1

- Good Morning! Let's get started!
- **
- Welcome to Mac Management: State of the Union and Where are we Going

---

2

- First a couple introductions
- Macintosh Systems Administrator at Meredith Corporation, home of Better Homes and Gardens
- Might know my work on Adobe Creative Cloud software
- Work in a team of four technicians and I've been there for over 13 years

---

3

- My co-presenter is Robert Hammen
- He's been managing Macs since before the days of Mac OS X
- As a consultant supported a diverse group of clients from Fortune 500 companies to education and businesses of various sizes
- Mac Team Lead at National Center for Biotechnology Information, NIH

4

- What are we going to be talking about today?
  - Well, a lot of things, but don't feel you have to copy everything down
  - The slides *and the presenter notes* will be available from the Penn State MacAdmins website
  - This presentation will be foundational in nature, giving an overview of popular Mac administration tools and the future

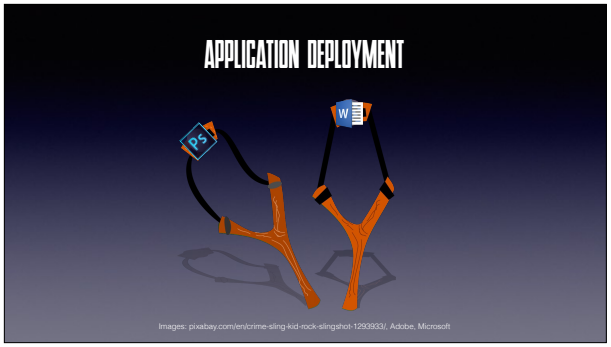
---

5

- Start with a history, how did we get here?
  - You might see parts of what you're doing today
  - At the very least you'll become familiar with the terms

6

- Imaging/Provisioning
  - What steps are necessary to welcome a Mac into your fold
  - Onboarding computers your organization is responsible for



7

- Application Deployment
  - Once the computer is part of your management system, how do you install software?
  - What is the user experience when the software is installed?

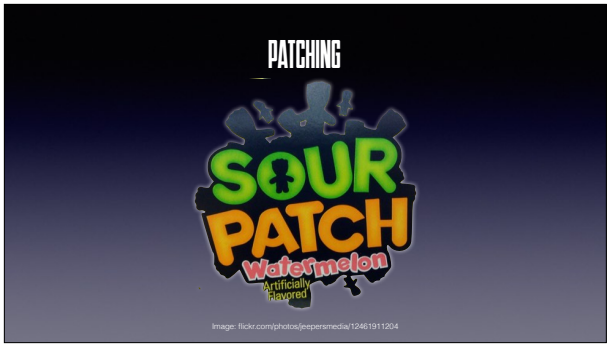
8

- Management, specifically Configuration Management
  - Once the software is installed, how do you manage settings?
  - Enforcing settings and providing a starting point for settings

---

9

- Packaging
  - How to distribute custom software and files
  - What to do, what to avoid



10

- Patching
  - Once software is installed, how to keep it up to date?
  - Discovering and distributing updates while keeping sane

11

- Where Are We Going?
  - How far can the various techniques go/last?
  - What steps can be taken now to minimize rework

12

- Time for a history lesson
  - Try to determine where you are on this scale
  - The sooner you recognize yourself in this history, the greater the amount of work ahead

Image: Twentieth Century Fox

13

- Oldest method is by hand
- Basically prehistory at this point


Image: pixabay.com/p-158130/?no_redirect, clipart.coolclips.com/480/vectors/tt05033/CoolClips_cart0474.png

14

- Slow, tedious, error-prone
- Able to adapt fairly quickly
    - Once an error is found…
    - Multiple possible errors by computers deployed
- Scales incredibly poorly
- Given the amount of work…

Image: Paramount Pictures

15

- Yeah, exactly

Image: Twentieth Century Fox

16

- Next is the Imaging Age
    - Been going on for a while
    - Breaks into three major categories


Image: family-guy-the-quest-for-stuff.wikia.com/wiki/Golden_Peter_Statue

17

- The first category is the Golden Master Image, the one true Mac
    - Including all your software, all your settings, all your errors, and when you're done you make an image
    - Perfectly, every time, for each new forked build
- Clean up every settings file
    - Network ports, keychains, users, default printers, recent servers, browser history
- Discover where each new setting is stored, figure out how (or if) you can remove it
    - Likely get to do this each time a new Mac model is released, due to forked builds
- Or wait for unified builds, which could be months
- Our original Golden Master Build documentation was 66 printed pages comprising 132 outline topics
- In fact, in today's world if you told someone you use Golden Master Imaging

---

Image: Twentieth Century Fox

18

- Hand setup and Golden Master both assume a fully booted Mac
    - Which doesn't lend itself to reproducibility
    - Doesn't lend itself to automation
    - And does lend itself to errors
- Since Apple never shipped a tool like Sysprep for Windows, Golden Master imaging is out
- What if we never booted macOS

19

- Modular
  - Introduced in Mac OS X 10.5 by InstaDMG
  - Torch taken up by AutoDMG around OS X 10.9
  - Install macOS onto a disk image, then install packages
  - Packages can be whatever is desired
- The packages need to install properly on a non-boot disk
  - And best if they only do one thing or a handful of tightly-related things
  - Suddenly no random issues due to random errors, the image is reproducible
  - Still need to rebuild with each new piece of hardware and each new forked macOS
  - And have "builds" for each group/department/software combination
  - But...what if we didn't bake *anything* in?

20

- You'd get to the last stage of imaging, Thin Imaging
  - Needs a software management system
  - Only include enough software on the "image" to connect it to your management system
  - Suddenly the "image" isn't much of anything
  - Still needs to be rebuilt with new hardware, but it's not much to rebuild

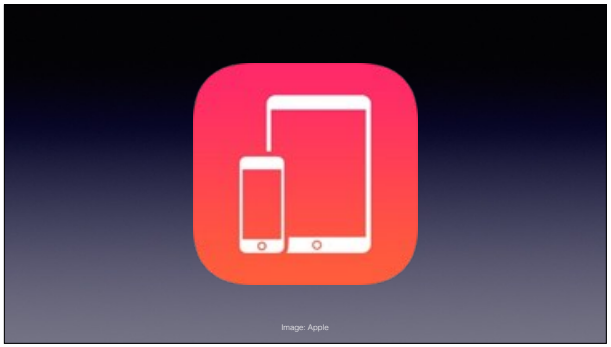
---

21

- But of course, every Mac comes with a perfectly good copy of macOS already installed, why not use it?
- We then move to the world of Beyond Imaging

22

- No-Imaging
- Need to boot the computer from something else, perhaps NetInstall or an external disk, and install the packages to attach the computer to your management system
- What if you could (remotely) tell the Mac to install those packages automatically?



23

- Device Enrollment Program (D-E-P) and Mobile Device Management (M-D-M)
    - If the Macs you purchase are Device Enrollment Program-eligible, they can automatically enroll in a Mobile Device Management system upon startup
    - Then preferred packages would install, and hook into management system

---

24

- So where are we?

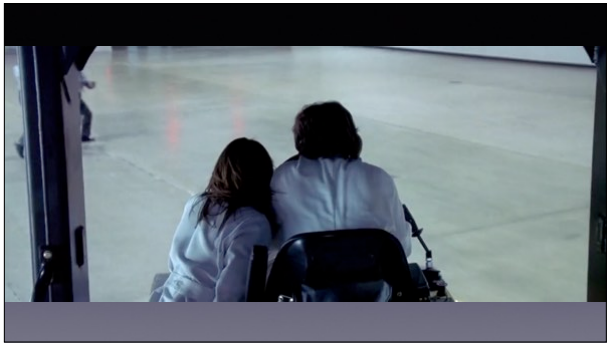
25

- We're in a mature imaging/beyond imaging and into the age of Device Enrollment Program and Mobile Device Management
- I'm going to discuss some imaging tools here, since I recognize not everybody is starting with the latest and greatest
    - If you're getting started, don't immediately plan on making images
- Keep an eye toward using packages and Profiles, both things we'll talk about later
- If you can't do what you want to do with Profiles, file an enhancement request with Apple
    - In the meantime use other tools mentioned later in this workshop
- Focus less on workflows that require an imaged OS

26

- In order to take advantage of Apple's Device Enrollment Program, your organization should be purchasing computers either from Apple or from a reseller who participates in Apple's Device Enrollment Program
- You'll also want to be evaluating Mobile Device Management services to find one that's the right fit for your organization
- You'll want to do both of these, regardless of where you are in the Mac management spectrum

---



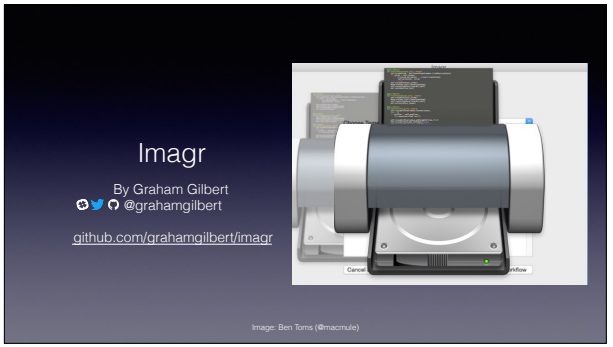
27

- In any event, this may seem like a slow burn
- And it has been
    - Mobile Device Management and Profiles since OS X Lion, July 2011
    - Device Enrollment Program since February 2014
- But these are changes to get ahead of
- Start testing now

28

- How do technicians bootstrap computers into your management system
- You may have heard of a product named DeployStudio
  - DS is closed source, slow to update, hard to diagnose issues
  - Requires a Mac in the server room
- Imagr doesn't replicate all the features of DeployStudio, just a core set

---



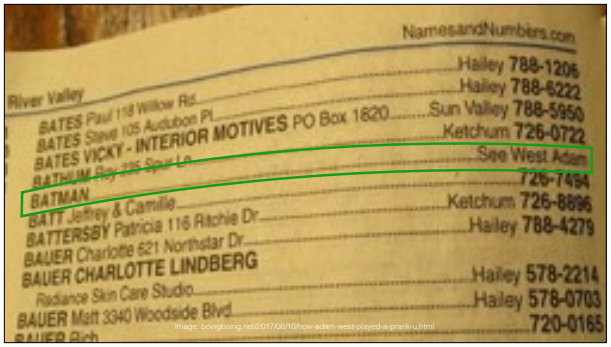
29

- Imagr is free, open-source software written by Graham Gilbert
- Designed to be run from a NetInstall image to perform actions on a disk
  - **Pleasant GUI that's approachable to non-administrators
  - Pick a target disk, a workflow, and run that workflow
  - Unlike DeployStudio, does not need a macOS server
    - Also unlike DeployStudio, it's open source
  - Settings file (as well as packages and scripts) hosted on a webserver
  - Only information the client needs is a URL to a settings file
- Create workflows of components
  - Each workflow being a significantly different end result
  - Sierra for employee computers, El Capitan for labs, Yosemite for that plugin you can't give up

---

30

- What components are available
  - Erase a disk
  - Partition a disk
  - Image a drive (Note I'm not advocating starting out imaging)
  - Install a package
    - Packages can be installed during Imagr runtime or during first boot of a target computer
  - Run script
    - Also can run the script at Imagr runtime or during first boot of a target computer
  - Set computer name



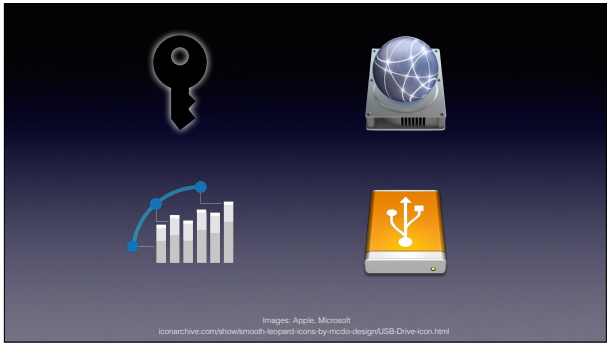
31

- Also the option of an "Included" workflow to refer to other workflows
  - Allows embedding workflows inside one other
  - And of course if you have included “utility” workflows that may not be standalone, you need…

32

- Hidden workflows
  - Workflows you don’t want to see
  - Useful for included workflows that you don't want run on their own

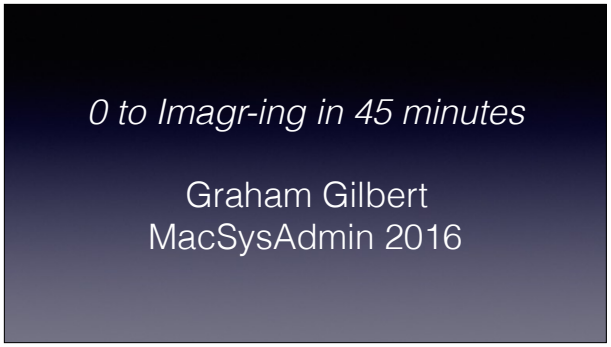
---



33

- Password support
  - Prevent folks from accidentally blowing up their computer, not to hide configurations or keep anything secret
- Includes CLI tools to create NetInstall set
  - Alternately, Ben Toms’ (macmule) AutoImagrNBI can be used to build a more customized NetInstall Image
- Reporting of actions taken by Imagr via POST request or to a syslog server
- If a Netboot server isn't an option, external projects can build a bootable drive for Imagr use

☆ AutoImagrNBI https://macmule.com/projects/autoimagrnbi/



34

- To learn more about Imagr, watch Graham Gilbert's presentation "0 to Imagr-ing in 45 minutes" from MacSysAdmin 2016

35

- Imagr support can be found first by reading the Imagr wiki
- Also, there's the Imagr channel on MacAdmins Slack

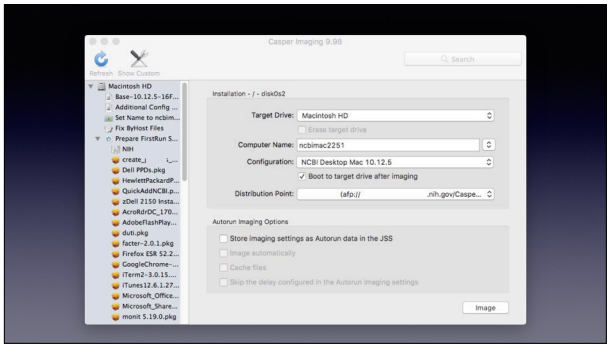
36

37

Casper Imaging

Casper Imaging is one component of the Jamf Pro suite (formerly known as Casper Suite). Its main purpose is to install software and (optionally) an OS on a Mac.

Casper Imaging is a standalone application. It's typically run from some kind of external drive or NetBoot image, although you can also use Thunderbolt + Target Disk Mode to image a number of machines quickly.



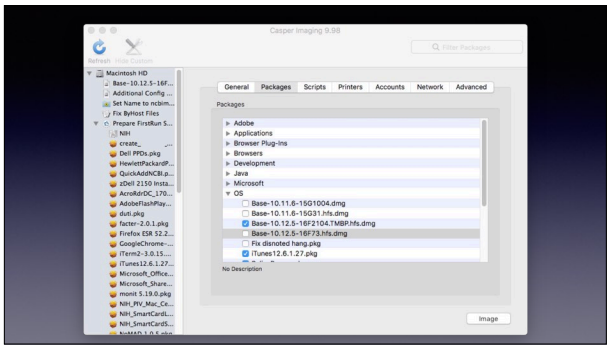
38

When the application is launched, first one typically has to authenticate to the Mac with administrative credentials, then log into the Jamf Pro server.

Once logged in, you're presented with a window like the one on this slide. It lets you choose your destination volume, whether that volume should be erased, specify a name for the computer, select an imaging configuration, etc.

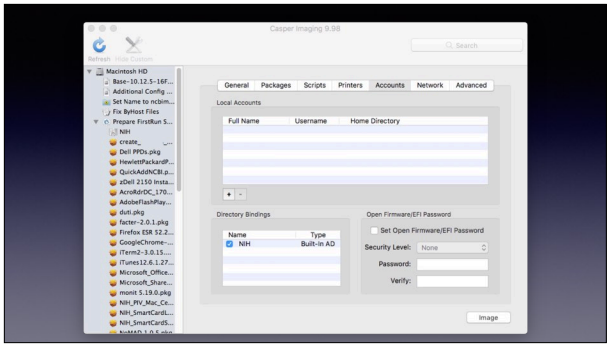
On the left side, you'll see presented a "recipe" of software that will be installed on the client, in the order that things will be installed.

If you are erasing the hard drive, the first thing that should be installed is a clean, basic OS image, usually created with AutoDMG. If not... you're going to have a bad day.



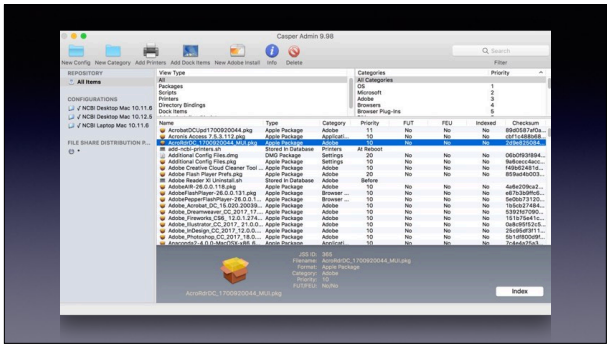
39

Note also the "Custom" option in the tool bar – this lets you create imaging configurations "on-the-fly", with the caveat that you cannot save these.



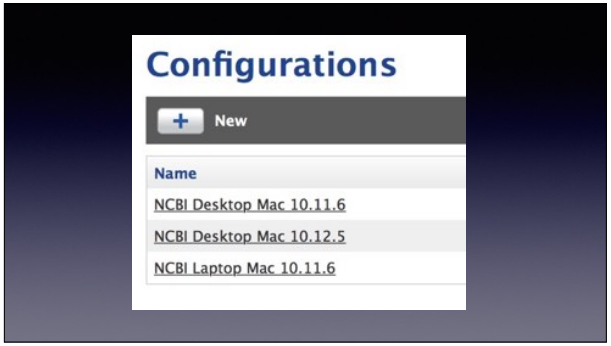
40

Also note the "First Boot" options. DMG files are copied to the drive first, then pkg files and, typically, scripts, are also copied to the new system. A first boot script is created, and this is what does the actual work of installing these packages and running these scripts. Assuming, of course, you have a valid OS on the device.



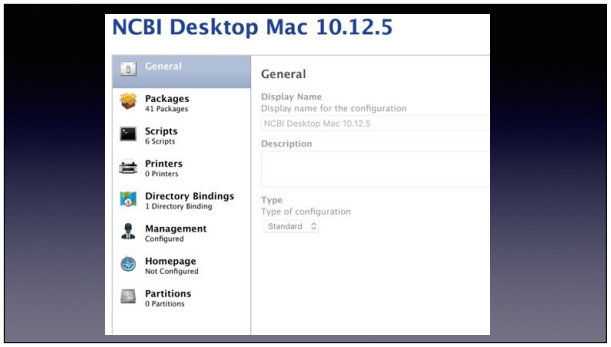
41

Imaging configurations are typically created in another one of the parts of the Casper Suite, Casper Admin, by creating a new configuration and then dragging and dropping OS images, packages, dmg files, and scripts to the configuration. Casper Admin is a tool used to add packages and dmg files to Jamf Pro, and we'll talk about it later.

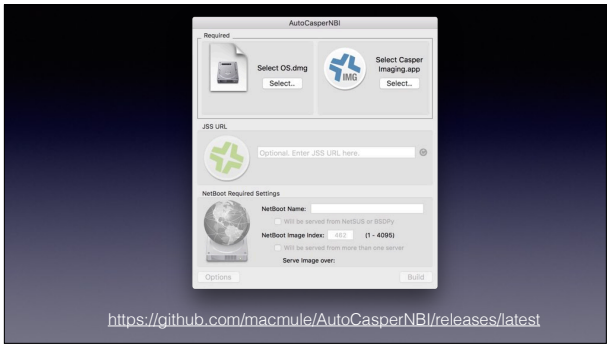


42

There is also a way to create and edit these configurations using the web interface of Jamf Pro. One thing about the web interface is that it is easy/easier to duplicate configurations and potentially make quick edits (swap out a package, etc) this way than using the Casper Admin app.



43



https://github.com/macmule/AutoCasperNBI/releases/latest

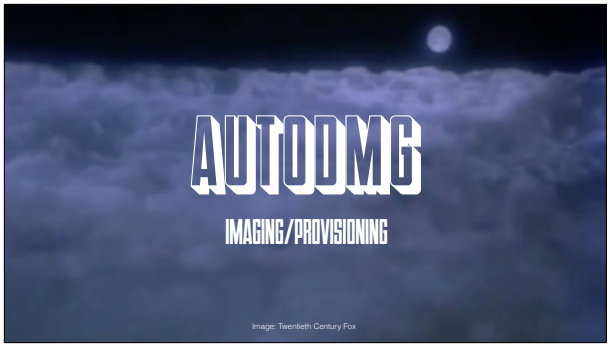
44

If you need to create a NetBoot image (PXE boot, lets you start up over a network), an invaluable tool is Ben Toms' AutoCasperNBI. You point it to a base OS, your JSS, your copy of Casper Imaging, and have some optional installs, and you can build a NetBoot image (which can also be restored to an external hard drive or USB flash drive).

#jamfnation
https://www.jamf.com/jamf-nation/

45

- Further discussion in the #jamfnation channel on MacAdmins Slack, and on https://www.jamf.com/jamf-nation/



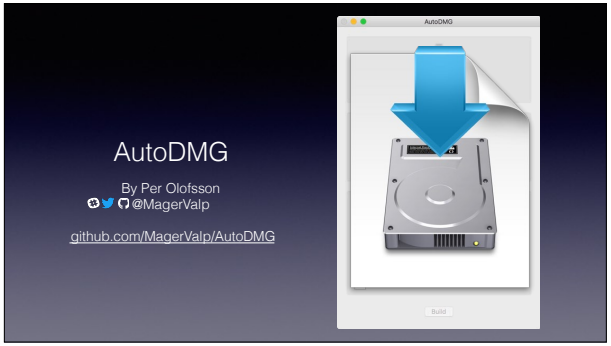
46

- Auto Damage is a GUI image creation tool
- Well, this is odd

47

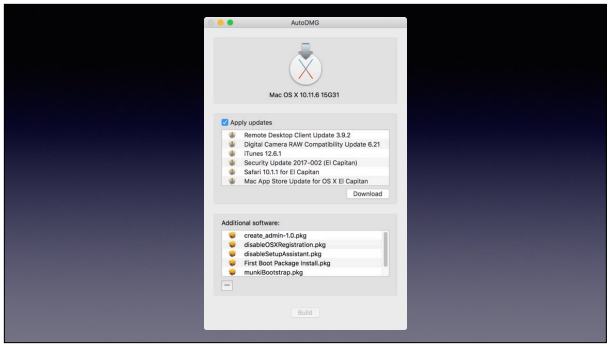
- And by odd I don't mean the pronunciation, that's the author's choice
- But…an image creation tool? Why talk about this
    - Realize everybody isn't necessarily jumping on the newest thing
    - Not everybody has the budget to be there
    - So a short diversion may be helpful for some folks in the audience

---



48

- AutoDMG is a free, open-source GUI tool written by Per Olofsson to create a "standard" macOS image
- Need an macOS install application (i.e. from the Mac App Store)
- **Supported on OS X 10.9 through macOS 10.12
    - Host computer needs to be on same major version of macOS that you're building
    - Great use of VM
- Make sure to Update Profiles in AutoDMG to make sure you have the latest Apple updates available for your macOS installation app



49

- Drag an macOS install application to the top section
- In the middle section you can download and apply updates, if desired
- Drag additional software to the bottom section
- Be simple! It's tempting to include Office, other things
- Don't, unless you have a really good reason
    - Gigantic software loads
    - Incredibly short turnaround times
    - Incredibly large numbers of computers to rebuild
- Only include enough software to attach to your management system

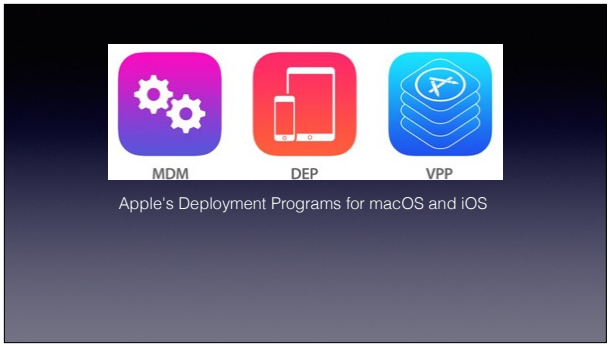
---

50

- Support is available through the AutoDMG channel on MacAdmins Slack

---

51



52

Also known to some as the Three Horsemen of the Apocalypse

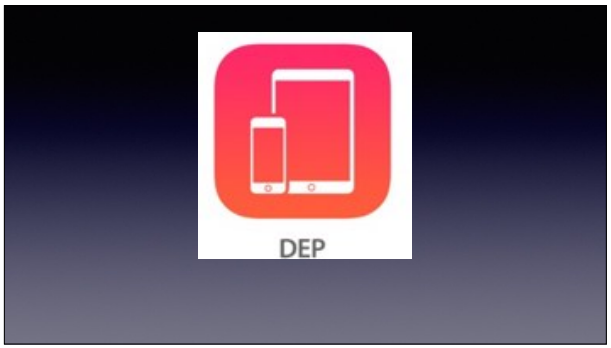
53

Mobile Device Management

Introduced by Apple in OS X 10.7 Lion, Mobile Device Management offered a standard framework of management commands that can be configured via profiles. These profiles can be installed in numerous ways, including by direct attachment (Configurator on iOS), via email or package, or “over-the-air” using an MDM server.

Apple’s implementation of Mobile Device Management is Profile Manager, which is part of OS X Server. It’s definitely OK to use as a “proof of concept” for testing and prototyping, but Profile Manager is not enterprise grade software, and it’s not particularly stable or reliable.

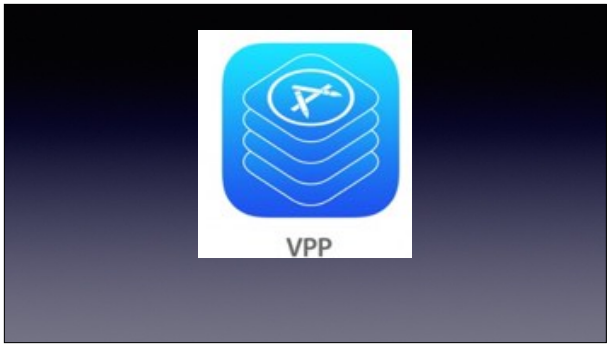
There are many commercial implementations of Mobile Device Management software in use by Mac and iOS admins today, including Jamf Pro, AirWatch, SimpleMDM, LanRev, Meraki, FileWave, MobileIron, Maas360, Windows InTune, etc. We’ll talk more about the kinds of things you can do with MDM when we talk about Profiles later on.



54

Device Enrollment Program (DEP)

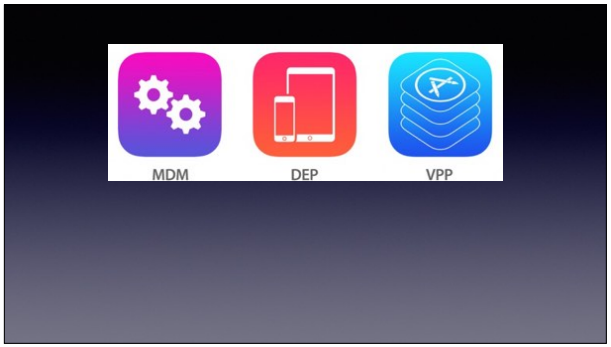
- This program is offered by Apple, and allows you to automate enrollment of Apple devices into a mobile device management solution in order to streamline deployment. For those of you who have had the joy of taking a new macOS or iOS device out of the box, you know the setup assistant that you have to step through to set up a new device? The goal of DEP is to allow you to customize this environment to pre-configure some of these steps, and reduce the complexity of the environment. For educational institutions, take a look at Apple School Manager (which we will not be discussing in this workshop).
- This works for most devices purchased from Apple after some date in 2011, and from resellers that support DEP. Please note that, for legal reasons, this is not offered in every country.



55

Volume Purchase Program (VPP)

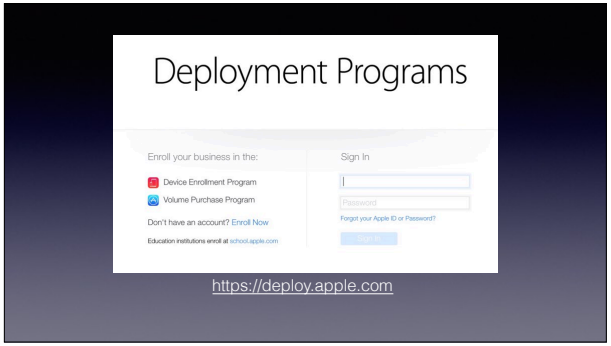
- This program is also offered by Apple. It allows you to deploy apps and books. You can distribute apps to devices (with iOS 9 or later, or OS X 10.11 or later) or apps and books to users.
- The same concern about VPP not being offered in every country.



56

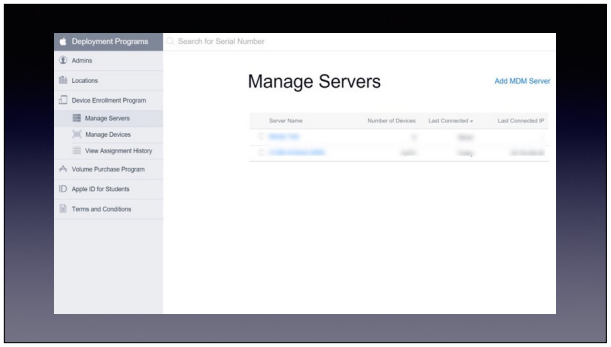
How does it work?

DEP and VPP are dependent upon you having an MDM solution. Depending on where you configure your devices and how your wireless network is set up, you probably want to have your MDM exposed to the Internet.



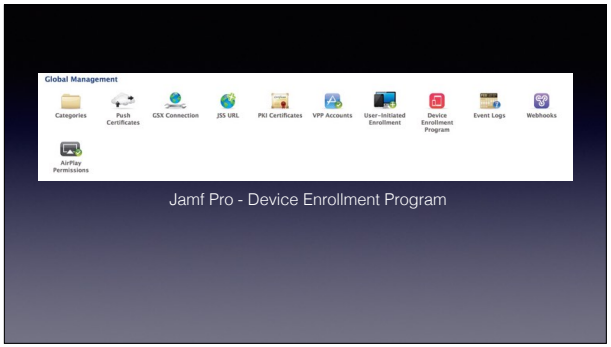
57

You need to visit http://deploy.apple.com to get signed up (i.e. Enroll Now). Do this step first: you'll need to create an email alias/distribution list, and, after enrollment, use that email address to create a master Apple ID (can't use an existing one), with two-step authentication, in order to use DEP. You'll also need your DUNS number. You'll also need to provide an organizational contact that can verify that you have signing authority for the organization, since you'll be agreeing to Apple's Terms and Conditions.



58

Once enrolled, you can define other DEP administrators, manage what MDM servers you have, and then assign Macs and iOS devices to an MDM server or different MDM servers.



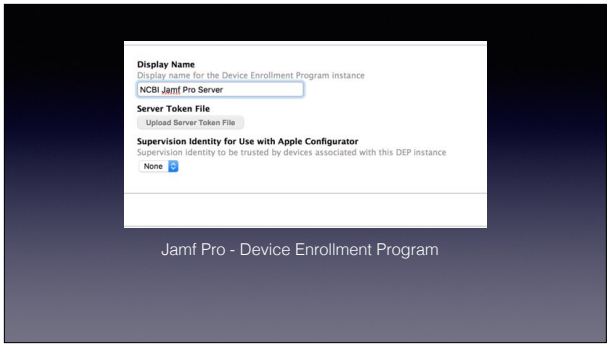
Jamf Pro - Device Enrollment Program

59

Once the servers are defined, you can download a token from the DEP portal that you will need to upload to your MDM server in order to associate it with Apple and establish trusted communication. Let's take a brief look at this for Jamf Pro:

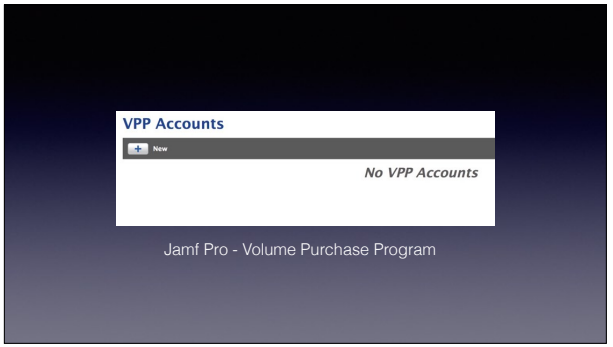
Jamf Pro - Device Enrollment Program

60



Jamf Pro - Device Enrollment Program

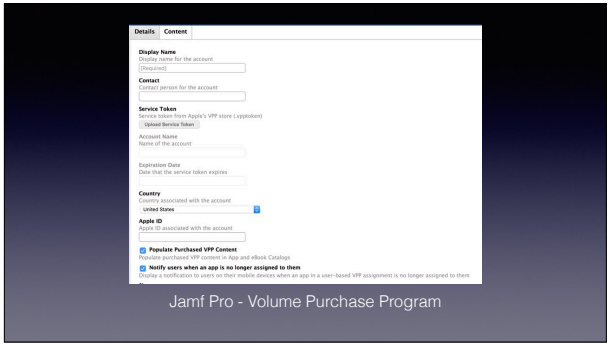
61



Jamf Pro - Volume Purchase Program

62

Same story with VPP, except there's also an Apple ID associated with the account, and it's called a service token.

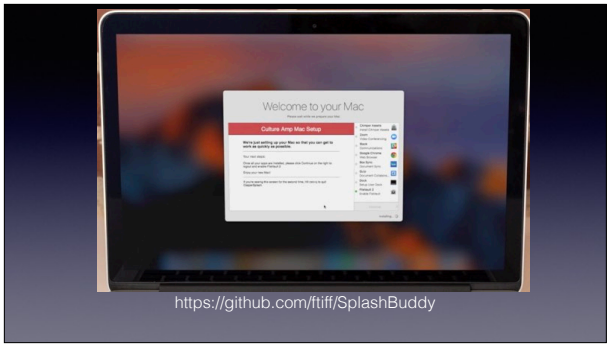


Jamf Pro - Volume Purchase Program

63

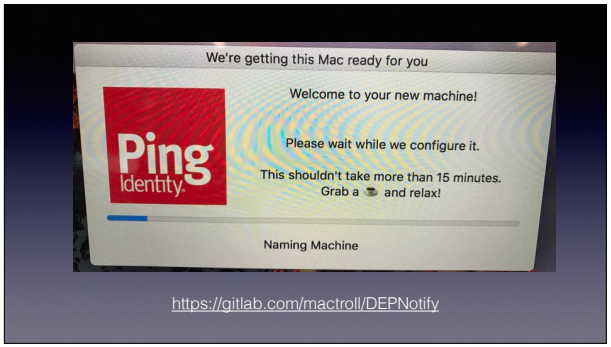
DEP on iOS allows you to lock down devices. Assuming your MDM server is "in the cloud", and you are supervising them, an iOS device that grows legs and leaves your network can be wiped and locked down, essentially "bricking" the device.

Sadly, the same is not true of OS X DEP today, as users can opt out, though you can force a "nag". There is no supervision for Macs today. This is an area that is likely to change in future versions of the OS.



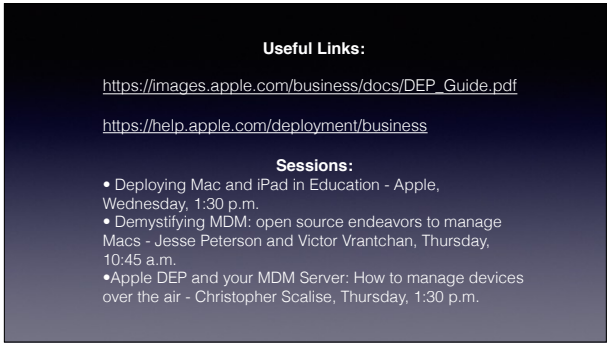
64

For macOS, there are a couple of useful third-party apps to customize the DEP experience. One is called SplashBuddy:
https://github.com/ftiff/SplashBuddy



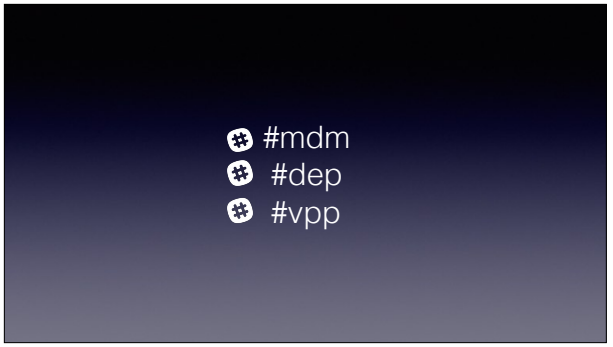
65

Another is called DEP Notify:
https://gitlab.com/Mactroll/DEPNotify



66

Resources



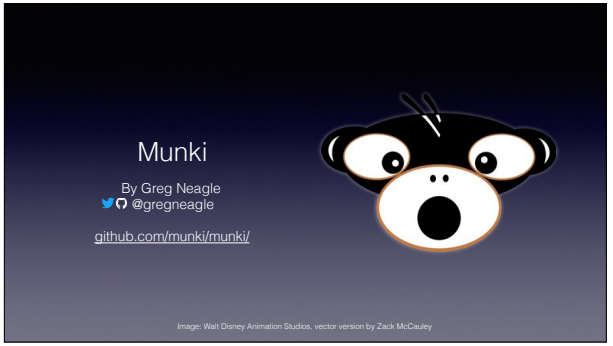
67

- Further discussion in the #mdm, #dep, and #vpp channesl on MacAdmins Slack

---

68

- How can you deploy software to your fleet of Macs?
- You can use Munki!
- What is Munki?
    - Software distribution framework



69

- Munki is free and open source written by Greg Neagle
- It is mostly client-side, there isn't a Munki "Server"
    - A web server, any web server
- This presentation will cover the basics of client configuration
- And the user-facing side of Munki, Managed Software Center
- Install Packages, copy item(s) from a "drag and drop" disk image, Adobe software packages, Configuration Profiles, and Apple Software Updates
- Operates on desired state, not doing an action to a computer
    - "xyz" software is (or isn't) supposed to be installed, perform an action if it's not

Client Defaults

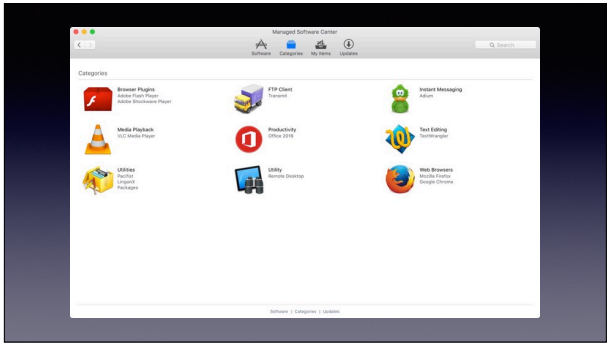
- SoftwareRepoURL
  Default: http://munki/repo
- ClientIdentifier
  Defaults: Fully-qualified hostname
  "Short" hostname
  Serial number
  "site_default"

70

- Configuration
  - There's a set of client defaults
    - Can leverage these defaults to your advantage
  - SoftwareRepoURL: URL to your Munki repo on your Munki server, defaults to http://munki/repo
  - ClientIdentifier: Fully-qualified hostname, "short" hostname, serial number, "site_default"
- Strongly, strongly recommended to set the SotwareRepoURL
  - It's a security risk to let the client find the server with defaults
- Also with security
  - Highly advisable to get https working before going live
  - Also some form of authentication, such as basic authentication or client certificates

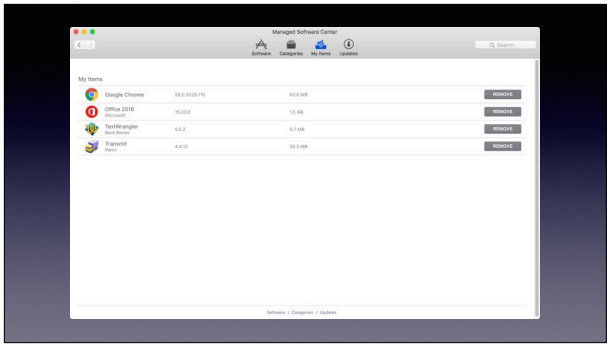
71

- Managed Software Center, the user-facing side of Munki
- Main "Software" page, list of optional installs and links on the right



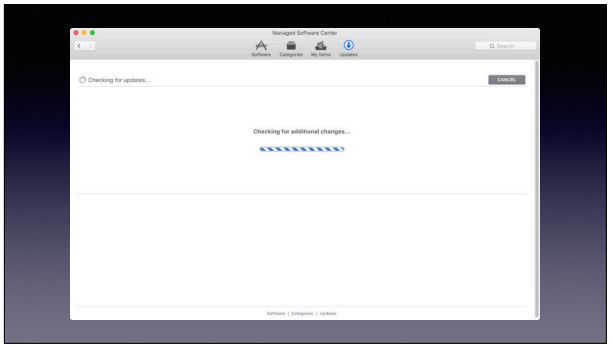
72

- Categories, for drilling down into optional installs



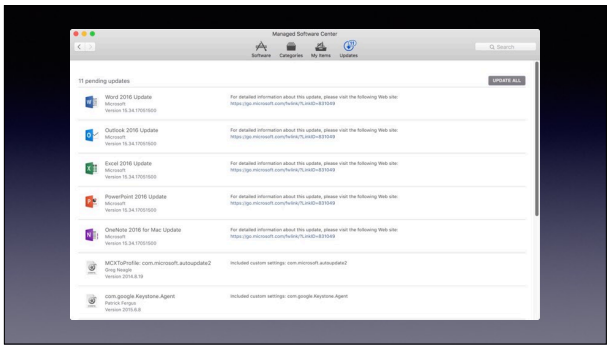
73

- My Items, for list of opted-into optional installs
  - Including removal, if possible



74

- Updates, where you can check for updates



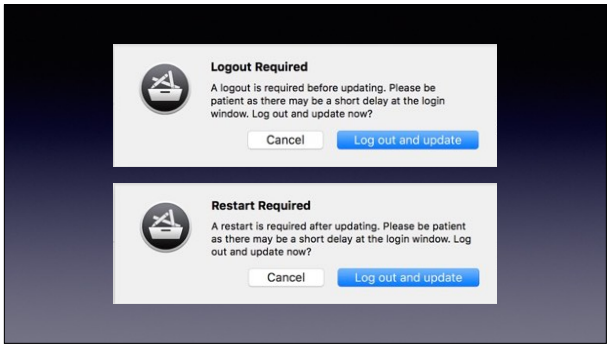
75

- And view items pending installation
- Munki client checks about every hour
- If updates are available, they can be installed silently if the admin sets the update as an unattended_install
- Otherwise MSC will open and the user will be prompted
  - By default, once a day

76

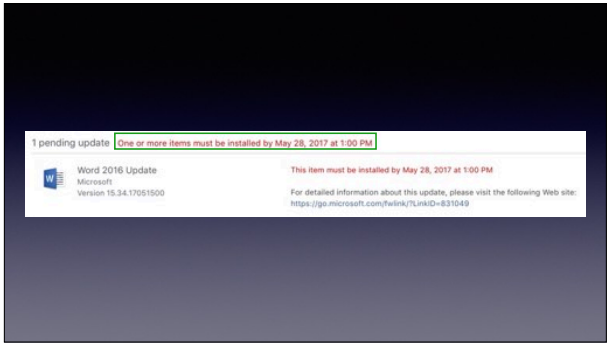
- Can coordinate requesting applications to quit
- This installation, possibly a Word update or a plugin for Word won't install unless Word is quit

---



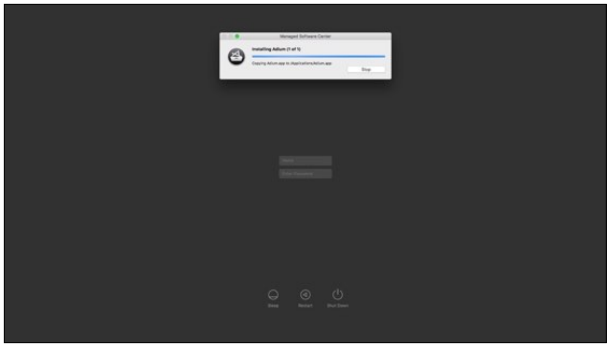
77

- Managed Software Center can also request a logout or a restart, and then handle restarting after installation
- The update could also just be applied at the login window



78

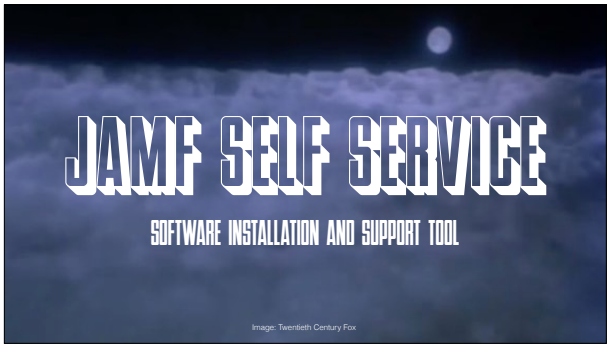
- If there is an update that's critical, MSC can tell the user (with increasingly pushy notifications) that it needs to be installed
- If the request to apply the update is ignored, MSC can forcibly log out the user and install the update



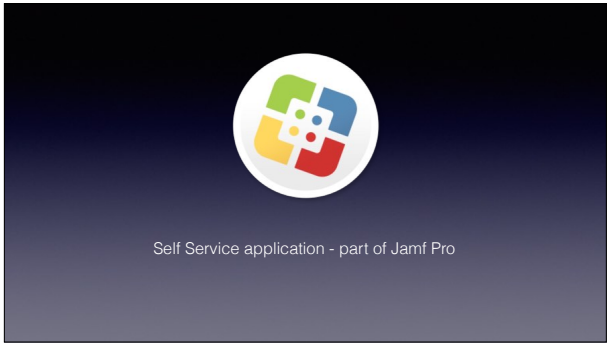
79

- And installations at the login window provide feedback about what's happening
- Exactly how the admin gets this done I'll cover in the "updates" section (a.k.a. patching) portion of the presentation

---



80

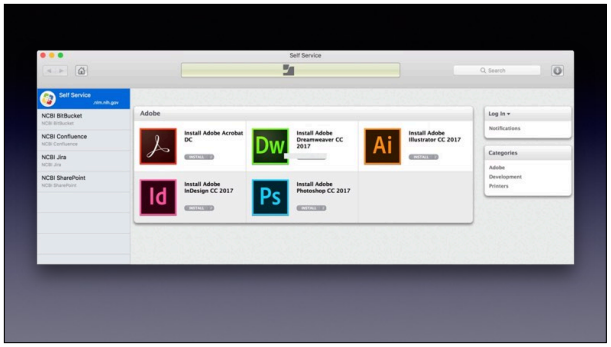


81

Self Service is another one of the applications in the Jamf Pro Suite. It allows end users to install software, printers, and perform tasks on their machine, whether or not they are administrative users.

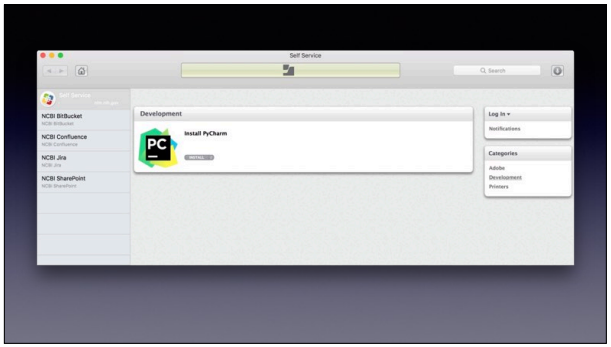
Self Service is typically installed by default in the Applications folder of each Mac that is enrolled in Jamf Pro. You can optionally brand it, although it's a bit involved. A totally rewritten, more "brand-able" version of Self Service is a feature in version 10.

The items that appear in Self Service can be customized, either by computer/device or by user. Not everyone's Self Service needs to appear alike.

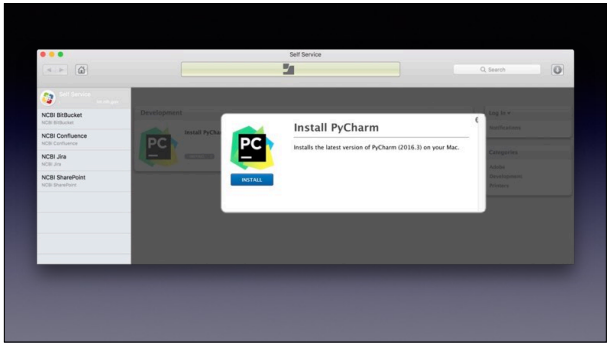


82

On the left side you can post web links to various other servers and sites. Back in my consulting days, I managed to convince most organizations to stop trying to manage web bookmarks in Safari, Firefox, and Chrome, and just resorted to putting internal sites in the sidebar.

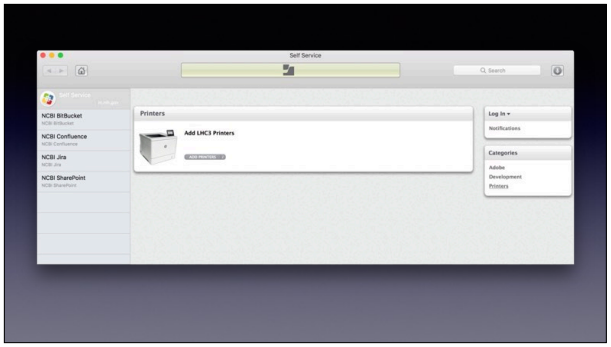


83



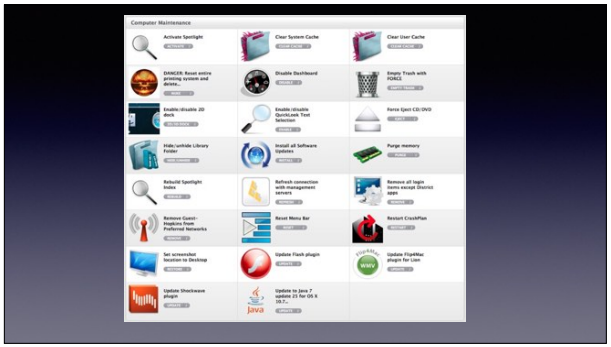
84

Applications can be featured, and you can force users to read the description before installing an application.



85

Another great use of Self Service is to offer printers to users. You can even have printers be made available to the user depending on what building and floor their computer is located. We'll talk more about policies and Smart Groups a little later on when we go into further depth on Jamf Pro.



86

Lastly, many organizations will put self-help tools into Self Service. No, not for psychoanalysis, but to do things like delete caches, delete Keychains, run Software Update, reboot the Mac, etc. The usual things that your help desk may want to try first. If you can get users to do some initial troubleshooting themselves, you can reduce the number of help desk tickets, while empowering users.

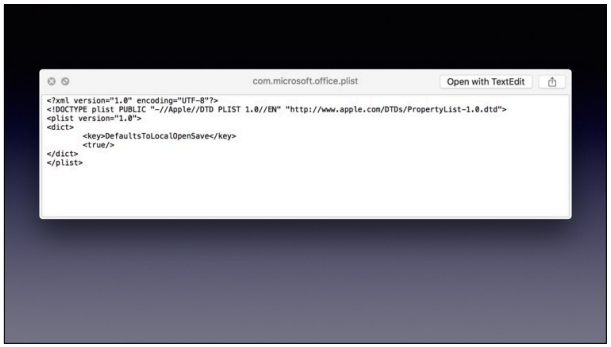
One of the cooler threads on Jamf Nation is the "Show Off Your Self Service"

#jamfnation
https://www.jamf.com/jamf-nation/

87

- Further discussion in the #jamfnation channel on MacAdmins Slack, and on https://www.jamf.com/jamf-nation/

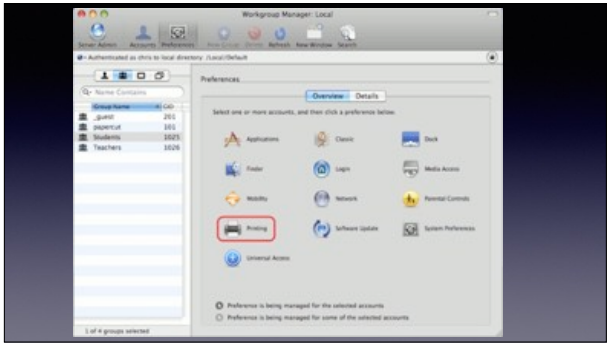
88



89

- One of the architectural design points of OS X was that preference files should be stored in a human readable property list (.plist) file, which used a standard XML format (NeXTStep, which Mac OS X was based on, used a similar (ASCII-based) format for these files). Again, this was a strong recommendation from Apple; developers (i.e. Adobe, Microsoft) have been known to use their own preference formats.

- There were a couple of advantages to this format:
- 1) Preferences would be “human-readable”
- 2) Preferences could be copied/moved between users, or installed in a default user profile (a/k/a User Template) when new users were created, so that they would “inherit” these default values.
-

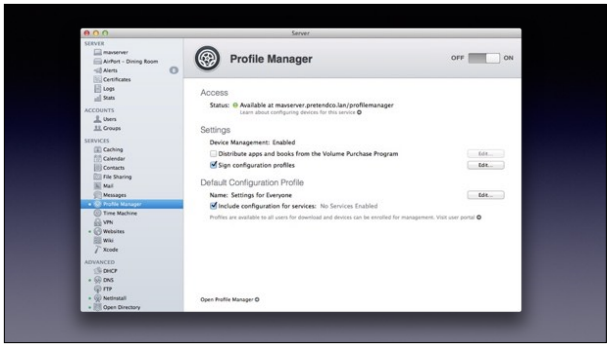


90

- Things got even more interesting for admins when Apple introduced MCX, Managed Client for OS X, in Mac OS X Server 10.2. This allowed client computers/users to have their preferences managed by OS X Server (and, later, Active Directory).

- We did so through the use of Workgroup Manager, an application that came with OS X Server. Workgroup Manager allowed you to apply settings to Users, Groups, Computers, or Computer Lists. In addition, you could specify whether these settings were applied Always, Once, or Often. These preferences were locally cached on the Mac (in /Library/Managed Preferences).

- It was very easy to create confusion (i.e. apply different Dock settings to a specific user, a group, and a computer = what do you get?). Open Directory was also very “fragile” (sensitive to DNS issues, date/time issues, corruption).

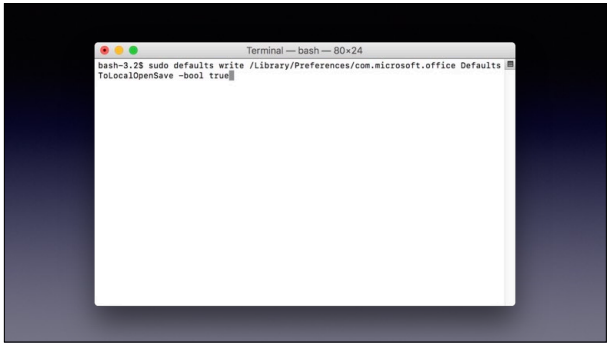


91

- When Apple introduced OS X 10.7 “Lion” they began the migration away from MCX to Mobile Device Management (which, as we discussed earlier, works for both Macs and iOS devices). MCX began to be deprecated In favor of Configuration Profiles, or Profiles for short. Apple began phasing out Workgroup Manager, pushing users instead to the web-based Profile Manager, which provided a GUI for some of the most commonly-used system settings.
- Many different MDM providers (Jamf, AirWatch, etc) use a similar user interface for configuration. But, you are able to create custom profiles, either via a GUI, or command-line tools,
-

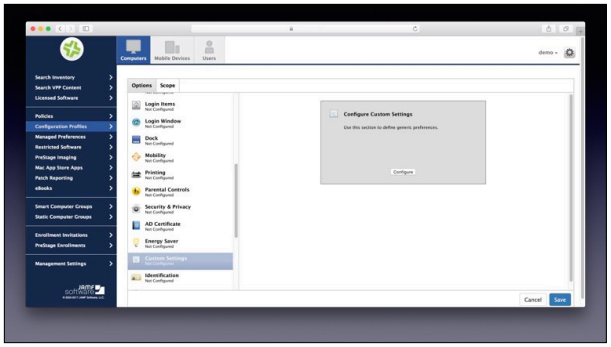
92

- The profiles are often installed “over-the-air” using an MDM. However, these profiles can also be saved as .mobileconfig files, that can be emailed, installed using an installer (Tim Sutton’s “make-profile-package”, installed via the “profiles” command-line tool, etc.



93

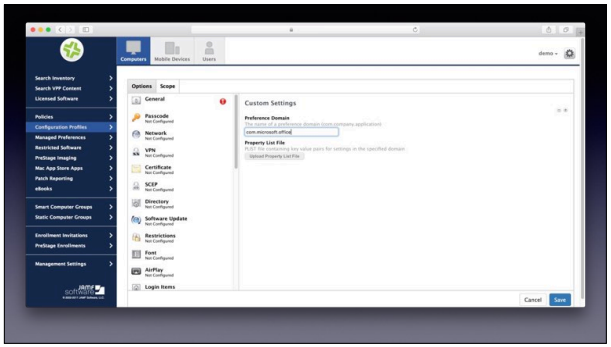
- The "old school" method of setting preferences via defaults write
- Note that, if this file does not exist, running this command will create a preferences plist file, in this case in the /Library/Preferences folder.



94

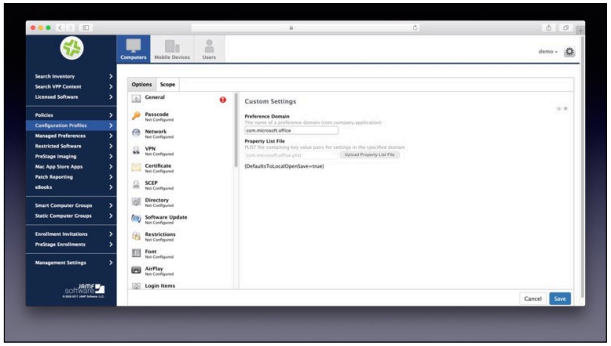
If we want to convert this plist into a profile, we have a number of options. One of them, Tim Sutton's MCX to Profile, is being discussed by Patrick a little later on.

- Here, we are in the Jamf Pro Configuration Profiles section. We are creating a new configuration profile, and we've navigated down to **Custom Settings.**



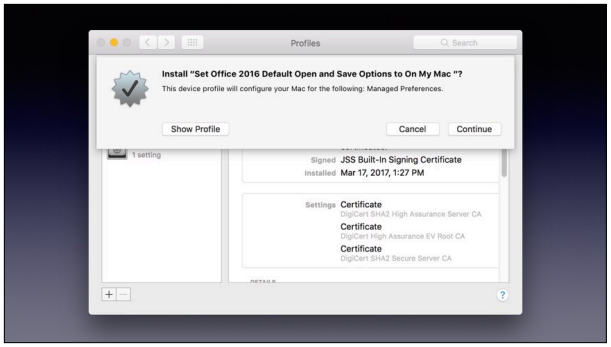
95

- First I must specify the preference domain. The domain is usually in the com.companyname.application format, in this case com.microsoft.office, since we are setting an office-wide preference.



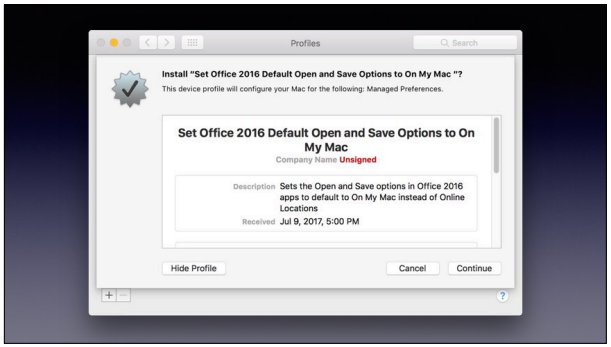
96

- Next, I'll click the button to navigate to the preference file that was created when we ran the defaults write command. In this case, it was in the "All Users" /Library folder at the base level of your drive, in that Preference folder. Note that, once uploaded, you'll see the actual preference key (DefaultsToLocalOpenSave) and the value (true). As long as I've given this configuration profile a proper name, I'll be able to Save it



97

- Once I've saved this profile, I can click the Download button in Jamf Pro, to download a copy of this .mobileconfig file to my Mac. Depending on how my browser is configured, it may actually try to load this profile on my Mac, by opening it in the Profiles System Preference pane.



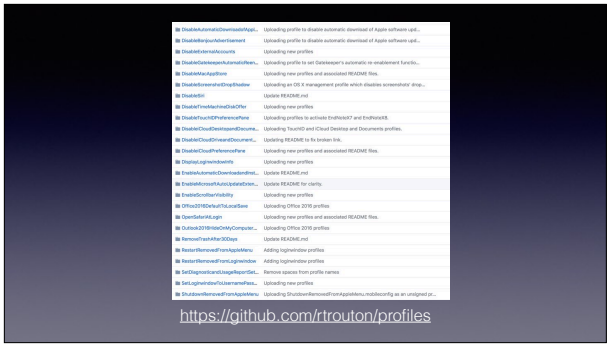
98

- If I click the "Show Profile" button, I can get more information about this profile, and whether the profile is signed. Profiles can and probably should be signed. Signing performs 2 functions:
- 1) tells you who created it 2) prevents it from being modified. A signed profile also may prevent your MDM from attempting to interpret and potentially change/corrupt the profile.
- https://osxdominion.wordpress.com/2015/04/21/signing-mobileconfig-profiles-with-keychain-certificates/

99

- Profiles are usually scoped to Computer Level or User Level. Gone is the ability to set items Once, or Often; if you need to replicate that functionality, consider Outset, which Patrick is going to talk about a little later

- Profiles can be locked with a password to prevent users from removing them, although users who are admins on their machines can remove all profiles (including MDM) by deleting the contents of /private/var/db/ConfigurationProfiles.



100

- Don't reinvent the wheel. Many of your Mac Admin brethren may have already created profiles to do things like disable iCloud prompts on a new system. There are a lot of pre-configured config profiles available on GitHub, on MacAdmins Slack, posted to Jamf Nation, etc.

101

- You don’t have to be using profiles today. The “copy preferences in to the default user template” approach, while discouraged, still works. MCX, though deprecated, still kind of, sort of works. You can always use the “defaults write” command to set preferences (either in the users’ ~/Library/Preferences folder, or /Library/Preferences for all users, assuming the app honors profiles there). However, it’s clear that Apple is moving towards an MDM/Profiles-only future for macOS and at some point in time these legacy methods (user template, MCX, defaults write) will likely stop working.
- Application Sandboxing potentially changes, or at least complicates, the location of preference files (~/Library/Containers, ~/Library/Group Containers
- Consider it your challenge to learn about/work with/master the use of Profiles, and replace all of the old methods you may be using now, over the next 12-15 months.
-

Useful Profile Tools

https://github.com/timsutton/make-profile-pkg
https://github.com/timsutton/mcxToProfile

Signing Your Configuration Profiles
https://osxdominion.wordpress.com/2015/04/21/signing-mobileconfig-profiles-with-keychain-certificates

Configuration Profile Reference
https://mosen.github.io/profiledocs

102

Previous PSU MacAdmins Sessions

- Implementing Configuration Profiles for OS X Management - Sergio Aviles, Jeremy Reichman, PSU 2016 https://www.youtube.com/watch?v=13oTDFcEOJ8
- Preference Management with Profiles – Greg Neagle, PSU 2016 https://www.youtube.com/watch?v=gTw3U-vxung
- Manage Third Party Applications with Profiles - Bill Smith, PSU 2014 https://www.youtube.com/watch?v=l7dGaULnY4o

103

#profiles

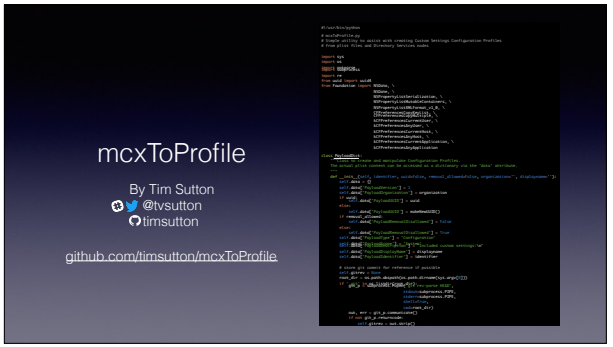
104

- Further discussion in the #profiles channel on MacAdmins Slack

105

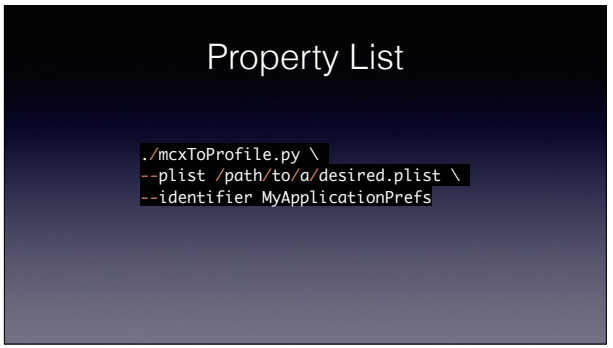
106

- If you have a collection of settings, how do you create Profiles?
- You may need to create a couple dozen Profiles to mimic your existing MCX management
- You can use mcxToProfile—what is mcxToProfile?



107

- mcxToProfile is a free, open-source script written by Tim Sutton to ease Profile creation
- A couple of caveats before using mcxToProfile
- If there is a built-in Profile (visible in Profile Manager), use it to manage things, rather than the Custom Settings Profiles mcxToProfile generates
    - But download and examine these built-in Profiles
    - Profile Manager (and tools that try to mimic it) can set too many keys, which can lead to conflicts between Profiles
- I stress you do this process additively, don't forklift move
    - Take the opportunity to evaluate what you want to manage (or not manage)



108

- Generate a Profile from a property list (plist)
    - Create a plist with defaults with *just* the settings you wish to manage and import that
    - This is probably the primary usage of mcxToProfile

Open Directory

```
./mcxToProfile.py \
--dsobject /LDAPv3/od.my.org/ComputerGroups/StandardPreferences \
--identifier MyBasePreferences
```

109

- Generate a Profile from an Open Directory node
  - Perhaps if you have a dual directory or Magic Triangle setup
  - This is less advised—it will grab *everything* and slam it into one Profile
  - Not granular, but possibly a starting point for deciding what you want to keep

---

Often & Once

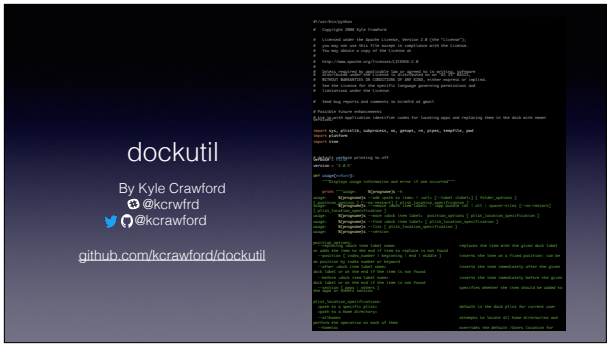
```
./mcxToProfile.py \
--plist /path/to/a/plist \
--identifier MyApplicationPrefs \
--manage [Often|Once]
```

110

- Always/Often/Once
  - “Always” (noted as “Forced in a Profile) depends on applications properly using Apple’s preferences APIs
  - “Often” and “Once” actually write to the preference file, which makes them more compatible with poorly-behaved applications
- I mention Always/Often/Once not to recommend you use it, but instead consider not using it
  - These features are unsupported by Apple
  - Strongly recommend using Outset
  - If you do want to use it, test, test, test

111

- For those moments when extra Dock flexibility is in order, there’s DockUtil
- Sometimes you need to edit the Dock, and Apple’s Dock Profile is not enough



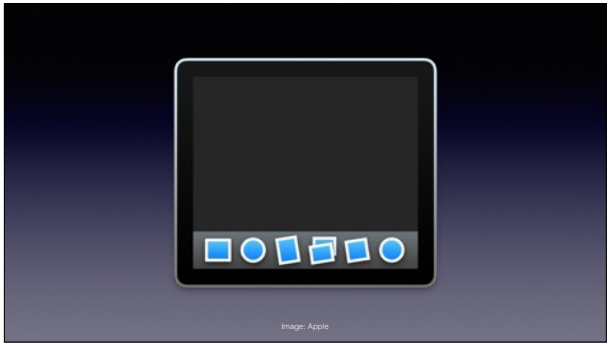
112

- dockutil is a free, open-source Dock editing utility by Kyle Crawford
  - Like peanut butter and jelly, dockutil goes great with Outset
- But there's a Dock Profile!
  - It's not flexible

113

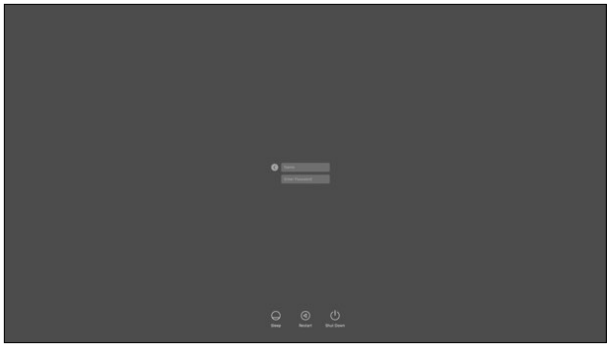
- List of things in the Dock, just a list
- No scripting tolerance of nonexistent applications
  - No tolerance of much of anything, really
  - Dock items are forced to be there, can't be removed
- All of this leads to user unhappiness with the Dock Profile
- Unless you're in a very locked-down environment (schools?) you'll eventually end up with Dock Profile exceptions
- All of this leads to the Dock Profile being pretty inflexible**
  - So let's ignore it for now—the Dock Profile is out there and you can use it if you like

---



114

- What makes dockutil so flexible and so preferable
  - Supports Applications, Folders, Stacks, and URLs
  - Add and remove items
  - Add items before an item or after an item
  - Replace an existing item
  - Folder view options (grid/fan/list/auto, display as folder/stack, sort name/dateadded/datemodified/datecreated/kind
  - Specific user or all users
- Consider whether "all" users might be better handled with Outset
  - Since it would only affect users who already exist on the computer
- Use cases



115

- First login
    - To start, dockutil has a built-in option to empty the Dock
    - Then add the desired items, and then (likely) never do it again

---

116

- During software upgrades
    - **If upgrading from Office 2011 to 2016
    - Or upgrading Creative Cloud software versions
- This is a time where dockutil's "all" users might be a good solution--edit the Dock for folks who have added Excel or Photoshop in the Dock

---

117

- But be considerate
    - Note dockutil's "replace" command will *add* items even if the previous item doesn't exist
    - Can write a script to "defaults read com.apple.Dock" and invoke dockutil to replace Dock icons only if they already exist
- dockutil by default will restart the Dock
    - Don't bounce Dock without letting folks know it's going to happen
    - Does anyone know what happens when the Dock is quit in this scenario?
    - A little video to show you what might happen if the Dock is killed while the user is logged in**

118

- One side trip from dockutil
  - After an OS upgrade, ever wondered how all *that* got there? Where did it come from?
  - How did Front Row and Game Center disappear during macOS upgrades?
  - How did Maps and iBooks get added?
  - Apple has one more file that edits the contents of the Dock

---

com.apple.dockfixup.plist

- Apple KBase HT207568
- tinyurl.com/modify-dockfixup

119

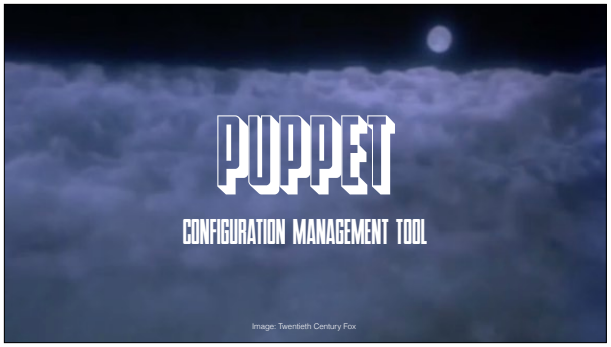
- That file is com.apple.dockfixup.plist
- Apple added a Configuration Profile setting, AllowDockFixupOverride, to control the DockFixup feature
  - And Owen Pragel wrote about it at the link above
- Setting this option in Sierra and beyond causes macOS to read a different DockFixup file
  - Or don't provide the override file, but do set the preference, which causes DockFixup to not run
  - May result in unsightly question marks in the Dock, if you're not handling things that are moved or deleted
- Be sure to test against future macOS and incorporate changes when Apple rolls out a new version

- https://medium.com/@opragel/how-to-customize-stop-dockfixup-on-macos-f349cd3c7b15 (or http://tinyurl.com/modify-dockfixup)
- https://support.apple.com/en-us/HT207568

#dockutil

120

- dockutil support can be found in the dockutil MacAdmins Slack channel



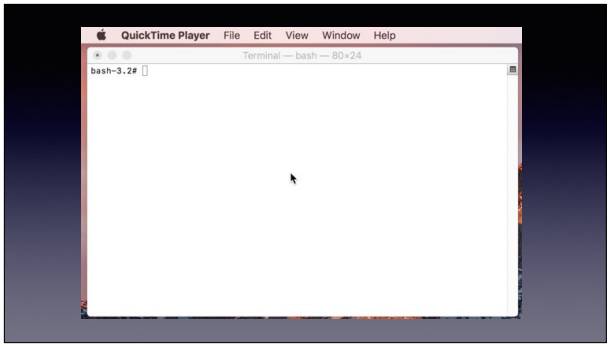
121

122

- Let's talk about Puppet. What is Puppet? Is it used for deploying software? No. Puppet is, more specifically, a configuration management tool.
- It allows you to maintain a specific configuration across a variety of systems. At my place of employment, Puppet is used to manage several settings on the Macs, including firewall rules, DNS settings, config files for tools like splunk, postfix, etc.
- But puppet is more widely used than just in the Mac environment. Where I work, my colleagues run the US government's most popular website, 11 months out of the year (IRS.gov always wins April). And, with over 4000 physical and virtual Linux servers, puppet is used to maintain them in a desired state.
-

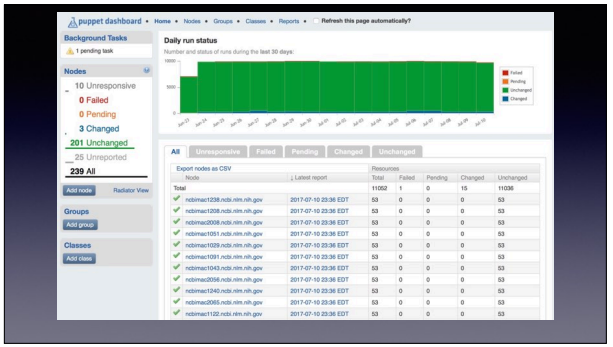
123

- Now, you could use something like Jamf Pro to do some of this work. Let's say you wanted to deploy a config file for splunk. That config file likely has the hostname embedded in it. So you'd have to deploy the file, then write a script using sed or awk commands to edit the file and change the hostname, and hope that you didn't mangle it.



124

- Let's give a real-world example. I accidentally deleted splunk's config file. I'll run puppet manually (by default it runs every 15 minutes). Boom, in under 20 seconds, config file recreated/restored.



125

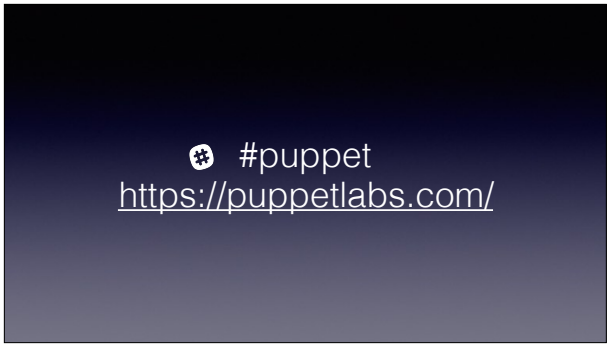
- Puppet is extensible – facter. If you know Ruby, you can write code to learn more about machines, and then conditionally make changes.
- Puppet can be used through the GUI, but at my work nearly everything is done using git. Git allows for source control – code and changes can always be identified, reviewed, tested, and, if something wasn't done 100% right, reverted.
- Puppet is very powerful, and it can be very "fun", but you have to have the programmer/development mindset. Start small, baby steps. I've been using it for about a year and I know enough to be dangerous.

Additional Resources

https://www.youtube.com/watch?v=GqerWmKU1Js
https://www.youtube.com/watch?v=Z2quMhdgILo

126

Graham Gilbert 2 previous presentations, at PSU MacAdmins and at MacAD.UK



127

- Further discussion in the #puppet channel on MacAdmins Slack

---

128

129

Similar to puppet in that it allows configuration via code (usually Ruby, also Python)
Meet your needs – simple configurations to creating users to dynamically creating configuration profiles to managing FileVault keys
Everything is done via code – no GUI
Automation
Repeatable
Consistent
Cross-platform (i.e. Facebook Chrome Cookbook manages both Mac and Windows

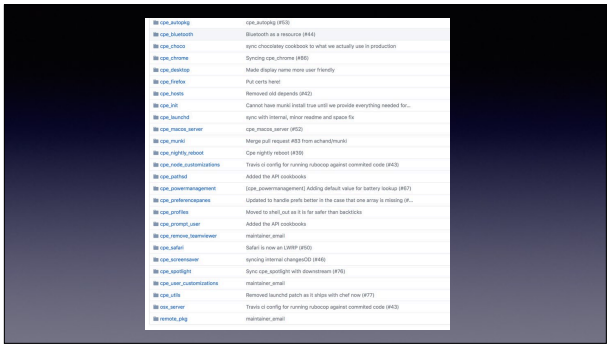
130

Chef supports Cookbooks

A cookbook is the fundamental unit of configuration and policy distribution. A cookbook defines a scenario and contains everything that is required to support that scenario:

- Recipes that specify the resources to use and the order in which they are to be applied
- Attribute values
- File distributions
- Templates
- Extensions to Chef, such as custom resources and libraries

takes the analogy further - kitchens, knife, etc.



131

- Facebook is one of the biggest proponents of Chef across multiple platforms
- https://github.com/facebook/IT-CPE/tree/master/chef
-

Session:

- Easy macOS Management with Chef - Nate Walck, Friday, 10:45 a.m.

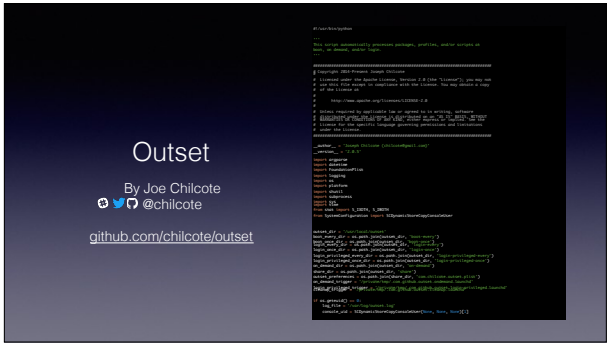
132

133

- Further discussion in the #chef channel on MacAdmins Slack, and information at https://www.chef.io/chef/

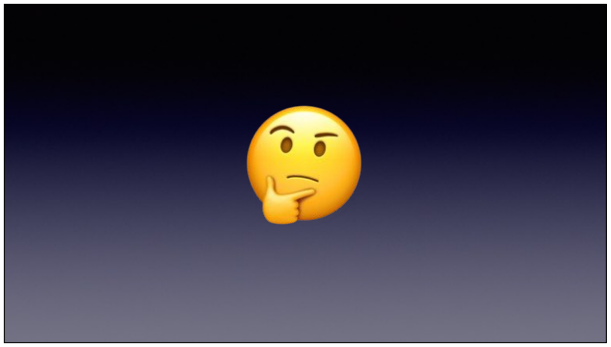
134

- Outset—what is it?
  - Outset doesn't do anything in particular by itself
  - It is a tool to do other things
  - For example, if you need to do a thing that Apple doesn't directly support, or doesn't support in a way you'd like
- If Apple doesn't support what you're trying to do, file an enhancement request
- Once you've done that, let's take a look at Outset



135

- Outset is a free, open-source tool written by Joe Chilcote
- Outset can
  - Run script
  - Install pkg
  - Install Profile
- When can Outset do these things?
  - At startup or login
  - Do the action a single time or every time
  - With user privileges or elevated privileges
  - Can do a thing on demand in the user's context
  - Can exclude particular users
- But wait, isn't this a lot like LaunchAgents and LaunchDaemons?



136

- Yes, there is overlap between a LaunchAgent/LaunchDaemon and what Outset can do
  - Outset won't replace every LaunchAgent/Daemon, but can limit the number you have to write
  - Outset covers additional use cases—package and Profile installation, and elevated actions at user login
- Well, what about loginhooks, maybe you've heard about those?
  - loginhooks are deprecated, since the release of Mac OS X 10.4 "Tiger" on April 29, 2005

Image: Interscope

137

- Ever since Gwen Stefani told us that "This stuff is bananas, B-A-N-A-N-A-S.", loginhooks have been deprecated
- LaunchAgents and LaunchDaemons also can have interesting trigger conditions, ones that aren't part of Outset
  - Watching a path for a file
  - Keeping a process alive
  - Starting on a timed interval

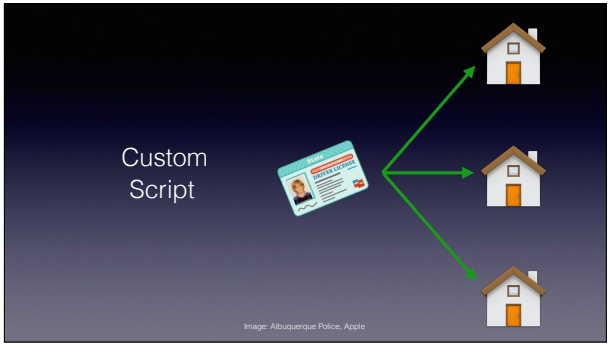
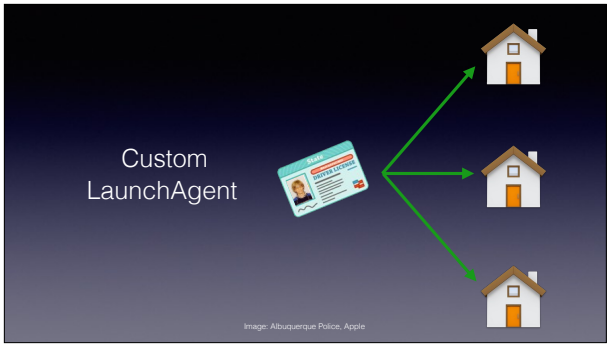


Image: Albuquerque Police, Apple

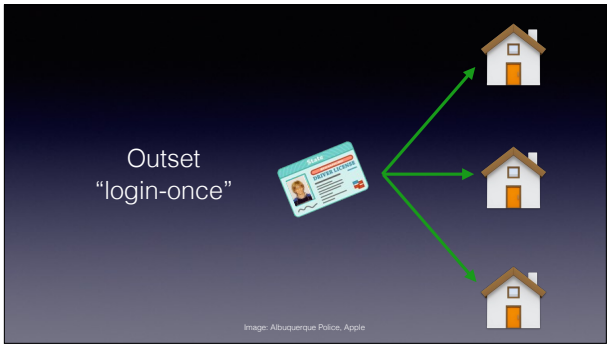
138

- Let's say a license file needed to be copied into each user's home directory
  - Could write a script that walks every home directory
  - Which doesn't help anyone who was migrated in after the script was run
  - And doesn't help network homes (if that's even a thing any more)
  - In short, doesn't help anyone who doesn't already exist when the script is run



139

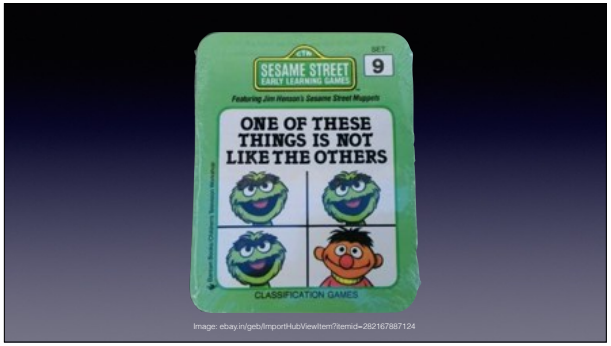
- Could create a LaunchAgent to run at login, then need to create the logic to handle only running once
- Which is completely dependent on your scripting chops



140

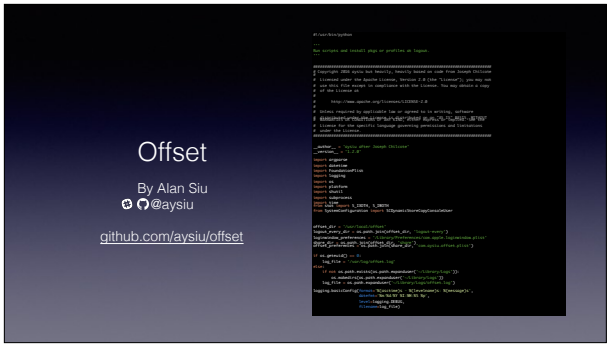
- Instead, write the license copying script and put it in Outset’s login-once directory
- Outset will run the script at login and record that it was completed
    - They will not be run again, even if the content of the script changes (although this behavior can be overridden)
- Instead of writing boring, not-unique things (run at login, run only once), you just get to write the unique part (copying the license) and let Outset handle the rest

---



141

- What about some differences between Outset and LaunchAgents, LaunchDaemons, and loginhooks?
    - Unlike a loginhook, Outset doesn’t cause the login process to pause
        - Don’t depend on LaunchD jobs to run immediately at login, they’re lazy
        - Test to make sure environment is as you expect, say if you’re using DockUtil
        - Note an Outset boot-once script will wait for a network connection and delay the login window
- What about logouthooks



142

- Logouthooks, also deprecated in the late Gwen Stefani period, run processes at user logout
- Fortunately there's Offset, by Alan SEEYOU
- This LaunchAgent can run short-lived processes at user logout
- If you need to do something at logout, take a look at it

143

- Back to Outset, what workflows could you dream up
  - Install Profiles
  - Gently enforce settings
    - Since Profiles that set something "Once" aren't official supported by Apple
    - Make sure to test *changing* that setting a few subsequent times, since Apple may protect a setting and only let it be set with a Profile
    - One such protected setting is Safari's homepage
  - Installing a poorly-written package that installs into user space and can't be reformed
  - Configuring the Dock with dockutil
  - Reenable ARD/SS remote control

144

- One more common usage of Outset is to set default applications
  - This can be achieved with either duti by Andrew Mortensen, or cdef by FREN-swa la-VO
  - These are two projects I haven't covered, but they can set a default application at the command line
  - Can set a default for a file extension, or a protocol such as http or mailto
- But overall, remember—if there's a Profile that satisfies your needs, use that instead of rolling your own solution

145

- Outset support can be found in the Outset channel of MacAdmins Slack

---

146

- Inevitably you're going to need to create your own package
    - Your own custom license file, script, or application
    - How do you land it on a computer?

147

- What will you make?
- But before we dive in, a few comments

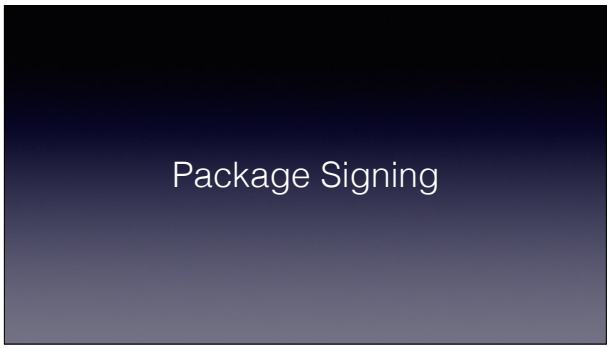
148

- There's some ground rules to packaging
  - Keep them simple, every new addition is more QA
  - A lot of bad ideas, but for simple packages the main two are:
  - 1) Don't use scripts unless you have to
  - 2) If you do use scripts to don't use them to do things you could do with a file-based payload
  - Three links in the presenter notes on packaging best practices, including the "Commandments of Packaging"

☆ Commandments of Packaging
https://www.afp548.com/2010/06/03/the-commandments-of-packaging-in-os-x/
☆ Packaging Guidelines for macOS
https://themacwrangler.wordpress.com/2017/04/28/packaging-guidelines-for-mac-os/
☆ "Introduction to Packaging"
https://www.youtube.com/watch?v=oKxjxi9Eny8

149

- Strongly consider AutoPkg
- Especially if you're building a package that's not custom to you
  - The work may already be done
  - If this is to package (or repackage) software that'll have future releases, AutoPkg limits the chances for mistakes
- And if it's not custom, write an AutoPkg recipe and share it
  - Help others in the community



150

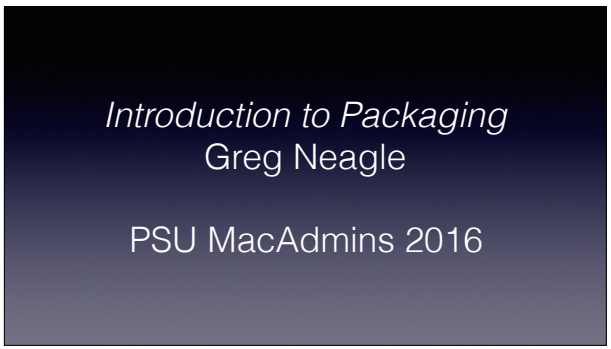
- Get a developer account from Apple
- It's $99, and will get you a certificate from Apple to sign packages
- If sharing packages with others, signing allows them to install packages without disabling security features
- Signing isn't absolutely required in every situation
  - Still important to learn and test
  - A future time with only signed packages
- For a guide on retrieving a singing certificate from Apple, links to Erik Gomez's and Greg Neagle's instructions on retrieving a signing certificate from Apple in the presenters notes

☆ http://blog.eriknicolasgomez.com/2017/03/08/Custom-DEP-Part-2-Creating-a-custom-package-and-deploying-Munki/#obtaining-your-certificates
☆ http://macadmins.psu.edu/wp-content/uploads/sites/24696/2016/07/psumac2016-33-PSUMacAdmins2016-Packaging-Workshop.pdf

151

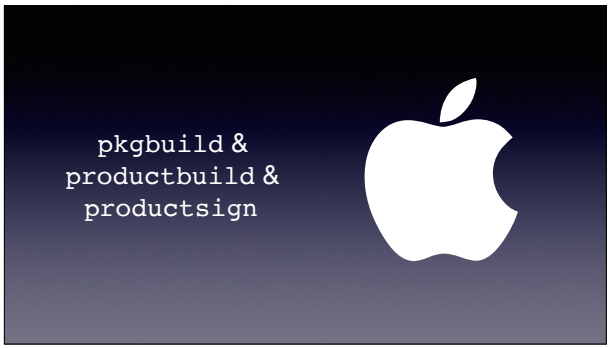
- Component Packages and Product Archives, these are the orange and brown installer package icons you're used to seeing
- Component Package
    - Installs one component, an application or plugin
    - Usually part of a product archive
        - But can be installed alone
    - No control over installer's behavior, but optional preinstall and postinstall scripts
- Product Archive (a.k.a. Distribution Package)
    - Contains one or more component packages
    - Can control the behavior of installer, OS and target disk requirements
- Signed Product Archives are required for direct MDM distribution
    - And likely other things Apple will dream up in the future

---



152

- For more on the ground rules of packaging, watch Greg Neagle's presentation "Introduction to Packaging" from Penn State MacAdmins 2016



153

- A mention of Apple's command-line solutions, pkgbuild, productbuild, and productsign
    - However they're not necessarily user-friendly, at least not for direct use
    - And for landing a handful of files on a computer, possibly quite complicated
    - I'm not saying they're *bad*, but we'll put them aside for now

154

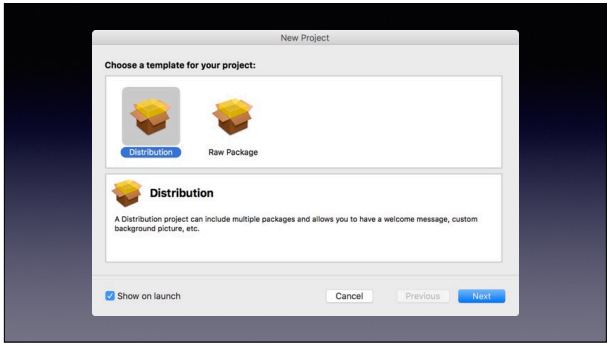
- The first and most approachable option is WhiteBox Packages
- While the official name is “Packages”, most will refer to it with the company name (WhiteBox) to make it clearer

---

155

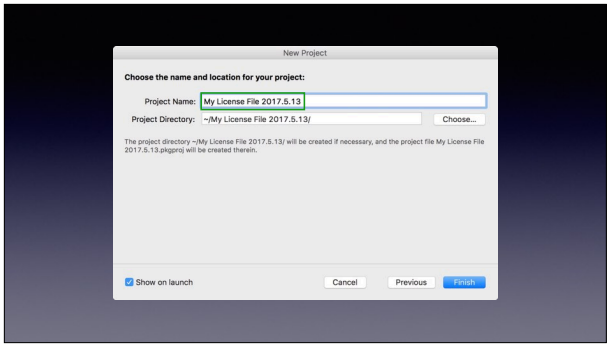
- Whitebox Packages is a free packaging utility from STEF-en su-DRA
- But this isn’t his first packaging utility
- You may have heard of Iceberg
    - In case you see mention of Iceberg, it was a great product, technology needs and features have moved on
- We’ll continue with Packages

☆ Packages Information Page
http://s.sudre.free.fr/Software/Packages/about.html



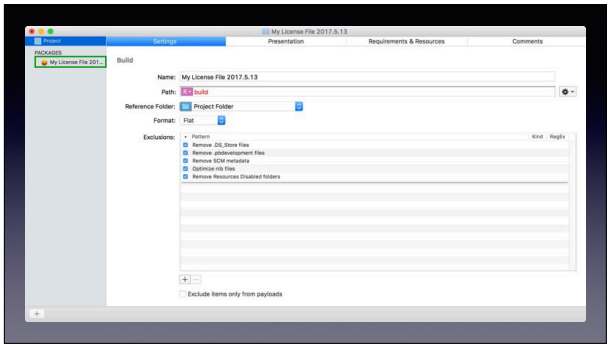
156

- When Packages opens you’re asked whether to create a Distribution package (a.k.a. a Product Archive) or a Raw Package (Component Package)
- For simplicity, we’ll talk about the Distribution package (Product Archive), since many things are applicable to the Raw package (Component Package)



157

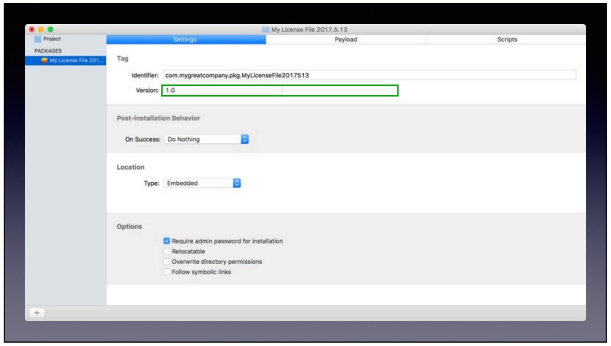
- Let's give the project a name



158

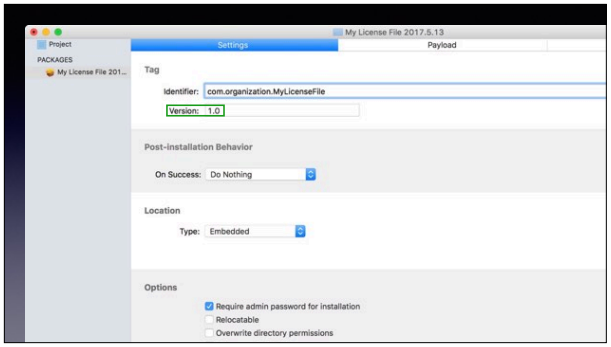
- The default settings for the Project are fine, but we need to look at the embedded package
- **Let's click on the orange package icon in the left-hand column

---



159

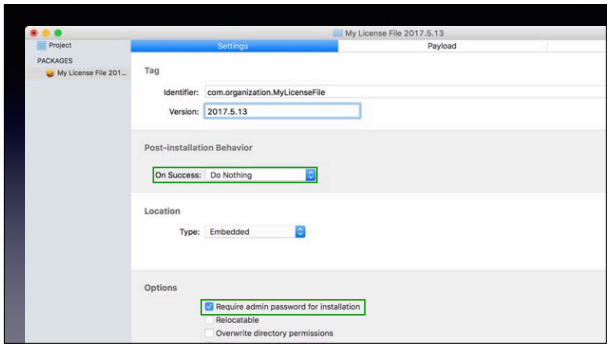
- Let's take a look at the settings pane first
- **The default settings for Identifier and version need to be fixed
- The Identifier should be a reverse DNS reflection our current organization, and shouldn't include a version number
    - Except if the software could be installed alongside an older version, then a major version number (CC 2017, 13, whatever) should remain
- We'll fix that



160

- Since this version doesn't reflect reality, let's fix that too

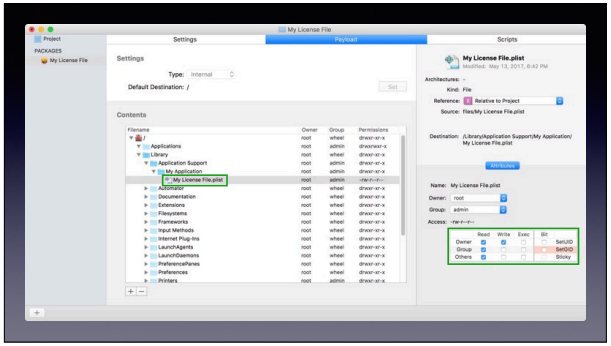
---



161

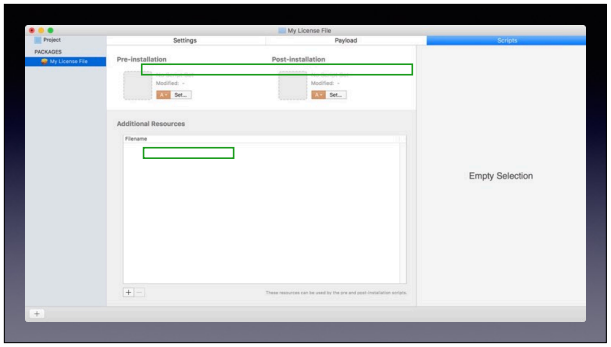
- A few more things to look at
- Post-Installation Behavior: Can require a restart, logout, or shutdown after package installation, but use it sparingly
- **Require admin password for installation: Given you're likely installing in protected locations, reasonable to leave this on

---



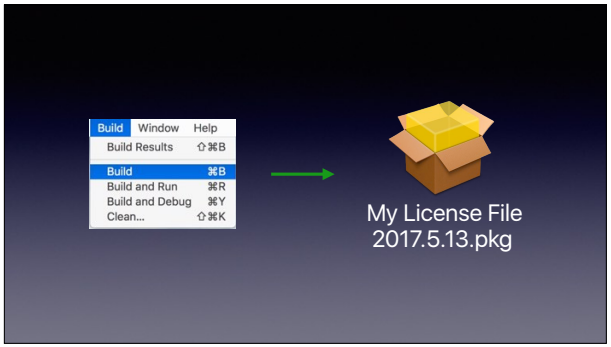
162

- Following that is the Payload tab
- Where target directories would be added and files provided
- **In this case "My License File.plist" is added under /Library/Application Support/My Application
- As files are added, Packages offers to keep permissions or inherit from the destination, inheriting is *generally* ok
- **If ownership and permissions changes to files or directories are needed they can be applied in the lower right-hand corner



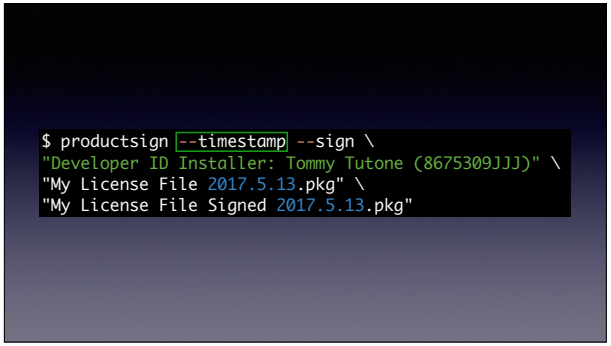
163

- Finally, the scripts tab
- **Can add preinstall and postinstall scripts
    - Again, you *can* add scripts. There is no *must* add scripts
- **One other curiosity is the “Additional Resources” section
    - This is where package resources that *aren’t* installed can be placed
    - Maybe a script requires a specific binary to run or to reference a license file
    - Or maybe a third-party installer that can run successfully (this is rare)



164

- Once you’re satisfied, Build the package
- The package will be dropped in a “Build” folder inside the Packages project
- And of course, once you have the package test, test, test



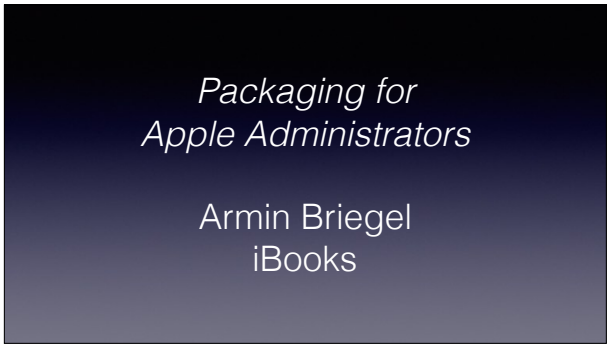
165

- Signing the newly-built package requires productsign and an Apple signing certificate
- **Note the timestamp option—this should make your packages be considered signed past the point when the actual signing certificate expires
    - The timestamp option (or the lack of it) caught some developers unaware when their packages expired

166

- General support for packaging can be found through #packaging
- Or possibly inside the MacAdmins Slack channel for your management system
- If you think you've found a bug in Whitebox Packages, you can tweet @packagesdev

---



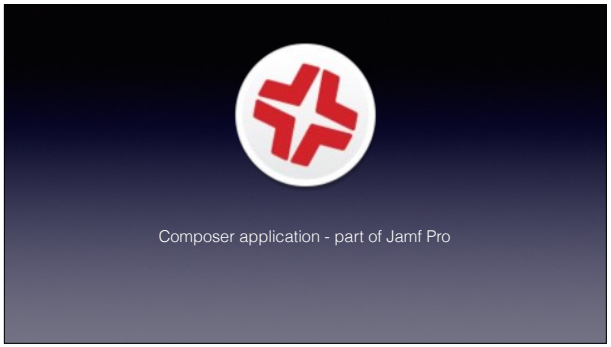
167

- For a more structured approach to learning about packaging, also check out "Packaging for Apple Administrators", by Armin Briegel—available on the iBooks store

---

168

169



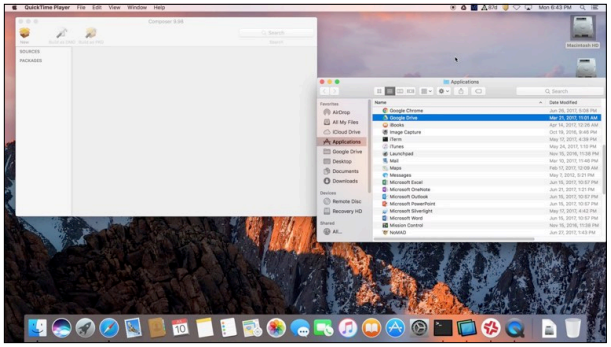
170

- Composer is one of the applications that's part of the Jamf Pro suite. Its main claim to fame is the ability to build packages.
- Composer has a couple of different modes: one where it will build "snapshots" both before and after you install a piece of software, then build a list of files that have been added or changed; and a second, less-known method where you can drag and drop files into the Composer window.

Let's take a look at both of these options:

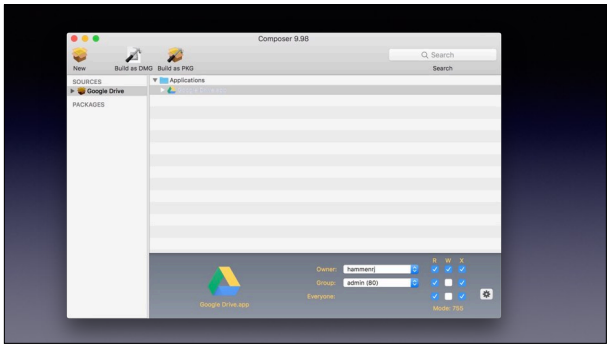
171

- Stop. Note that your snapshot is capturing files unrelated to the installer package.
- Careless use of Composer is called Composer-itis. Don't contribute to composer-itis.
- Snapshot packaging should be your last resort. Be careful. Snapshot with no other applications open if at all possible. Potentially use a virtual machine that you can easily roll back.
-



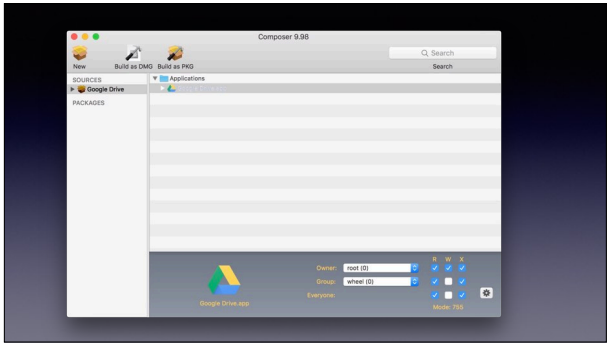
172

The other method that Composer can do is drag and drop



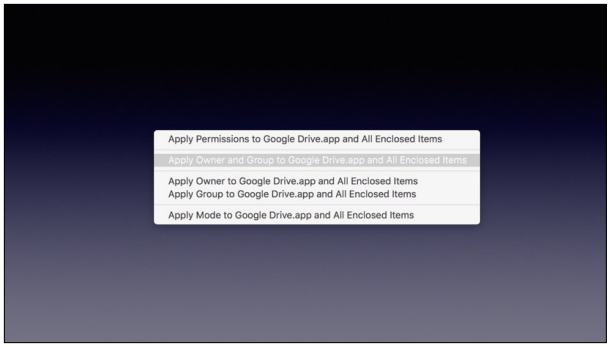
173

- Now, I may want to make some changes. Look at the application owner and group in the lower right corner. That's me. I probably don't want to push the application to end users this way.



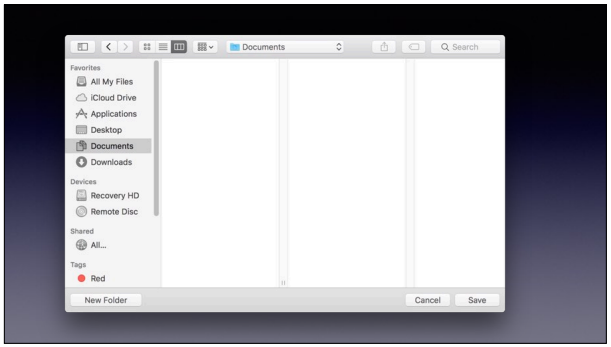
174

- I can click on the owner and group and change them to recogized users on the system



175

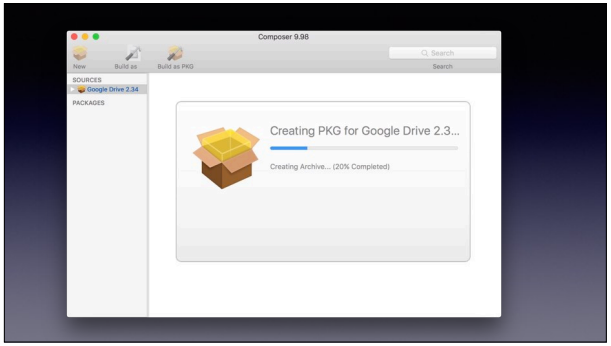
- I can then apply this owner and group to the app and all files enclosed inside it.



176

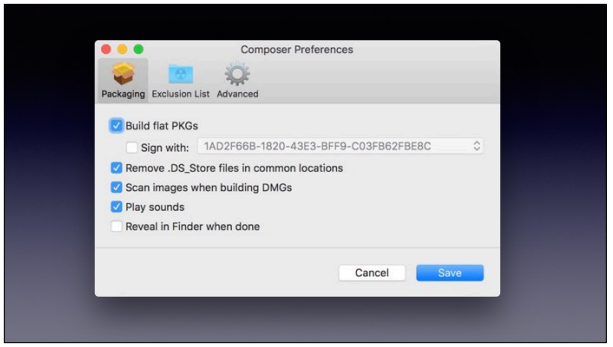
- When I'm satisfied, I click Build Package and am prompted where to save it

---



177

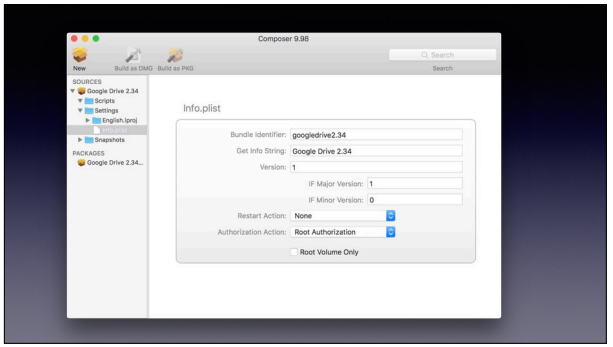
- You'll see the progress as the package is built



178

- Note that, in Composer's preferences, you can digitally sign your package with an Installer certificate issued from Apple's Developer Certificate Authority. Investing $99/year to join the Mac Developer Program is probably a really, really good idea

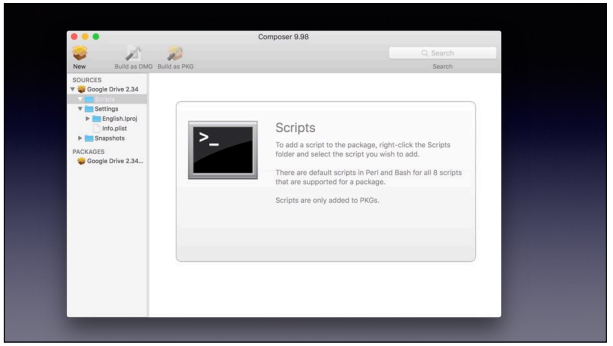
---



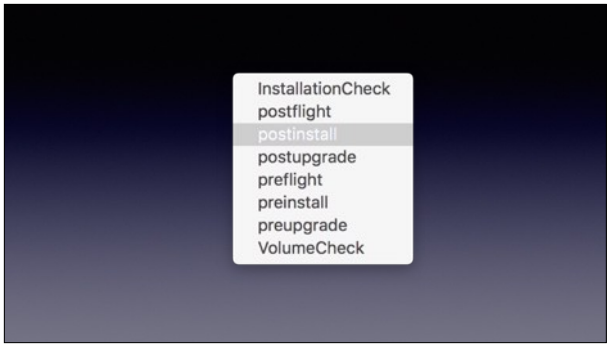
179

- Let's take a look at some advanced packaging info.
- You can set the Get Info string, version, specify if the package requires a restart, requires admin credentials, etc

---



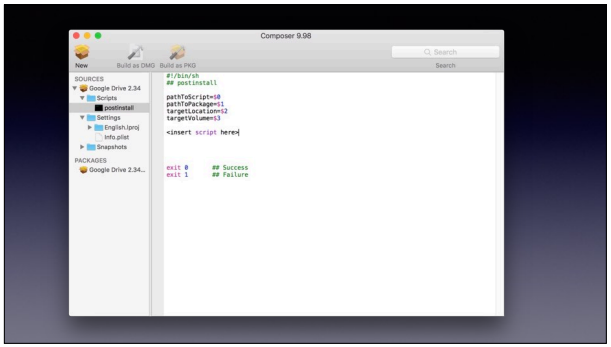
180



181

Note the various types of scripts that can be added to a package.

The modern (flat) package only allows for preinstall and postinstall scripts, so I'd recommend only using one of these types of scripts.



182

Here's where you can enter your script. Be sure to test it first before adding it to your package.

183

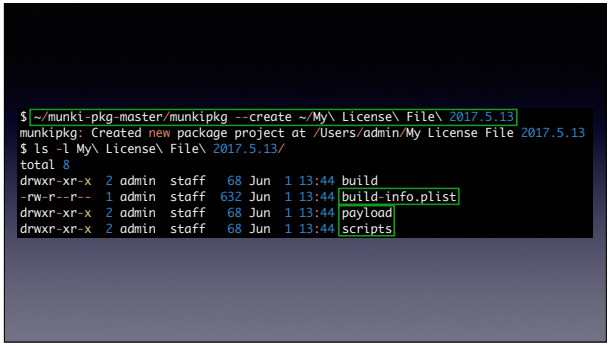
- Further discussion in the #jamfnation channel on MacAdmins Slack, and on https://www.jamf.com/jamf-nation/

184

- Perhaps there's interest in a more flexible process to build packages
- Or maybe you enjoy the command line more than the GUI
- Enter munkipkg

185

- munkipkg is a free open-source tool written by Greg Neagle to build packages
- Note before we dive in, packages made by munkipkg are not specific to Munki—the can be used wherever you'd normally use packages
- munkipkg projects aren't in an opaque binary format
    - They're a hierarchy of payload files and a collection of text configuration files
- Offers package signing features



186

- Use munkipkg's "create" option to start a new munkipkg project
- When you create a munkipkg project, there's a **payload and a scripts directory, much like the tabs in Whitebox Packages
- **Let's take a look at the build-info.plist file

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/
DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.github.munki.pkg.MyLicenseFile2017.5.13</string>
    <key>install_location</key>
    <string>/</string>
    <key>name</key>
    <string>MyLicenseFile2017.5.13-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action</key>
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
```

187

- A typical Apple XML plist, with some familiar keys from the earlier look at Whitebox Packages
- **There's the identifier, we can change that to match what we used in WhiteBox Packages, including removing the version information

---

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/
DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.organization.MyLicenseFile</string>
    <key>install_location</key>
    <string>/</string>
    <key>name</key>
    <string>MyLicenseFile2017.5.13-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action</key>
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
```

188

- And a couple lines below that, the name**
    - There's no reason to include the version number twice in the package name
    - Let's make the name match what we had before

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/
DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.organization.MyLicenseFile</string>
    <key>install_location</key>
    <string>/</string>
    <key>name</key>
    <string>My License File-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action</key>
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
```

189

- And let's look toward the end of the file**
- Let's change the version to match our desired version

```
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.organization.MyLicenseFile</string>
    <key>install_location</key>
    <string>/</string>
    <key>name</key>
    <string>My License File-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action</key>
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>2017.5.3</string>
</dict>
</plist>
```

190

- That's looking better
- Let's flag to munkipkg that we want this built as a Product Archive (Distribution Package) so we can properly sign and distribute it
    - **To do that, change the distribution_style key to true

---

```
    <key>name</key>
    <string>My License File-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action</key>
    <string>none</string>
    <key>signing_info</key>
    <dict>
        <key>identity</key>
        <string>Developer ID Installer: Tommy Tutone (8675309JJJ)</string>
        <key>timestamp</key>
        <true/>
    </dict>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>2017.5.3</string>
</dict>
</plist>
```

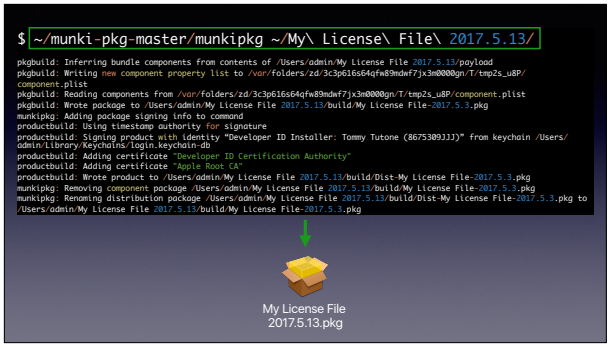
191

- And to properly sign we need to add a section of signing information**
- Note the timestamp option in the signing information
    - This causes the package to include the trusted time stamp
- OK, the build-info.plist looks good
- Next, let's add the "My License File.plist" to the project

```
$ mkdir -p ~/My\ License\ File\ 2017.5.13/payload/Library/
Application\ Support/My\ Application

$ cp Licenses/My\ License\ File.plist ~/My\ License\ File\
2017.5.13/payload/Library/Application\ Support/My\ Application

$ ls -l ~/My\ License\ File\ 2017.5.13/payload/Library/Application\
Support/My\ Application
total 0
-rw-r--r--  1 admin  staff  0 May 14 21:16 My License File.plist
```

192

- We need to create the "Library", "Application Support", and "My Application" directories under the "payload" directory
- **Then we copy the license in
- **And confirm it's there



193

- And now we can build the package
- Note munkipkg is calling pkgbuild, productbuild, and productsign, it's a wrapper around those tools

---

194

- So the big question
  - Why munkipkg when we have Whitebox Packages or Composer?
- Command line build process
  - One that can be started and controlled by other processes
- The source files are not an opaque binary, but instead can be more easily integrated into a version control system
  - Build options are not command-line flags, but instead an audit-able configuration file

---

195

- If you do plan to use munkipkg with git, make sure to read "Important git notes" on the munkipkg home page
- git has some unique interactions with various Mac-specific data

196

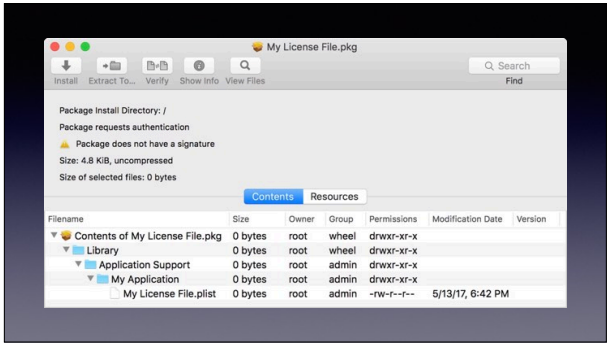
- A quick hit for two of my favorite tools
- If we're talking about packaging we need to mention two tools to help tear apart packages
  - Note there are command-line equivalents
- These tools do the same basic things, may have some slight twists

---

197

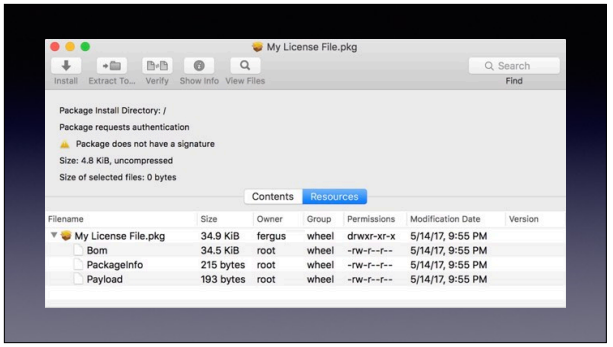
- First is Pacifist, a shareware tool written by Charles SIR-ska
- It has been around a long time, at least since 2002
- Can vouch that if I've had difficulties, I've sent Charles a package and received an update to address it

---



198

- Pacifist can visually show the payload of the package
- Can extract files out of the payload as well
  - An alternative to installing a package, especially if it is sizable
- Can also Quick Look a file

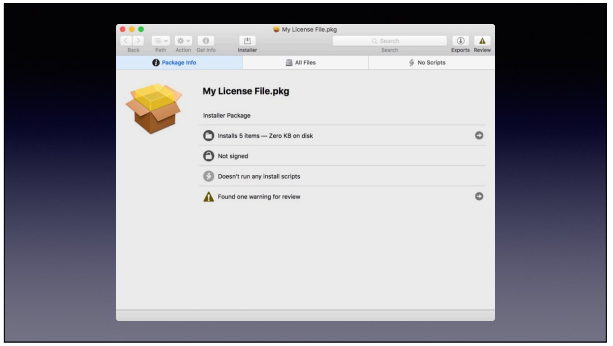


199

- The Resources tab can show various metadata files inside the package
- Most importantly, it can show the “Scripts” directory
- Quick Look-ing a script is really, really useful

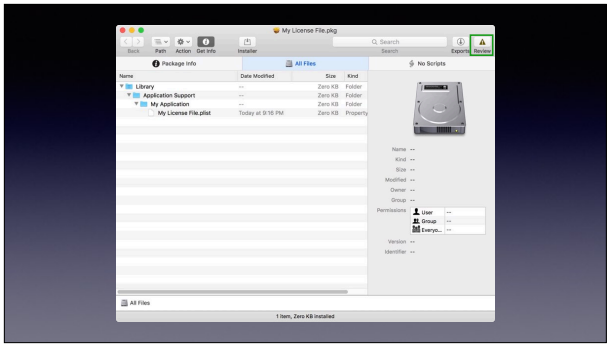
200

- The second tool is Suspicious Package, free software from Randy Saldinger at Mothers Ruin Software
- A bit newer and revised frequently



201

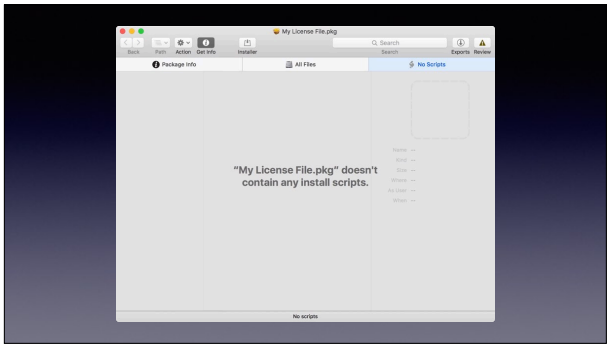
- A lot of the same information as Pacifist
- Suspicious Package has an overview of the package itself



202

- And a view of files to install
- Also note the **“Review” button in the upper right, for items that Suspicious Package’s developer considers concerning

---



203

- And there’s a view of the scripts, although this package doesn’t have any
- Good to keep both tools around, one tool may be able to handle a package the other cannot
- Note both of these have the ability to search a package payload, if you’re looking for a particular file
- Also good as a second and third opinion when auditing self-made packages

---

Pacifist
@ support@charlessoft.com

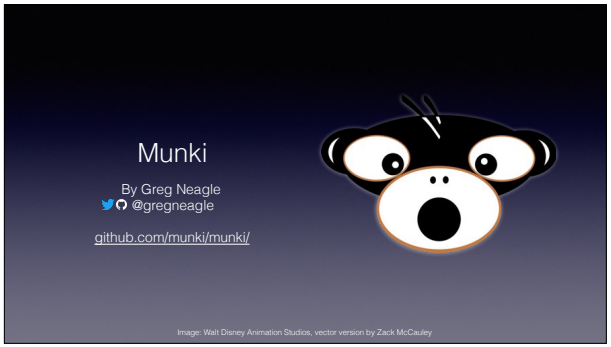
Suspicious Package
@ support@mothersruin.com

204

- Support for each of these titles is via email

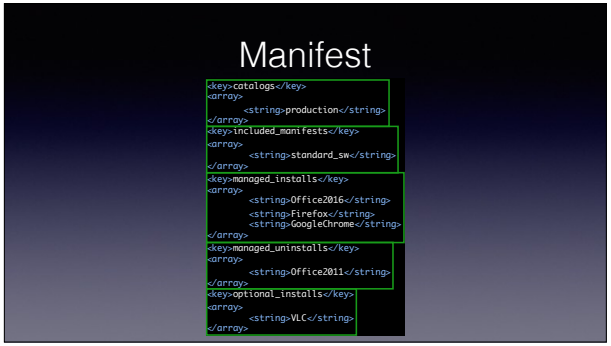
205

- Now, managing updates with the server-ish side of Munki



206

- Munki is free, open-source software still written primarily by Greg Neagle
- And you've seen this slide before, so to spice it up, I give you my youngest son's impression of the Munki logo**



207

- Munki clients retrieve software per a manifest
- Peeking inside a Munki manifest
    - **Managed installs are software titles you want to keep installed on the computer
    - **Managed uninstalls are software titles you want to keep uninstalled from the computer
    - **Optional installs are software titles the user can install on their computer if they so choose
    - **Included manifests allow one manifest to be nested in another
    - **But there's this peculiar "catalogs"
- Updating software with Munki is built on the development, testing, production workflow
    - This workflow is based on catalogs
    - Think of the manifest as a shopping list
        - Catalogs are stores
    - Let's go shopping for wine like a Munki client

208

- The primary catalog is typically called the “production” catalog
- No sharp edges, everything here works and is approved for distribution
- Production should be safe for anyone, kinda like your nearby SuperTarget
- **If you were shopping for wine in the production manifest you might find a nice KEY-yon-tee

---

209

- The testing catalog is for the more adventurous sorts
- Items in this store might not be ok, but they need to be checked before reaching production
- Shopping the testing catalog is closer to visiting a Trader Joe’s
- **As far as wine, folks shopping the testing catalog are more of a Two Buck Chuck crowd

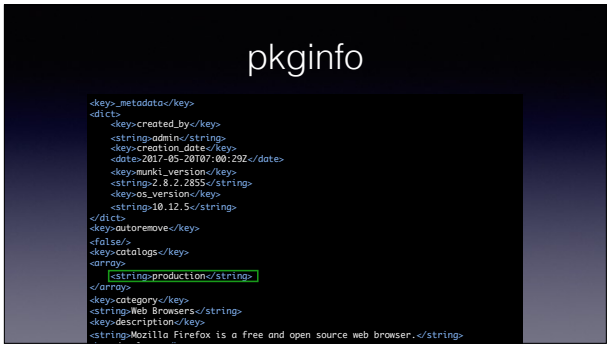
---

210

- The development catalog is only for the truly adventurous
- Folks shopping from the Eight-Twelve here might find things that could be hazardous to your health
- **The wine from the development catalog may be of questionable origin

211

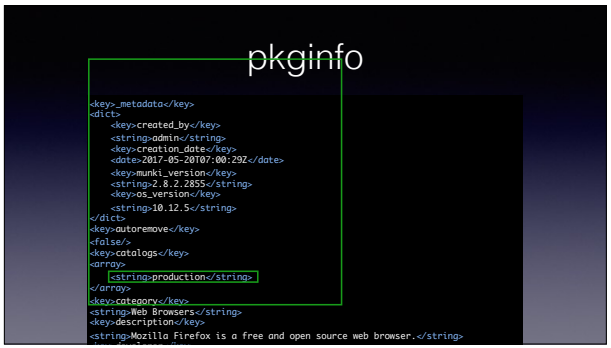
- Munki clients shop their manifest's catalog list in the order they're listed in the manifest
- If they find a matching product, they stop searching and evaluate for installation
- *Most* computers will have just a "production" catalog in their manifest
- **A handful of interested testers' computers will have the testing catalog listed before the production
- **And one or two computers (or willing victims) will have the development catalog
- Where do the catalogs come from?



212

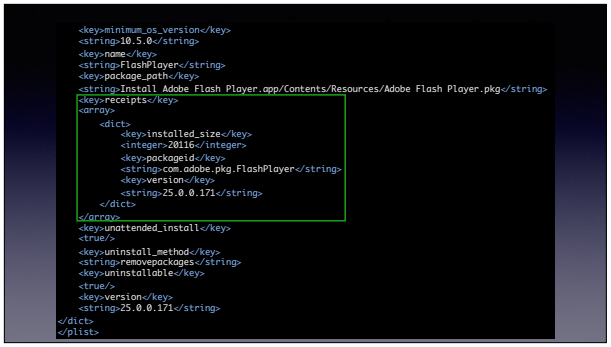
- Catalogs are generated from a second important structure in Munki, the pkginfo
- A pkginfo is a series of metadata about an installation package
  - There's information used by MSC (such as category, display name, description) as well as information about the actual package installation procedure (Restart required, applications that should block installation)
  - There's also the software catalogs the package should be part of, **this Firefox package is part of the production catalog
- Note the manifest we saw earlier doesn't list a software *version*
  - Because versioning is the duty of the catalog
- Software updates become less about remembering who has what, and more about an update's march to the production catalog

---



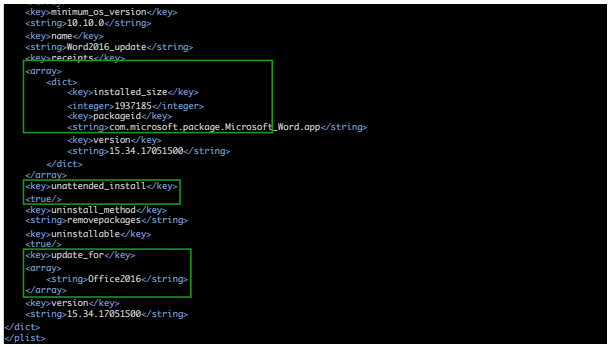
213

- Another important part of the update process is how Munki determines if installation is needed
  - One option, an Installs array, validates the version of an application or bundle at a path



214

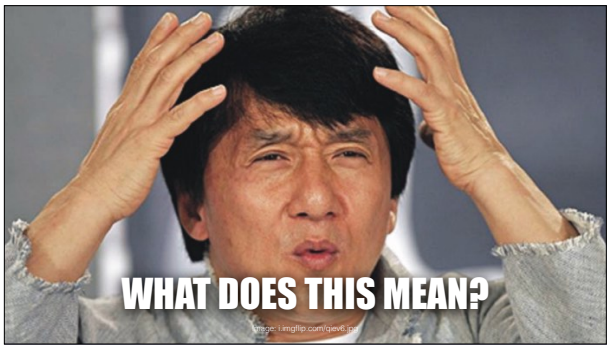
- Also, versioned package receipts can validate installation as in this receipt for Adobe Flash Player
    - Installs arrays and receipts, since they understand versions can tolerate updates that are applied from outside Munki
    - This is key—Munki is happy if the installed version is the *same* or *newer* than what's in the repo
    - This allows Munki to intelligently handle computers that might be upgraded outside of your control
    - Keep this tolerance in mind when evaluating receipts and creating installs arrays--how might the *next* software version "look" to Munki?



215

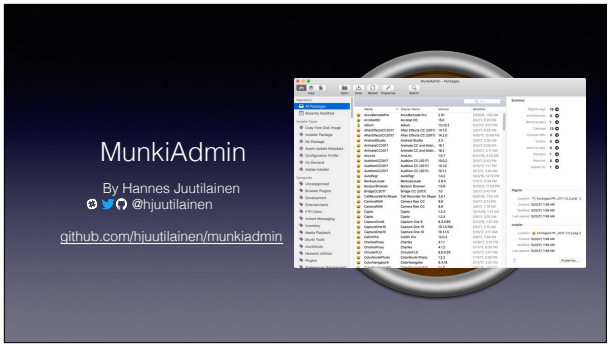
- Also important to updating is Munki's concept of an update_for
- This is used when something that's imported into Munki updates (or adds to) and existing installation
- An update_for is not added to a manifest, it's discovered by the Munki client at runtime
    - Word 2016 update is an update_for Office 2016
    - update_for also works great for things like software licenses, you're not modifying the original package
- **This installation is also marked as an unattended_install
    - If possible, it will install without bothering the user
    - But what happens if Word is open
- **Munki's blocking_applications allows the installation to intelligently be skipped, and the user potentially notified of the pending installation

---



216

- So what does this mean?
- Software updates can be
    - Checked out in development and approved
    - Rolled out to test groups and verified
    - Promoted to production
- Built-in version checking can determine if a software update needs to be installed
    - Can tolerate updates coming from other places besides Munki
- An update_for can tether updates to existing items in the Munki repo
- All by changing around six lines of text



217

- A couple things before I'm done, there is a GUI that covers many of Munki's options
- MunkiAdmin by HOHN-ez YOU-tee-lie-nen** aims to streamline Munki repo operations
- I'd still suggest learning the command line, and then once comfortable you can use MunkiAdmin if desired

---

218

- Another thing—we should cover what Munki is *not*
- Munki is a tremendous software installation mechanism, but there are things it is not

---

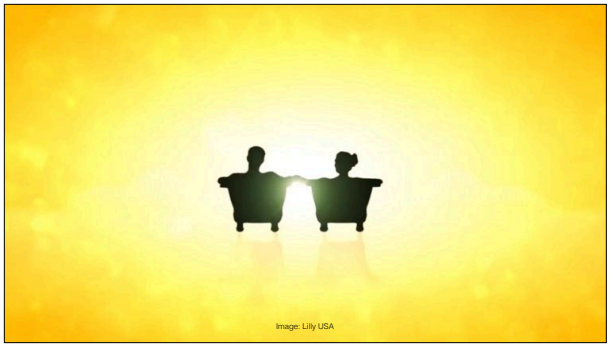
~~MDM~~

- VMware AirWatch
  air-watch.com
- Simple MDM
  simplemdm.com
- MicroMDM
  micromdm.io

219

- Munki is not an MDM
- If you want to deploy Munki with a MDM solution, there are some options
- There are links in the presenter notes for further reading about all of these
  - VMware AirWatch has worked with Erik Gomez to provide installation of the Munki client as part of their MDM suite
  - Simple MDM is another commercial MDM that also supports installing Munki as part of their MDM suite
  - MicroMDM is an open source MDM project that provides the option of installing Munki
    - Note Victor Vrantchan and Jesse Peterson, two MicroMDM developers, have a conference session later this week

- AirWatch links
  https://my.air-watch.com/help/9.1/en/Content/Release_Notes/Help_Release_Notes.htm
  http://blog.eriknicolasgomez.com/2017/04/27/Custom-DEP-Part-6-Vendor-Announcement-and-Presentation/
- Simple MDM
  https://simplemdm.com/2017/03/07/deploy-munki-apple-dep-mdm/
  https://simplemdm.com/2017/03/22/distribute-macos-product-archive-packages/
- MicroMDM
  https://micromdm.io



220

- Or talk to your MDM vendor about whether distributing Munki is right for them

~~Inventory System~~

221

- Munkireport-PHP
  github.com/munkireport/munkireport-php
- Sal
  github.com/salopensource/sal
  salsoftware.com

- Inventory
  - MunkiReport-PHP, by AR-e-n VON BOOK-hoven is an open source project provides a wide variety of reporting information from Munki clients
  - Sal is provided in two flavors, an open source version by Graham Gilbert and a paid version from Sal Software

☆ MunkiReport-PHP
https://github.com/munkireport/munkireport-php
☆ Sal Open Source
https://github.com/salopensource/sal
☆ SalSoftware
http://salsoftware.com

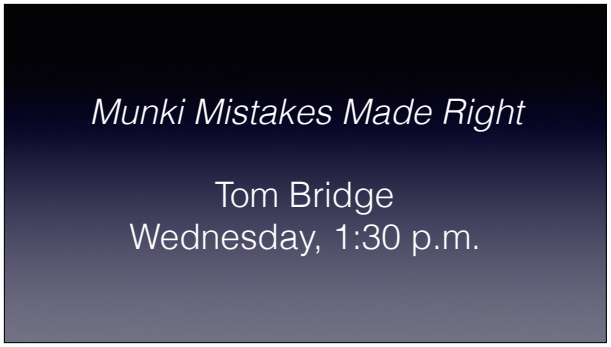
Jamf?

222

- jamJAR
  github.com/dataJAR/jamJAR

- Or possibly you're considering mixing Munki with Jamf Pro
- jamJAR by dataJAR from London provides an open source option for driving Munki with the Jamf Pro suite

☆ https://macmule.com/projects/jamjar/



223

- To learn more about Munki, be sure to attend Tom Bridge's session "Munki Mistakes Made Right"

224

- For Munki support, you'd want to check out the Munki Wiki
- #munki on MacAdmins Slack
- And the Google Groups mailing lists of munki-discuss and munki-dev
- Note the Wiki also lists options for paid support, if that's an avenue you'd like to investigate

  ☆ Munki Paid Support Starting Points
  https://github.com/munki/munki/wiki/Professional-Support

225

- Or…Greg is at the conference this week so you could possibly ask him your Munki questions directly

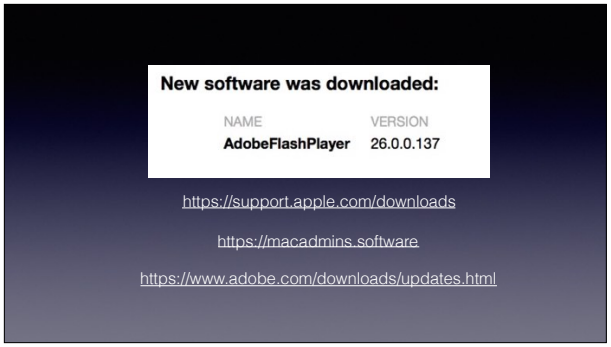
226

227

- What is patch management? It's the process of making sure that updates are applied in a timely basis.
- (updates, not necessarily upgrades, i.e. Microsoft Office 15.34->15.35)

228

- Why Patch?
- If it ain't broke, don't fix it, right? Well… if an update addresses security vulnerabilities, it should be scheduled, tested, and applied everywhere. How quickly depends on the seriousness of the vulnerability.



229

- How do you find out about updates?
- AutoPkg, Patch Management, Jamf Nation, etc. Some vendors schedule updates on a regular basis (Adobe and Microsoft have monthly/quarterly “Patch Tuesdays” for some products. Apple releases them as needed).

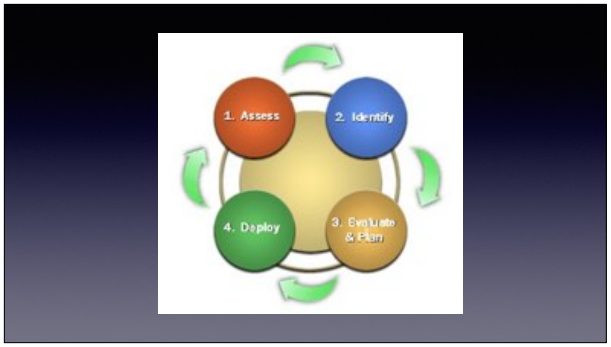
  https://support.apple.com/downloads, https://macadmins.software, https://www.adobe.com/downloads/updates.html
-

230

- How do you deploy patches?
- You must have a test plan. At some organizations this is a single department (i.e. Systems/IT). At others, a cross-functional group of “power users” from different departments. Different departments use different software, or use the same software differently.
- Do the patches require a restart (typically the case for OS patches)? Do they require the applications be terminated? How do you notify the user that there are updates waiting to be applied?
-

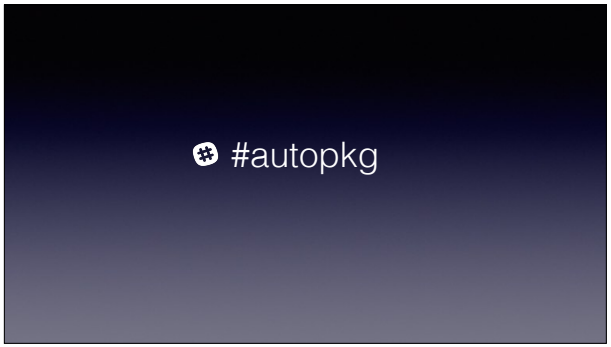
231

- Do the patches require a restart (typically the case for OS patches)? Do they require the applications be terminated? How do you notify the user that there are updates waiting to be applied?
-



232

- How long do you test? Some orgs want to wait at least a week in testing before widely deploying. Also, if your org has a Change Management process, patches must be submitted and approved (and have a back-out plan) before they can be widely deployed. No YOLO applies, unless it's your last day ;-)
- i.e. where I work there is one day a month (Thursday following the 2nd Tuesday) where I can actively reboot systems. If an update can be deployed without a reboot, as long as I can script to check if any "blocking" apps are running, and defer execution until they are not running. Or, run updates either at logout or login.

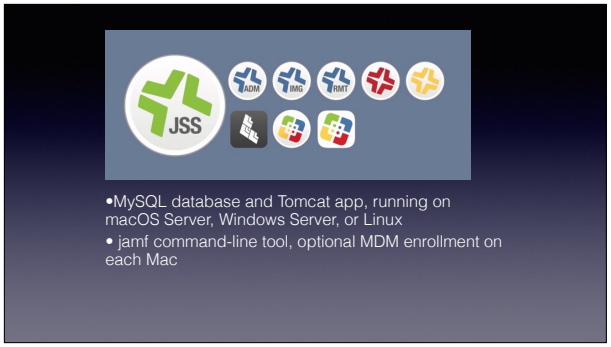


233

- Further discussion in the #autopkg channel on MacAdmins Slack

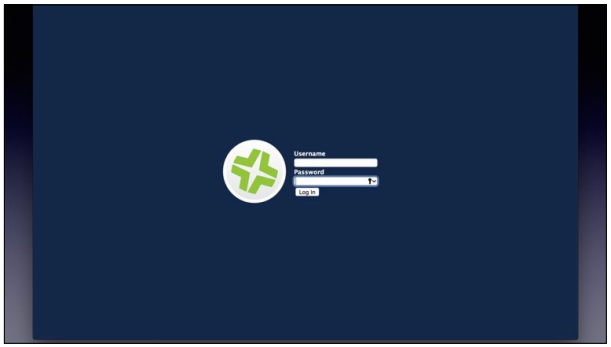
---

234

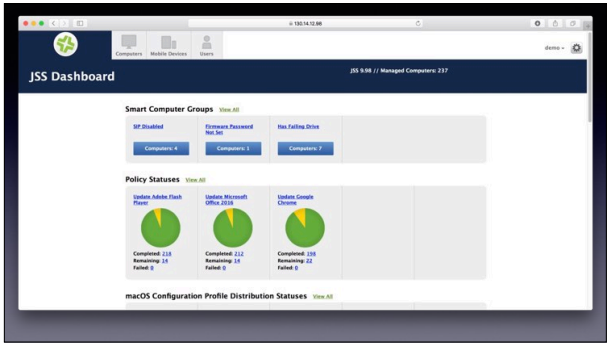


235

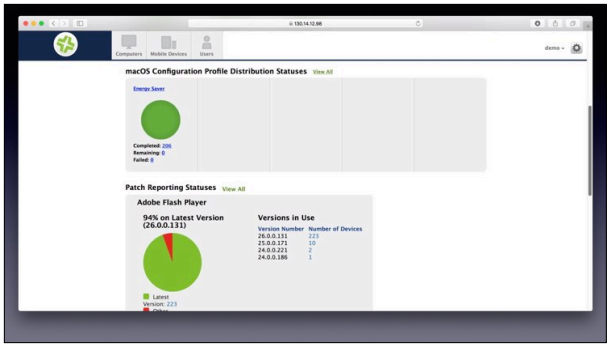
- Jamf Pro, formerly the Casper Suite, is a comprehensive suite of tools for managing (imaging, deploying, patching) macOS and iOS devices. It's the most widely used commercial package for managing Macs.
- At its heart, the Jamf Pro server runs a MySQL database and a Tomcat (Java) webapp on the server, and a jamf command-line binary agent on each client (as well as offering MDM enrollment). Because MySQL and Tomcat are available across all major platforms – Mac, Windows, Unix – you can run your Jamf Software Server (JSS) on whatever platform you are most comfortable with.



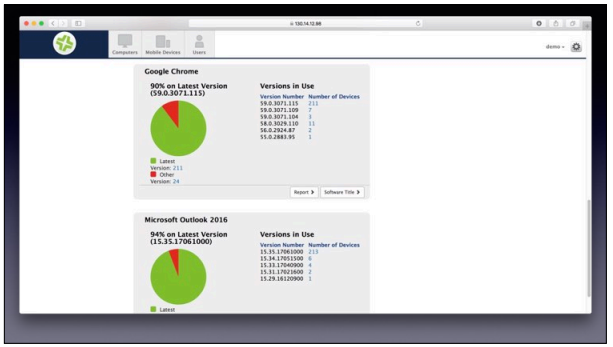
236



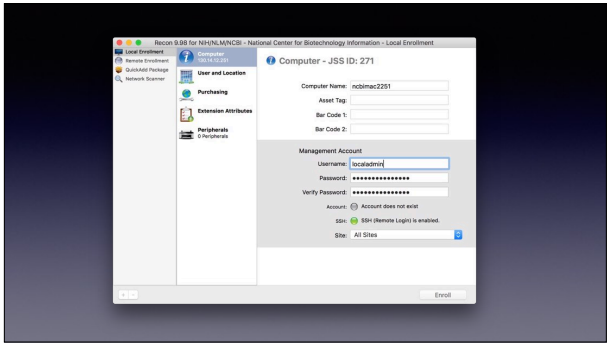
237


macOS Configuration Profile Distribution Statuses
Patch Reporting Statuses
Adobe Flash Player
94% on Latest Version (26.0.0.131)
Versions in Use

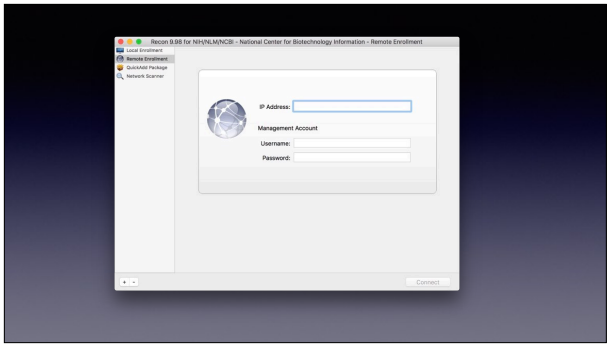
238


Google Chrome
90% on Latest Version (59.0.3071.115)
Versions in Use
Microsoft Outlook 2016
94% on Latest Version (15.35.17061000)
Versions in Use

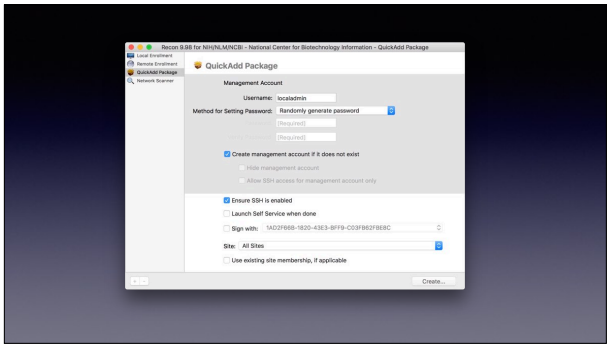
239


Computer - JSS ID: 271
Computer Name: ncbimac2251
Management Account
Username: localadmin
Enroll

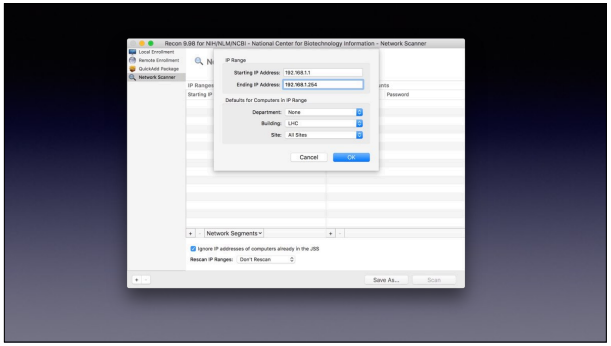
240


Recon 9.98 for NIH/NLM/NCBI - National Center for Biotechnology Information - Remote Enrollment
Local Enrollment
Remote Enrollment
QuickAdd Package
Network Scanner
IP Address:
Management Account
Username:
Password:
Connect

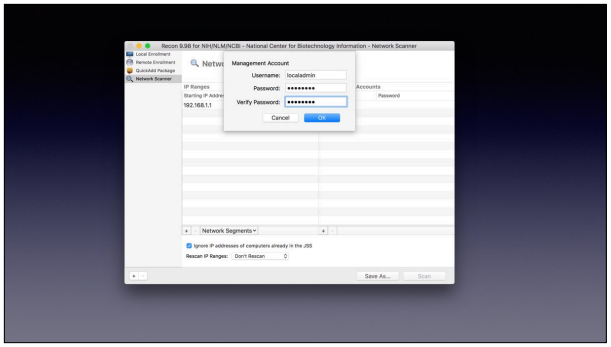
241


Recon 9.98 for NIH/NLM/NCBI - National Center for Biotechnology Information - QuickAdd Package
QuickAdd Package
Management Account
Username: localadmin
Method for Setting Password: Randomly generate password
Create management account if it does not exist
Hide management account
Allow SSH access for management account only
Ensure SSH is enabled
Launch Self Service when done
Sign with:
Site: All Sites
Use existing site membership, if applicable
Create...

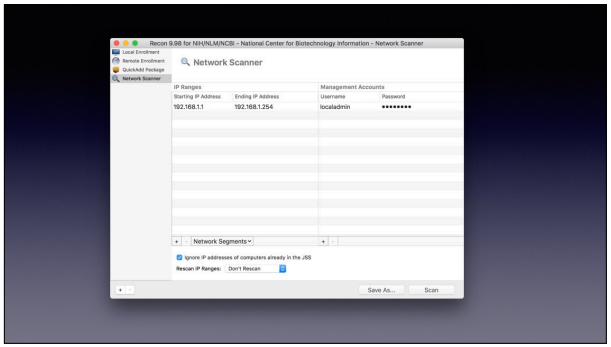
242


Recon 9.98 for NIH/NLM/NCBI - National Center for Biotechnology Information - Network Scanner
IP Range
Starting IP Address: 192.168.1.1
Ending IP Address: 192.168.1.254
Defaults for Computers in IP Range
Department: None
Building: LHC
Site: All Sites
Cancel
OK
Network Segments
Ignore IP addresses of computers already in the JSS
Rescan IP Ranges: Don't Rescan
Save As...
Scan

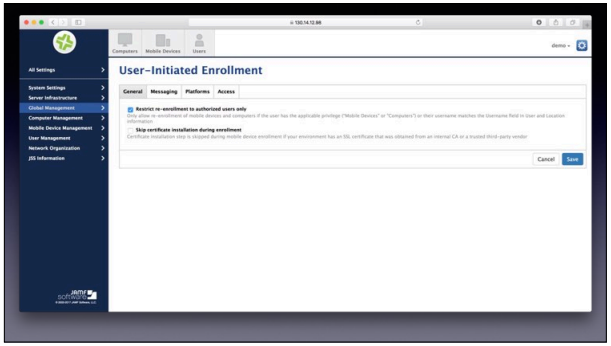
243


Recon 9.98 for NIH/NLM/NCBI - National Center for Biotechnology Information - Network Scanner
Management Account
Username: localadmin
Password:
Verify Password:
Cancel
OK
Network Segments
Ignore IP addresses of computers already in the JSS
Rescan IP Ranges: Don't Rescan
Save As...
Scan

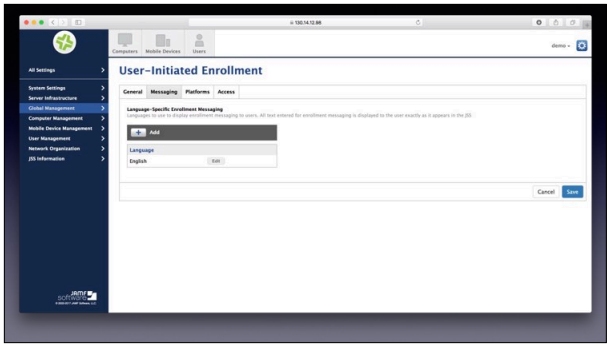
244


Recon 9.98 for NIH/NLM/NCBI - National Center for Biotechnology Information - Network Scanner
Network Scanner
IP Ranges
Starting IP Address
Ending IP Address
192.168.1.1
192.168.1.254
Management Accounts
Username
Password
localadmin
Network Segments
Ignore IP addresses of computers already in the JSS
Rescan IP Ranges: Don't Rescan
Save As...
Scan

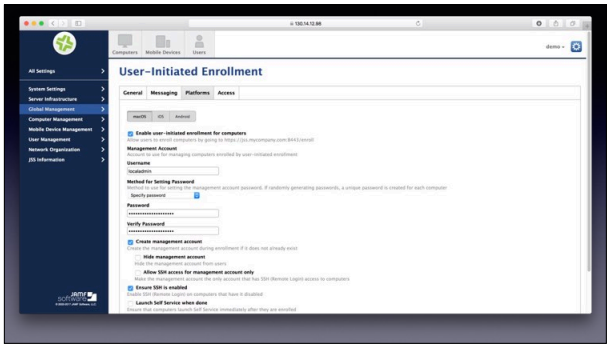
245


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
User-Initiated Enrollment
General
Messaging
Platforms
Access
Restrict re-enrollment to authorized users only
Skip certificate installation during enrollment
Cancel
Save

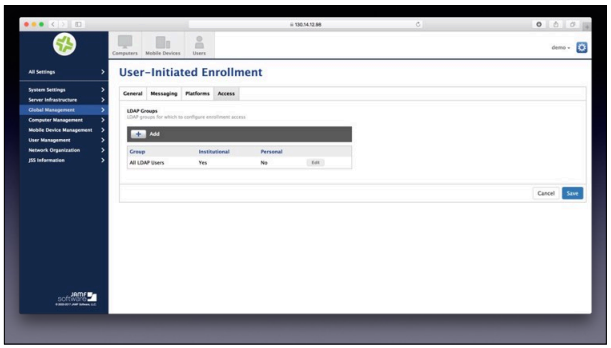
246


User-Initiated Enrollment
General
Messaging
Platforms
Access
Add
Language
English
Edit
Cancel
Save

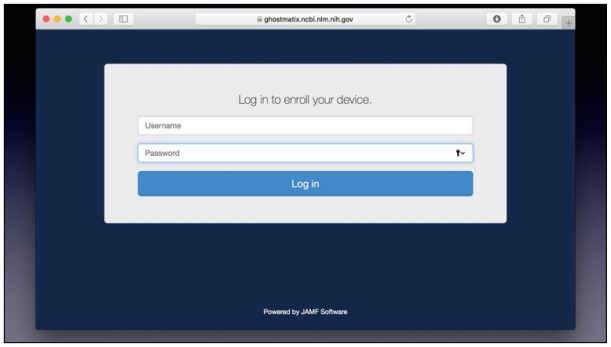
247


User-Initiated Enrollment
General
Messaging
Platforms
Access
Enable user-initiated enrollment for computers
Management Account
Username
Method for Setting Password
Password
Verify Password
Create management account
Hide management account
Allow SSH access for management account only
Ensure SSH is enabled
Launch Self Service when done

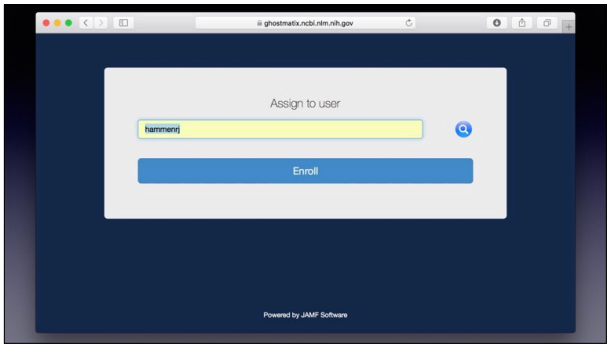
248


User-Initiated Enrollment
General
Messaging
Platforms
Access
LDAP Groups
Add
Group
Institutional
Personal
All LDAP Users
Yes
No
Edit
Cancel
Save

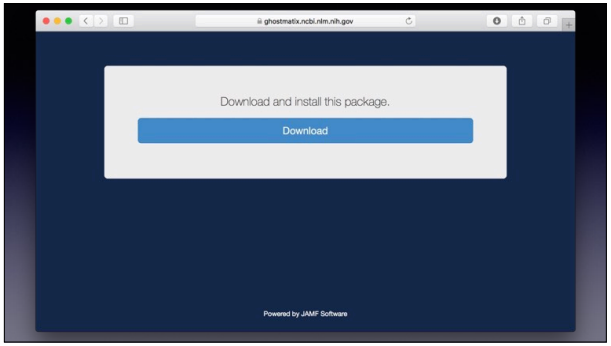
249


ghostmatix.ncbi.nlm.nih.gov
Log in to enroll your device.
Username
Password
Log in
Powered by JAMF Software

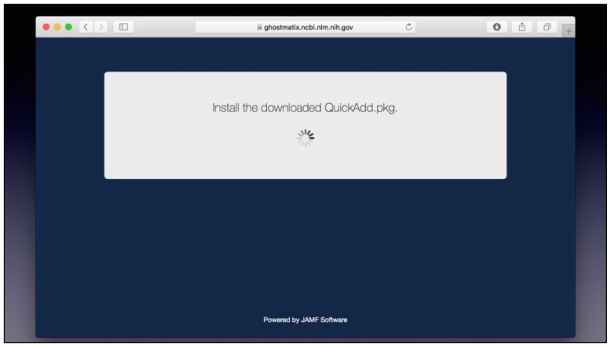
250


ghostmatix.ncbi.nlm.nih.gov
Assign to user
hammenrj
Enroll
Powered by JAMF Software

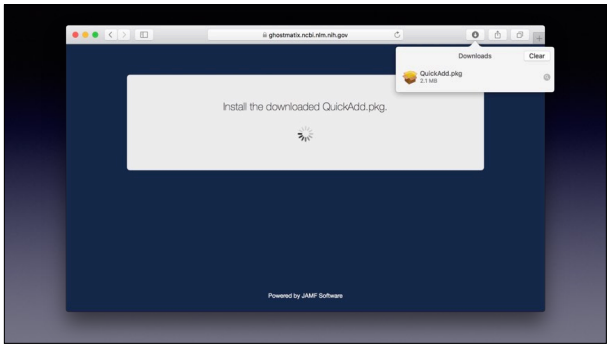
251


ghostmatix.ncbi.nlm.nih.gov
Download and install this package.
Download
Powered by JAMF Software

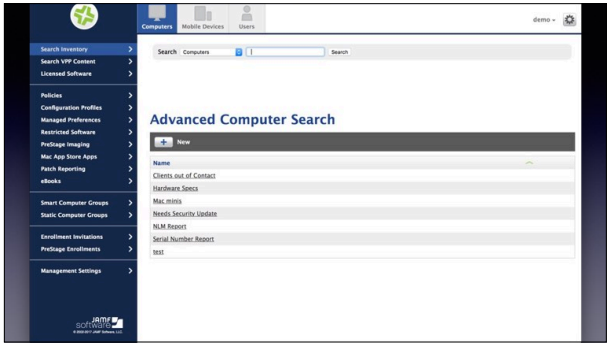
252


ghostmatix.ncbi.nlm.nih.gov
Install the downloaded QuickAdd.pkg.
Powered by JAMF Software

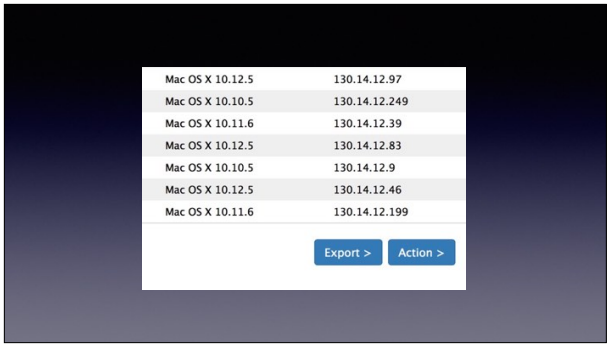
253


ghostmatix.ncbi.nlm.nih.gov
Downloads
Clear
QuickAdd.pkg
2.1 MB
Install the downloaded QuickAdd.pkg.
Powered by JAMF Software

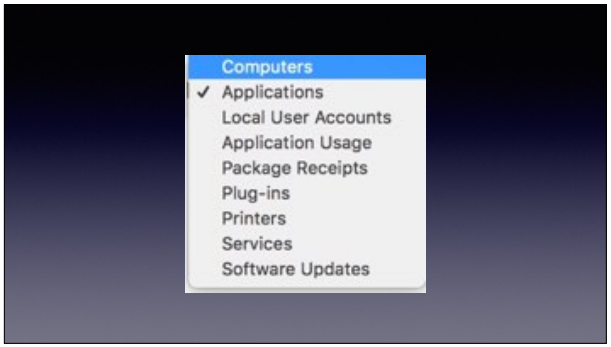
254


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Search
Computers
Search
Advanced Computer Search
New
Name
Clients out of Contact
Hardware Specs
Mac minis
Needs Security Update
NLM Report
Serial Number Report
test

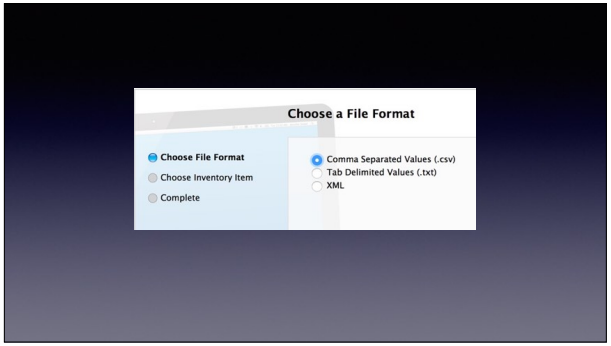
255


Mac OS X 10.12.5 130.14.12.97
Mac OS X 10.10.5 130.14.12.249
Mac OS X 10.11.6 130.14.12.39
Mac OS X 10.12.5 130.14.12.83
Mac OS X 10.10.5 130.14.12.9
Mac OS X 10.12.5 130.14.12.46
Mac OS X 10.11.6 130.14.12.199
Export >
Action >

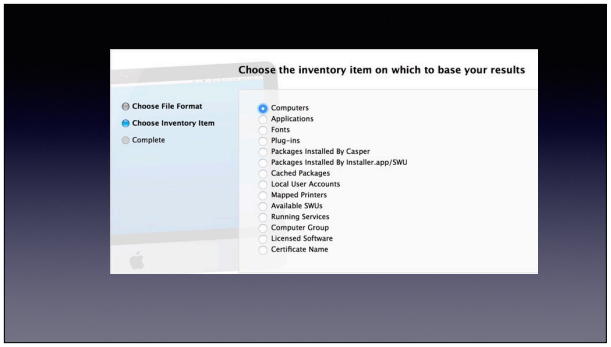
256


Computers
Applications
Local User Accounts
Application Usage
Package Receipts
Plug-ins
Printers
Services
Software Updates

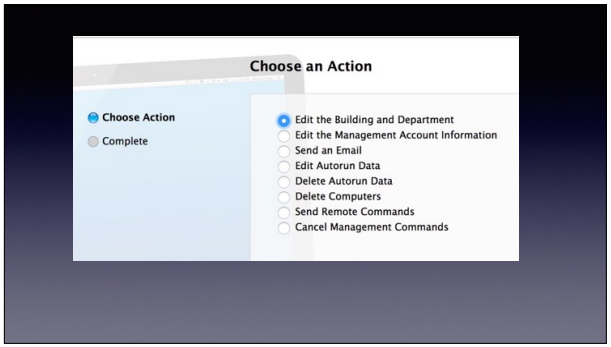
257


Choose a File Format
Choose File Format
Choose Inventory Item
Complete
Comma Separated Values (.csv)
Tab Delimited Values (.txt)
XML

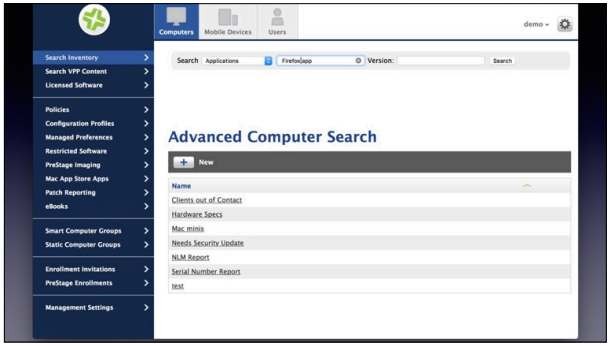
258


Choose the inventory item on which to base your results
Choose File Format
Choose Inventory Item
Complete
Computers
Applications
Fonts
Plug-ins
Packages Installed By Casper
Packages Installed By Installer.app/SWU
Cached Packages
Local User Accounts
Mapped Printers
Available SWUs
Running Services
Computer Group
Licensed Software
Certificate Name

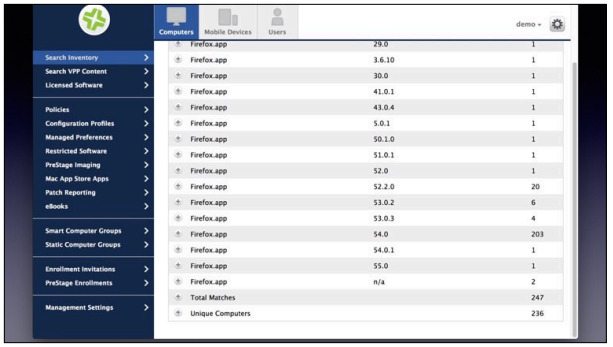
259


Choose an Action
Choose Action
Complete
Edit the Building and Department
Edit the Management Account Information
Send an Email
Edit Autorun Data
Delete Autorun Data
Delete Computers
Send Remote Commands
Cancel Management Commands

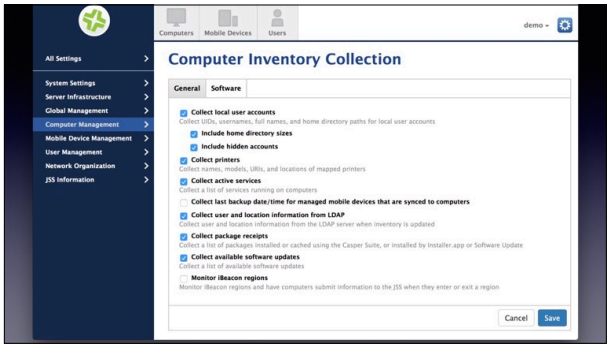
260


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Search
Applications
Firefox.app
Version:
Search
Advanced Computer Search
New
Name
Clients out of Contact
Hardware Specs
Mac minis
Needs Security Update
NLM Report
Serial Number Report
test

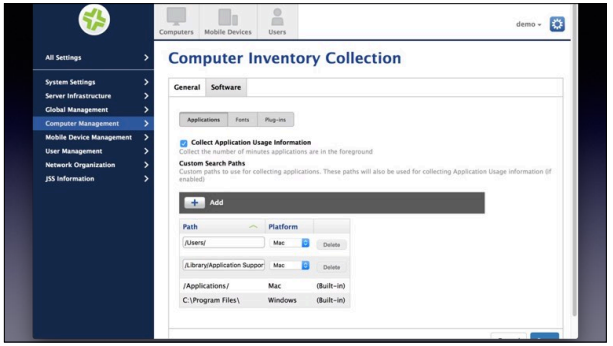
261


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Firefox.app 29.0 1
Firefox.app 3.6.10 1
Firefox.app 30.0 1
Firefox.app 41.0.1 1
Firefox.app 43.0.4 1
Firefox.app 5.0.1 1
Firefox.app 50.1.0 1
Firefox.app 51.0.1 1
Firefox.app 52.0 1
Firefox.app 52.2.0 20
Firefox.app 53.0.2 6
Firefox.app 53.0.3 4
Firefox.app 54.0 203
Firefox.app 54.0.1 1
Firefox.app 55.0 1
Firefox.app n/a 2
Total Matches 247
Unique Computers 236

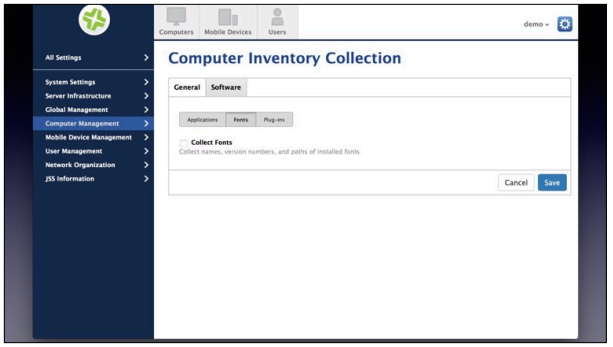
262


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer Inventory Collection
General
Software
Collect local user accounts
Collect UIDs, usernames, full names, and home directory paths for local user accounts
Include home directory sizes
Include hidden accounts
Collect printers
Collect names, models, URIs, and locations of mapped printers
Collect active services
Collect a list of services running on computers
Collect last backup date/time for managed mobile devices that are synced to computers
Collect user and location information from LDAP
Collect user and location information from the LDAP server when inventory is updated
Collect package receipts
Collect a list of packages installed or cached using the Casper Suite, or installed by Installer.app or Software Update
Collect available software updates
Collect a list of available software updates
Monitor iBeacon regions
Monitor iBeacon regions and have computers submit information to the JSS when they enter or exit a region
Cancel
Save

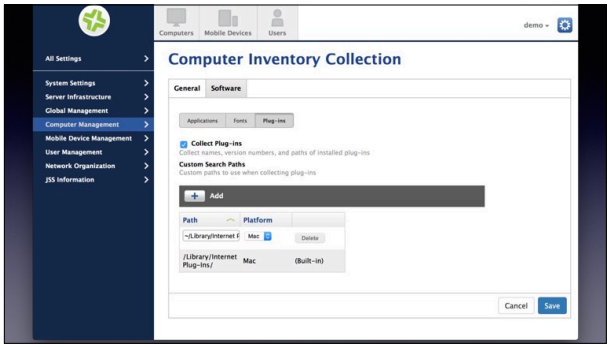
263


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer Inventory Collection
General
Software
Applications
Fonts
Plug-ins
Collect Application Usage Information
Collect the number of minutes applications are in the foreground
Custom Search Paths
Custom paths to use for collecting applications. These paths will also be used for collecting Application Usage information (if enabled)
Add
Path
Platform
/Users/
Mac
Delete
/Library/Application Suppor
Mac
Delete
/Applications/
Mac
(Built-in)
C:\Program Files\
Windows
(Built-in)

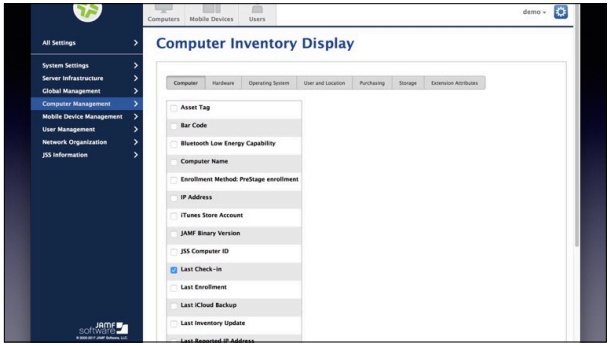
264


Computers
Mobile Devices
Users
demo
All Settings
Computer Inventory Collection
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
General
Software
Applications
Fonts
Plug-ins
Collect Fonts
Collect names, version numbers, and paths of installed fonts
Cancel
Save

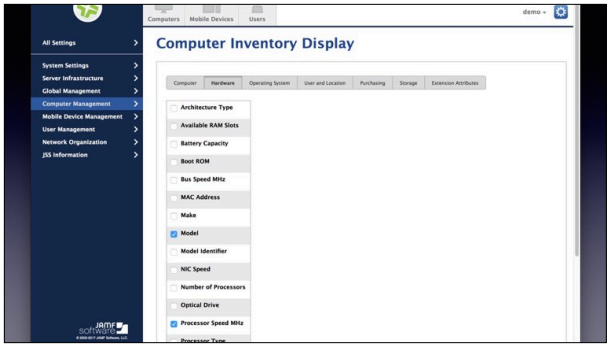
265


Computers
Mobile Devices
Users
demo
All Settings
Computer Inventory Collection
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
General
Software
Applications
Fonts
Plug-ins
Collect Plug-ins
Collect names, version numbers, and paths of installed plug-ins
Custom Search Paths
Custom paths to use when collecting plug-ins
Add
Path
Platform
~/Library/Internet F
Mac
Delete
/Library/Internet Plug-Ins/
Mac
(Built-in)
Cancel
Save

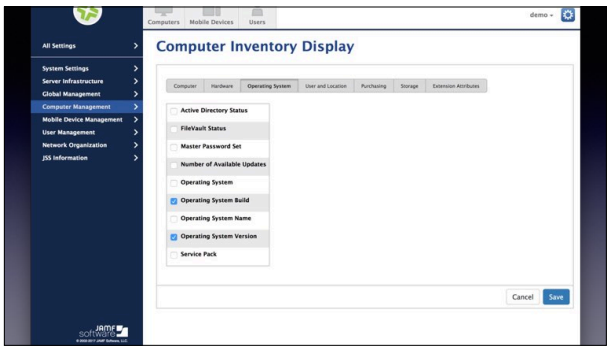
266


Computers
Mobile Devices
Users
demo
All Settings
Computer Inventory Display
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer
Hardware
Operating System
User and Location
Purchasing
Storage
Extension Attributes
Asset Tag
Bar Code
Bluetooth Low Energy Capability
Computer Name
Enrollment Method: PreStage enrollment
IP Address
iTunes Store Account
JAMF Binary Version
JSS Computer ID
Last Check-in
Last Enrollment
Last iCloud Backup
Last Inventory Update
Last Reported IP Address

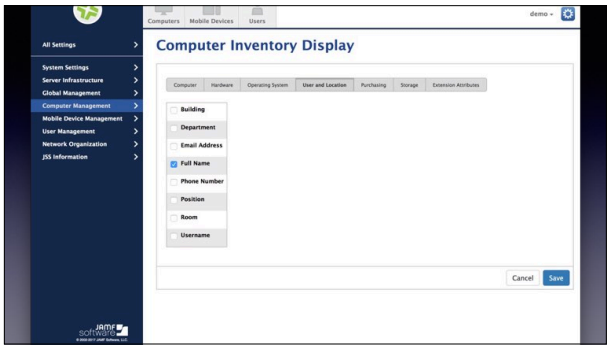
267


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer Inventory Display
Computer
Hardware
Operating System
User and Location
Purchasing
Storage
Extension Attributes
Architecture Type
Available RAM Slots
Battery Capacity
Boot ROM
Bus Speed MHz
MAC Address
Make
Model
Model Identifier
NIC Speed
Number of Processors
Optical Drive
Processor Speed MHz

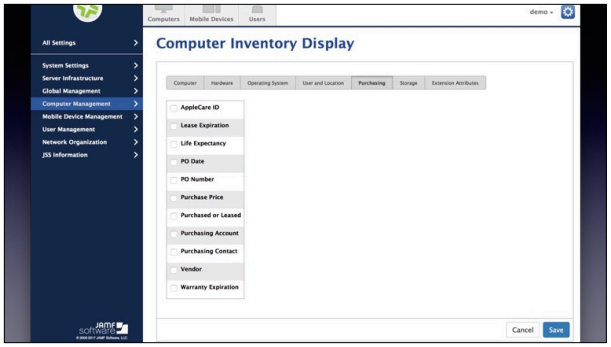
268


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer Inventory Display
Computer
Hardware
Operating System
User and Location
Purchasing
Storage
Extension Attributes
Active Directory Status
FileVault Status
Master Password Set
Number of Available Updates
Operating System
Operating System Build
Operating System Name
Operating System Version
Service Pack
Cancel
Save

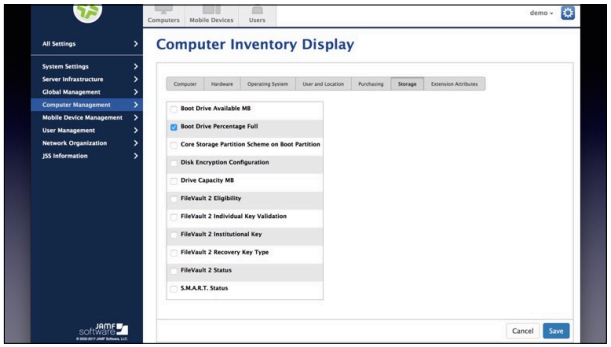
269


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer Inventory Display
Computer
Hardware
Operating System
User and Location
Purchasing
Storage
Extension Attributes
Building
Department
Email Address
Full Name
Phone Number
Position
Room
Username
Cancel
Save

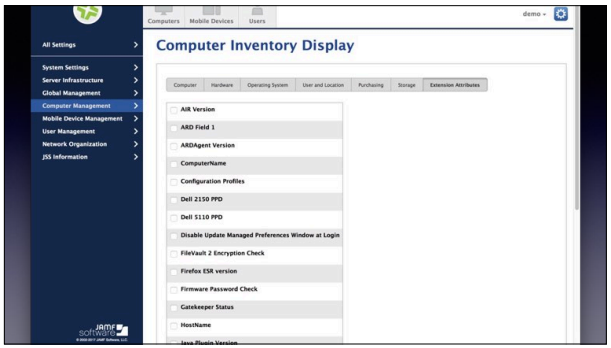
270


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer Inventory Display
Computer
Hardware
Operating System
User and Location
Purchasing
Storage
Extension Attributes
AppleCare ID
Lease Expiration
Life Expectancy
PO Date
PO Number
Purchase Price
Purchased or Leased
Purchasing Account
Purchasing Contact
Vendor
Warranty Expiration
Cancel
Save

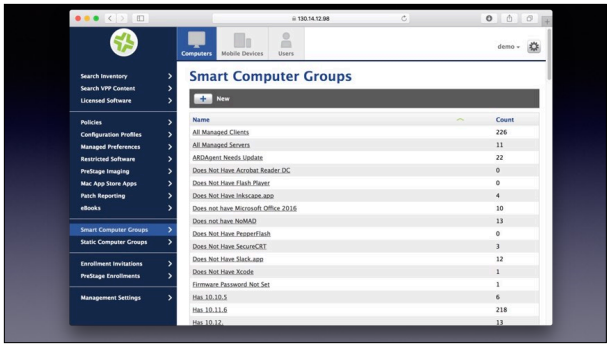
271


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer Inventory Display
Computer
Hardware
Operating System
User and Location
Purchasing
Storage
Extension Attributes
Boot Drive Available MB
Boot Drive Percentage Full
Core Storage Partition Scheme on Boot Partition
Disk Encryption Configuration
Drive Capacity MB
FileVault 2 Eligibility
FileVault 2 Individual Key Validation
FileVault 2 Institutional Key
FileVault 2 Recovery Key Type
FileVault 2 Status
S.M.A.R.T. Status
Cancel
Save

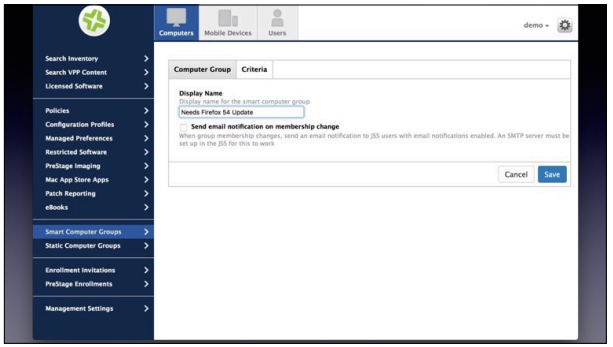
272


Computers
Mobile Devices
Users
demo
All Settings
System Settings
Server Infrastructure
Global Management
Computer Management
Mobile Device Management
User Management
Network Organization
JSS Information
Computer Inventory Display
Computer
Hardware
Operating System
User and Location
Purchasing
Storage
Extension Attributes
AIR Version
ARD Field 1
ARDAgent Version
ComputerName
Configuration Profiles
Dell 2150 PPD
Dell 5110 PPD
Disable Update Managed Preferences Window at Login
FileVault 2 Encryption Check
Firefox ESR version
Firmware Password Check
Gatekeeper Status
HostName

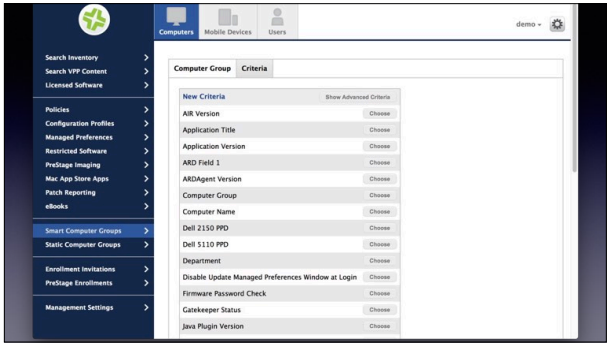
273


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Smart Computer Groups
New
Name
Count
All Managed Clients 226
All Managed Servers 11
ARDAgent Needs Update 22
Does Not Have Acrobat Reader DC 0
Does Not Have Flash Player 0
Does Not Have Inkscape.app 4
Does not have Microsoft Office 2016 10
Does not have NoMAD 13
Does Not Have PepperFlash 0
Does Not Have SecureCRT 3
Does Not Have Slack.app 12
Does Not Have Xcode 1
Firmware Password Not Set 1
Has 10.10.5 6
Has 10.11.6 218

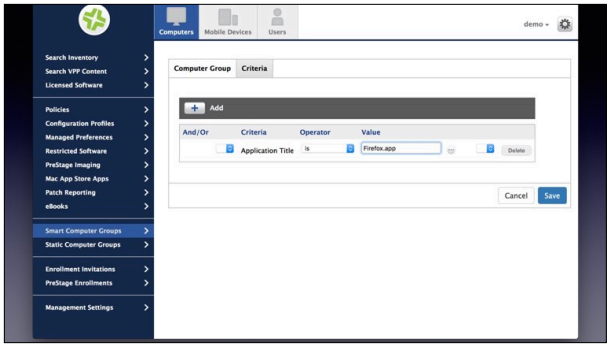
274


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Computer Group
Criteria
Display Name
Display name for the smart computer group
Needs Firefox 54 Update
Send email notification on membership change
When group membership changes, send an email notification to JSS users with email notifications enabled. An SMTP server must be set up in the JSS for this to work.
Cancel
Save

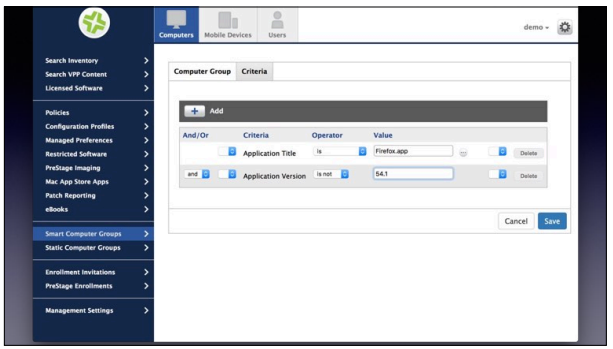
275


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Computer Group
Criteria
New Criteria
Show Advanced Criteria
AIR Version
Application Title
Application Version
ARD Field 1
ARDAgent Version
Computer Group
Computer Name
Dell 2150 PPD
Dell 5110 PPD
Department
Disable Update Managed Preferences Window at Login
Firmware Password Check
Gatekeeper Status
Java Plugin Version
Choose

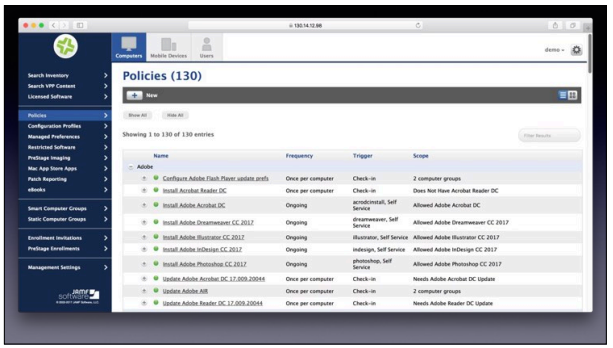
276


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Computer Group
Criteria
Add
And/Or
Criteria
Operator
Value
Application Title
is
Firefox.app
Delete
Cancel
Save

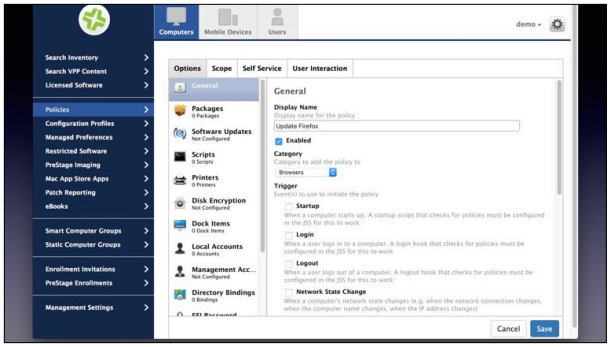
277


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Computer Group
Criteria
Add
And/Or
Criteria
Operator
Value
Application Title
is
Firefox.app
Delete
and
Application Version
is not
54.1
Delete
Cancel
Save

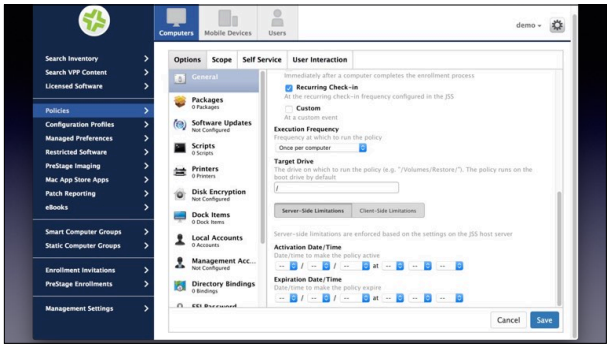
278


Computers
Mobile Devices
Users
demo
Policies (130)
New
Showing 1 to 130 of 130 entries
Name
Frequency
Trigger
Scope
Adobe
Configure Adobe Flash Player update prefs
Once per computer
Check-in
2 computer groups
Install Acrobat Reader DC
Once per computer
Check-in
Does Not Have Acrobat Reader DC
Install Adobe Acrobat DC
Ongoing
Check-in
Allowed Adobe Acrobat DC
Install Adobe Dreamweaver CC 2017
Ongoing
Check-in
Allowed Adobe Dreamweaver CC 2017
Install Adobe Illustrator CC 2017
Ongoing
Check-in
Allowed Adobe Illustrator CC 2017
Install Adobe InDesign CC 2017
Ongoing
Check-in
Allowed Adobe InDesign CC 2017
Install Adobe Photoshop CC 2017
Ongoing
Check-in
Allowed Adobe Photoshop CC 2017
Update Adobe Acrobat DC 17.009.20044
Once per computer
Check-in
Needs Adobe Acrobat DC Update
Update Adobe AIR
Once per computer
Check-in
2 computer groups
Update Adobe Reader DC 17.009.20044
Once per computer
Check-in
Needs Adobe Reader DC Update

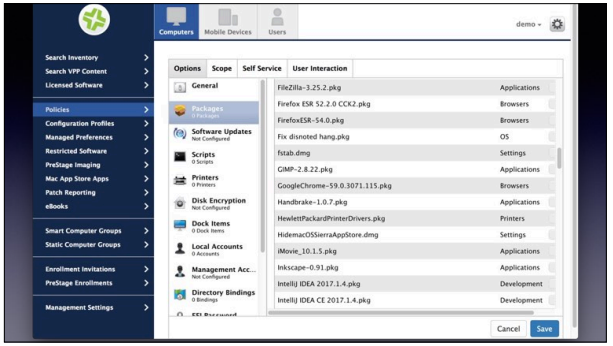
279


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Options
Scope
Self Service
User Interaction
General
Packages
Software Updates
Scripts
Printers
Disk Encryption
Dock Items
Local Accounts
Management Acc...
Directory Bindings
Display Name
Update Firefox
Enabled
Category
Browsers
Trigger
Startup
Login
Logout
Network State Change
Cancel
Save

280


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Options
Scope
Self Service
User Interaction
General
Packages
Software Updates
Scripts
Printers
Disk Encryption
Dock Items
Local Accounts
Management Acc...
Directory Bindings
Recurring Check-in
Custom
Execution Frequency
Once per computer
Target Drive
/
Server-Side Limitations
Client-Side Limitations
Activation Date/Time
Expiration Date/Time
Cancel
Save

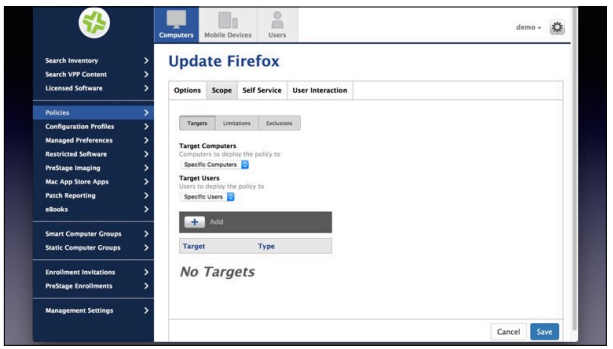
281


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Options
Scope
Self Service
User Interaction
General
Packages
Software Updates
Scripts
Printers
Disk Encryption
Dock Items
Local Accounts
Management Acc...
Directory Bindings
FileZilla-3.25.2.pkg
Applications
Firefox ESR 52.2.0 CCK2.pkg
Browsers
FirefoxESR-54.0.pkg
Browsers
Fix disnoted hang.pkg
OS
fstab.dmg
Settings
GIMP-2.8.22.pkg
Applications
GoogleChrome-59.0.3071.115.pkg
Browsers
Handbrake-1.0.7.pkg
Applications
HewlettPackardPrinterDrivers.pkg
Printers
HidemacOSSierraAppStore.dmg
Settings
iMovie_10.1.5.pkg
Applications
Inkscape-0.91.pkg
Applications
IntelliJ IDEA 2017.1.4.pkg
Development
IntelliJ IDEA CE 2017.1.4.pkg
Development
Cancel
Save

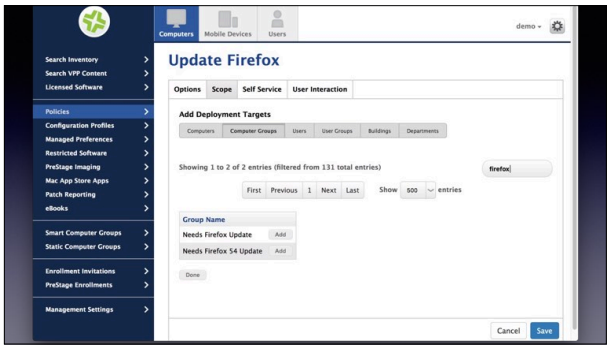
282


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Update Firefox
Options
Scope
Self Service
User Interaction
General
Packages
1 Package
Software Updates
Not Configured
Scripts
0 Scripts
Printers
0 Printers
Disk Encryption
Not Configured
Dock Items
0 Dock Items
Local Accounts
0 Accounts
Management Acc...
Not Configured
Directory Bindings
Maintenance
Update Inventory
Force computers to submit updated inventory information to the JSS
Reset Computer Names
Change the computer name on computers to match the computer name in the JSS
Install Cached Packages
Install packages cached by the Casper Suite
Fix Disk Permissions (Not compatible with macOS v10.12 or later)
Fix ByHost Files
Flush System Caches
Flush caches from /Library/Caches/ and /System/Library/Caches/, except for any com.apple.LaunchServices caches
Flush User Caches
Flush caches from ~/Library/Caches/, ~/.jpi_cache/, and ~/Library/Preferences/Microsoft/Office version #/Office Font Cache
Verify Startup Disk
Cancel
Save

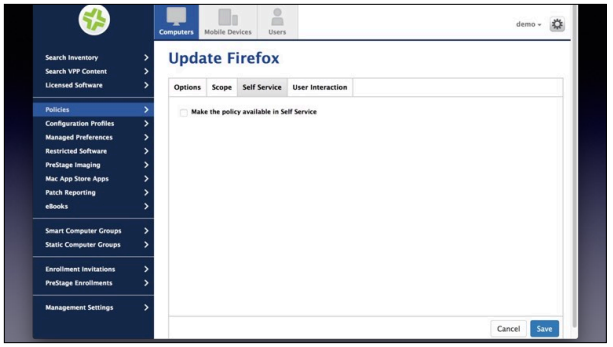
283


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Update Firefox
Options
Scope
Self Service
User Interaction
Targets
Limitations
Exclusions
Target Computers
Computers to deploy the policy to
Specific Computers
Target Users
Users to deploy the policy to
Specific Users
Add
Target
Type
No Targets
Cancel
Save

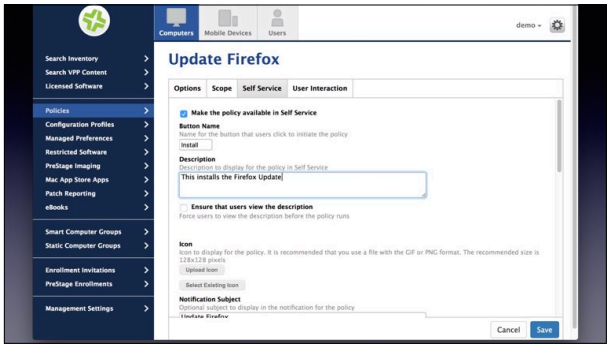
284


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Update Firefox
Options
Scope
Self Service
User Interaction
Add Deployment Targets
Computers
Computer Groups
Users
User Groups
Buildings
Departments
Showing 1 to 2 of 2 entries (filtered from 131 total entries)
firefox
First
Previous
1
Next
Last
Show
100
entries
Group Name
Needs Firefox Update
Add
Needs Firefox 54 Update
Add
Done
Cancel
Save

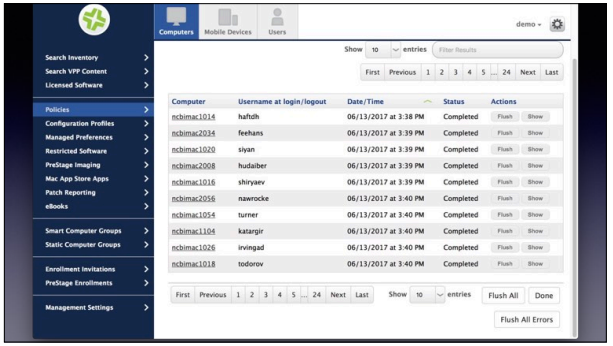
285


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Update Firefox
Options
Scope
Self Service
User Interaction
Make the policy available in Self Service
Cancel
Save

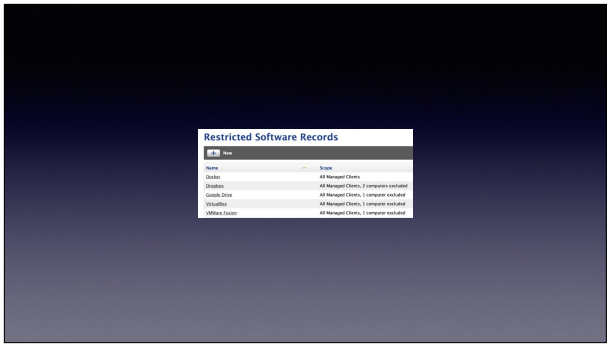
286


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Update Firefox
Options
Scope
Self Service
User Interaction
Make the policy available in Self Service
Button Name
Name for the button that users click to initiate the policy
Install
Description
Description to display for the policy in Self Service
This installs the Firefox Update
Ensure that users view the description
Force users to view the description before the policy runs
Icon
Icon to display for the policy. It is recommended that you use a file with the GIF or PNG format. The recommended size is 128x128 pixels
Upload Icon
Select Existing Icon
Notification Subject
Optional subject to display in the notification for the policy
Update Firefox
Cancel
Save

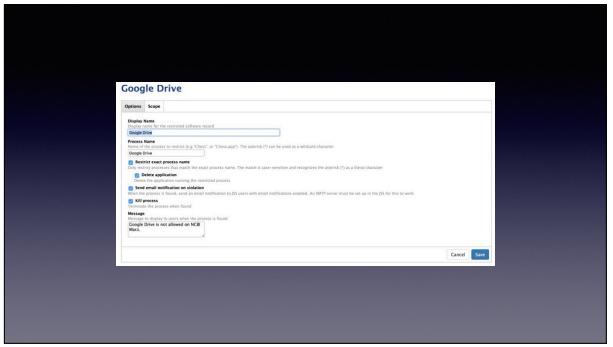
287


Computers
Mobile Devices
Users
demo
Search Inventory
Search VPP Content
Licensed Software
Policies
Configuration Profiles
Managed Preferences
Restricted Software
PreStage Imaging
Mac App Store Apps
Patch Reporting
eBooks
Smart Computer Groups
Static Computer Groups
Enrollment Invitations
PreStage Enrollments
Management Settings
Show 10 entries
Filter Results
First Previous 1 2 3 4 5 ... 24 Next Last
Computer
Username at login/logout
Date/Time
Status
Actions
ncbimac1014 haftdh 06/13/2017 at 3:38 PM Completed Flush Show
ncbimac2034 feehans 06/13/2017 at 3:39 PM Completed Flush Show
ncbimac1020 siyan 06/13/2017 at 3:39 PM Completed Flush Show
ncbimac2008 hudaiber 06/13/2017 at 3:39 PM Completed Flush Show
ncbimac1016 shiryaev 06/13/2017 at 3:39 PM Completed Flush Show
ncbimac2056 nawrocke 06/13/2017 at 3:40 PM Completed Flush Show
ncbimac1054 turner 06/13/2017 at 3:40 PM Completed Flush Show
ncbimac1104 katargir 06/13/2017 at 3:40 PM Completed Flush Show
ncbimac1026 irvingad 06/13/2017 at 3:40 PM Completed Flush Show
ncbimac1018 todorov 06/13/2017 at 3:40 PM Completed Flush Show
First Previous 1 2 3 4 5 ... 24 Next Last
Show 10 entries
Flush All
Done
Flush All Errors

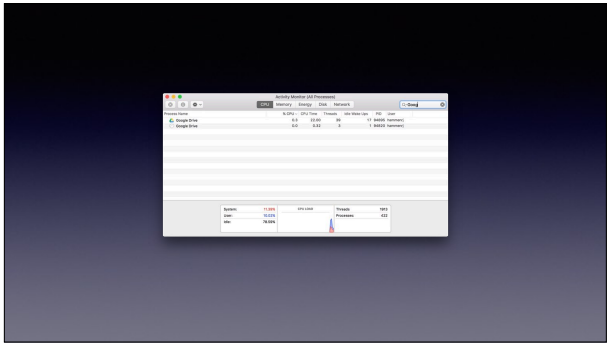
288


Restricted Software Records
New
Name
Scope
Docker
All Managed Clients
Dropbox
All Managed Clients, 2 computers excluded
Google Drive
All Managed Clients, 1 computer excluded
VirtualBox
All Managed Clients, 1 computer excluded
VMWare Fusion
All Managed Clients, 1 computer excluded

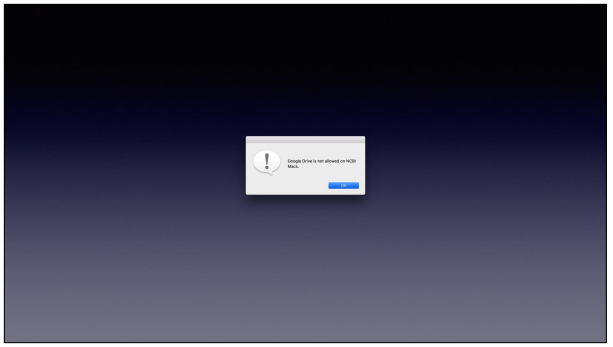
289


Google Drive
Options
Scope
Display Name
Google Drive
Process Name
Google Drive
Restrict exact process name
Delete application
Send email notification on violation
Kill process
Message
Google Drive is not allowed on NCB Mail.
Cancel
Save

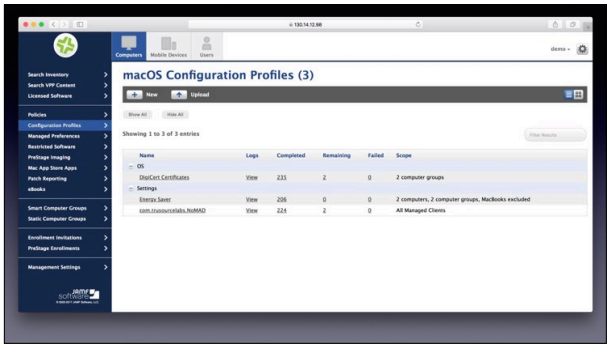
290


Activity Monitor (All Processes)
CPU
Memory
Energy
Disk
Network
Google Drive
Google Drive

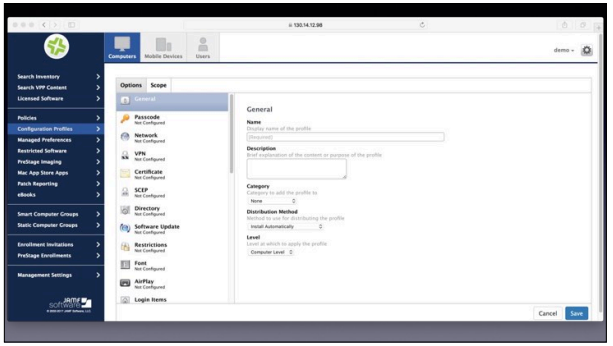
291



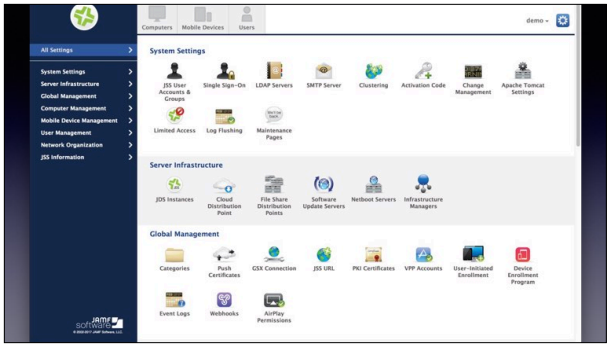
292


macOS Configuration Profiles (3)
New
Upload
Showing 1 to 3 of 3 entries
Name
Logs
Completed
Remaining
Failed
Scope
OS
DigiCert Certificates
2 computer groups
Settings
Energy Saver
2 computers, 2 computer groups, MacBooks excluded
All Managed Clients

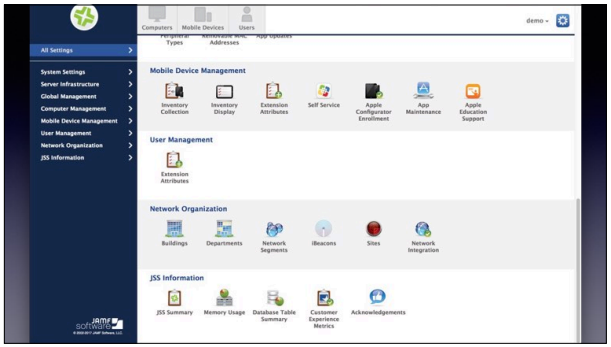
293


Options
Scope
General
Passcode
Network
VPN
Certificate
SCEP
Directory
Software Update
Restrictions
Font
AirPlay
Login Items
Name
Description
Category
Distribution Method
Level
Cancel
Save

294



295



296

#jamfnation
https://www.jamf.com/jamf-nation/

297

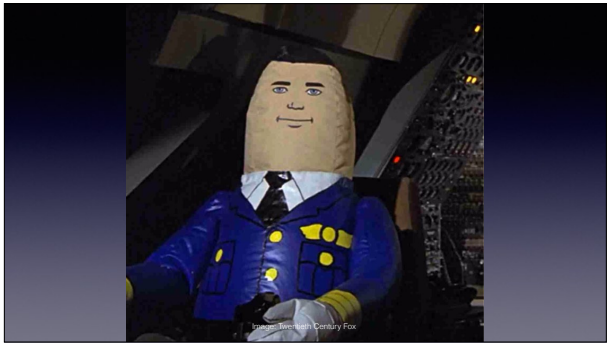
- Further discussion in the #jamfnation channel on MacAdmins Slack, and on https://www.jamf.com/jamf-nation/

298

299

- Finding updates
    - **Tedious keeping track of updates for software you do use frequently
    - Not to mention software that you use infrequently
    - Once discovered, you need to download and import the software into management system
        - Remember metadata, name, test groups
    - **That's a lot of work
    - What if there was a concierge to obtain updates and make them available

---

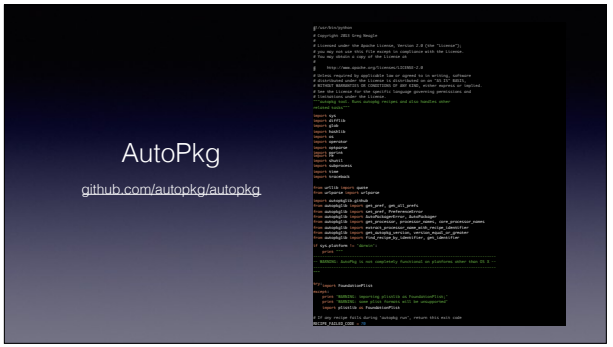


300

- Kinda like putting your update checking on automatic pilot
- It turns out there is

301

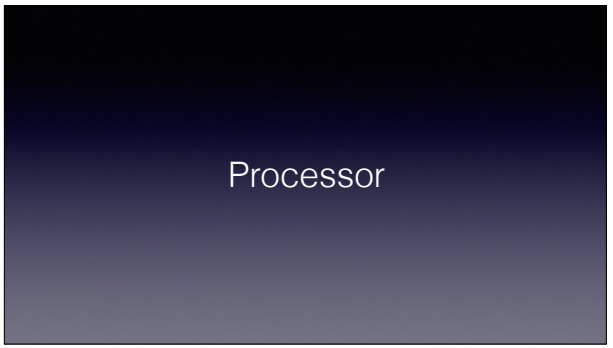
- AutoPkg
  - Software framework to handle retrieving, possibly transforming, and importing software updates
  - Community-extensible free open source software



302

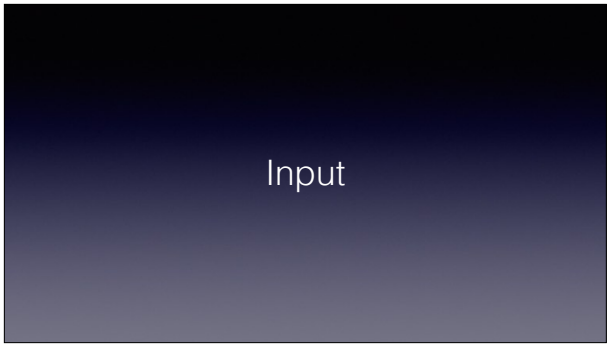
- AutoPkg is a framework, a set of tools
  - It doesn't do anything by itself, it needs a set of blueprints
- Before diving into AutoPkg, let's run through a glossary of terms

---



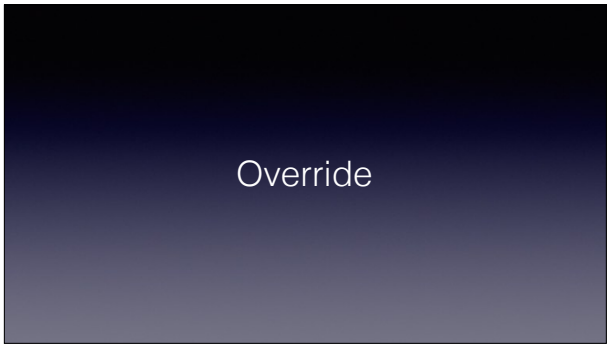
303

- Processor: A piece of code that performs a particular procedure for AutoPkg
  - Each of the processors in the AutoPkg organization has a Wiki page
  - Processors offer discovery, download, transformative, packaging, and verification features



304

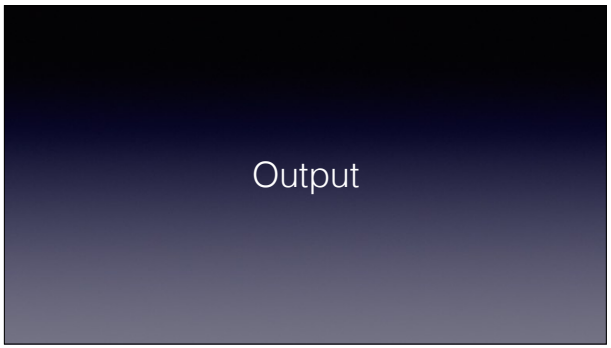
- Input: Variables supplied to a processor
  - Fed in at the beginning of an AutoPkg run
    - Software download URL
  - Input can also be generated during recipe runtime
    - What's this new software's version number?



305

- Override: Collections of substitute input which make a recipe run uniquely for your environment
  - Since different organizations have different needs, AutoPkg makes heavy use of variables that can be overridden
  - AutoPkg is designed that each recipe would have an override
  - Overrides also hold "trust" information, more about that later

---



306

- Output: Variables set by the result of running a processor
  - File paths, version numbers, information extracted from the download

307

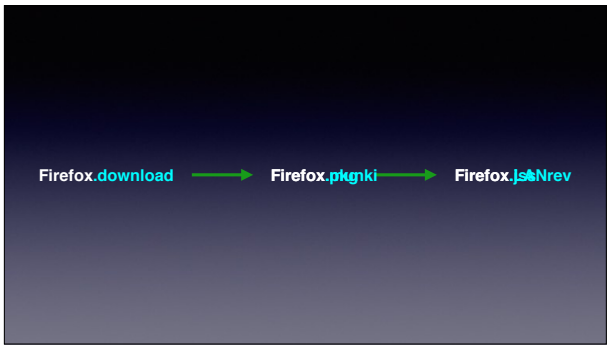
- Recipe: A procedure list of processors that perform a specific workflow
  - Each recipe has a particular type, denoted by extension
  - Such as download, pkg, munki, jss, lanrev, sccm, and others

---

308

- (Recipe) Repos: Collections of recipes, typically shared by others
  - Public repos are usually on Github in the AutoPkg organization
  - But a repo could be anywhere git is used (private, internal, etc)
  - Note there are repos that aren't in the AutoPkg org (such as Facebook's)
  - A quick Google search won't hurt, I've found recipes that are not listed in the AutoPkg org
- Ok, enough of the glossary, a quick AutoPkg concept to be aware of
  - AutoPkg recipes are built to be ***modular***, output of one recipe flows to the input of the next

---



309

- The Firefox.munki recipe runs Firefox.download and then Firefox.munki
- **Firefox.jss runs the same Firefox.download recipe, but then runs Firefox.pkg, then Firefox.jss
- **Firefox.LANrev runs the same Firefox.download and Firefox.pkg recipes, and then Firefox.LANrev
- You only need to run the last recipe in the hierarchy, AutoPkg can find the parents
- When running a recipe, it's helpful to think of the parent recipes being concatenated together as one recipe

Installation

- AutoPkg
  github.com/autopkg/autopkg/releases/latest
- Management System Tools
  github.com/autopkg/autopkg/wiki/Noteworthy-Processors
- git

310

- Installation
- AutoPkg itself
  - 10.6 or later, probably the minimum would be 10.10
  - AutoPkg *can* run on a Mac serving other things, I'd suggest running AutoPkg on a separate computer or VM
- Tools to support your management system
  - Munki admin tools (note it doesn't have to be a Munki client), Jamf Pro, etc
  - Many can be found on the AutoPkg Wiki "Noteworthy Shared Processors" page https://github.com/autopkg/autopkg/wiki/Noteworthy-Processors
- git
  - Xcode CLI tools probably easiest—just type "git" at the command line
  - Knowledge of git not required

```
$ autopkg search skype

Name                                 Repo                      Path
----                                 ----                      ----
Franz.munki.recipe                   andrewvalentine-recipes   Franz/Franz.munki.recipe
Skype.sccm.recipe                    cgerke-recipes            Applications/Skype.sccm.recipe
CallRecorderforSkype.munki.recipe    foigus-recipes            Ecamm/CallRecorderforSkype.munki.recipe
CallRecorderforSkype.pkg.recipe      foigus-recipes            Ecamm/CallRecorderforSkype.pkg.recipe
CallRecorderforSkype.download.recipe foigus-recipes            Ecamm/CallRecorderforSkype.download.recipe
Skype-Win.download.recipe            hansen-m-recipes          Skype/Skype-Win.download.recipe
AudioHijack.munki.recipe             homebysix-recipes         RogueAmoeba/AudioHijack.munki.recipe
Skype.jss.recipe                     jss-recipes               Skype/Skype.jss.recipe
Microsoft Lync.jss.recipe            jss-recipes               Microsoft Lync/Microsoft Lync.jss.recipe
Skype.jss.recipe                     novaksam-recipes          Skype/Skype.jss.recipe
Skype for Business.jss.recipe        novaksam-recipes          Skype for Business/Skype for Business.jss.recipe
SkypeForBusiness.pkg.recipe          novaksam-recipes          Recipes - pkg/SkypeForBusiness.pkg.recipe
```

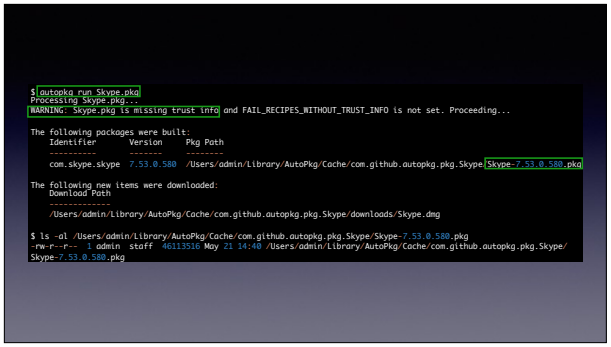
311

- We want to make a Skype package, so let's look for Skype using AutoPkg's search option**
  - Finding recipes others have written in the AutoPkg org

```
$ autopkg search skype | grep pkg

Name                                 Repo                      Path
----                                 ----                      ----
CallRecorderforSkype.pkg.recipe      foigus-recipes            Ecamm/CallRecorderforSkype.pkg.recipe
SkypeForBusiness.pkg.recipe          novaksam-recipes          Recipes - pkg/SkypeForBusiness.pkg.recipe
Skype.pkg.recipe                     recipes                   Skype/Skype.pkg.recipe

$ autopkg repo-add recipes
Attempting git clone...

Adding /Users/admin/Library/AutoPkg/RecipeRepos/com.github.autopkg.recipes to RECIPE_SEARCH_DIRS...
Updated search path:
  '.'
  '~/Library/AutoPkg/Recipes'
  '/Library/AutoPkg/Recipes'
  '/Users/admin/Library/AutoPkg/RecipeRepos/com.github.autopkg.recipes'
```

312

- Let's filter the search results with grep, looking for pkg recipes
  - **There's Skype.pkg, that looks likely
- **Let's add the recipe repo via AutoPkg's repo-add command

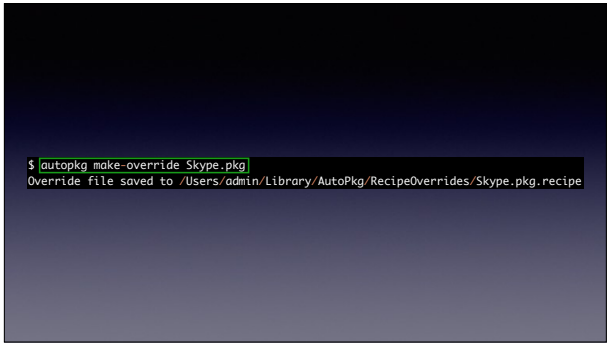


313

- Now that we have the repo added, let's run the Skype.pkg recipe
- **And the product of running the Skype recipe is a package, ready to install
- **Note the trust warning, we'll revisit that later

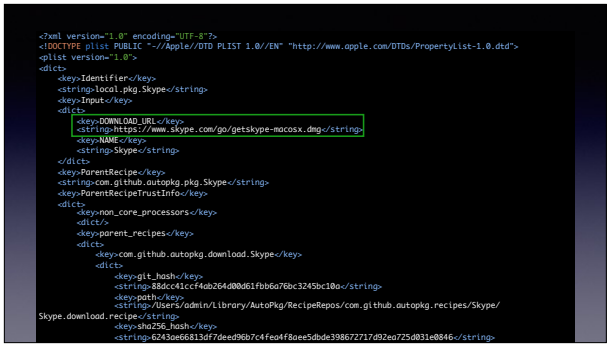
314

- The intent is that you can run AutoPkg recipes over and over and over
  - You get new software when it's available
- That's fantastic and all, but in my organization we call it Microsoft Skype!



315

- Let's take a look at making AutoPkg your own, through the creation and curation of overrides
- Creates an alternate set of Input to feed into AutoPkg
- Use the "make-override" option to create an override

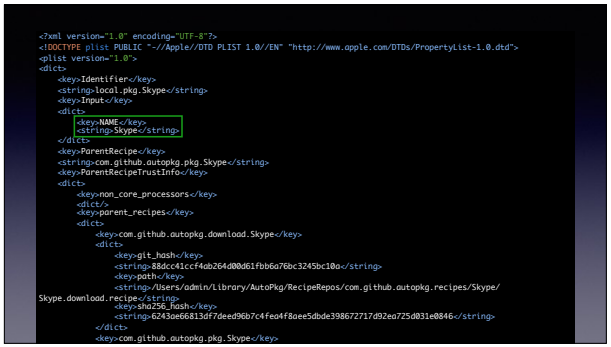


```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Identifier</key>
    <string>local.pkg.Skype</string>
    <key>Input</key>
    <dict>
        <key>DOWNLOAD_URL</key>
        <string>https://www.skype.com/go/getskype-macosx.dmg</string>
        <key>NAME</key>
        <string>Skype</string>
    </dict>
    <key>ParentRecipe</key>
    <string>com.github.autopkg.pkg.Skype</string>
    <key>ParentRecipeTrustInfo</key>
    <dict>
        <key>non_core_processors</key>
        <dict/>
        <key>parent_recipes</key>
        <dict>
            <key>com.github.autopkg.download.Skype</key>
            <dict>
                <key>git_hash</key>
                <string>88dcc41ccf4ab264d00d61fbb6a76bc3245bc10a</string>
                <key>path</key>
                <string>/Users/admin/Library/AutoPkg/RecipeRepos/com.github.autopkg.recipes/Skype/
Skype.download.recipe</string>
                <key>sha256_hash</key>
                <string>6243ae66813df7deed96b7c4fea4f8aee5dbde398672717d92ea725d031e0846</string>
```

316

- An override is a normal plist
- Looking at the “Input” you can override, set things you do need to override, remove what you don't need to override
    - **In this case the download URL doesn’t need to be overidden, so we can remove it

---



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Identifier</key>
    <string>local.pkg.Skype</string>
    <key>Input</key>
    <dict>
        <key>NAME</key>
        <string>Skype</string>
    </dict>
    <key>ParentRecipe</key>
    <string>com.github.autopkg.pkg.Skype</string>
    <key>ParentRecipeTrustInfo</key>
    <dict>
        <key>non_core_processors</key>
        <dict/>
        <key>parent_recipes</key>
        <dict>
            <key>com.github.autopkg.download.Skype</key>
            <dict>
                <key>git_hash</key>
                <string>88dcc41ccf4ab264d00d61fbb6a76bc3245bc10a</string>
                <key>path</key>
                <string>/Users/admin/Library/AutoPkg/RecipeRepos/com.github.autopkg.recipes/Skype/
Skype.download.recipe</string>
                <key>sha256_hash</key>
                <string>6243ae66813df7deed96b7c4fea4f8aee5dbde398672717d92ea725d031e0846</string>
            </dict>
            <key>com.github.autopkg.pkg.Skype</key>
```

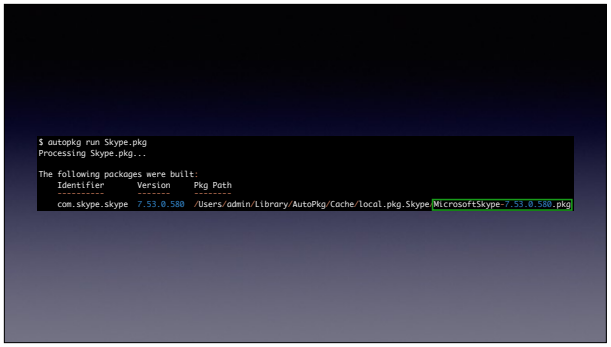
317

- There also is the “NAME” key
- Our organization wants this changed to “Microsoft Skype”

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>Identifier</key>
    <string>local.pkg.Skype</string>
    <key>Input</key>
    <dict>
        <key>NAME</key>
        <string>MicrosoftSkype</string>
    </dict>
    <key>ParentRecipe</key>
    <string>com.github.autopkg.pkg.Skype</string>
    <key>ParentRecipeTrustInfo</key>
    <dict>
        <key>non_core_processors</key>
        <dict/>
        <key>parent_recipes</key>
        <dict>
            <key>com.github.autopkg.download.Skype</key>
            <dict>
                <key>git_hash</key>
                <string>88dcc41ccf4ab264d00d61fbb6a76bc3245bc10a</string>
                <key>path</key>
                <string>/Users/admin/Library/AutoPkg/RecipeRepos/com.github.autopkg.recipes/Skype/
Skype.download.recipe</string>
                <key>sha256_hash</key>
                <string>6243ae66813df7deed96b7c4fea4f8aee5dbde398672717d92ea725d031e0846</string>
            </dict>
            <key>com.github.autopkg.pkg.Skype</key>
```

318

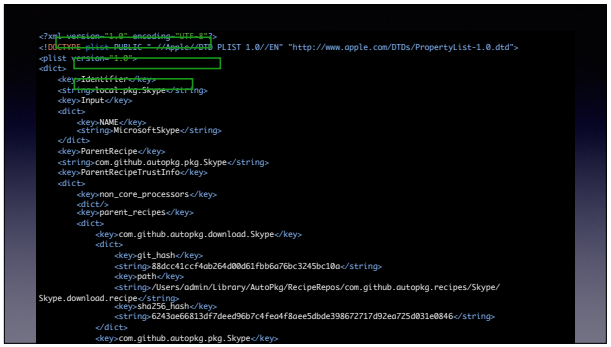
- Now let’s see what happens when the recipe is run



319

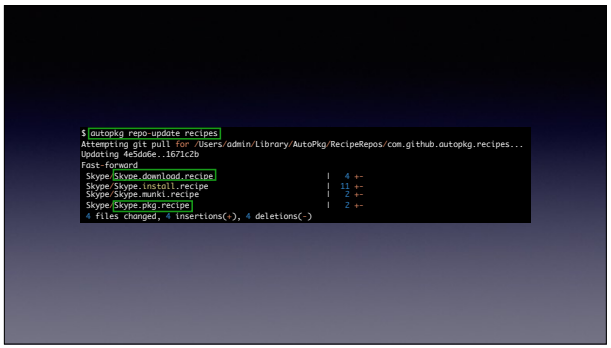
- Hooray, a Skype package was built
- But note the name change
- This package can be imported into your management system
    - If a recipe exists (or you write one) the package could be directly imported into your management system by AutoPkg
    - This would be the munki, or jss, or sccm, or whatever system recipe

---



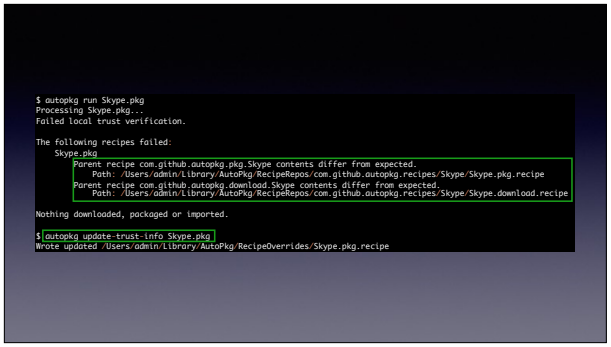
320

- Back to the override for a minute
- An important part of an override is the trust-related keys
    - **Scroll down to take a look at these
- Under the “ParentRecipeTrustInfo”, these keys record the state of the recipe at override creation time
    - **This includes the state of the third party processors
    - **As well as the state of the parent recipes
- The concept of “trust” in AutoPkg means that the administrator has reviewed and understands what the recipe is doing
    - The creation of the override adds this trust information
    - Why is this important—AutoPkg can do bad things



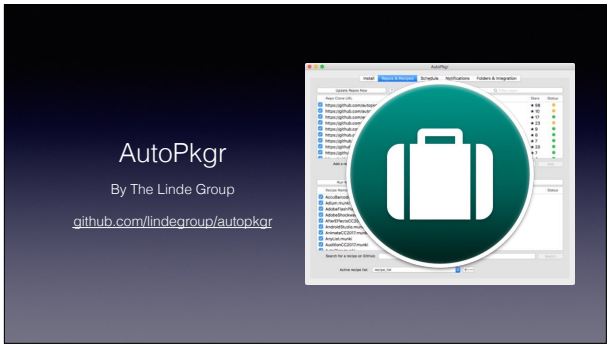
321

- It is important to be able to update AutoPkg recipes to handle changes
    - The “repo-update” command is used to obtain recipe updates
        - **In this case both the Skype download and package recipes were updated
    - But it’s also a potential vector for unintended consequences



322

- When an overridden recipe has parent recipes or processors change, it will refuse to run
  - ***This is a good thing,*** it notifies the conscientious admin to check the changes applied to a recipe
  - Once the admin has evaluated the changes and is comfortable with them, the **“update-trust-info” command will update the trust information in your override
  - You don’t directly modify the override’s trust information, let AutoPkg do it
- So after all this, let’s say you’re not a fan of the command line, what to do?



323

- There’s one other tool of note in the AutoPkg world
  - AutoPkgr is a free, open source GUI wrapper around AutoPkg, written by The Linde Group
  - Can install the various AutoPkg ecosystem pieces
  - Can run AutoPkg recipes on a schedule
  - And my favorite, can send notification emails when new software is retrieved
- **While it is a powerful tool, I’d start with AutoPkg at the command line
  - It’s better to start out at a lower level, understanding how AutoPkg pieces connect
  - Then move up to AutoPkgr, or even just use it for the automation and notification features

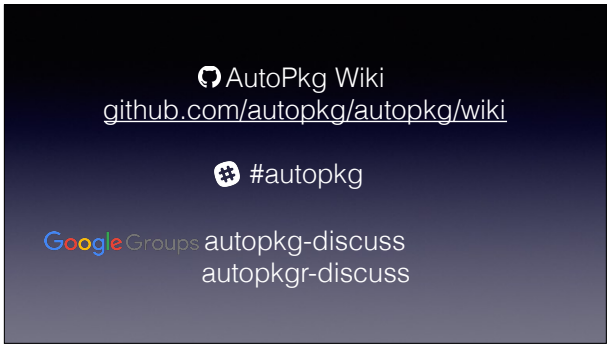
---

324

- To learn more about AutoPkg, be sure to attend Elliot Jordan’s session “How not to do bad things with AutoPkg”

325

- And if you can't make that, catch the recording of Damon Vogel's presentation from last year's Penn State MacAdmins, "Autopkg, Munki & You: Automating 3rd party application updates"



326

- If you need support
  - AutoPkg Wiki: https://github.com/autopkg/autopkg/wiki
  - #autopkg on MacAdmins Slack, good for both AutoPkg and AutoPkgr
  - autopkg-discuss and autopkgr-discuss https://groups.google.com/forum/#!forum/autopkg-discuss https://groups.google.com/forum/#!forum/autopkgr-discuss

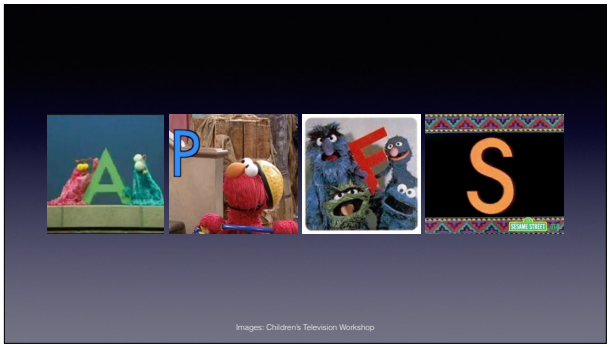
---

327

- You've now seen where we are, what options you have today
- A high level overview of the technologies you should be looking at
- So where are we going

328

- It's a bit of Sesame Street to determine it
- Your planning needs to include the following letters

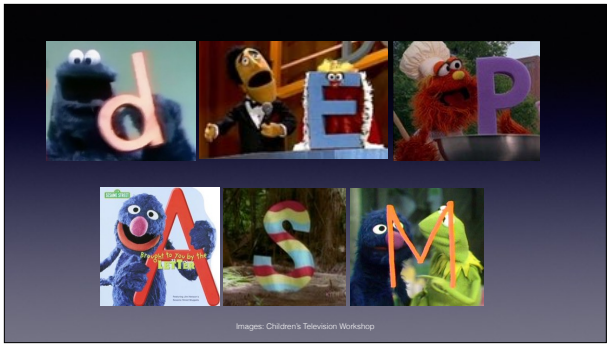


329

- APFS
  - Stands for Apple File System
  - Replaces the previous default macOS file system, HFS+
  - HFS+ was built for kilobytes and dozens of files on a floppy and doesn't cut it in a world of terabytes and millions of files on flash
  - APFS is a new disk format for a new era
- APFS's predecessor, HFS+, is based on roots so old that the file system is able to vote…and has been able to vote since 2002

330

- MDM
  - Mobile Device Management
  - Your MDM solution needs to meet *your* needs, evaluate carefully
    - Education-centric: Classroom productivity, testing and assessments, presentations, and integration with Student Information Systems
    - Business-centric: Auditing, Active Directory/LDAP integration
  - A pure MDM solution *today* might be enough, depending on needs, scale
  - A MDM solution paired with an agent might be needed to successfully manage your Macs



331

- DEP/ASM
  - Device Enrollment Program and Apple School Manager
  - Business uses Device Enrollment Program while education should be looking at Apple School Manager
  - Device Enrollment Program and Apple School Manager rely on Mobile Device Management
  - These device on boarding programs start with Apple's knowledge of an organization's device ownership

---

332

- But...
  - You may see reports of legacy things still working!
- macOS 10.13 appears to be a transitional release
  - Legacy items (HFS+, NetBoot, NetInstall) apparently are still functioning
  - But let's not depend on it
- Let's look at other Apple transition timelines

Change	Announced	Primary	Dead
Mac OS X	1998	2003	2007
Intel	2005	2009	2011

333

- We'll take a look at the change, when it was announced, when it became the primary option, and when the predecessor died
- **Mac OS 9 -> Mac OS X
  - Announced: May 1998
  - Primary: A bit fluid, but I'll pick the middle of the dates where Apple released hardware that didn't boot Mac OS 9, Mid-2003
  - Dead: Mac OS X Leopard, which didn't include the Classic environment, October 2007
- **PowerPC -> Intel
  - Announced: June 2005
  - Primary: Snow Leopard, which booted only on Intel Macs, August 2009
  - Dead: OS X Lion, no Rosetta, July 2011

334

Change	Announced	Primary	Dead
Mac OS X	1998	2003	2007
Intel	2005	2009	2011
Profiles/MDM/DEP	2011	2014	?
APFS	2016	2017	?

- **MCX -> Profiles/MDM/DEP
  - Announced: MDM and Profiles since OS X Lion, July 2011
  - Primary: DEP since February 2014
  - Dead: ?
- HFS+ -> APFS
  - Announced: WWDC last year, June 2016
  - Primary: With release of macOS High Sierra in roughly September 2017
  - Last macOS that can boot from HFS+: ?
- What's more interesting is the rate of change between these stages, let's calculate that in months

---

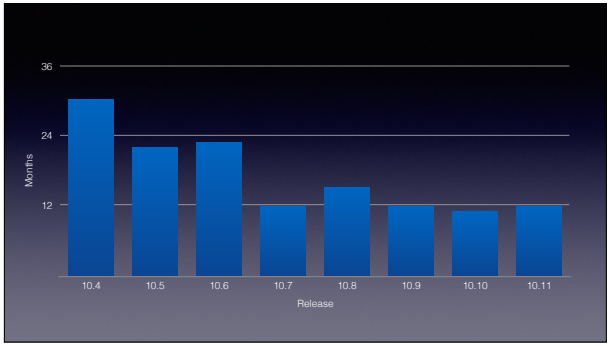
335

Change	Announced	Primary Δ*	Dead Δ*
Mac OS X	-	61	113
Intel	-	50	73
Profiles/MDM/DEP	-	31	?
APFS	-	15	?

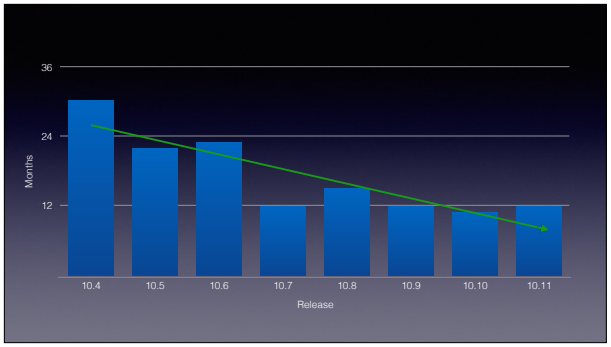
*Months

- As Apple proceeds through each of these transitions, the time has been *shortening*
- Apple's rate of change is increasing
- Another way to look at this is the longevity of releases of macOS

336

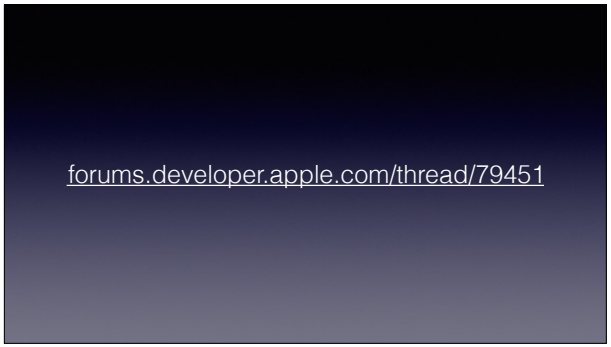


- Looks like this
- In case the trend isn't clear…



337

- Apple isn't going to be waiting around for you
- One quick diversion before the next segment

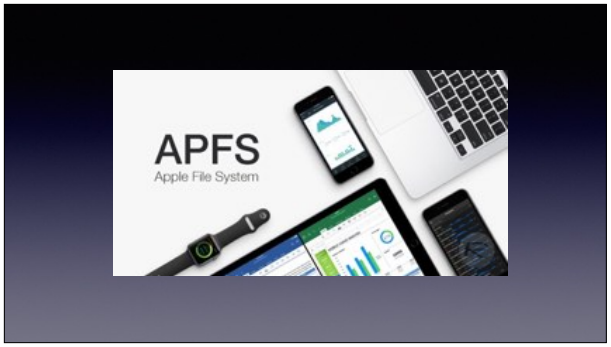


338

- The last couple of years, Rich Trouton has collected community questions and asked them at WWDC
  - Rich didn't win the WWDC lottery this year, but some others donated their time and effort to research community-fed questions
  - Rich collected the responses into one Apple Developer Forums meta-post
  - This post would basically be cited in every piece of the following presentation, so we'll do it here
- Note that some Apple employees may be misinformed, so take things with a grain of salt
  - It's worth reading, might answer some questions you didn't know you had

---

339



340

- What is APFS? It's the brand new Apple File System)
- APFS is one of the biggest changes coming in macOS High Sierra this fall.
- For the first time in nearly 20 years, the Mac is getting a new filesystem. People with iOS (and watchOS and tvOS) devices running the latest major revision of software are running APFS already.

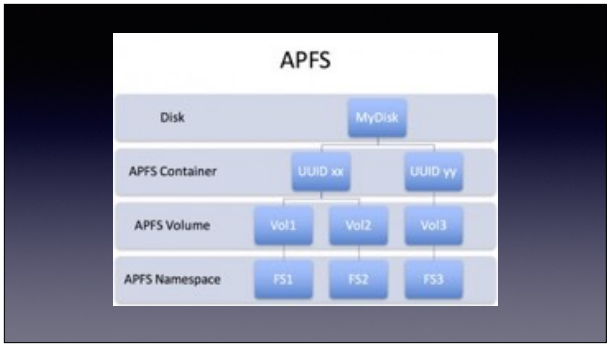
341

- What does this mean?
- The HFS Plus filesystem (which, in slightly modified form, is still the drive format for most Mac storage today) was introduced in 1998.
- Back when dinosaurs roamed the ea-, uh, I mean, back when we ran good old Mac OS 8.1 on PowerPC-based Macs.
- Was extended to add journaling, CoreStorage was invented to handle encryption and Fusion drives, etc.
-

342

HFS Plus (currently, Extended Journaled) has a number of issues, including:

1) File date limitation – February 6, 2040 due to 32-bit integer. Wasn't a concern when designed, but today?
2) Maximum of about 4 billion files
3) 1 second timestamp granularity – most other OSes can go to micro or even nanoseconds
4) single-threaded writes – meaning if one process is writing to disk, others need to wait to "take their turn"
5) doesn't support snapshot/rollback features common on more modern filesystems
6) designed for a processor architecture that handles data completely opposite of the current processor-architecture (big endian PPC vs little endian Intel)



343

- APFS addresses all of these, and more.

- 1) 64-bit block allocation means larger storage sizes
- 2) maximum of about 9 quintillion files
- 3) nanosecond timestamp granularity
- 4) multi-threaded writes with Atomic Safe-Save (i.e. the write happened or it didn't, no partial writes)
- 5) snapshots!
- 6) File duplication (cloning). Two copies of a file don't take up 2x the same amount of space. If one of the files is modified, only the changes are saved to disk.

344

HFS Plus (currently, Extended Journaled) has a number of issues, including:

1) File date limitation – February 6, 2040 due to 32-bit integer. Wasn't a concern when designed, but today?
2) Maximum of about 4 billion files
3) 1 second timestamp granularity – most other OSes can go to micro or even nanoseconds
4) single-threaded writes – meaning if one process is writing to disk, others need to wait to "take their turn"
5) doesn't support snapshot/rollback features common on more modern filesystems
6) designed for a processor architecture that handles data completely opposite of the current processor-architecture (big endian PPC vs little endian Intel)
7) Space Sharing. Volumes on APFS containers (like CoreStorage's logical volume groups) will show the same size and free space. Ever partitioned a drive with HFS Plus and needed to make the first partition larger without backing up, deleting, and restoring the second partition? Yeah. Not an issue any more.

345

Most productivity apps work at a much higher level and won't/don't care what the disk format is. It's only tools that create/manipulate/repair the filesystem that will need to be revised to support it. And those include imaging tools.

Do you have to move to APFS? No, not right away. At some point in time after High Sierra ships, Apple will start shipping Macs with APFS-formatted drives straight from the factory.

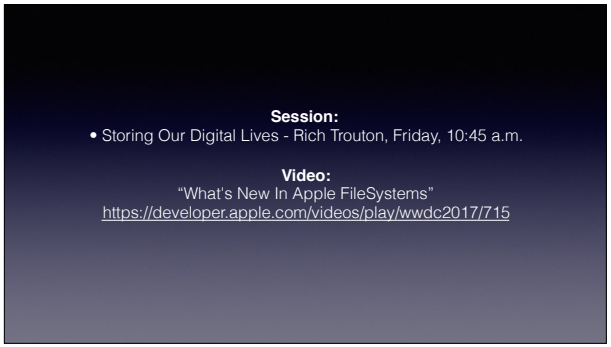
Any of the imaging tools will have to be revised to deal with APFS, or you'll have to do some trickerly to destroy the APFS volume and recreate it as HFS Plus Extended (Journaled).

346

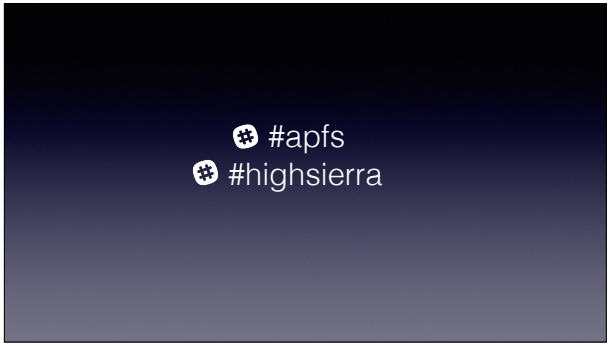
Apple says that asr restores of APFS volumes will be possible, but it doesn't work in the current seeds of macOS High Sierra. There isn't a ton of developer documentation out there.

But, you will likely want to move to APFS in the future. The potential performance increases are stunning – up to 17x faster in some cases. You can test APFS the creation and use of APFS volumes using VMWare Fusion 7.5.8 and later.

When macOS High Sierra ships, or preferably before, you're going to want to test your deployment methodologies with APFS, and come up with an adoption plan.



347



348

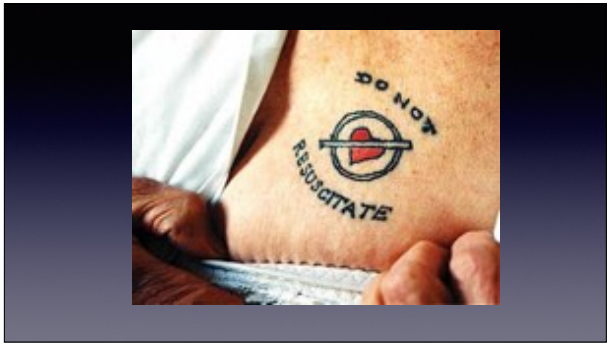
- Further discussion in the #apfs and #highsierra channels on MacAdmins Slack

349

350

You may have seen or received marketing emails from vendors that sell Mac management tools that proclaim “Imaging is Dead”. There were many warnings that macOS 10.13 would prevent us from imaging. But is it true? Is imaging really dead?

Right now, no. At Apple’s World Wide Developer’s Conference, questions were asked as to whether asr (Apple Software Restore - dates back to the 1990's), the fundamental, under-the-hood method to “copy” an OS image to storage media) would be supported for APFS volumes. There is no such capability today. But, it seems likely that there will be, by the time that macOS High Sierra ships this fall.



351

So, the patient might be in the ER, but is still alive. Although they have granted medical power-of-attorney and signed a Do Not Rescuscitate order.

Apple is already limiting kernel extensions in macOS High Sierra, to the detriment of antivirus vendors and other security software developers worldwide.

https://www.apple.com/business/docs/iOS_Security_Guide.pdf

"Each step of the startup process contains components that are cryptographically signed by Apple to ensure integrity and that proceed only after verifying the chain of trust. This includes the bootloaders, kernel, kernel extensions, and baseband firmware. This secure boot chain helps ensure that the lowest levels of software aren't tampered with."

352

Some of the things found lurking under the hood of macOS High Sierra have shown references to "SecureBoot". For a corollary, let's look at the iOS Security Guide: https://www.apple.com/business/docs/iOS_Security_Guide.pdf

If Apple does implement something like this for macOS, as it seems likely, that is probably the point in time where "imaging is dead". That date is perhaps 15 months off.

353

Combine cryptographically signed components and APFS snapshots. You know how you can "Erase all Content and Settings" on an iOS device and return it to factory settings? My guess is that we're probably headed down the same path for macOS.

354

For every major macOS version of late, Apple seems to introduce new security features that further increase security. From SIP to signed kernel extensions to APFS to potentially SecureBoot, the trend is very clear to see.

355

As new-to-the platform Mac admins, you need to be prepared for these changes. Settle on an MDM provider. Utilize DEP for out-of-the-box configuration. Use VPP to deploy App Store apps. Use configuration profiles to establish settings (don't push plists, use defaults write commands, or modify the user template). Sign your package and profiles.

356

If there are things that you cannot do today via MDM or custom config profiles, let Apple know (contact your sales engineers, contact AppleCare support if you have a contract, or open a "radar" at https://bugreport.apple.com).

357

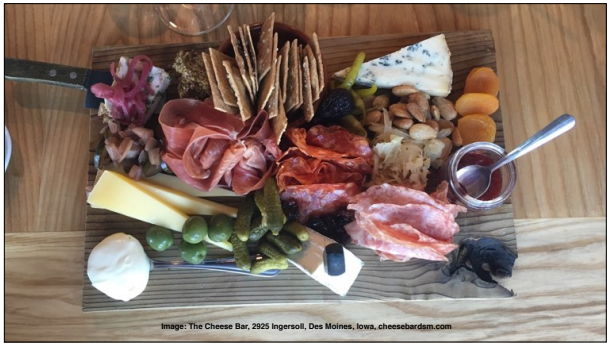
no MDM – may be limited as to what settings you can control
MDM-only – may be limited as to what you can do
MDM+agent – probably the best-case scenario

358

I hope you like change, because it seems like we are in for a lot of it in the future. (Aside: if you don't like change, you're in the wrong career field working in IT). You can protest, but you're not going to change Apple's path. The sooner that you can fully adopt these methodologies, the better off that you will be.

359

- Now for a mini smorgasboard
- What is this?



360

- It's not this kind of smorgasbord
    - These are a few other important tidbits that didn't really fit in anywhere else
    - Important for those testing macOS High Sierra

361

- macOS 10.13 High Sierra is last macOS to fully support 32 bit apps
- 32 bit applications may fail to run beyond macOS 10.13 High Sierra
  - Audit for 32 bit applications in System Information, under “Software” and then “Applications”
  - Contact your vendor for an upgrade or plan a migration off of them

---

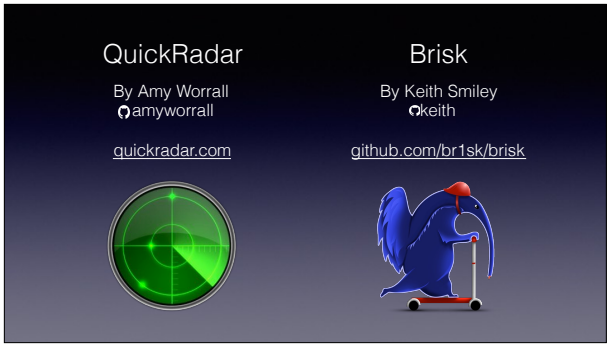
362

- Bad ideas
  - Gatekeeper can be disabled
    - Gatekeeper verifies downloaded applications before allowing them to run, reducing the chance of running malicious software
  - System Integrity Protection (SIP) can be disabled
    - SIP protects files and directories on the computer from inadvertent or intentional modification
  - User Template NOT locked down
    - The User Template is the collection of files that are copied into a new user home directory
- But stop doing disabling Gatekeeper and SIP, and stop editing the User Template, Apple could take these away
  - Complain to software vendors who make you do these things

---

363

- High Sierra no longer supports startup Profiles
- If you’re installing Profiles on a non-startup disk, Apple recommends you use the “profiles” command line tool to install Profiles on the next startup



364

- Filing Bugs
  - Use QuickRadar or a relatively new program, Brisk
  - These applications to easily let you file bugs through Apple's bug reporting system
    - And duplicate existing bugs

☆ https://derflounder.wordpress.com/2015/08/26/using-quickradar-to-file-bug-reports-with-apple/

365

- Search and use OpenRadar
  - An open database of submitted radars, allowing you to see others' radars and others to see yours
  - Note that Brisk allows you to optionally cross-post bugs to OpenRadar

☆ https://derflounder.wordpress.com/2015/08/26/using-quickradar-to-file-bug-reports-with-apple/

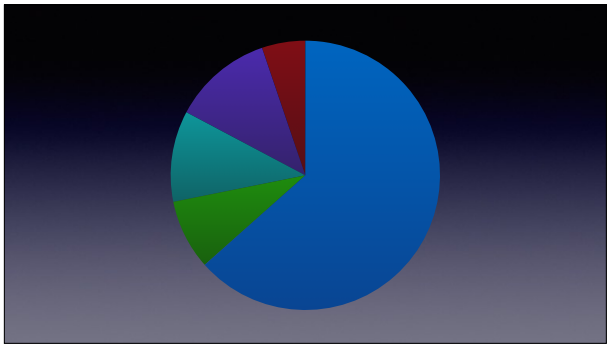
366

- Wait, Apple didn't announce we'd only have MDM
- Why are we talking about this, it hasn't happened

367

- No, Apple didn't, but it isn't hard to imagine Apple deciding to go down this road
  - Prep for a world that *doesn't* have agents
  - Maybe it'll never occur, but if you're going to be reworking your processes you'll be examining these sorts of questions anyways
- A blog post worth reading on this topic
  - “m(DM)acOS” is written by Mike Lynn, frogor
  - It discusses many of the same topics I’ll touch on in the next few minutes

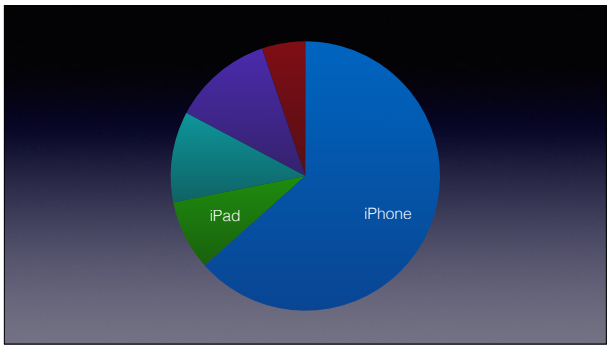
☆ Michael Lynn (frogor), “m(DM)acOS”, http://tinyurl.com/mDMacOS, http://michaellynn.github.io/2016/10/04/mDMacOS/
☆ Eric Holtam (eholtam), "APFS Disk Roles” http://tinyurl.com/apfs-disk-roles, https://osxbytes.wordpress.com/2017/04/10/apfs-disk-roles/



368

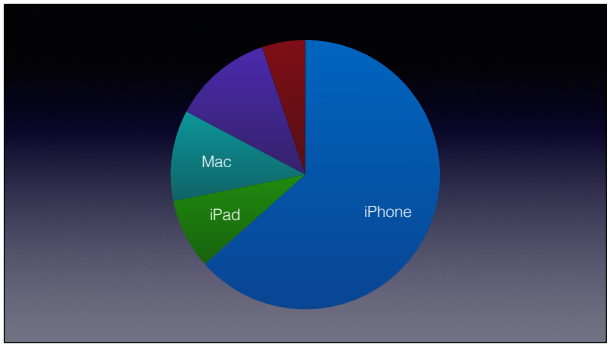
- Let's take a look at the evidence
  - Here’s Apple’s total revenue by segment over the last four quarters

---



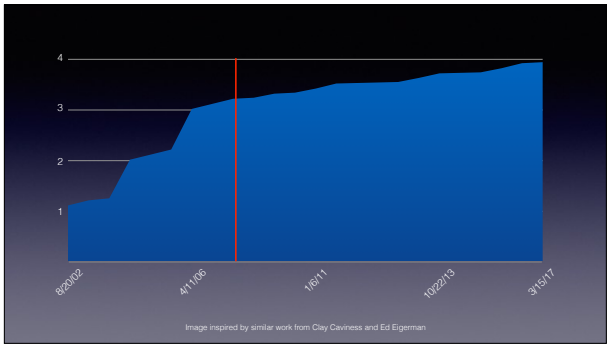
369

- iOS here is about 70%



370

- While the Mac is 10% (it's really 71/11)
- What if Apple looked at the big pile of money from iOS and inferred
    - "Folks really like the closed model! MDM works for that—it's all they really need!"
    - "Let's move the Mac to that model!"
- What about Apple's support of it's own management tools (besides DEP and MDM)?



371

- Here's an Apple Remote Desktop release graph for the last 15 years, inspired by a similar graph from two of Google's Mac administrators
    - For emphasis, **here's where the iPhone was released
    - But you might claim "Apple Remote Desktop is still supported!"

372

- There's a shirt for that
- **Let me zoom in
- Apple's interest in updating non-MDM tools definitely has waned

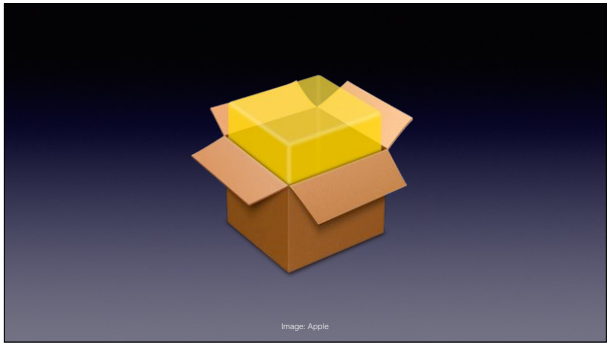
373

- Beyond this, in Server under macOS Sierra, Apple's Software Update Service was deprecated
  - Control of macOS updates will disappear
  - New in macOS High Sierra, software updates can be delayed for up to 90 days
    - If the Mac is managed via DEP and MDM

374

- One additional thing
  - Apple has said APFS is an attempt to unify the file system across multiple lines of devices
  - It wouldn't be a stretch to see DEP and MDM as an attempt to unify device management across multiple lines of devices

---

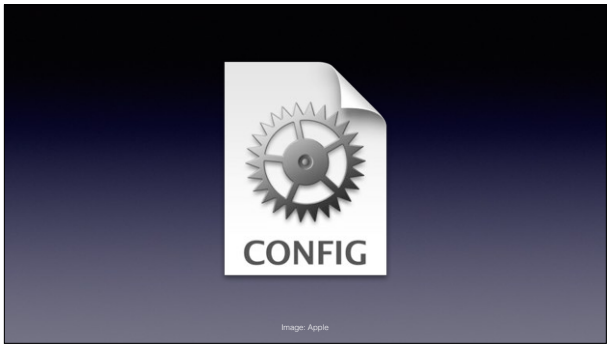


375

- Let's review what pure MDM needs and can do
- MDM requires signed Product Archive package
  - Or if the item exists in the Mac App Store, Volume Purchase Plan
  - Drag and drop installation isn't compatible with MDM
  - And large packages might be an issue, depending on infrastructure
- Older packages?
  - Unsigned, Component, bundle-style, mpkg need not apply
  - Varying levels of difficulty to address these package formats
- If your vendors are using the older packages, tell them to stop

376

- Besides being unusual in 2017
  - They're indicative of other development problems
  - Signed Product Archives, every time
- Note something near and dear to my heart
  - Adobe Creative Cloud Packager packages are bundle style packages
  - Just something to think about



377

- MDM can let you set Apple's blessed Profile settings, and Custom Settings Profiles
  - Not every application cares about managed settings
  - They may have their own preference handling
  - Some Apple Profiles aren't a la carte and instead manage lots of seemingly unrelated settings
  - Some features (such as setting a preference once) aren't supported or don't exist

---

378

- What do *you* need to deploy your infrastructure—can MDM do it today?
  - Do you need package dependencies? Including chaining independent packages together for a cohesive set of software?
  - Possibly you need installer choice changes, to enable and disable optional package choices?
  - Perhaps you need to manage a preference that MDM doesn't support?

379

- What can you do? Note this list is sourced from Mike Lynn's blog post mentioned earlier
- Inventory your existing management settings
    - Attempt to find Configuration Profile solutions for all of them
    - If they don't exist, file enhancement requests at bugreport.apple.com listing the following things:
        - Identify the setting and configuration options you need
        - Explain why the setting is important
        - Explain the impact of being unable to change this setting
        - Explain whether alternative methods exist currently to change it
        - Explicitly identify the number of impacted Macs / possible sales this could affect

---

380

- Then get about doing things with some of the tools from this workshop
- If Apple's Custom Settings Profiles *could* provide what you're looking for but the third-party software won't allow it, engage your software vendor
    - Shining example is Microsoft, who through engaging Mac admins in MacAdmins Slack has added manage-able preferences in response to customer feedback
- What if you need to modify the User Template?
    - No, wait…

381

- …that's still a bad idea, stop doing that anyways
- But seriously
    - This isn't to sound alarmist, Apple doesn't necessarily seem to be headed this way, *yet*
    - But why would they build out MDM management of Macs, and not try to complete its capabilities?

382

- For an additional take on these thoughts, watch Mike Lynn's presentation from last month's MacDevOps YVR

☆ Macnarök: The end times of our workflows - Mike Lynn, MacDevOps YVR 2017 https://www.youtube.com/watch?v=4Qv8CE6D_Rc&t=90s

---

383

- Time for the short closing ramble

384

- If you're not there already, join MacAdmins Slack
  - #gettingstarted, #jamfnation, #munki, #autodmg et. al.
  - To learn, to share, and to teach
  - Everyone knows something someone else doesn't—share your knowledge
  - The Mac admin community is amazing, it works because people give back
    - Blog posts
    - Software
    - Forum posts
    - Conference presentations
    - Answering questions in MacAdmins Slack
- When you return back to work after the conference, look forward to working with your fellow administrators

385

- Talk to people here—introduce yourself, say thanks
    - Maybe you owe Greg Neagle a frosty drink for Munki
    - Or Rich Trouton a cold Diet Coke for that helpful blog post
    - Or possibly you're going to leave a roadmap folded incorrectly in front of Mike Lynn (frogor) and see if he writes some sweet-ass Python to fix it
- Thank you all for coming, hopefully this workshop provided some useful information about:
    - Where we've been
    - Where we are
    - And where we're going
- **Feedback can be provided at the above URL