192256 SPARSE "8.10 Core-s001.vmdk"
RW 4192256 SPARSE "8.10 Core-s002.vmdk"
RW 4192256 SPARSE "8.10 Core-s003.vmdk"
RW 4192256 SPARSE "8.10 Core-s004.vmdk"
RW 4192256 SPARSE "8.10 Core-s005.vmdk"
RW 4192256 SPARSE "8.10 Core-s006.vmdk"
RW 4192256 SPARSE "8.10 Core-s007.vmdk"
RW 4192256 SPARSE "8.10 Core-s008.vmdk"
RW 4192256 SPARSE "8.10 Core-s009.vmdk"
RW 4192256 SPARSE "8.10 Core-s010.vmdk"
RW 20480 SPARSE "8.10 Core-s011.vmdk"

# The Disk Data Base
#DDB

ddb.virtualHWVersion = "7"
ddb.uuid = "60 00 C2 99 ab ce aa 92-d3 3e 55 a9 30 e7 c6 dd"
ddb.geometry.cylinders = "2610"
ddb.geometry.heads = "255"
ddb.geometry.sectors = "63"
ddb.adapterType = "lsilogic"
ddb.toolsVersion = "7460"
```

We want to change the paths so they'll work from other locations. Since I keep all my virtual machines in /Users/etung/Virtual Machines/, I can do something like the following:

```
# Disk DescriptorFile
version=1
encoding="UTF-8"
CID=242293ca
parentCID=ffffffff
createType="twoGbMaxExtentSparse"

# Extent description
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s001.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s002.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s003.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s004.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s005.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s006.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s007.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s008.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s009.vmdk"
RW 4192256 SPARSE "../8.10 Core/8.10 Core-s010.vmdk"
RW 20480 SPARSE "../8.10 Core/8.10 Core-s011.vmdk"

# The Disk Data Base
```

```
#DDB
ddb.virtualHWVersion = "7"
ddb.geometry.cylinders = "2610"
ddb.geometry.heads = "255"
ddb.geometry.sectors = "63"
ddb.adapterType = "lsilogic"
ddb.toolsVersion = "7460"
```

Another option would be to use an absolute path instead (e.g. instead of "../8.10 Core/8.10 Core-s001.vmdk", use "/Users/etung/Virtual Machines/8.10 Core/8.10 Core-s001.vmdk", etc.). I'm not sure if it's necessary, but I also took out the ddb.uuid line to be safe against duplicate uuids.

Write Protect

Finally, make sure the .vmdk files are **not** writable - this provides another layer of defense against accidental changes. You could do this by doing a Get Info on each file, or by running the following Terminal command:

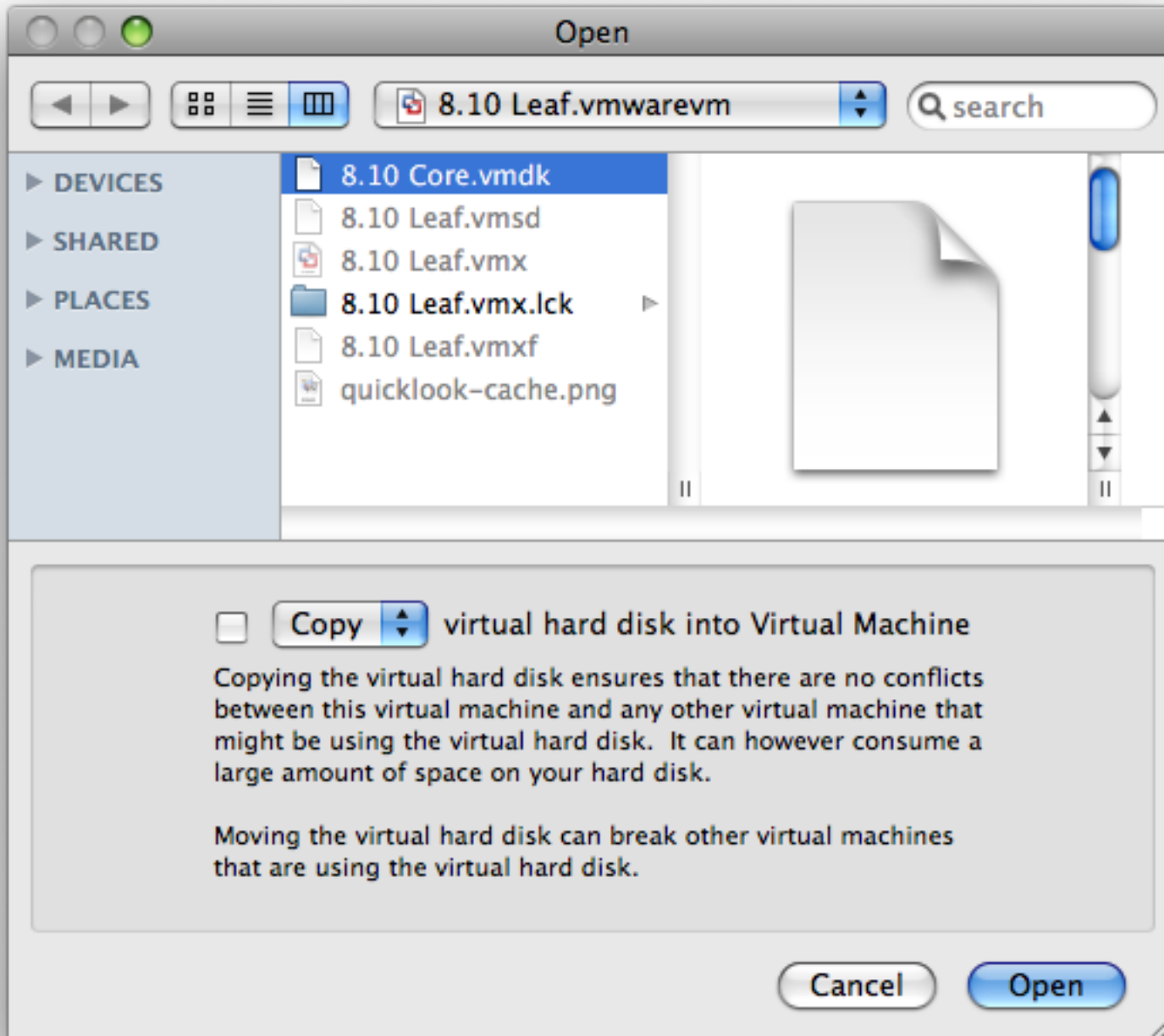
```
chmod a-w
/Users/${USER}/Virtual\ Machines/8.10\ Core/*.vmdk
```

Of course, replace the path as appropriate for your setup.

Create a Clone

We're now ready to clone. Create a new virtual machine in Fusion. Continue without the disk, and create a custom virtual machine. Select the appropriate guest OS type (in this example, Linux/Ubuntu). Choose to customize the default settings. In this example, I'll call the new virtual machine "8.10 Leaf" and save it in my usual location, /Users/etung/Virtual Machines/.

In the Hard Disk settings pane, delete the existing virtual disk by pressing the minus button. Go inside the 8.10 Leaf.vmwarevm bundle and delete the .vmdk files (optional). Copy the small plaintext metadata file from /Users/etung/Virtual Machines/8.10 Core/8.10 Core.vmdk to the 8.10 Leaf.vmwarevm bundle. Back in the Hard Disk settings pane, press the + button, then selecting "Choose existing disk..." for the file name. Choose the copied metadata file (the one in 8.10 Leaf.vmwarevm, not the one in the 8.10 Core folder) and uncheck the checkbox - we're happy with the file where it is, we don't want to copy or move it.



The reason we copied the metadata file ourselves instead of letting Fusion do it is that Fusion would have also copied all the slices, defeating the space saving point of this technique. The reason we made a copy of the metadata file instead of leaving it where it is because the lock file is created in the same place as the metadata file - if we left it in the 8.10 Core folder, all the clones would be trying to use the same lock file, which would defeat the simultaneous point of this technique.

Snapshot

Take a snapshot of the virtual machine, which will tell Fusion not to try to write to the original disk. I named mine "Base" with the comment "Don't delete!". Even if you run without the snapshot, because of the read-only permissions on the .vmdk files you shouldn't be able to change the original vmdk, but the leaf virtual machine will not be happy.

You should now be able to run the clone. For more clones, repeat the Create a Clone section.