Munki Around

A Hands On Lab


2012 Greg
Neagle Lovefest
ImageChef.com

What is Munki?

Munki is a set of tools that can manage software installs on OS X Client Machines.

What can it do?

- Apple Packages
- Drag-and-Drop .app installs
- Adobe CS3/4/5 deployment “packages” and updates
- Apple Software Updates (From Apple or Internal SUS)
- Many other Stupid Munki Tricks

What is needed?

- Web Server (Apache, IIS, etc)
- Proper repo folder structure
- Clients

Web Server

- Any web host should work
- Easily accessible by Admin machine
- Ability to do SSL/Auth is desirable

Repo Structure

What does a Munki Repo look like?

catalogs

- Used for filtering packages (testing, production, etc)
- Clients look at catalog files when told to install things via manifest
- Helpful for testing updates before being deployed widely

manifests

- List of software to install, uninstall, update or offer as an optional install
- Specifies which catalog the software should be installed from (testing, production, etc)
- Can be nested

pkgs

- pkgs is a directory that contains all software installers
- Best practice to sort them by vendor rather than one folder with everything in it.

pkgsinfo

- pkgsinfo is a directory that contains metadata about all software installers
- pkgsinfo's structure is mirrored from the pkgs directory

Lab Time!

- Complete the "Initial Server Setup" section
- Look over your settings and make sure they match the guide if you run into errors
- If you have questions, ask!

More Details

- pkginfo files
- manifests
- ManagedInstalls.plist

pkginfo files

- Each key and value gives munki information about the package
- managedsoftwareupdate uses these keys to figure out to install, where and how
- Most are named well and self-explanatory

Name

- Munki uses the Name key to identify the software package in manifests
- Do not put version numbers or dashes in the name unless it is something like PhotoshopCS5, CS6, etc.

Name

- Simple Example

```
<key>name</key>
<string>Firefox</string>
```

- Another possibility

```
<key>name</key>
<string>PhotoshopCS5</string>
```

Version

- Version of the software being installed
- Can be used to specify version in manifests:
  - Firefox-10.0.3 vs Firefox-11.0.1
- When specifying just “Firefox,” munki always picks the newest version available in the catalogs for that client’s manifest

Version

- Firefox Version:

```
<key>version</key>
<string>10.0.4</string>
```

Installs

- Tells Munki what the package installs
- Could be a .app, a plist file with version info or even a file, its location and its hash value
- Munki uses this to determine if a package needs to be installed or not
- Install loops occur without these in some cases

Installs

```
<key>installs</key>
<array>
    <dict>
        <key>CFBundleIdentifier</key>
        <string>org.mozilla.firefox</string>
        <key>CFBundleName</key>
        <string>Firefox</string>
        <key>CFBundleShortVersionString</key>
        <string>10.0.4</string>
        <key>minosversion</key>
        <string>10.5</string>
        <key>path</key>
        <string>/Applications/Firefox.app</string>
        <key>type</key>
        <string>application</string>
    </dict>
</array>
```

unattended install/ uninstall

- Will install or uninstall silently
- If a blocking_applicaton is open, will delay install until later

unattended install/ uninstall

```
<key>unattended_install</key>
<true/>

<key>unattended_uninstall</key>
<true/>
```

blocking_applications

- Munki will not install the update until the specified apps are closed
- Works hand in hand with unattended_installs for completely safe and silent installs.
- See http://code.google.com/p/munki/wiki/BlockingApplications

blocking_applications

```
<key>blocking_applications</key>
<array>
    <string>Safari</string>
    <string>Firefox</string>
    <string>Opera</string>
</array>
```

force_install_after_date

- Will forcibly log out users and install software after date specified
- Warns users progressively often until date is reached
- Excellent for critical security updates

force_install_after_date

```
<key>force_install_after_date</key>
<date>2012-05-04T13:00:00Z</date>
```

Detailed List

- http://code.google.com/p/munki/wiki/SupportedPkginfoKeys
- Wiki pages for specific keys and their implementation

manifests

- Lots of options
- Makes your deployments more flexible **and** smart
- An array of strings, except for conditionals

included_manifests

- Allows you to nest manifests
- Good for separating out common software installs from special cases

included_manifests

```
<key>included_manifests</key>
<array>
    <string>common</string>
    <string>universalprintqueues</string>
    <string>MunkiCerts</string>
    <string>MunkiScripts</string>
</array>
```

managed_installs/ uninstalls

- Will try to install every time munki runs until they are successfully installed
- Depending upon the pkginfo, can install silently without user interaction

managed_updates

- Like managed_installs, but does not initially install the software
- Great for updating software that is already on clients but isn't a mandatory install

optional_installs

- Self Service
- Software is maintained by munki after being installed
- User Initiated OS Upgrades - upgrade from 10.6 to 10.7 using InstallLion.pkg

conditional_items

- Set any install types based upon specific conditions
- Laptop vs Desktop
- 10.6 vs 10.7
- Many others: http://code.google.com/p/munki/wiki/ConditionalItems

conditional_items

```
<key>conditional_items</key>
<array>
  <dict>
      <key>condition</key>
      <string>os_vers BEGINSWITH "10.6"</string>
      <key>managed_installs</key>
      <array>
         <string>JavaForMacOSX106</string>
      </array>
      <key>optional_installs</key>
      <array>
         <string>OSXLion</string>
      </array>
  </dict>
</array>
```

conditional_items

```
<key>conditional_items</key>
<array>
  <dict>
      <key>condition</key>
      <string>os_vers BEGINSWITH "10.6"</string>
      <key>managed_installs</key>
      <array>
         <string>JavaForMacOSX106</string>
      </array>
      <key>optional_installs</key>
      <array>
         <string>OSXLion</string>
      </array>
  </dict>
</array>
```

conditional_items

```
<key>conditional_items</key>
<array>
  <dict>
      <key>condition</key>
      <string>os_vers BEGINSWITH "10.6"</string>
      <key>managed_installs</key>
      <array>
         <string>JavaForMacOSX106</string>
      </array>
      <key>optional_installs</key>
      <array>
         <string>OSXLion</string>
      </array>
  </dict>
</array>
```

conditional_items

```
<key>conditional_items</key>
<array>
  <dict>
      <key>condition</key>
      <string>os_vers BEGINSWITH "10.6"</string>
      <key>managed_installs</key>
      <array>
         <string>JavaForMacOSX106</string>
      </array>
      <key>optional_installs</key>
      <array>
         <string>OSXLion</string>
      </array>
  </dict>
</array>
```

ManagedInstalls.plist

- Can install Apple Updates
- Suppress User Notification
- InstallRequiresLogout
- Many other options: http://code.google.com/p/munki/wiki/configuration

Repo Management

- makepkginfo
- munkiimport
- makecatalogs
- manifestutil

makepkginfo

- Used to generate whole pkginfo files
- Can also generate pieces of a pkginfo file (installs item, postinstall script, etc)
- No configuration necessary

munkiimport

- Imports installers to the repo and generates pkginfo files
- Accepts pkgs, mpkgs, dmgs and apps
- Will transfer pkgsinfo keys from previous versions of the software
- Must be configured before use

makecatalogs

- Regenerates the catalog files
- Always run after tweaking a pkginfo file

manifestutil

- Tool to manage manifests
- No hand-editing of xml necessary
- Has tab autocomplete for various items (software names, manifests, etc)
- Must be configured before use

Lets dig in!

- Complete the remaining sections
- Don't be afraid of Terminal
- The repo tools will make this relatively painless
- If you have questions, ask!
- Complete the Extra Credit if time permits

Lab Files

- Slides:
- Lab Guide: http://db.tt/SJDBWu6V
- Lab Files: http://db.tt/FZ9nyCCh

Additional Resources

- Munki Wiki - http://code.google.com/p/munki/wiki/GettingStartedWithMunki?tm=6
- http://groups.google.com/group/munki-dev
- pkginfo examples - https://github.com/arubdesu/BarrelOfPkginfos
- ##osx-server on irc.freenode.net