# Deploying 1400+ computers in 3 weeks? Are you nuts?!?

Sean Kaiser

Northmont City Schools

# Who am I?

- systems integration specialist
- server, network, Mac, Google apps domain administrator
- level 3 tech support

# What this session is not

- A how-to on:
  - DeployStudio
  - the luggage
  - munki
  - puppet
  - reposado

# What this session is

- How I configured, used, and extended:
  - DeployStudio
  - the luggage
  - munki
  - puppet
  - reposado

  to complete a major deployment of machines in a short amount of time by a small group of staff

# Brief Project Background

- needs
  - replace various old platforms (iBooks, eMacs, iMacs)
  - prepare for mandated online testing
- why during school year
  - financial deals from Apple during Oct-Dec quarter
  - warehouse space limited for huge shipment of computers
- timeline ~2 months from initial discussion until go time

# History of deployment methods

- manual
- firewire target disk mode
- netboot
  - netrestore
  - deploystudio
- monolithic/golden master vs. modular deployment

# Getting the pieces in place

- Web Help Desk
- DeployStudio
- the luggage
- munki
- puppet
- reposado
- take delivery of machines

# Getting the pieces in place - Web Help Desk

- custom field for puppet classes, add options for munki


- performance issues due to frontbase database and tie-in with puppet
- refactor existing lookup scripts to query mysql database instead of frontbase

# Getting the pieces in place - DeployStudio

- updated existing workflow for 10.7.5 image
- created new workflow for 10.8.2 image
- created new NBI for new hardware
- created new workflow for modular deployment (“Northmont-ize”)
- full imaging (wipe/reinstall) workflows include same steps as the Northmont-ize workflow

# Getting the pieces in place - DeployStudio

- Northmont-ize workflow
  - set time server; computer name; anonymous bind to OD; install facter, puppet, and config files/service; create local user accounts; install munki; enable ARD; join appropriate WLAN; split hard drive; set munki bootstrap file
  - all postponed installs (so runs at first boot)
  - workflow takes ~2 minutes

# Getting the pieces in place - DeployStudio

- first boot process takes <5 minutes
- longest part is the split of the hard drive due to diskutil verifying drive before doing the split
- split hard drive into 2 partitions
  - 200 GB for boot partition
  - rest for userHD which is mounted via /etc/fstab at /Users (by UUID)
  - allows reimaging of boot drive without affecting user's data

# Getting pieces in place - the luggage

- (re)package installers
  - Flash player
  - other applications that need customization beyond provided installer
- build printer installers

# Getting pieces in place - puppet vs munki

puppet

- Adobe Reader
- Chrome
- FileMaker Pro
- Firefox
- Flash player
- Google Earth
- Google SketchUp
- iLife
- local users
- Office
- puppet/facter installers
- services (scripts/plists)
- VLC

# Getting pieces in place - puppet vs munki

## puppet

- local users
- services (scripts/plists)

## munki

- Adobe Reader
- Chrome
- FileMaker Pro
- Firefox
- Flash player
- Google Earth
- Google SketchUp
- iLife
- Office
- puppet/facter installers
- VLC

# Getting pieces in place - puppet

- disable installs that we were moving to munki
- manage munki services and preflight script

# Getting pieces in place - puppet

munki.pp

```
# /etc/puppet/manifests/classes/munki.pp

class munki {

    service { "com.googlecode.munki.logouthelper":
        enable => true,
        ensure => running,
    }

    service { "com.googlecode.munki.managedsoftwareupdate-check":
        enable => true,
        ensure => running,
    }

    service { "com.googlecode.munki.managedsoftwareupdate-install":
        enable => true,
        ensure => running,
    }

    service { "com.googlecode.munki.managedsoftwareupdate-manualcheck":
        enable => true,
        ensure => running,
    }

    file { "/usr/local/munki/preflight":
        source => "puppet:///files/macosx/usr/local/munki/preflight",
        ensure => present,
        owner => root,
        group => wheel,
        mode => 755,
    }

}
```

# Getting pieces in place - puppet

- what was left for puppet to manage?
  - users (2 admins and 1 non-admin/no password student account)
  - services
    - remote logon (sshd)
    - puppet (launchd plist, service)
    - munki (launchd plists, services)
  - scripts
    - munki's preflight and restart puppet launchdaemon

# Getting pieces in place - munki

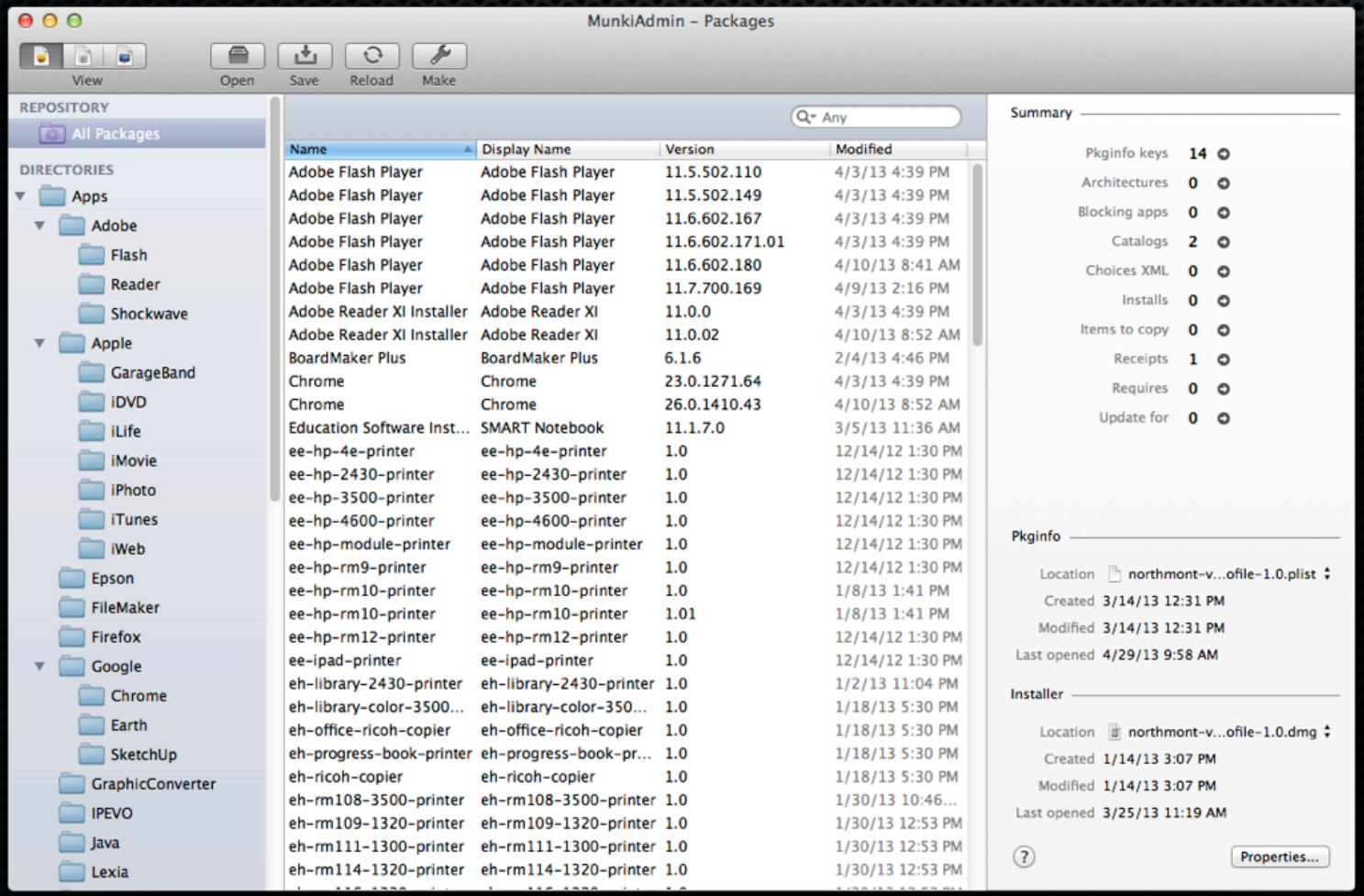
- learn... via MacTech articles, munki wiki
- what software to install and how to install it
  - managed or optional install
  - attended or unattended install

# Getting pieces in place - munki

- design manifest “groups”
  - use web help desk options for puppet & munki custom field


- build munki configuration
  - MunkiAdmin

# Getting pieces in place - munki (via MunkiAdmin)



# Getting pieces in place - munki (preflight script)

- dynamically configures manifest option (ClientIdentifier) from data in web help desk
  - combination of location and munki role fields
  - sets CatalogURL in com.apple.softwareupdate.plist and disables scheduled checks
- installed as part of DeployStudio workflow

# Getting pieces in place - munki (preflight script)

preflight

```
# define the hw_address
hw_address=`/sbin/ifconfig en0 | /usr/bin/awk '/ether/ {gsub(":",""); print $2 }'`

# get the location field value
location=`/usr/bin/curl -s -G -d "mac_address=${hw_address}" -d "field=6" ${whdurl} | tr ' ' '\0' | sed 's/^bldg_//g'`

# get the puppet extra classes field values (the munki_* ones, at least)
classes=`/usr/bin/curl -s -G -d "mac_address=${hw_address}" -d "field=9" ${whdurl} | tr ';' ' '`

# parse classes for major role and test role
for class in ${classes}; do
  if [[ `echo ${class} | grep "^munki_" | wc -l` -gt 0 ]]; then
    role=`echo ${class} | cut -d'_' -f2`
    if [ "${role}" == "test" ]; then
      swupdatecatalogurl="${swupdateurl}/index_test.sucatalog"
    else
      munkirole="${munkirole}_${role}"
    fi
  fi
done

if [ "x${munkirole}" != "x" ]; then
  munkirole="${location}${munkirole}"
fi

if [ "x${swupdatecatalogurl}" != "x" ]; then
  munkirole="${munkirole}_test"
else
  swupdatecatalogurl="${swupdateurl}/index_production.sucatalog"
fi

if [ "x${swupdatecatalogurl}" != "x" ]; then            # should never be 'x' as if not a tester it's production
  /usr/bin/defaults write "${swupdatepref}" CatalogURL "${swupdatecatalogurl}"
  /usr/bin/defaults write "${swupdatepref}" ScheduleFrequency -int -1
  logger "configured software update's catalog url as ${swupdatecatalogurl}"
fi

if [ "x${munkirole}" != "x" ]; then
  /usr/bin/defaults write "${munkipref}" InstallAppleSoftwareUpdates -bool false
  /usr/bin/defaults write "${munkipref}" SoftwareRepoURL "${munkiurl}"
  /usr/bin/defaults write "${munkipref}" ClientIdentifier "${munkirole}"
  /usr/bin/defaults write "${munkipref}" SuppressAutoInstall -bool false
  logger "configured munki to use ${munkiurl} and manifest ${munkirole}"
fi

exit 0
```

# Getting pieces in place - reposado

- want to mirror Apple software update service internally due to internet bandwidth, and have the ability to control what updates are available
- but how to manage?
  - regular command line tool (repoutil)
  - web interface

# Getting pieces in place - reposado (& margarita)

- margarita (web interface)
- no authentication though, so need to secure otherwise
  - anyone who can get to web page can modify catalogs
  - run on a non-standard port
  - work being done to get authentication working, though

# Getting pieces in place - reposado (& margarita)

Margarita | Apply queued changes 0 | Hide commonly listed updates | Filter...

| Software Update Product | | Version | Post Date | Apple branch | production branch | testing branch |
|---|---|---|---|---|---|---|
| Thunderbolt Firmware Update | | 1.2 | 2013-05-09 | Listed | Unlisted | Unlisted |
| Chinese Word list Update | | 1.6 | 2013-05-08 | Listed | Unlisted | Unlisted |
| Xerox Printer Software Update | | 2.3 | 2013-05-02 | Listed | Unlisted | Unlisted |
| Digital Camera RAW Compatibility Update | | 4.05 | 2013-04-17 | Listed | Unlisted | Unlisted |
| iPhoto Update | | 9.4.3 | 2013-04-16 | Listed | Unlisted | Unlisted |
| Canon Laser Printer Software Update | | 2.0 | 2013-04-16 | Listed | Unlisted | Unlisted |
| Safari | | 5.1.9 | 2013-04-16 | Listed | Unlisted | Unlisted |
| HP Printer Software Update | | 2.14 | 2013-04-16 | Listed | Unlisted | Unlisted |
| EPSON Printer Software Update | | 2.14 | 2013-04-16 | Listed | Unlisted | Unlisted |
| Safari | | 6.0.4 | 2013-04-16 | Listed | Unlisted | Unlisted |
| Java for OS X 2013-003 | | 1.0 | 2013-04-16 | Listed | Unlisted | Unlisted |
| Java for Mac OS X 10.6 Update 15 | | 15.0 | 2013-04-16 | Listed | Unlisted | Unlisted |
| Safari | | 6.0.4 | 2013-04-16 | Listed | Unlisted | Unlisted |
| Aperture Update | | 3.4.4 | 2013-04-16 | Listed | Unlisted | Unlisted |
| ProApps QuickTime codecs | | 1.0.2 | 2013-03-28 | Listed | Unlisted | Unlisted |
| Chinese Word li Update Deprecated | | 1.5 | 2013-03-26 | Unlisted | Unlisted | Unlisted |

# Getting pieces in place - taking delivery

- 1250 13” MacBook Pros (5 packs)
- 60 21.5” last model iMacs
- occupied a lot of our warehouse

# Getting pieces in place - inventory

- serial number, MAC addresses (ethernet and wifi) scanned from shipping boxes into a spreadsheet
- asset number, client name, location, wlan, munki role added
- imported spreadsheet into web help desk

# Ready to go!

DeployStudio first boot

(less than 5 minutes)

munki bootstrap

(hours... uh oh...)

# Back to the drawing board

- what went wrong?
  - munki and reposado repos at district NOC
  - 20 mbps WAN link
  - 10 simultaneous machines
  - ~400 MB of installation dmgs
  - 10.8.2 combo update and iTunes 10.7 (800+ MB)

# Options

- move setup operation to high school (NOC location)
  - pro: bandwidth not an issue (local LAN)
  - con: no space for all of the new machines; have to move them twice; no time
- change configuration/mirror repo
  - pro: fairly quick fix; future-proof
  - con: half a day of imaging “wasted” getting local repository configured

# The fix

- replicate munki repo to local server
- change munki preflight script and deploy via puppet
  - set PackageURL to local server (based on subnet)
  - set InstallAppleSoftwareUpdates to false
- result: munki bootstrap = about 8-10 minutes (even with 10 machines at a time)

# The fix

preflight

```
# what's our location based on ip address?
ip_address=`/usr/sbin/netstat -rn | /usr/bin/grep default | /usr/bin/grep en | /usr/bin/head -1 | /usr/bin/awk '{ print $2 }'`
if [ "`echo ${ip_address} | cut -d. -f1-3`" == "10.57.0" ]; then        # at northview/preschool site
  pkgurl="10.57.0.11"
elif [ "`echo ${ip_address} | cut -d. -f1-2`" == "10.11" ]; then        # on the district's network
  oct3=`echo ${ip_address} | cut -d. -f3`
  case ${oct3} in
    0) pkgurl="10.11.1.250" ;;
    1) pkgurl="10.11.1.250" ;;
    2) pkgurl="10.11.1.250" ;;
    3) pkgurl="10.11.1.250" ;;
    4) pkgurl="10.11.1.250" ;;
    5) pkgurl="10.11.1.250" ;;
    6) pkgurl="10.11.1.250" ;;
    7) pkgurl="10.11.1.250" ;;
    128) pkgurl="10.11.129.250" ;;
    129) pkgurl="10.11.129.250" ;;
    130) pkgurl="10.11.129.250" ;;
    131) pkgurl="10.11.129.250" ;;
    132) pkgurl="10.11.129.250" ;;
    133) pkgurl="10.11.129.250" ;;
    134) pkgurl="10.11.129.250" ;;
    135) pkgurl="10.11.129.250" ;;
    32) pkgurl="10.11.33.250" ;;
    33) pkgurl="10.11.33.250" ;;
    34) pkgurl="10.11.33.250" ;;
    35) pkgurl="10.11.33.250" ;;
    36) pkgurl="10.11.33.250" ;;
    37) pkgurl="10.11.33.250" ;;
    38) pkgurl="10.11.33.250" ;;
    39) pkgurl="10.11.33.250" ;;
    160) pkgurl="10.11.161.250" ;;
    161) pkgurl="10.11.161.250" ;;
    162) pkgurl="10.11.161.250" ;;
    163) pkgurl="10.11.161.250" ;;
    164) pkgurl="10.11.161.250" ;;
    165) pkgurl="10.11.161.250" ;;
    166) pkgurl="10.11.161.250" ;;
    167) pkgurl="10.11.161.250" ;;
    40) pkgurl="10.11.41.250" ;;
    41) pkgurl="10.11.41.250" ;;
    42) pkgurl="10.11.41.250" ;;
    43) pkgurl="10.11.41.250" ;;
    44) pkgurl="10.11.41.250" ;;
    45) pkgurl="10.11.41.250" ;;
```

# The fix

preflight

```
    *) pkgurl="munki.northmontschools.com" ;;
  esac
fi

# parse classes for major role and test role
for class in ${classes}; do
  if [[ `echo ${class} | grep "^munki_" | wc -l` -gt 0 ]]; then
    role=`echo ${class} | cut -d'_' -f2`
    if [ "${role}" == "test" ]; then
      swupdatecatalogurl="${swupdateurl}/index_test.sucatalog"
    else
      munkirole="${munkirole}_${role}"
    fi
  fi
done

if [ "x${munkirole}" != "x" ]; then
  munkirole="${location}${munkirole}"
fi

if [ "x${swupdatecatalogurl}" != "x" ]; then
  munkirole="${munkirole}_test"
else
  swupdatecatalogurl="${swupdateurl}/index_production.sucatalog"
fi

if [ "x${swupdatecatalogurl}" != "x" ]; then            # should never be 'x' as if not a tester it's production
  /usr/bin/defaults write "${swupdatepref}" CatalogURL "${swupdatecatalogurl}"
  /usr/bin/defaults write "${swupdatepref}" ScheduleFrequency -int -1
  logger "configured software update's catalog url as ${swupdatecatalogurl}"
fi

if [ "x${munkirole}" != "x" ]; then
  /usr/bin/defaults write "${munkipref}" SoftwareRepoURL "${munkiurl}"
  /usr/bin/defaults write "${munkipref}" SuppressAutoInstall -bool false
  /usr/bin/defaults write "${munkipref}" InstallAppleSoftwareUpdates -bool false
  if [ "x${pkgurl}" != "x" ]; then
    /usr/bin/defaults write "${munkipref}" PackageURL "http://${pkgurl}/munki/repo/pkgs"
  else
    /usr/bin/defaults delete "${munkipref}" PackageURL
  fi
  logger "configured munki to use ${munkiurl} and manifest ${munkirole}, pkgurl ${pkgurl}"
fi

exit 0
```

# But... what about Apple updates?

- didn't fix the Apple swupdate issue, just disabled them
- no way to specify different locations for CatalogURL and actual packages (like you can with munki's SoftwareRepoURL versus PackageURL, CatalogURL, and ManifestURL)
- solution: use apache's mod_rewrite to redirect content/downloads (but not catalog) requests to local repo host based on ip address
- supplements OS version agnostic CatalogURL setting on client

# Multi-site reposado

- synchronize master reposado repository to local servers' web directory
  - only the reposado/html/content/downloads folder is necessary
  - use synchronization tool of choice
- why not set CatalogURL to local reposado repo URL?
  - portable machines not always awake to run puppet and munki
  - some staff move between buildings, so reconfiguration necessary prior to running softwareupdate

# Multi-site reposado

- .htaccess on master reposado server (not on clones)

.htaccess

```
Options +FollowSymLinks
RewriteEngine On

# sc untrusted
RewriteCond %{REMOTE_ADDR} ^10\.11\.([0-7])\.([0-9]|[1-9][0-9]|1([0-9][0-9])|2([0-4][0-9]|5[0-5]))$
RewriteRule ^content/downloads/(.+) http://updates.sc.northmont.k12.oh.us/content/downloads/$1 [R=302,L]

# co untrusted
RewriteCond %{REMOTE_ADDR} ^10\.11\.([8-9]|1[0-5])\.([0-9]|[1-9][0-9]|1([0-9][0-9])|2([0-4][0-9]|5[0-5]))$
RewriteRule ^content/downloads/(.+) http://updates.co.northmont.k12.oh.us/content/downloads/$1 [R=302,L]

# .sucatalog stuff
RewriteCond %{HTTP_USER_AGENT} Darwin/8
RewriteRule ^index(.*)\.sucatalog$ content/catalogs/index$1.sucatalog [R=302,L]
RewriteCond %{HTTP_USER_AGENT} Darwin/9
RewriteRule ^index(.*)\.sucatalog$ content/catalogs/others/index-leopard.merged-1$1.sucatalog [R=302,L]
RewriteCond %{HTTP_USER_AGENT} Darwin/10
RewriteRule ^index(.*)\.sucatalog$ content/catalogs/others/index-leopard-snowleopard.merged-1$1.sucatalog [R=302,L]
RewriteCond %{HTTP_USER_AGENT} Darwin/11
RewriteRule ^index(.*)\.sucatalog$ content/catalogs/others/index-lion-snowleopard-leopard.merged-1$1.sucatalog [R=302,L]
RewriteCond %{HTTP_USER_AGENT} Darwin/12
RewriteRule ^index(.*)\.sucatalog$ /content/catalogs/others/index-mountainlion-lion-snowleopard-leopard.merged-1$1.sucatalog
[R=302,L]
```

# Monitoring client machines' state

- need to know if software is updated, what version(s)
- munkiwebadmin
  - very easy to install (< 30 minutes)
  - munki preflight/postflight scripts
  - shows installed applications' version
  - add-on script includes internet plug-ins, too!
  - use puppet to deploy preflight/postflight scripts
    - didn't include directly in pre/postflight scripts, but rather as standalone scripts that get called

# Munkiwebadmin

MunkiWebAdmin

munki.northmontschools.com/report/overview

Apple iCloud Facebook Twitter Wikipedia Yahoo! News Popular dvr

MunkiWebAdmin

**Errors** 46

**Warnings** 1271

**Activity** 518

| Machine | User | IP | Latest Run |
|---|---|---|---|
| nwprincipal-23829 | mrichardson | 10.11.43.141 | 2013-05-14 08:24 2 pending installs |
| or28-23244 | acozad | 10.11.211.38 | 2013-05-14 08:24 1 pending install |
| nm104-22506 | mschreiber | 10.11.195.42 | 2013-05-14 08:24 2 pending installs |
| nm216-22695 | awhite | 10.11.195.56 | 2013-05-14 08:24 4 pending installs |
| ms117-23567 | sryan | 10.11.148.233 | 2013-05-14 08:24 2 pending installs |
| studentservices-23863 | afolkening | 10.11.170.78 | 2013-05-14 08:23 3 pending installs |
| studentservices-23891 | tforest | 10.11.154.88 | 2013-05-14 08:23 5 pending installs |
| hs-rm103-18195 | ktroehler | 10.11.27.194 | 2013-05-14 08:23 1 package installed |
| ms219-23624 | amcguinness | 10.11.149.12 | 2013-05-14 08:22 3 pending installs |
| studentservices-23864 | scolucci | 10.11.211.70 | 2013-05-14 08:22 1 pending install |
| studentservices-23906 | tdisabatino | 10.11.154.34 | 2013-05-14 08:20 3 pending installs |

Open "http://munki.northmontschools.com/lookup/detail/a8:20:66:1a:72:55" in a new tab

# Munkiwebadmin

MunkiWebAdmin

munki.northmontschools.com/inventory/items?name=Flash%20Player&version=11.6.602.171

Reader

Apple iCloud Facebook Twitter Wikipedia Yahoo! News Popular dvr

MunkiWebAdmin

## Flash Player

### Machines 4

Show 20 entries

Search: flash player

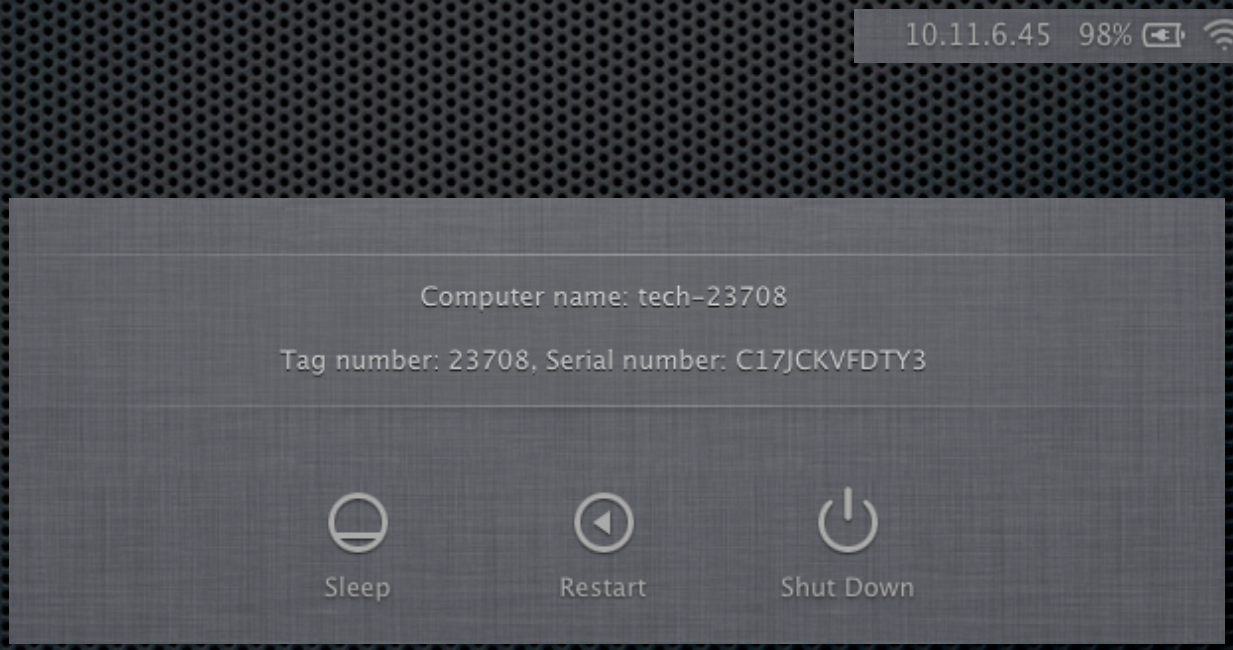
| | | | | |
|---|---|---|---|---|
| Flash Player | | | 9.0.45.0 | 1 |
| | | | 11.7.700.169 | 1288 |
| | | | 11.6.602.180 | 26 |
| | | | 11.6.602.171 | 4 |
| | | | 11.5.502.149 | 1 |
| | | | 11.5.502.110 | 1 |
| | | | 10.1.102.64 | 2 |
| 23107 | | Player.plugin | | Player.plugin |
| orcart3-23357 | 11.6.602.171 | com.macromedia.Flash Player.plugin | Flash Player | /Library/Internet Plug-Ins/Flash Player.plugin |

Showing 1 to 4 of 4 entries

← Previous 1 Next →

# Changes since initial deployment

- login window info (replaces cycling info area pre-10.7), deployed via puppet in updated machine naming script



# Changes since initial deployment

- ARD privileges rollback
  - DeployStudio's enable ARD step enables for all local users... and we have a non-admin, no password account
- printer administration - lpadmin group
  - staff can add printers at home on their assigned machine
  - any staff member can resume printing on any machine

# Changes since initial deployment

- lots of application updates
  - flash player, adobe reader, java mainly
- vpn configuration profile
  - launchagent to ensure vpn menu item is loaded
- we’re a school district and schools love to make end of the year videos on DVD

# Changes since initial deployment

- “critical” mode for munki updates (for attended installs)
  - turn “bug the heck out of the users” mode on/off
  - file on munki server is the controller

# Changes since initial deployment

preflight

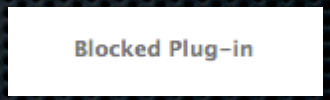
```
      munkirole="${munkirole}_${role}"
    fi
  fi
done

if [ "x${munkirole}" != "x" ]; then
  munkirole="${location}${munkirole}"
fi

if [ "x${swupdatecatalogurl}" != "x" ]; then
  munkirole="${munkirole}_test"
else
  swupdatecatalogurl="${swupdateurl}/index_production.sucatalog"
fi

if [ "x${swupdatecatalogurl}" != "x" ]; then          # should never be 'x' as if not a tester it's production
  /usr/bin/defaults write "${swupdatepref}" CatalogURL "${swupdatecatalogurl}"
  /usr/bin/defaults write "${swupdatepref}" ScheduleFrequency -int -1
  logger "configured software update's catalog url as ${swupdatecatalogurl}"
fi

if [ "x${munkirole}" != "x" ]; then
  /usr/bin/defaults write "${munkipref}" SoftwareRepoURL "${munkiurl}"
  /usr/bin/defaults write "${munkipref}" ClientIdentifier "${munkirole}"
  /usr/bin/defaults write "${munkipref}" SuppressAutoInstall -bool false
  /usr/bin/defaults write "${munkipref}" InstallAppleSoftwareUpdates -bool false
  if [ "x${pkgurl}" != "x" ]; then
    /usr/bin/defaults write [illegible] PackageURL "http://${pkgurl}/munki/repo/pkgs"
  else
    /usr/bin/defaults delete "${munkipref}" PackageURL
  fi
  if [ "x`curl ${munkiurl}/critical`" == "xYES" ]; then
    logger "critical flag is set... changing preferences to bug user"
    /usr/bin/defaults write "${munkipref}" SuppressUserNotification -bool false
    /usr/bin/defaults write "${munkipref}" DaysBetweenNotifications 0
    /usr/bin/defaults write "${munkipref}" SuppressStopButtonOnInstall -bool true
  else
    logger "critical flag is not set... changing preferences to nice mode"
    /usr/bin/defaults write "${munkipref}" SuppressUserNotification -bool true
    /usr/bin/defaults write "${munkipref}" DaysBetweenNotifications 1
    /usr/bin/defaults write "${munkipref}" SuppressStopButtonOnInstall -bool false
  fi
  logger "configured munki to use ${munkiurl} and manifest ${munkirole}, pkg[illegible] ${pkgurl}"
fi

exit 0
```

# What’s left

- nested manifests in munki
  - most manifests contain same applications
  - modify one manifest instead of 30+
- update munki preflight script to reenable installing Apple software updates
- set some users up as “test” users and do early app deployments to that group
  - set via checkbox in web help desk

# What's left

- put munki and reposado in the dmz
  - school is almost out
  - users won't be "on campus" but will get xprotect updates
  - uh oh...

So you probably want to see the physical result of replacing 1500+ iBooks, eMacs, Mac Minis, and iMacs with these new machines, right?

# The results

13” MacBook Pros 10.4/10.5

eMacs running 10.8

when eMacs go out to pasture

ready to go over the rainbow

# Questions?

# Supplemental info

- blog article with links to projects, code, etc.
  - http://seankaiser.com/redirects/psumacadmins2013/
- contact me
  - @seankaiser
  - seankaiser.com