Enterprise Imaging Mac OS X

Quick Start Guide to creating an Enterprise Image of Mac OS X. Feel free to contact me for more information or to offer me a job :) via chris.gerke@gmail.com

This guide uses the concept of Package-based thin imaging. The same process can be used to make a monolithic image if you so choose.

Some great resources:

http://www.macenterprise.org/mailing-list
http://managingosx.wordpress.com/
http://derflounder.wordpress.com/
http://krypted.com

Getting Started

Generally speaking you can create a single “universal” image that can be used across all Mac models, provided you keep it simple and don't include anything model specific. However its not uncommon for Apple to introduce new hardware throughout the year that requires a slightly different “fork” of the Mac OS X operating system.

So the first thing I do when I receive a shipment of new macs is to capture a copy of the InstallESD.dmg file from Apple for each different model. This is to ensure the “build” number matches my image, if it does not, then you don't really have an alternative other than to wait for the next iteration of the Mac OS X operating system to be released before you can resume using a single DMG image for your systems.

(There are alternatives, such as not using DMG imaging and instead use a PKG only workflow, but this is a quick start guide so I am not going into it).

InstallESD.dmg

“Electronic Software Distribution”, just a disk image including all the pieces needed to install a Vanilla copy of Mac OS X.

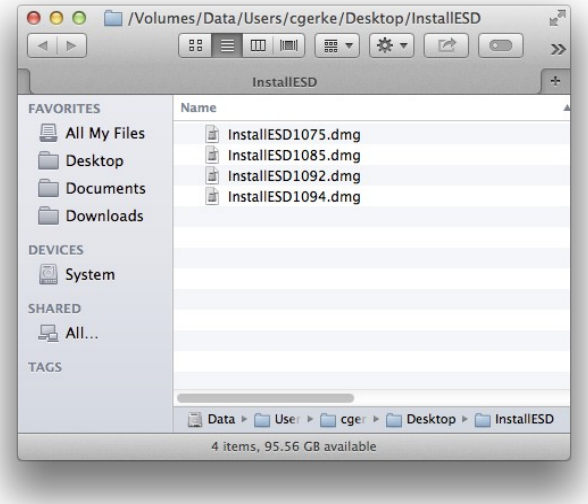
1. Download the OS X installer from the AppStore.
2. Extract the InstallESD.dmg from the following location.

```
/Applications/Install\ OS\ X\ Mavericks.app/Contents/SharedSupport/InstallESD.dmg
```

3. Check the build version by mounting the InstallESD.dmg and using the following command;

```
defaults read "/System/Library/CoreServices/SystemVersion" ProductBuildVersion
```

4. I generally rename the InstallESD.dmg file to include the OS version and build number if I need to manage multiple OS revisions.



Create a vanilla Disk Image

Run this, passing the path to your required InstallESD file.

https://github.com/will-i-am-79/osx/blob/master/system_builder.bash PATH_TO_INSTALLESD

This is a bash script i've written that will create a basic Mac OS X install, it has the option to also install extra PKG files it finds in the current working directory of the script.

Take a look at the comments for an explanation, but as a summary, all that is happening is;

1. A 20g sparse disk is created with a volume name “System” (these variables can be changed to your liking).
2. /Volumes/OS\ X\ Install\ ESD/Packages/OSInstall.mpkg is run targeting the sparse volume.
3. Any pkg files found in the script working directory are installed, targeting the sparse volume.
4. The sparse disk is compressed and asr scanned so it can be restored.

It's really that simple, the tricket part is understanding how to build PKG files but even that is a trivial exercise.

In a typical enterprise image you would probably only need packages that do the following (and these really depend on your environment).

1. Create a local admin user (or two).
2. Disable Registration Wizard
3. Disable Setup Assistant
4. Disable iCloud Setup
5. Enable and configure Remote Assistance
6. Enable auto login to run some localisations if necessary.
7. Enable a first boot script.
8. Localisations

But as I say these can differ depending on your infrastructure. An example scenario would be;

1. Create a thin image that is simply a vanilla Mac OS X install with a single package included that incorporates your software management system agent.
2. Restore dmg
3. System boots
4. First boot configures some items such as (keep it minimal, get your software management system to do the heavy lifting);

```
#++ disable gatekeeper
/usr/sbin/spctl -master-disable
```

5. System auto login
6. Provide a localisation interface so the tech/user can set a hostname that utilises a naming convention (again this can be automated, why not have a mac address lookup via your asset management system then you can name the device automatically).

```
#++ hostname
scutil --set ComputerName "${device_name}"
scutil --set LocalHostName "${device_name}"
hostname "${device_name}"
scutil --set HostName "${device_name}"
defaults write /Library/Preferences/SystemConfiguration/com.apple.smb.server NetBIOSName "${device_name}"
```

7. The system reboots
8. Your software management system agent takes over and configures your system based on your hostname.

```
#++ time
sudo systemsetup -settimezone "${site_timezone}"
/usr/sbin/systemsetup -setusingnetworktime on
sudo systemsetup -setnetworktimeserver "${domain_fqdn}" #++ this command will clear /etc/ntp.conf and add the
primary as the first line.
echo "server time.asia.apple.com" | sudo tee /etc/ntp.conf
sudo systemsetup -setusingnetworktime on
```