

Grok Local MCX

And Managed Preferences In General

This outline was created for the PSU Mac Admins Conference, May 12th, 2011

Intro - Separate the Dirty from the Quick

Getting Reacquainted with Prefs

Pre-History: MacOSXHints and ye olde 'defaults write'
'Spray and Pray' - More of a craft than an art
Just Because You Can... (and even if you think you can't...)
The Directory Service is inside you!

What is LocalMCX?

Breakdown of all the moving parts

Refresher, Once-Often-Always-Never
Continued, User-Group-Computer-ComputerGroup
The Parental Controls Paradigm
Where the rubber meets the road
Better-than-we-did-yesterday practices™

Bootstrapping

Zero to Sixty

Start with just three steps - download, customize, and deploy
Apple's unfortunate endorsement message
Package one breakdown
Package two blow-by-blow
Season to taste - Word 2008/2011 default save format example

Take-Home

Where do we go from here?

Recap - Why, What, How

Q&A - Thanks!

Grok Local MCX

And Managed Preferences in General

Allister Banks, @sacrilicious - Point Consultants, NYC

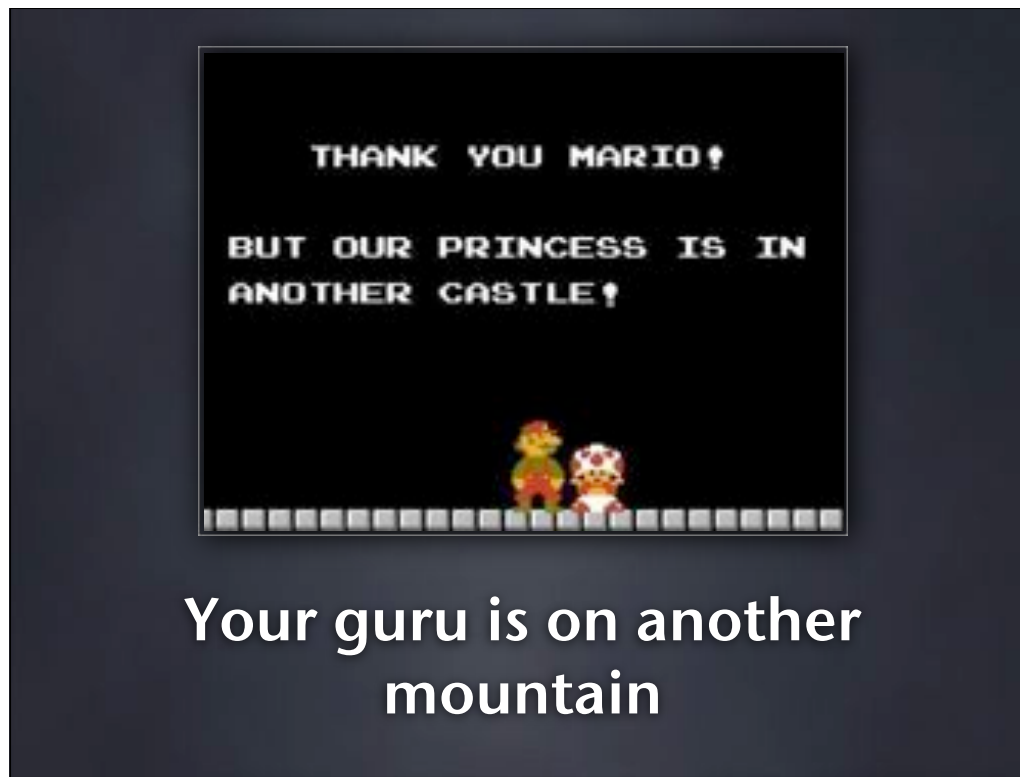
1

My name is Allister Banks and I'm lead engineer for an outsourced IT consulting company in NYC. For those who came to John DeTroye's talk, pardon me for covering much of the same ground. But even with as much as I learned by attending his talk, I'll be covering a specific subset of deploying Managed Preferences. And just to set some ground rules - the respect I want to show presenters whose sessions I'm attending means I want to be in 'pencils down' mode - which means:

Please, do not use your computers except for two things - taking notes or using the poll widget to submit your questions. I do, however, value your feedback immensely as it is critical for this to be as rewarding an experience as possible, so please don't hesitate and open yourselves up to the possibility of forgetting a question, just enter it when the spirit moves you.

One of the things that this poll widget is going to help us avoid is hands going up, and when the next slide has the answer to the question you're about to ask, the both of us are going to feel we're not being respectful of everyone's time. However, do not hold out hope I will cover the point or topic you're concerned about - put your question in the system and damn the torpedoes. And I can't be the judge the effectiveness of my communication skills from this end, so you'll be helping me a lot as well. As Chris Tucker's character said in Rush Hour, "Do you understand the words that are coming out of my mouth?" If I'm seeing question marks on peoples faces, I hope the next action you'll perform is submitting a question so my communication can improve.

Based on the feedback from my poll that comes in, plus q&a time at the end, hopefully we'll have enough of what you're here for to satisfy everyone, so let's get started. First a disclaimer,



Your guru is on another
mountain

2

Elvis has left the building. And I don't just mean John DeTroye went out to get a soda, I mean I am not, by any stretch of the imagination, the guru on this topic. All I've actually done is adapt the teachings(meaning blog posts) of Greg Neagle, and used his recent collaboration with Ed Marczak on a book for Apress called Enterprise Mac Managed Preferences as my sacred text.

[Write EMMP on chalk/dry-erase board]

So, a guru I am not, just an enthusiast who wanted a cleaner way to alter preferences. This is in part due to the fact that Microsoft deemed it necessary to make docx, pptx and xlsx, which are xml-enhanced and therefore less compatible formats the default for 2008 office and beyond, put out a confusing conversion tool. And I could've just come up with a hack to get that in my instaDMG images.

Ye Olde defaults write



3

In fact, if you guys were anything like me until recently, I would probably have read some quick and dirty hack on MacOSXHints on how to fix this ‘silently’ via the command-line and by gosh it worked on a test box and then I’d figure out some way to put it in production. Just to give you the scope of this practice, if you search MacOSXHints you’ll get 333 hits for the exact phrase “defaults write”. So obviously we have demand, just like in basic economics, and this site was filling that need for those who were brave enough to venture into the terminal. And then we come to the brute force way of affecting changes on the systems under your purview:

Spray & Pray

Less an art than a craft

- `/System/Library/User Template`
- Apple Remote Desktop
+ Task Server /
Send Unix Command
- etc.



© Threadless / m 2 designs

4

Again, this is not to say I am above the fray, as if I believe what I'm implementing is best practices, instead of just better-than-we-did-yesterday practices. But, to start, there are places in unix you are not supposed to go, and Apple has designated that the root `/System/Library/User Template` folder is one of them; that's why there's that illustrative 'prohibited' icon on it. When they ask you nicely not to mess around in there and you do, along comes an OS update that replaces the functionality you wanted and you're re-doing your work, or just otherwise violating the terms of the contract. Also it's not particularly convenient, because you need access to the system before the user logs in and takes that template and builds their own customization on top of it. And just in general, though, we know we'd rather not have to mess with the global template.

In the Managed Preferences book I'll be referring to throughout this talk, they admit they are not an Apple-approved guide, but while you may 'get away' with certain techniques, there's got to be a better way to do things, especially at scale. And a task server or just plain old Apple Remote Desktop and 'send unix command', paired with your favorite snippets, may just be enough of a reliable distribution method for you. ARD alone certainly had been getting me most of the way for a time. Even harder to admit is the amount of times I'm not breaking Apple's rules, but breaking my own. And that is when I copy a quote-unquote known-good plist from one of my systems to overwrite the same one on workstations I manage and just kind of hope I'm not unnecessarily exposing a fleet of machines to risk because of my laziness.

And doing filedrop of plists, as I call that last technique requires a certain amount of guesswork when it comes to timing, because swapping out a certain file at the wrong time could lead to application instability, hence the name of this section, "spray and pray".

I do this with just my local admin account in images still, and I don't want to sound too harsh, since I know that even revered packaging tools like the luggage has stanzas for targeting the global template. I'm just trying to get the point across that it's unfortunate when we make compromises instead of what we know is more warrantable and therefore supportable methods.

Just because you can... (and even if you think you can't)

- `/usr/libexec/PlistBuddy`
- `NSDictionary` via Scripting Bridges (e.g. `PyObjC`)
- ... but there is the 'approved' way

5

Most of what we pride ourselves on in our environment usually deals with infrastructure. So taking the time to optimize those facilities that often have low hurdles to implementation is a much better idea than veering off the path into global template or defaults write land. Another side note/slash/factoid is that the `defaults` command may not function in the same capacity going down the line, so even if you needed this manual or ad-hoc route, `PlistBuddy` or a scripting language that can alter the plist XML data using a bridge to `NSDictionary` (like `MacRuby` or `PyObjC`) is probably a better horse to back anyway.

[Write `/usr/libexec/PlistBuddy` and `NSDictionary` on chalk/dry-erase board]

And at this point, however many minutes I am into this presentation, I'm finally bringing up how Apple says you should apply preferences;

**Managed
Preferences
(aka MCX)
+
Directory Services
(e.g. AD, OD)**

6

via a facility called Managed Client for OS X, or MCX for short, heretofore officially referred to as managed preferences. And for storing/distributing them, the location any computing platform says is the ONE TRUE WAY to deploy settings for a fleet of computers is ... a Directory Service. Don't fear the reaper, centralized can mean consistent, and when you're doing these changes in one place they are certainly more audit-able if you're into that type of thing, and while that is stricter when it comes to accountability, there is an obvious efficiency gain. I'm going to connect the dots between defaults write and managed preferences over the next section, but what I want to move on to say now is that there are situations where distributing managed prefs from a directory service is impractical, either logistically or due to the political capital required. And I'm here to tell you fret not, unbound masses!

The Directory Service is *inside you!*



7

When the reign of netinfo, which had carried over from the days of NeXT finally ended in Leopard, we ended up with an equivalent to OS X Server's Open Directory on a standard Client OS install. The long and the short of that is: you don't need to hook into a centralized service to manage preferences in the same manner using the same tools. That means Workgroup Manager can help you edit your plist, designate how the preference is going to be applied, and once you reach out and touch your fleet, all those prefs will be applied – locally. This fulfills a bunch of things you would have needed binding and therefore access to the LDAP domain to do centrally, and I have personally done what I can to make this as simple to implement as possible with a project on github we'll go into later. The goal state is my InstaDMG built-image can be restored anywhere in the world and the customizations I want applied to that image happen with zero manual interaction and no connection to the home office needed. And tweaking or expanding what you manage can be as easy as opening Workgroup Manager, changing it in one place, and distributing it to as many similarly configured places as you see fit.

Moving Parts

Adverbs

-

Nouns

- Never
- Once
- Often
- Always

- User
- Group
- Computer
- ComputerGroup

8

<DRINK WATER>

So – let’s get in to the glossary of terms we’ll be dealing with when connecting the dots between defaults write and directory service–managed preferences, and it starts out with two sets of four parts each that will interact with what we’re trying to manage

On this side we have the Nouns, the person place or thing, and on the other we have the Adverbs, the timing with which it’s going to be enforced or applied with. Now let’s talk about a theoretical example of doing things the old way: running a defaults write to turn off the silly traffic light buttons in iTunes

Moving Parts

Adverbs

-

Nouns

- Once
- User

9

we're affecting that change once and that preference is stored in our user folder, so its being managed on the user level. I want to remind us of my goal of managing the save formats in Office 2008, and we'll get to that later on, after I've revealed my deployment-ready solution.

Moving Parts Adverbs

- Never
- Once
- Often
- Always
- User
- Group
- Computer
- ComputerGroup

10

Moving on to the fast and hard rules of this game

Moving Parts

Adverbs

- Never
- Once
- Often
- Always

11

let's throw never out, because when I tell my boss that we can have that meeting never, or 'look at your calendar, how does never work for you', he knows nothing is going to happen. Reclaim that space in your brain now, and we're left with three. We just affected iTunes with the Once adverb in our example, because we could run the same command as our currently-logged in user to reverse or change the preference we specified. Once means malleable or changeable.

Always and Often are the more powerful adverbs, and touching on the more obvious of the two, Always manages Always – no touch, you break you buy(okay, you can't break). If, as an admin, you are the bastard operator from hell, you call the customers 'users' and lock the preferences up tight.

So then we're left with the special case, often – which is hidden from the WorkgroupManager GUI, but here's an applicable scenario for when you'd want to use it: Say you've decided how the dock should be setup and a teacher wants the class to be able to rearrange things for this one period. Often would theoretically reapply the preference on every login, ensuring we start each session with the preferences admins intended.

Last thing about adverbs, and especially because Once is so attractive; Be wary of updating or tweaking the preferences you're managing Once – when workstations receive and apply that update, they have no memory of what happened before – it will therefore affect a change and undo customizations users may have set since it was first applied! A bad thing™

Moving Parts Nouns

- Never
- Once
- Often
- Always

12

Now when I said the LDAP structure of Open Directory is present in a OS client install, we can talk about these other nouns that are present on each workstation, and user is straightforward so on to Group and Computer. We all know what the default group is for local users or on the server, which is staff. Looking through the LDAP structure we want to refer to this object of a user or a group, and computers as a whole are also helpful, but Active Directory takes that computer name and ties it to DNS, which makes interacting with it a relatively heavier process, whereas on the Mac there's three different names you can interact with on a workstation so changes can be more granular and lighter-weight and precise. And internally when we're talking about localMCX we're even less concerned with a specific name, because we don't answer to the entire domain, just the local system itself. And another optimization for Macs is the grouping of computers into another object called a computergroup, so many computers can belong to one or many computer groups. Which can get confusing, and you do need to be equally wary of when there is a conflict of managing the same preference in multiple nouns with different adverbs. We'll get to the bottom of why smoke is coming out of people's ears in Q&A, thank you kindly for your patience.

The Parental Controls Paradigm



13

So far this has been sounding like a good amount to learn, and hard work in general for those not accustomed to going into Workgroup Manager for anything except adding users, but there's an analogue to LocalMCX that has been staring us in the face from SystemPreferences for quite some time: Parental Controls. Like certain common uses of defaults write, it operates at the user level but in a location that's not accessible to the user (otherwise it wouldn't be all that enforceable), and it is not compatible with Managed preferences that come from a directory service.



**Where
it's at**

14

So something is happening behind the scenes, and it's finally time to dive all the way in to what LDAP looks like on the client OS, and the three directories that are the key players in MCX:

Where the nouns be at

```
bash-3.2$ cd /private/var/db/dslocal/nodes/
bash-3.2$ sudo ls -al Default/
Password:
total 8
drwx----- 11 root wheel 374 Sep  4 2010 .
drwxr-xr-x  3 root wheel 182 Jul  2 2009 ..
drwx----- 18 root wheel 348 Jul  2 2009 aliases
drwx----- 13 root wheel 442 Sep  6 2010 computer_lists
drwx----- 13 root wheel 442 Sep  6 2010 computergroups
drwx-----  5 root wheel 178 Sep  5 2010 computers
drwxr-xr-x  7 root wheel 238 Oct 17 2010 config
drwx----- 95 root wheel 3230 Apr  4 14:02 groups
drwx-----  3 root wheel 182 Jul  2 2009 networks
drwx-----  3 root wheel 182 Sep  4 2010 presets_computer_groups
drwx----- 61 root wheel 2074 Apr  4 14:07 users
```

15

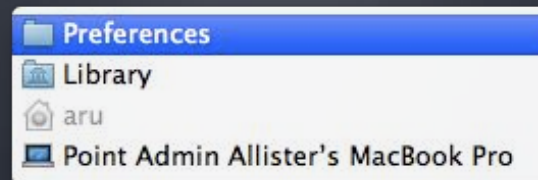
1. (/Private)/var/db/dslocal/nodes/ is where our nouns stored on our local system actually live, and in the case of Parental Controls, preferences we want to manage for a particular user are nested in their user record

Now that's a biggie right there, and will be the main stage LocalMCX plays it's part on, but I just wanted to mention you can punt on the whole rest of this presentation by doing everything with Parental Controls – but please don't. It's limiting to be too micro and specific about what the user is we're going to, for example, apply itunes preferences to. Using the computer group noun is the practice LocalMCX is built on, which we'll follow up with shortly. Moving on through the other directories:

Once policy done been applied



- Always



- Once
- Often

16

/Library/Managed Preferences are where preferences are cached that are managed for any noun with the adverb “always”

and ~, or any given users homes directory's /Library/Preferences is where preferences managed for any noun with the adverb once or often would be stored, just like with ye olde defaults write So OK – we have why we want to use LocalMCX, what the moving parts in play and directories are that are where this takes place in a system, but I've got a bit of a bombshell to drop now: I have been privy to non-sensitive footage from deep inside Apple that proves...

Circa August 2009



Intro

Video/Audio:

From the producers of the iPod, iPhone and iPad come...

Local Management Policy

Setup Steps

- Create local computer account
- Set policy with Workgroup Manager
- Export policy with `dscld -mcxexport`

Main thrust

Video/Audio:

Shiny `dscld -mcxexport`, doing the heavy lifting

Automating Deployment of Local Managed Preferences in Mac OS X

17

This is totally apple-authorized. Just be mindful of the fact that this was, y'know, 2009, 10.5, or to put it another way, yesterday. And I'm a big proponent of not 'best practices', but better-than-we-did-yesterday practices. Still, the demonstration given is LocalMCX and uses the same tools you can. That video can be found here:

[http://seminars.apple.com/
seminaronline/mcx/apple/
index.html?s=301](http://seminars.apple.com/seminaronline/mcx/apple/index.html?s=301)

18

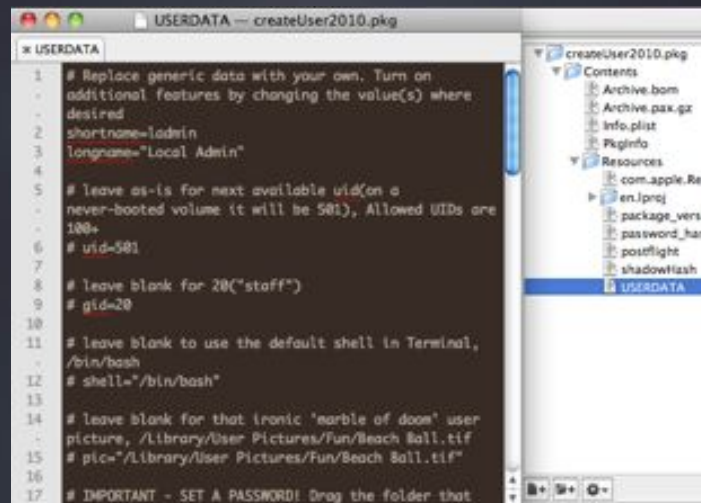
But before I touch on the specifics of why the technique we'll be using differs significantly from this one, I'll get us started on the three-step process to deploy MCX via packages, which was intended for and tested with InstaDMG.



19

So, step one, we go to github.com and download from my repository. <https://github.com/arubdesu/One-Stop-LocalMCX/zipball/luggs>

If you're familiar with source code branches and savvy with theLuggage there is a luggs branch, but since everyone just saw that talk in 302 I'm not going to touch upon what that entails. I've only half-implemented it anyway, just throwing it out there. What you'll see in the zip when you're downloading from THIS branch is two packages, createUser2010 and LocalMCXstarter

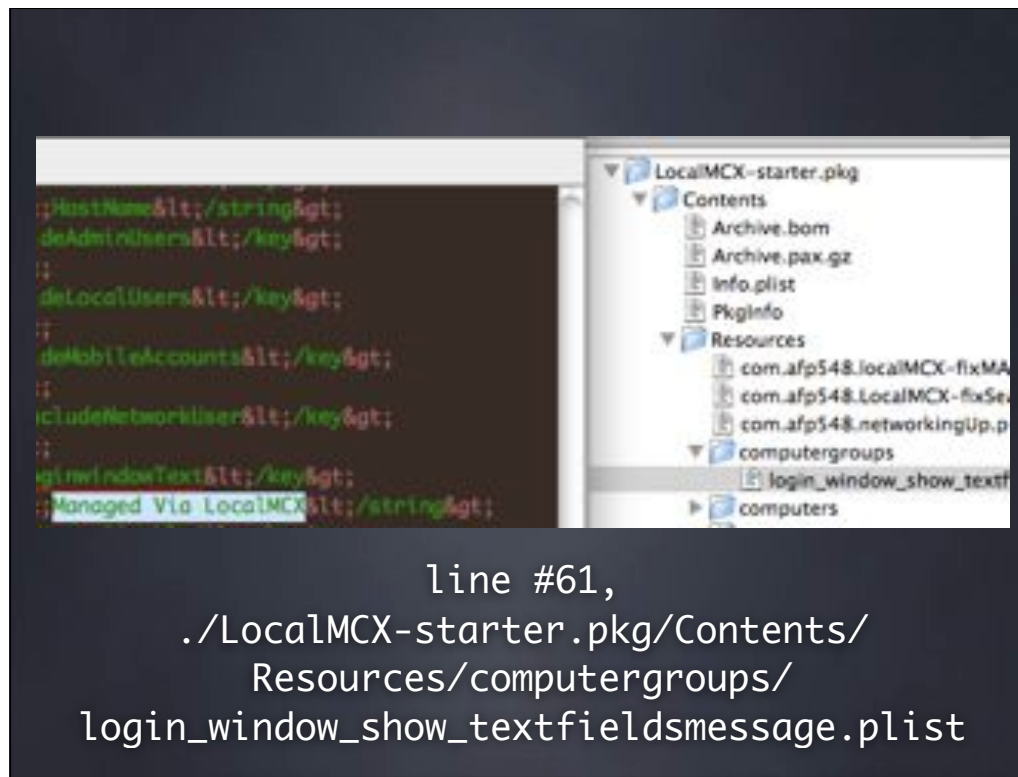


```
1 # Replace generic data with your own. Turn on
2 additional features by changing the value(s) where
3 desired
4 shortname=ladmin
5 longname="Local Admin"
6
7 # Leave as-is for next available uid(on a
8 never-booted volume it will be 501), Allowed UIDs are
9 100+
10 # uid=501
11
12 # leave blank for 20("staff")
13 # gid=20
14
15 # leave blank to use the default shell in Terminal,
16 /bin/bash
17 # shell="/bin/bash"
18
19 # leave blank for that ironic 'marble of doom' user
20 picture, /Library/User Pictures/Fun/Beach Ball.tif
21 # pic="/Library/User Pictures/Fun/Beach Ball.tif"
22
23 # IMPORTANT - SET A PASSWORD! Drag the folder that
```

If you've used createUser with InstaDMG, it's... familiar

20

Since we're all savvy and familiar with InstaDMG and want to try this in an image, I'm breaking this step down into three parts since we'll immediately start getting this customized for our environment: We modify the USERDATA file located within CreateUser2010.pkg/Contents/Resources so it creates a Local MCX admin in a non-default node among other options, and we create the password we'd like this local admin user to have with the bundled shadowhash tool



21

And just one last thing you probably want to customize is what the managed preference causes the login window to look like, by default line 61 of the example managed preference `login_window_show_textfieldsmessage.plist` just has it read `Managed Via LocalMCX`, so since we're theoretically taking the time to build an image with this we'll spiffy that up as well. All of the steps above have generic defaults in place, but the only barrier to entry is editing text, so why not.

Step three is we load it into our `instaDMG` build train, and when that image is restored and booted you will see the login screen change from a default list to of the users on the system to empty text fields with our message on top.

Not horrible, right? Minus the `instaDMG` robot building an image for us that's less than 5 minutes to have a working `localMCX` implementation.

Now, back to pointing out where Apple's steps are less than optimal



- 'localhost' computer w/127.0.0.1 IP
- -mcximport various into computer
- ... in the Default node

22

First, they create a computer noun called localhost in the "Default" node and gives it the IP address 127.0.0.1. Which was probably fine in Leopard, until Snow Leopard came along and has a computer object called localhost in the Default node with that same loopback IP, but also Kerberos and IPv6 info in it. Then they associate the management of a bunch of prefs to that one record, and finish by scripting the creation of a fresh record at startup (besides the fact that this is a computer record that's good, totally valid) and importing the MCX data to associate it with the local computer (kind of eggs in a basket). All of that is intended to occur at startup time, and while I haven't tried it, I hope at this point it would fail instead of harming what the system intended to use there. Greg Neagle writes, in *Enterprise Mac Managed Preferences*, which is the book I used as the primary reference for this talk, randomly settings wouldn't apply, which he traced to the deletion of his local computer record that had the managed prefs nested inside of it. He deduced it was due to the system coming into conflict with a record in that space, and he also saw his logs filled with errors regarding conflicts with directory services that were used for authentication but not MCX, since every mac can utilize multiple directory services. Just like we all have opinions about anything with a chip or a cable, Default is not where it's at, and of course the state of the art is going to make advancements, but it was still a good video for the clear endorsement of LocalMCX. Now I can move on to why Greg Neagle, and by extension my project made several different decisions, and here they are in the order of how the quickstart I previously went through applied them. But first, a quick intermission to let our brains ooze out of our ears for a second

Intermission



Create the MCX 'node', and a hidden mcxadmin

```
205 if [ $localmcx -eq 1 ]; then
206
207     /bin/mkdir -p "$LocalMCX_DB/computers"
208     /bin/mkdir "$LocalMCX_DB/computergroups"
209     /usr/sbin/chown -R 0:0 $LocalMCX_DB
210     /bin/chmod 700 $LocalMCX_DB
211
212     $serviceCmd $LocalMCX_DB $suffix -create $mcxAdminPath || exit 1
213     $serviceCmd $LocalMCX_DB $suffix -create $mcxAdminPath realName "$MCX_Admin"
214     $serviceCmd $LocalMCX_DB $suffix -create $mcxAdminPath gid 80 # only group necessary is admin,
    # instead of default 'staff'
215     $serviceCmd $LocalMCX_DB $suffix -create $mcxAdminPath uidKey 444
216     $serviceCmd $LocalMCX_DB $suffix -create $mcxAdminPath home "${TARGET_DIR}/tmp"
217     $serviceCmd $LocalMCX_DB $suffix -merge $mcxAdminPath authentication_authority ";ShadowHash;"
218     $serviceCmd $LocalMCX_DB $suffix -create $mcxAdminPath passwd ""
219     $serviceCmd $LocalMCX_DB $suffix -create $mcxAdminPath shell "/dev/null"
220
221     # auto-generate a genuid for mcxAdmin
222     MCXAdminGenUID= $serviceCmd $LocalMCX_DB $suffix --read $mcxAdminPath generateduidawk '{print
    $2}'
223
224     # use same password for mcxAdmin's shadow hash file and set perms
225     /bin/in "${TARGET_DIR}/var/db/shadow/hash/$MCXAdminGenUID"
226     "${TARGET_DIR}/var/db/shadow/hash/$MCXAdminGenUID"
227     /bin/chmod 600 "${TARGET_DIR}/var/db/shadow/hash/$MCXAdminGenUID"
228
229     # hide mcxAdmin
230     /usr/bin/defaults write "${TARGET_DIR}/Library/Preferences/com.apple.loginwindow"
    hiddenUserlist -array-add mcxAdmin
231 fi
```

24

So going through that three-step quickstart, I'll break down how the Greg Neagle popularized method of LocalMCX manifests itself with those two packages in my github-hosted project. First, we talked about the first directory that MCX interacts with, which is (/private)/var/db/dslocal/nodes, and there is that Default node there for local users and groups. We want a node of our own, and Mac OS X obliges by allowing us to create a new MCX node with our package, and not only do we create an admin user to modify records in there, we symlink the password to be the same as our regular local admin, so they stay in sync. I'll be referring to this new node, which has the full path (/private)/var/db/dslocal/nodes/MCX as just the MCX directory. Creating a directory from the command line in a script isn't the biggest of deals, but creating users, especially on an un-booted system like instadmg operates on, takes heavy machinery like the directory service command line utility, aka dscl, hallowed be thy name. Okay, so that's the first pkg, createUser2010. Please look through the postflight script contained inside it if you're not familiar with how createUser works, and this afternoon will be instaDMG-palooza if we can store up any questions for then so I hopefully have something to talk about for 3 hours.

The Real Nittany-Gritty

```
ditto -V "${PKG_DIR}/computergroups"  
"${LocalMCX_DB}/computergroups"
```

(Including login_window_show_textfieldsmessage.plist)

LocalMCX-fixSearchPath.sh

fixComputerRecord.sh

25

Now pkg two is where the lion's share of the localMCX specific goodness is, and while CreateUser in my mind is the archetypal payload-free package, I like to think of localMCXStarter as a textbook first-boot pkg, but I'll get to what I mean by that in a second. So I want to apply my login window preference, and I've decided to enforce it with the adverb always on the noun a computergroup called, equally un-inventively, login_window_show_textfieldsmessage.plist, so that is seeded by the localMCXstarter package in the MCX/computergroups directory. This is the same as we could do with Workgroup Manager, and I encourage maintaining and adding computergroups with WGM from this point forward. Managing a preference computer-wide means every user on the computer gets it as well, while that isn't obvious in this specific example, when we get to office2008 it becomes clear.

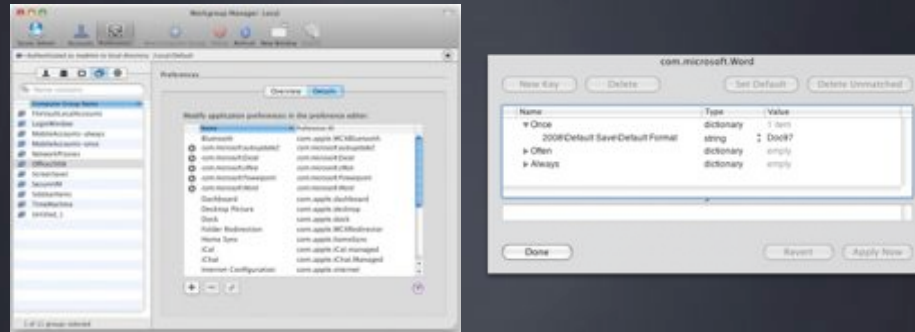
Now back to the rest of what the package does, with 2 launch daemons and a launchagent, And what are launchD jobs without actions to perform, so these are the scripts I'm going to describe now, localmcx-fixSearchPath and fixComputerRecord.

```
LocalMCX-fixSearchPath.sh
fixComputerRecord.sh
```

26

To both breakdown the order of operation of what was seeded by this firstboot, and why it acts the way it does, we started after createUser setup the MCX node for us. That is so we're in a different directory than the one named, inventively, 'Default'. we need to do the equivalent of adding that MCX node to our searchpath with Directory Utility, and again it's our friend dscl put to use in a launchdaemon-called script, and this helps the machine know to read in records from that directory and check that they apply to us. But before when I was talking about computergroup, we don't have any computers that are members this group yet. Our package seeds both a local_desktop and a local_laptop record in the MCX/computers directory. And the reason seeding two computer records is that therefore, through group membership, certain computergroups can contain laptop-specific settings. So this is just in case I didn't want to apply the same management settings that's used on desktops, and gives me granularity of what gets applied where, but I only need to deploy one package for either. This final script I call with a launchdaemon puts the correct MAC address in the applicable computer record – just so you know what the logic is, if system profiler finds a model name that contains 'book', its a laptop, and it wipes the desktop record. Then, for the coup de grace, it checks that there are no active user sessions, since there shouldn't be since we just barely turned on the computer, and it then kills the loginWindow so our plist immediately gets read in and applied, and voila! Managed preferences, applied.

MS Word 2008 Example

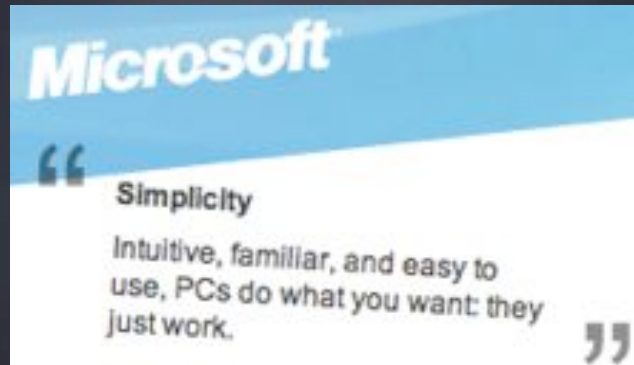


<http://managingosx.wordpress.com/2008/02/13/managing-office-2008/>

27

Phew! Thank you all for staying with me through all of that. Now we're going to add a computergroup to contain Office2011 default save prefs on a theoretical image deployed with our packages, and I'm using graphics from 2008 but the process is exactly the same. On our machine that has localMCX set up we install the server admin tools and open WGM, navigate to the MCX directory, and authenticate with the username mcxadmin with the local admin's password. Then we open word and change the default save format to just doc instead of docx and quit the program. With accounts selected at the top of WGM we go to computergroups and hit the plus sign to add a word_save_format group, I like to use hyphens or camelCase instead of spaces to make it easier to navigate in the commandline. You want to add both the local_computer accounts, laptop and desktop, to this computergroup. Then we switch to Preferences and the details tab to add the com.microsoft.word.plist file from your library to the computergroup we created. Now you want to modify what you imported so we hit the pencil edit icon and clean out all the other stuff imported besides the one called 14\Default Save\Default Format which corresponds to Doc97, and if it's not in the Once section you can drag-drop it in there. And when you want to roll out this plist to the rest of your machines that are using localMCX, you just need to copy out the word_save_format plist and drop it into the other machines MCX/computergroups directory. Donezo

Take-home



28

If there's a single mistruth that Mac admins dislike above any other, it's the phrase 'it just works'. Vendors make promises, and admins are the ones caught holding the bag, which leads to the less desirable aspects of the job. After having been around the block a couple times, theory can sometimes be presented in a way that is so wonderfully vague that when I need to get the work done its all the more challenging that implementation details are near-impossible to get right.

If nothing else, I hope this project gives you a glimpse at the moving parts involved with MCX, and empowers you to be able to adapt what I've provided for your needs without reinventing the wheel.

Why

- Easy as Workgroup Manager
 - Flexible
 - Supported

29

To recap, why we'd want to begin to use this method is that it's as easy to use as WGM, but flexible enough that whether or not your machines interact with directory services you can still take advantage of MCX properly. There are ways that involve scripting and twiddling too many knobs and levers, and are certainly not Apple-endorsed methods of getting the management you want delivered to the workstations under your purview. And as with any technique that makes admin's lives easier, the ones they utilize and support with their flavor of directory services is the one they're less likely to break, and more likely to be allowed to flourish.

What

- Nouns (like users) and Adverbs (like once)
- The three directories:
 - `/private/var/db/dslocal/nodes/`
 - `/Library/Managed Preferences`
 - `~/Library/Preferences`

30

A recap of what MCX is when coming from ANY node is that it is essentially plist files that contain management settings, which I broke down with the grammar terms nouns for objects like user/group/computers and computergroups, and adverbs for when we'd like to see a given preference managed on that object. The three directories I referred to as the stomping grounds of MCX were 1.

that long `/private/var/db/dslocal/nodes` which contains a `Default/users` folder for local users on the system

2, the `/Library/managed preferences` folder, which will contain a cache of plists when there is management applied to any part of the system with the adverb always

And 3, the regular users library folder will be affected by management applied with the adverb once or often

How

31

And taking it on home to how my localMCX project injects everything we need for a working implementation, I broke it down into the two packages involved. CreateUser2010 set up the local MCX node as a directory with an mcxadmin user. And localMCXstarter seeded the appropriate directories with the managed preference computergroup plist, the local computer record, and the scripts to kick off applying the managed pref, which alters the login window. I then explained the scenario of creating a new computergroup to house managed prefs for Microsoft Word from scratch, and hinted at how you might distribute it to the rest of the computers you'd like it applied to.

Thanks!

Allister Banks, @sacrilicious - Point Consultants, NYC

32

And that is all I've got. Thank you so much for your patience and attention, and please put feedback into the widget so I can get much better next time. Questions?