

InstaDMG

I/O Works So You Don't Have To

These guidelines were created for the PSU Mac Admins Conference, May 12th, 2011

Intro - Early Advancements in Mac Imaging

Technology that's advanced the state of Mac IT deployment over the years

- Pre-History: "The Service Drive"
- Cheating - one vendor, under unix
- Imaging Basics - A standardized base
- Distribute that base efficiently

What is InstaDMG?

Breakdown of the history and moving parts

- Sparse disk images create a ['Revolution'](#)
- Be agnostic(with your hardware), sidebar on ['Slipstreaming'](#)
- In search of Zero-Touch Deployment
- Batteries included

Bootstrapping

Zero to Sixty

- Start with just three steps
- All the ingredients in a customized blend
- A glimpse of the future - InstaUp2Date
- Blowing the doors off

Take-Home

Where do we go from here?

- Re-cap. Theory & Tools refresher
- Evaluating alternatives, common usage
- The genie is out of the bottle

Thanks! Q&A



InstaDMG

I/O Works So You Don't Have To

Devices Silent,
Pencils Down Please

Allister Banks, @sacrilicious - Point Consultants, NYC

1

We're ready to get started, can everyone hear me ok? For those who came to my LocalMCX talk, pardon me for saying the following again, my name is Allister Banks and I'm the lead engineer for Point Consultants, we're an outsourced IT consulting company in NYC. And Just to set some ground rules - the respect I want to show presenters whose sessions I'm attending means I want to be in what I call 'pencil's down' mode - which means:

Please, do not use your computers except for two things - taking notes or using the poll widget to submit your questions. On the handout that you received with the outline for this presentation you'll see two links to web forms - one is for an actual poll I hope you have already taken to let me know your familiarity with imaging, the other will be the way to give me feedback and ask questions. I do value your feedback immensely as it is critical for this to be as rewarding an experience as possible, so please 'vote early and vote often' as they say.

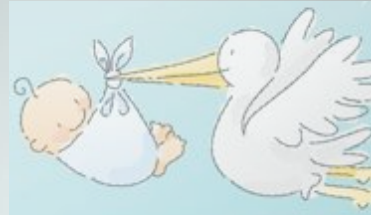
When it comes to questions I'd even go so far as to say ignore "next slide syndrome", for example when you ask a question that gets answered by the next slide. Do not hold out hope I will cover the point or topic you're concerned about - put your observation into the system and damn the torpedoes.

Now! I only have 3 hours, and even after the great job Mr. Trouton did in his Modular Deployment session, I've got a lot of material to cover to reflect the range of experience of everyone here with instaDMG. And after our break, if you'll indulge me I'll describe my personal involvement in its maintenance as well. Based on the feedback from my poll that comes in, plus q&a time at the end, hopefully we'll have enough of what you're here for to satisfy everyone. And, a disclaimer:



2

I'm about a 90% advanced instaDMG user since I have a limited scope of things I need to interact with, and have only been using it since early 2009, so from a developer or image-building master perspective I still have a long way to go. I also have very little experience with any tools that do similar things like Casper Imaging and System Image Utility. So just keep in mind while I'm making pronouncements we'll be teaching each other and proving what we know by testing that what I'm saying is current and still works - and from that proof from testing we'll know how to get the most out of our tools and time.



Where did we come from?

3

Now early in my career I had customers with a unique need; we were tasked with keeping iMacs that guests in a photo studio could use in a secure state and let the sorta tech-savvy manager perform whatever maintenance was necessary. This was probably before Deep Freeze, which they use in Apple stores for the demo models, where it reverts back to a saved state when you reboot the computer.

And back in 2004, the most basic, vital part of a break-fix toolbox was the 'service drive'. Meaning an external, bootable hard drive with a 'known-good' OS

[Write known (good) - warrantee-able state on chalk/dry-erase board]



The Service Drive

4

And back in 2004, the most basic, vital part of a break-fix toolbox was the 'service drive'. Meaning an external, bootable hard drive with a 'known-good' OS

Practice (Theory)

- Known (*Good, Warrantee-able*) State

5

Here we're starting to gather our guidelines of what we want to achieve, and I'll expand upon these as we go forward



The Service Drive

6

Apple's motherboard firmware since way back has supported this method of starting up the computer from another disk, which in essence separates out the internal software from hardware when troubleshooting behavior on a computer. This is a valuable advantage which improves the efficiency of technicians, and since 10.4's addition of partition resizing to add boot camp support, that flexibility is even greater - you could carve a slice out of the internal drive and install a fresh version of the system and you could do that with only the disc that came with the computer - there's the utilities menu that has Disk Utility in it. Just these points bring me to a primary tenet of my workstation management philosophy:



**Cheat since they
let you**

7

Apple cheats in ways that can greatly benefit you. There certainly hasn't been easy access to boot options on other vendors workstations, definitely not for consumers until very recently. So the early lessons we took from the simplicity of the service drive is that when that operating system is booted, you can almost guarantee certain things about the state of all of the software on it irregardless of what hardware its attached to. We'll go into how that is only a semi-warranty-able state of affairs from Apple, but just understand this has been maintained for years, with Apple supporting the core advantages for us. We'll touch on the engines under the covers relating to the options for how to move that operating system snapshot around, for example Disk Utility and the various free or commercial third party tools like Drive Genius and Carbon Copy Cloner, or CCC hallowed be its name.

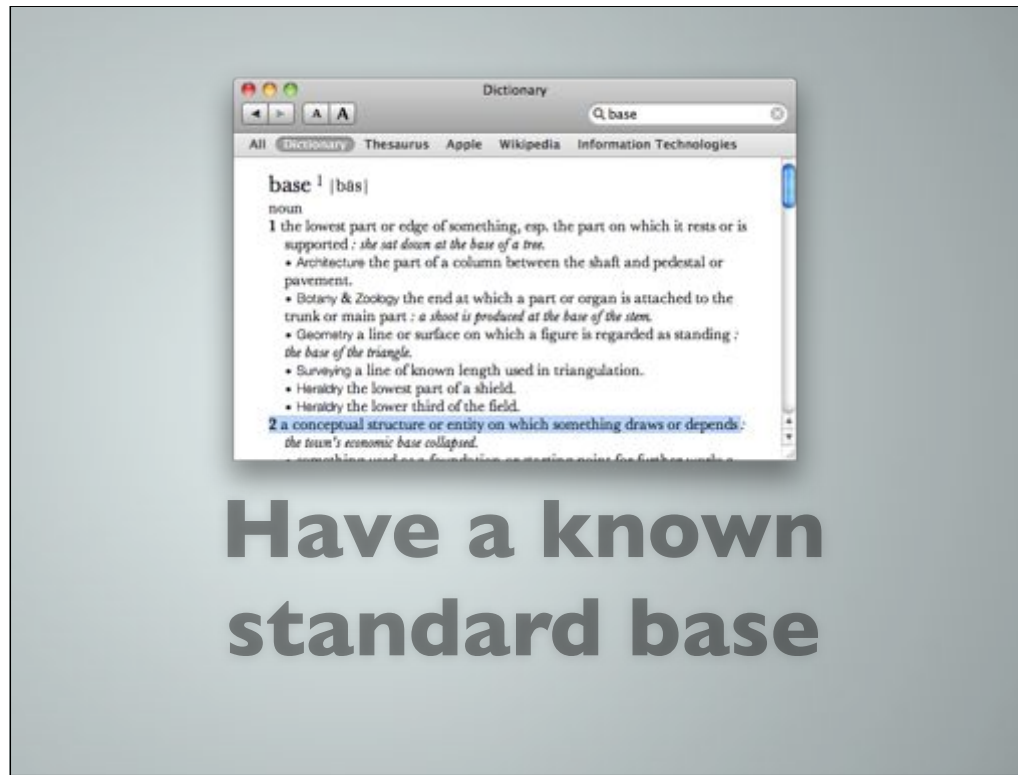
Tools



● Carbon Copy Cloner

8

Similarly to how we have practices to guide us, I'm leading a tour of the tools we interact with. Just the addition of Carbon Copy Cloner turns our humble service drive into a primitive way to deploy an image, a snapshot of the software state of a computer: when you fire it up and ask it to move all the bits on that drive to a slice on the internal hard disk, you are putting one image many places. And while there are more secure, scripted ways to go through the process of making that 'cloned' image clean and hospitable for a real deployment, we at least have a known state to start from.



Have a known standard base

9

Back to that photo studio client that needed to consistently roll out an image we'd prepare for them, this was the solution my former consultancy provided: the client bought an external firewire drive and manually went around running this half hour process per machine every Monday morning. CCC was our hero back then, but this was all the 'Golden Master' style of creating an image - you have the good with the bad, and when you're manually interacting with a live system that you'll then deploy multiple places you run into a broken-seal type situation. We'll mostly concentrate on how instaDMG solves that for us instead of going into why it's a poor compromise, but just to reinforce the idea that using a master image like this is sub-optimal, a little side-story:

When I used to weld steel and aluminum for theater sets, you'd need to wear specific safety gloves. I had a boss that wouldn't let anyone borrow his gloves, and his way of objecting was to say using another person's gloves is like using another man's dirty socks. Your take-away from that should be 'used is not the best'

Alright, diversion over. So as soon as we could we adopted another piece of software from Mike Bombich, and that's NetRestore.

Tools - ASR

(via NetRestore + NSR Helper)

A terminal window titled "Terminal - gotty - 78x23" showing the help text for the ASR (Apple Software Restore) command. The text includes the name, synopsis, and description of the tool. The synopsis lists several options: -source, -target, -server, -file, and -image. The description explains that ASR efficiently copies disk images and volumes, either directly or via a multiloast network stream, and can also accurately clone volumes without the use of an intermediate disk image. It also notes that in its first form, ASR copies source (usually a disk image, potentially on an HTTP server) to target, and source can be specified using a path in the filesystem, or an http or https URL. It can also be an asr:// URL to

```
ASR(8) BSD System Manager's Manual ASR(8)
NAME
  asr -- Apple Software Restore; copy volumes (e.g. from disk images)

SYNOPSIS
  asr [options]
  asr restore --source source --target target [options]
  asr server --source source --server server[:port] [options]
  asr restore --source source --file file [options]
  asr imagecopy --source image [options]
  asr help | version

DESCRIPTION
  asr efficiently copies disk images and volumes, either directly or via a
  multiloast network stream. asr can also accurately clone volumes without
  the use of an intermediate disk image.

  In its first form, asr copies source (usually a disk image, potentially
  on an HTTP server) to target. source can be specified using a path in
  the filesystem, or an http or https URL. It can also be an asr:// URL to
```

● Carbon Copy Cloner

● **ASR** - package and deploy block-level restorable images

10

NetRestore and its helper app converted the image you wanted to deploy to read-only, and could restore it in minutes instead of over half an hour, like CCC would normally take. But really, NetRestore was just a front-end for Apple Software Restore (or ASR for short), which Apple uses in manufacturing to load the operating system onto machines. ASR has been around since the beginning, and has the epic win or feature of being able to lay down an entire system, either over a network or locally, in minutes. Instead of doing checking and verification and other overhead when laying bits or blocks of data onto a disk, it just streams them in just about the most efficient method possible for hfs volumes. What NetRestore did was utilize ASR to both prepare a snapshot of your choosing to be restorable, and roll it out in an optimized fashion.



**get back to that
state pronto**

11

And this is really why we're here, we want to choose what's on the systems in detail, and get any machine fresh from the box into a customer's hands with a known-state in the most expedited, simple manner possible. And that's where instaDMG comes in

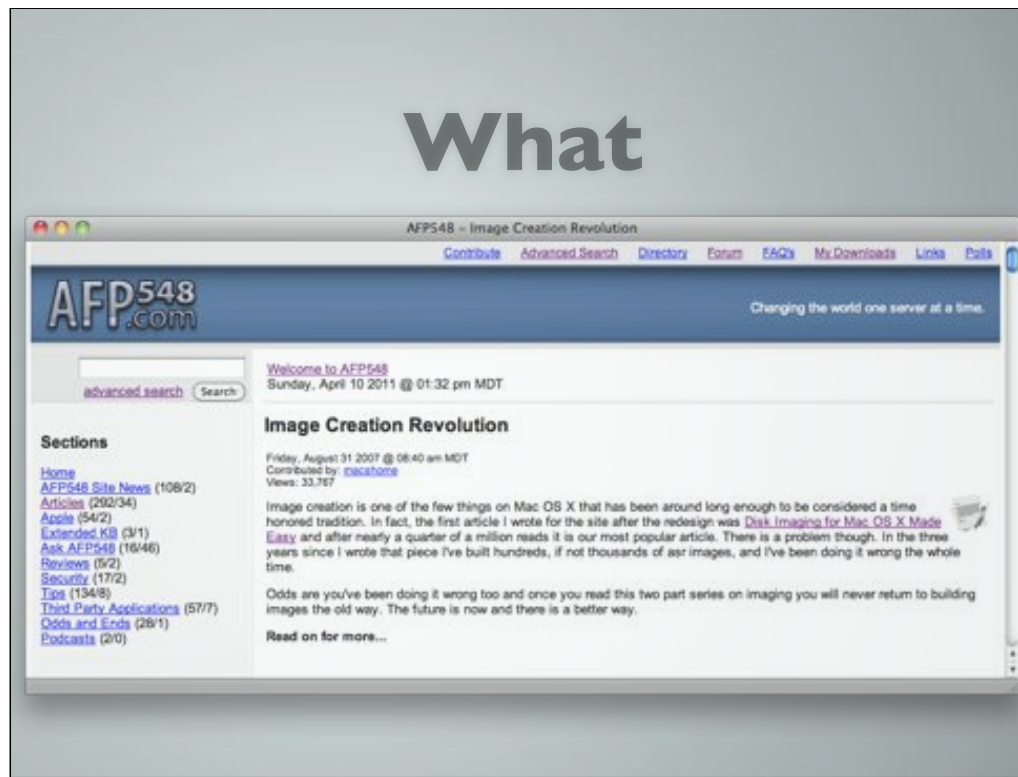
Practice (Theory)

- Known (*Good, Warrantee-able*) State
- Repeatable
- Automated
- Optimized

12

And when we're talking about building an image, the standard that the instaDMG project reaches for is: getting the operating system and many other software packages installed in a known state, and to do that in a repeatable, automated, and optimized way. And we'll come back to this to expand on these points.

What



13

Without continuing with too much of a lecture or history book tone, here is the briefest timeline of InstaDMG: Announced with his 2007 article “image creation revolution” on AFP548.com, Josh Wiesenbaker started collaborating with some of the brightest minds in the Mac IT space to create software that would automate the building of an image. Karl Kuehn took over the project in 2008, and in addition to other optimizations, he introduced a way of sharing common lists of software to install with an add-on tool called InstaUp2Date.

Underlying Tools



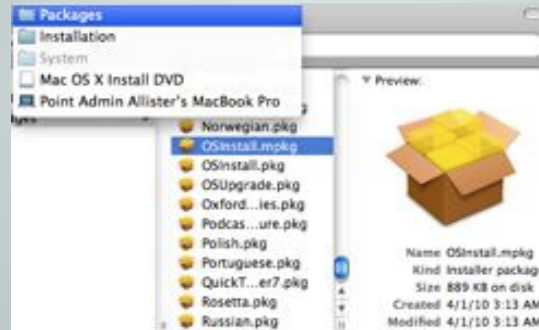
```
Terminal — grotty — 77x25
HDIUTIL(1)      BSD General Commands Manual      HDIUTIL(1)
}
NAME
  hdiutil -- manipulate disk images (attach, verify, burn, etc)
SYNOPSIS
  hdiutil verb [options]
DESCRIPTION
  hdiutil uses the DiskImages framework to manipulate disk images. Common
  verbs include attach, detach, verify, create, convert, compact, and burn.
  .
  The rest of the verbs are currently: help, info, checksum, chpass,
  unflatten, flatten, imageinfo, isencrypted, mountvol, unmount, plugins,
  ediffrez, edifferez, internet-enable, resize, segment, makehybrid, and
  gmap.
BACKGROUND
  Disk images are containers that emulate disks. Like disks, they can be
  partitioned and formatted. Many uses of disk images blur the distinction
  between the disk image container and its content, but this distinction i
  _
```

- ASR
- **hdiutil** - an invisible image to install into

14

And just to get right into the concepts behind it, it creates a disk image and installs the software you choose into it, just as if you were manually doing it booted from a retail install disc. That's hdiutil at the command line that is creating that image for us. This stable way of creating expandable disk images became possible back in 10.4 and Tiger is still supported, although it does require you to make that build on a 10.5 system. 10.5 can only be built on 10.5, 10.6 on 10.6 only.

Underlying Tools



- ASR
- hdiutil
- jail_installd/
chroot -
ADVANCED**

15

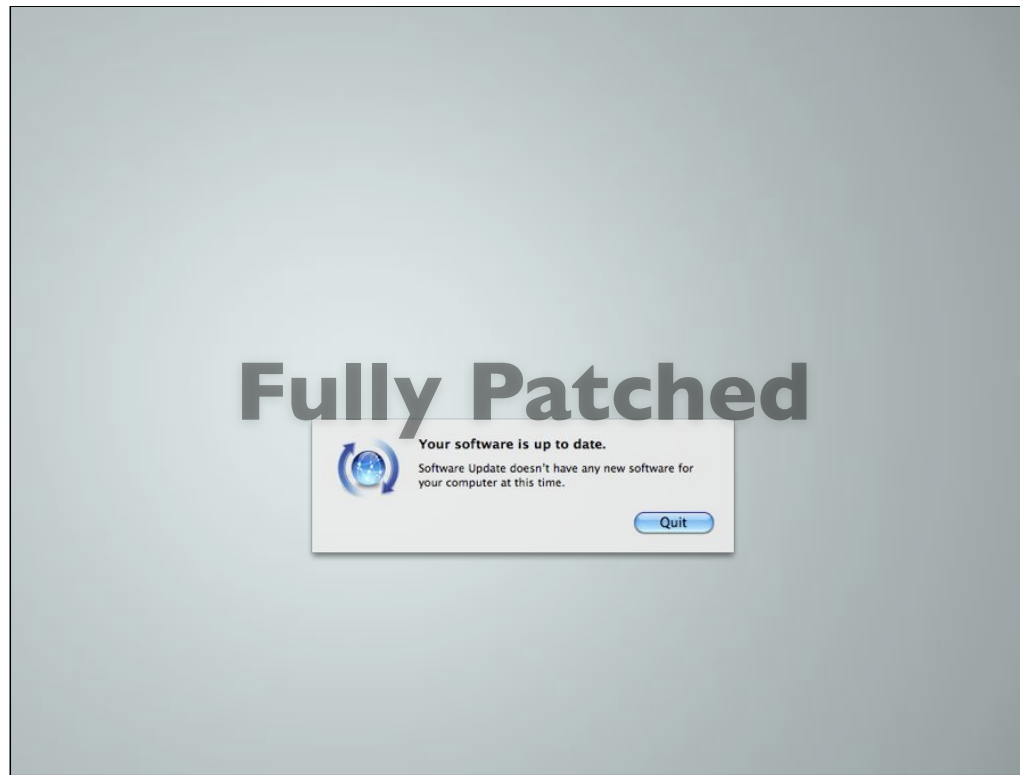
While we're adding to this list of the underlying tools, chroot jailing is a little advanced to mention at this point but I just wanted to add it to the list, because its the technique being used in instadmng to 'anonymize' or hide what the actual booted system is that instadmng is running on, since the installer doesn't actually need it. And to reiterate my point of Apple letting you cheat with optimized system maintenance abilities, howsabout the trick of when you have an installed disk mounted, using Go to Folder in the finder to launch OSInstall.mpkg and start an OS install without rebooting. It will allow us to point that install at just about any properly-formatted volume.



Hardware- Agnostic (Sparse) Images

16

Now this anonymized process being similar to an OS install from a retail install disc is worth mentioning because of the fact that a DVD is read-only: it can't modify or tailor much of what it installs to the hardware its installing to. And in that same way, when instadmng installs the operating system from the command line into a disk image, it kindof relies on the fact that it can't make any assumption about what system it's booted from while it runs.



17

And in just the same way with the installer application you can point the OS install elsewhere, all of the updates from apple are essentially capable of the same thing, they're in that same package format. Now, often you may see the the package insist "This must be installed on the startup volume"

So InstaDMG steps in and utilizes the command line version of the installer, and therefore throws caution to the wind and laughs at GUI restrictions. In earlier versions of OS X software update allowed you to get all your packages gathered via the "download only" menu option, which was one-stop shopping for all these patches, which you could then conceivably take and throw at the volume you just did the OS install on, and voila! Theoretically a fully patched image.

This Ain't Slipstreaming



- Windows-Specific
- Adds Drivers or Service Packs
- Still need to go through an install process

18

Now I want to take another sidebar to discuss slip-streaming, which is a windows-based concept of rolling service packs or drivers into an OS install. This is not the same as what we're doing, that process is manipulating the OS installer so at installer runtime it incorporates more of what we want. The resultant slipstreamed installer disc has a higher chance of working the way we want it to when we go through the manual install process with it, but otherwise we aren't gaining too much. Pardon me if it's obvious that running an installer is slower than restoring an image via ASR, but just to be clear, the InstaDMG robot runs the installer process to make the final result a patched OS that you can then roll out just about anywhere to the hardware of your choosing.

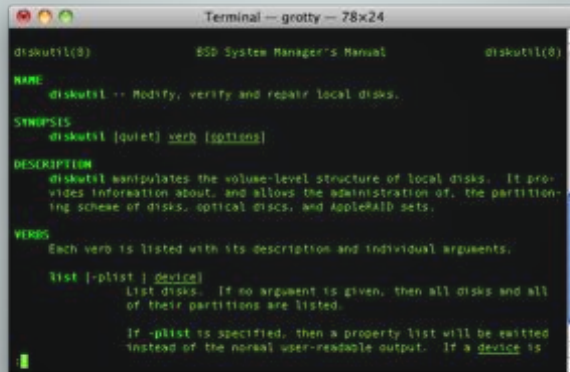
With Various Customizations



19

And moving on, that patched image is nice if all you needed was the built in software, and we saw how long that lasted for the iPhone... So the earliest addition was a way to create your default local admin user, and presto whamo, upon starting up your fresh image you get a login window with your user, and with small tweaks there's no waiting through the welcome to leopard intro movie or getting bothered for registration.

Underlying Tools



```
Terminal -- grotty -- 78x24
diskutil(8)      BSD System Manager's Manual      diskutil(8)

NAME
  diskutil -- Modify, verify and repair local disks.

SYNOPSIS
  diskutil [quiet] verb [options]

DESCRIPTION
  diskutil manipulates the volume-level structure of local disks. It provides information about, and allows the administration of, the partitioning scheme of disks, optical discs, and AppleRAID sets.

VERBS
  Each verb is listed with its description and individual arguments.

  list [-plist] [device]
    List disks. If no argument is given, then all disks and all of their partitions are listed.

    If -plist is specified, then a property list will be emitted instead of the normal user-readable output. If a device is
```

- ASR
- hdiutil
- jail_install
- diskutil

And InstaDMG wipes its feet afterwards, making that mix of customized goodness play well together and not compromise the stability of the system. Conceivably you could take it out of the box, netboot it, and automatically have it morph into a fully configured machine with no further effort required.

That Deploy Fast

21

When the Bombich-branded NetRestore was retired, he named DeployStudio, not SystemImageUtility, as its rightful successor. I'll touch upon some reasons why in the second half. Your InstaDMG-baked images now have a much more flexible GUI to utilize when restoring them to computers over the network, and just to brag, the average time a minimal image I've created can be thrown at a disk in literally less than 3 minutes.

The idea is you test and bang on your image ten ways to sunday, and then it's QualityAssurance-approved to blast out over a fleet the size of Google's, tens of thousands of macs worldwide. No waiting for a verification step, just simple blocks queued up and laid on disks as fast as it can write.

How - Bootstrapping

22

So up until now we've been high-level about the details of how I'm recommending we practice this craft of image-building, and when it comes down to brass tacks, we've got quite a list of tools that are carrying out the work underneath InstaDMG.

CLI-Curious?

Go CommandLine

credit: @thespider

- /usr/bin/installer - 'binary'
- /Applications/Utilities/Installer - GUI app
- command - what you type after the '\$' sign in Terminal

23

And while significant work has gone into hiding how the sausage is made, an early requirement is - the CLI or command line interface, which we access via the terminal application. No way around that, but you'll thank yourself later by harnessing its power if you aren't already. When I'm describing actions you'll perform in the CLI, I'll use the term 'binary' to differentiate from a GUI install and the command line counterpart "installer" (kudos to apple on the inventive name). Text that you enter in a terminal window is referred to as a command - everybody comfortable with that? All you folks cursing that you didn't really think this would be for beginners, I'd love to not leave anyone behind, so thanks for your patience. I've been using the term 'restore' as a verb which means essentially the act of getting a hard disk image copied to another volume... everybody understands partitions versus volumes versus partition map, right? If someone could explain partition maps to me that would be awesome.....anyway, moving on

Please pardon me because I'll be taking you straight into a quickstart that uses commands that do not contain the word 'instadmg' anywhere - just stick with me, all will be revealed. Please, submit questions early, and often.



Workflow demo - the quickstart

24

One word of caution - installing other software, mounting drives and putting the machine to sleep in the middle of the process is not recommended, luckily a reboot is usually all that's necessary to clear weird errors and start again.



Open Terminal, and paste:

`svn checkout http://instadmng.googlecode.com/svn/trunk instadmng`

A screenshot of a macOS Terminal window. The title bar reads "Terminal -- bash -- 105x53". The command prompt shows the user 'aru' at 'Point-Admin-Allisters-MacBook-Pro' executing the 'svn checkout' command. The output lists several files being checked out, each preceded by an 'A' character.

```
Point-Admin-Allisters-MacBook-Pro:~ aru$ svn checkout http://instadmng.googlecode.com/svn/trunk instadmng
A   instadmng/InstallerFiles
A   instadmng/InstallerFiles/BaseUpdates
A   instadmng/InstallerFiles/InstUpDatePackages
A   instadmng/InstallerFiles/InstallerDiags
A   instadmng/InstallerFiles/BaseOS
A   instadmng/InstallerFiles/CustomFWG
A   instadmng/instadmng.bash
A   instadmng/OutputFiles
```

25

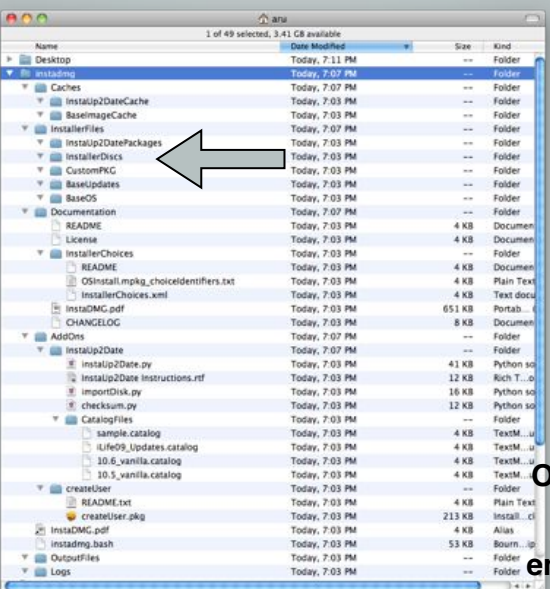
First, where's the software, right? The easiest way for me to remember the address for its location on google's code hosting service is the name of the project `instadmng.googlecode(one word).com`. There you can download a prettier version of the guide I wrote separately from the software, and a smaller file-size version is bundled with the svn checkout. Although not a lot has changed with the code recently, please do not download the zip you see at the top in this window. There are very few hurdles I'm presenting with these three steps, and I really want everyone to take advantage of the most up-to-date helper files that are bundled with the project.

Luckily, this is an easy hurdle to overcome, even Mr. President can do it - thank you google image search. Don't even worry about what I mean by svn checkout - we're three copy-and-paste steps away from getting the latest updates to the software and creating our first finished product.

I can't put it much better I do in that PDF guide, so here there are the three 'quickstart' steps, from memory:

Open terminal, in Applications/Utilities, and type or paste: `svn checkout http://instadmng.googlecode.com/svn/trunk instadmng`

Hitting return on that will show you it pulling down the latest version in less than 30 seconds, and it will tell you which revision it was when it's done. Next step:



**Insert a *retail-box* (important)
OS 10.6 (or 10.6.3) installer disk**

**Paste the following,
entering your (admin) password
when prompted:**

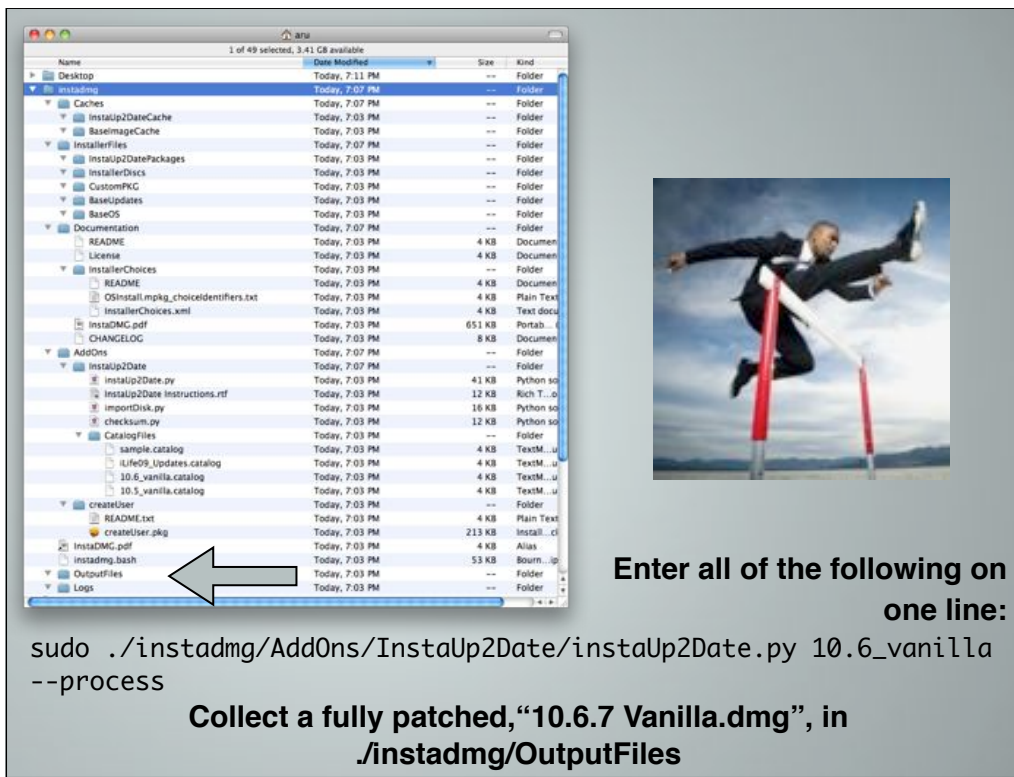
```
sudo ./instadm/AddOns/InstaUp2Date/importDisk.py --automatic
```

Let it process for about 45 minutes

26

We're doing an OS install, so we need an installer DVD - important to note here we're talking about a RETAIL BOXED COPY, just like you buy from the store, not a grey one bundled with a computer - insert it into the drive, and type or paste `sudo ./instadm/AddOns/InstaUp2Date/importdisk.py --automatic`

And this will take on average 45 minutes to convert that disc to an image in the location InstaDMG will depend upon it when trying to access it from that point forward. Just for comparison, for the new thunderbolt iMac restore disc, it took 33 minutes and two seconds from an external optical drive to my SSD. Grab a beverage, and when Terminal returns to an empty command prompt, we start the show for reals:



Enter all of the following on one line:

```
sudo ./instadmg/AddOns/InstaUp2Date/instaUp2Date.py 10.6_vanilla --process
```

Collect a fully patched, “10.6.7 Vanilla.dmg”, in `./instadmg/OutputFiles`

27

```
sudo ./instadmg/AddOns/InstaUp2Date/instaup2date.py 10.6_vanilla --process
```

We sit back and enjoy another beverage, read hacker news, absorb vitamin d by going outside, all the while our robot goes through all these steps: first, it downloads all the “approved for mass consumption” updates to get those in place, it sets up a disk image as if you’re doing a regular install to another hard drive inside your computer, it installs the OS and stows away an inbetween version of what that out-of-date install would look like before moving on

[Write repeatable automated optimized on chalk/dry-erase board

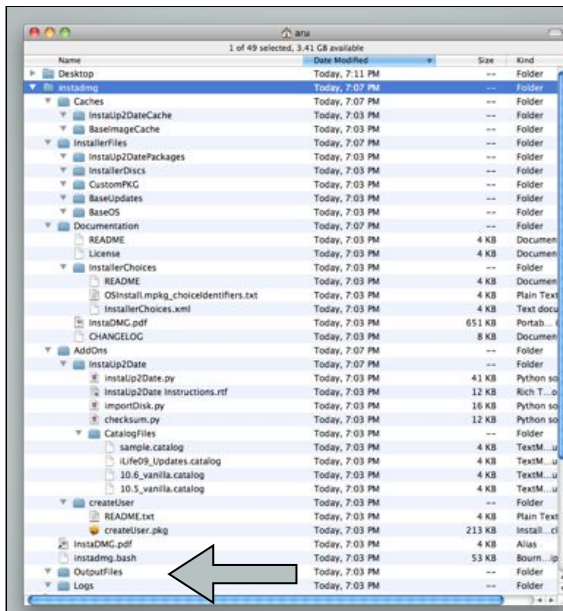
]We’re still going, we’re halfway through all the tasks its performing, but I’ll come back to reinforce these points. So okay, we’ve got this intermediate image stowed away, now it takes those updates it downloaded early on and throws them on in a specific order, does a little housekeeping to make sure I cleans up any toys it took off the shelf, so to speak, and then runs repair permissions since this is the only applicable use of that normally-vooodoo tool, makes the disk image its created of the final result read-only and finally -

See why there’s an automated tool now?

And finally it performs a process that prepares the disk image for ASR restoration. Phew! Okay! At this point its good to take a moment and acknowledge that there is the perspective that this is a non-trivial amount of time to spend but remember - this is computer time. A robot did this work, not a human with a wife to pay attention to and watch tv with. And why this method over the Golden Master? Just like that story about the welding gloves I said earlier, this image we’ve created is as good as straight from the factory, and is sealed in a read-only container - make the effort to see the robot build an optimal image for you.

Intermission





Enter all of the following on one line:

```
sudo ./instadm/AddOns/InstaUp2Date/instaUp2Date.py 10.6_vanilla --process
```

**Collect a fully patched, “10.6.7 Vanilla.dmg”, in
./instadm/OutputFiles**

29

Okay, back from the break, just to remind you where we were.

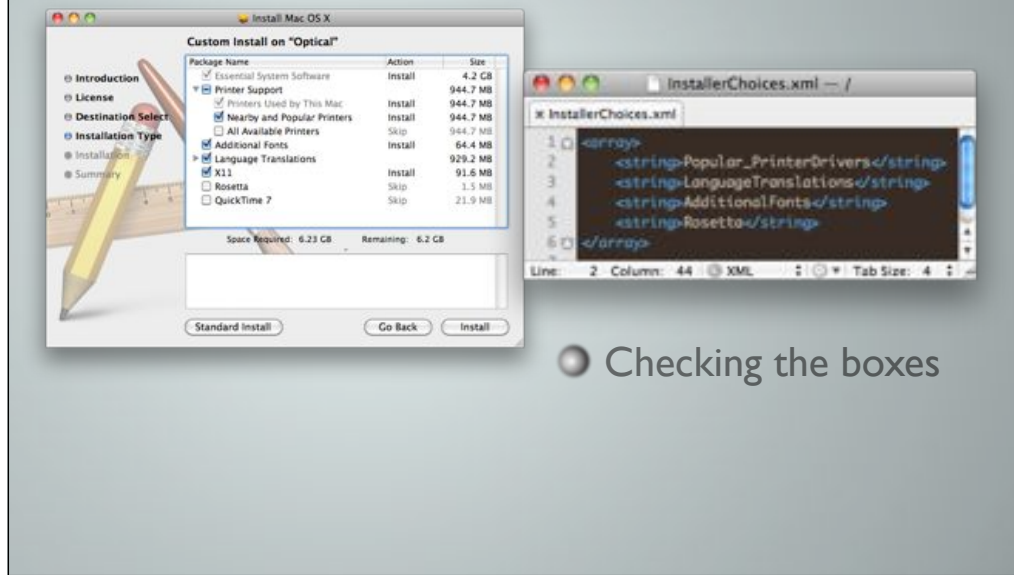
More perspective on how savvy this instaUp2Date robot has become is to look at how you used to have to interact with instadm.



30

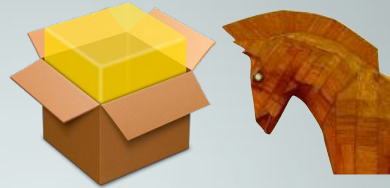
If you're watching lines of text fly by while that last command winds its way through all its steps, you may have seen instadmng get triggered by the command we ran, and maybe more text about it running an installer from a numbered folder. And that's exactly how it used to be done - it was up to you to make a disk image, download the updates, make numbered folders for them so there was an order implied, and enter a command to run instadmng, which is a bash script and therefore has the .sh extension. You'd put the most recent combo update in the first folder, and all the patches you wanted after that in any order that worked when running the installer from the GUI. Better believe instaUp2Date has more Quality Assurance than your gut instincts on what order to put stuff in.

● Before, the default



● Checking the boxes

Now speaking of the GUI, one common need which was a little more important before Snow Leopard had more sensible defaults about bundled printer drivers, is the way you can deselect or select language translations and other optional installs which are available on the OS install disk. In the terminal you can read the man page or manual on how the installer binary can utilize what's called a ChoiceChanges file, and there is some guidance on how to create and interact with one. InstaDMG considers this, and the creation of users, to be important enough to create a sample for you in the Documentation folder that's bundled with the download. If you'd like to go through the process of making further customizations from the ones in the sample, you can do so by following the readme which is also included. If anyone is similar with a kickstart file for certain linux distributions, a ChoiceChanges file is limited in scope but accomplishes a similar need.



CreateUser, a Payload-Free Pkg

32

Switching gears to another customization wish, createUser can essentially perform all the actions one would do in the system preferences accounts pane, and can let you store that password in a hashed or obscured format so that its that much safer while it sits in the package, waiting to be (air quotes) “installed”. Now I don’t want to go too much further down that road because I have my own version of this package with enhancements that I spoke about during my LocalMCX talk. And I did merge clearer instructions into the version of createUser bundled with instadmng, but it really is somewhat straightforward - you customize a USERDATA file in the createUser package, and use a bundled helper binary tool to get the password hashed.



Advanced - InstaUp2Date

33

Before I go too in-depth I'll make it clear - you can run `instaDMG` without `instaUp2Date`, but you shouldn't. The `instadmng.bash` script can be thought of as the engine that `InstaUp2Date` runs when helping you through the processing steps, and if you don't want or need that help, or if you're having a problem where the python error message doesn't make sense, you can manually create and populate folders in order to let `instaDMG` run.

`InstaUp2Date` shines when you have multiple images or want a more dynamic way to alter the order things are installed. It obviates the need to manage folders of stuff to install, and community support means you can 'crib notes' so to speak from the catalog files bundled in an svn checkout - By that I mean if you quote 'download' the Vanilla catalogs which are included in the svn repo, you'll get a tested list of all the updates you'd need to create a fully patched image, including `iLife` and `iWork` updates. The vanilla designation for a catalog file means that list which you'd need for just about every time you're making an image.

Now I'll touch on another bundled tool, `checksum.py` - and this is part of the paranoia you should have when scaling up your deployment, you need to know that the package included in the build was delivered 'as advertised' so to speak - and fingerprinting each package by running a tool to check its contents is how we accomplish that. The catalog file wants to show you a friendly name, find the package its looking for at either a local file path or an internet address, and then verify that what it pulls down has not gotten mangled in any way and is indeed what we were looking for. The moral of the `checksum.py` tools story is that it will help you generate that fingerprint so the catalog file has what it needs to both functionally do its job but also protect your build from something being corrupted or otherwise not what we expected.

And not to go too far down the rabbit hole, `InstaUp2Date` catalog files can reference each other in order to add all the packages in that catalog. So for example if I want to run `10.6_vanilla`, like in our quickstart example but also include `iLife 09`, I'd just need to take three steps -

1. get the `iLife09` dmg in a folder on the same volume as `instadmng` and have `checksum.py` run to generate a catalog file line for it,
2. put that line at the beginning of our `iLife` updates catalog file, and
3. Add the line "include-file colon tab and the full path the `ilife` updates catalog file with its name

So that looks like this. And now I just use the same exact command as the quickstart and `instaUp2Date` will follow the path to the included catalog and at the end a fully patched image with `iLife09` will pop out.

We could go into all the 3rd party software and other settings you could then incorporate, but we'll leave it at this for now.



Rev The Engine

34

So for this entire process we've talked about a quickstart way of using the tool in a basic fashion, some of the ingredients of common builds and the old manual way of what instadmng's moving parts are, and then a glimpse of the future, for how instaUp2Date makes the process easier to document and repeat.

To really get the most benefit in the shortest amount of time, I'm going to try to reinforce basic best practices, but then reward your patience optimizations that... well, may seem basic if you are already aware of them, but if you were curious I've done a good amount of experimentation to verify that they do deliver the goods.



Rev The Engine

35

Now, revisiting the last of the quickstart steps, it contains optimization #1, which is to trust the community - that 10.6 vanilla catalog we got from the svn repo is approved for mass consumption, and has been tested by at least one person before being posted. Ideally, you don't need to think about what updates are needed the next time you go to make a build, just check out a new svn, and it's there for you to modify and stack the software you need on top of.



Rev The Engine Deploy

36

#2 is to ride the SSD wave - this entire process is not memory or cpu bound, it's all about the I/O. So if you put your packages someplace that's great at reading like an SSD and maybe output to a striped RAID, your time to completion will be easily a third of what you'd get with laptop HDs that have little cache and modest speed.

Now for deployment, it's good to point out that InstaDMG is open source, (Apache licensed, patches accepted) and a gentleman from Canada was able to take a look at the code, and wrote a script to load the resultant image directly into DeployStudio. If we haven't talked about DeployStudio enough I will touch in it briefly later.

But taking it back a step with a hopefully simpler example, just the same you can use System Image Utility to prepare an instaDMG cooked image for rolling out via NetRestore. Now that NetRestore is not to be confused with the former Mike Bombich software, (again Apple with the innovative name). What that new-in-10.6 NetRestore function in SystemImageUtility essentially does is make a netinstall set that writes the image we've prepared to the booted system via ASR. Which gives you another network-based deployment method for testing, as fast and often more convenient than using the asr command line while locally attached or connected by firewire cable over target disk mode. So that's just a quick sidebar on deployment.

**Where do we go
from here?**

37

I think its obvious I'd like everyone to give instaDMG a try, and to clearly chart the path I'd like everyone to take,

Where do we go from here?

- Follow the quickstart
 - optional: make a installerChoices.xml
- Add createUser.pkg
- Sky's the limit! (Office, iLife, LocalMCX, etc.)

38

first I'd love it if you'd follow the quickstart to make yourself a vanilla image just to get the hang of it. Then, come up with the file I mentioned, installerChoices.xml, so the base install is customized and therefore the intermediate cache is created to our liking as soon as possible. Then getting your admin user on there. clear the registration prompts and skip the 'welcome to OS X movie', which I do all in one package. And from that point you have a launching pad to whatever you want to put in there and whichever other management tools you'd like to 'seed' to your organization.

Recap: Why - What - How

39

Why we want to use it, what instaDMG is and how we'd use it, in broad strokes.

I would hope you'd agree that imaging is a powerful tool to enhance your system administration.

The Practice of System and Network Administration says quite clearly, start all your hosts out in a known state, and have a process of getting them back to that state pronto. Having your organizations approved image decided upon and maintained efficiently is how we do that.

Theory + Tools

- Known (*Good, Warrantee-able*) State
- Repeatabe
- Automated
- Optimized
- *checksum.py
- installerChoices.xml
- hdiutil
- jail_installd
- diskutil
- ASR

40

What theories InstaDMG follows and how it performs it brings us back to this collection. And when we're talking about building an image, the standard that the instaDMG project reaches for is: getting the operating system and many other software packages installed in an agreed-upon way, and to do that in a repeatable, automated, and optimized way. And we'll come back to this to expand on these points.

Alternatives, When to use it



Apple 'S.I.U.'

● System Image Utility

● Casper Imaging Suite



imaging
SUITE

41

How it does the job versus similar tools is what I'll go over briefly here:

SIU's NetRestore functionality has plusses and obvious deficiencies, one of which is it has no impetus to make it easy to include any third party software or settings. But it is Apple supported, and we have every reason to believe it might improve at handling poorly written software. Casper's imaging suite does things the JAMF way, which I can only imagine is nice work if you can get it, but can't speak to plusses/minuses. However, I just wanted to call out that it seems to be the only commercial option for building images in the InstaDMG style, with an eye on incorporating 3rd party software.

I update the vanilla catalogs two to three times a month as Apple rolls out patches, and to speak to when you should refresh your builds, it's somewhat obvious that you'll want to do it before you need to image a machine. That would commonly break down into one of two events: when new hardware enters your environment, or when a refresh or re-provisioning for a new employee/customer/guest is deemed necessary. That first one I'll talk about later on, but the second one could theoretically be at any time, and is at your discretion. If you're using configuration or patch management tools, you will of course need much less interaction with InstaDMG, but the read-only, QA-friendly nature of an instadmg-prepared image has quantifiable value to me, so I cook new ones often.

Take-home

42

And as I've found through working to get the most out of this tool, getting good at this modular deployment philosophy will make you better at your job. Learning how to make a first-boot script work was the first time I really got into scripting, rolling localMCX settings into an image was the first time I really revved the engine on managed preferences, and maintaining three images for my company is now as easy as one command - I tell it the three catalog files of the images I want, and less than two hours later all three are in my DeployStudio repo - while I'm doing business email and not reading Twitter.

We should come to enjoy this process of iterative improvements to what we can include in the build and therefore guarantee about the state of our machines in the field, and I feel there's no turning away or compromising on the process once we've realize our goal.

Thanks!

Allister Banks, @sacrilicious - Point Consultants, NYC

43

Before we get into Q&A I wanted to thank you all for your patience and attention, this has been a lot of fun for me. More importantly your feedback will help me since I can't be the judge of my communication skills, so please let me know where I haven't been as clear as you'd hope. As Chris Tucker's character said in Rush Hour, "Do you understand the words that are coming out of my mouth?" Please, give me feedback so my communication can improve. And pardon me if I go so far as to say a great way of giving back to the community that fostered this tool and caused it to thrive is to help me train new people on using it. Thanks very much!