Creating Neutered Administrators

Session P223

Macworld Boston 2005

Dave Pooser
Alford Media Services
ACSA 10.3

Mike Sebastian
Splash of Color
ACSA 10.3

What We'll Cover

What We'll Cover

- Securing the system
  - Setting an Open Firmware password
  - Adding a hidden administrator account

And more…

- Adding capabilities to standard users
  - Editing `/etc/authorization`
  - Changing permissions using Access Control Lists
  - Editing `/etc/sudoers`

Why You Need To Know This

Murphy's Laws of System Administration:

- Everything a user CAN change, he WILL change
- Any changes will break something important
  - At the last second, so you have no time to fix it
  - As far away from you as possible
  - The first things broken will be the tools you need to fix systems remotely

And somehow, it's still YOUR fault!

Why You Need To Know This

- Apple doesn’t offer granular permissions
- Two options:
  - Local administrators rule their own boxes
  - Standard users can’t even change time zone
- We’re looking for a middle ground…

Where this is useful

- Road warriors– laptop users need control over network, time zone, and similar
- Remote sites/branch offices– may not have IT staff on hand
- Management– sometimes the folks who sign the checks want to feel independent

Securing the system

- Setting up an Open Firmware password
  - Prevents users' changing boot device
  - Prevents booting in single user mode
  - Prevents startup in Target Disk mode
- Easily defeated; just add or remove RAM

Demo 1

- Add an Open Firmware password using Apple's Open Firmware Password 1.1
  - Get it off your install disk in / Applications/Utilities
  - 1.0.2 will **not** work with Tiger!
  - See <http://docs.info.apple.com/ article.html?artnum=106482> for details

Securing the system

- Adding a hidden administrator account
  - A “back door” if primary admin cracked
  - Hidden to avoid user confusion
  - Can be disguised as (unused) system user to minimize chance of detection
    - e.g. mailman, cyrus or postfix users
  - Easily detected in NetInfo Manager

Demo II

- Use NetInfo Manager to delete user "cyrus"
- Create new user "cyrus" via Accounts pane
- Edit user "cyrus" with NetInfo Manager:
  - change UID to 98; change GID to 80
  - change home to `/var/imap`
  - delete SharedDir

Demo II

- Using Terminal:

```
sudo mv /Users/cyrus /var/imap

sudo chown -R 98 /var/imap
```

- Log out
- On login, use down arrow to select user; then Option-Enter to get to user/password entry blanks
- Log in as cyrus

Upgrading users

- Create a group for users who'll have some administrative rights
  - Include administrators!
- Reassign some admin group privileges to this powerusers group

Upgrading users

- `/etc/authorization` is a collection of rights and rules
  - Example: the right *system.burn* matches the rule *allow*; by default anyone can burn CDs
  - Open `/etc/authorization` with Property List Editor (from Xcode) to view all rights and rules

New: 32% more rights!

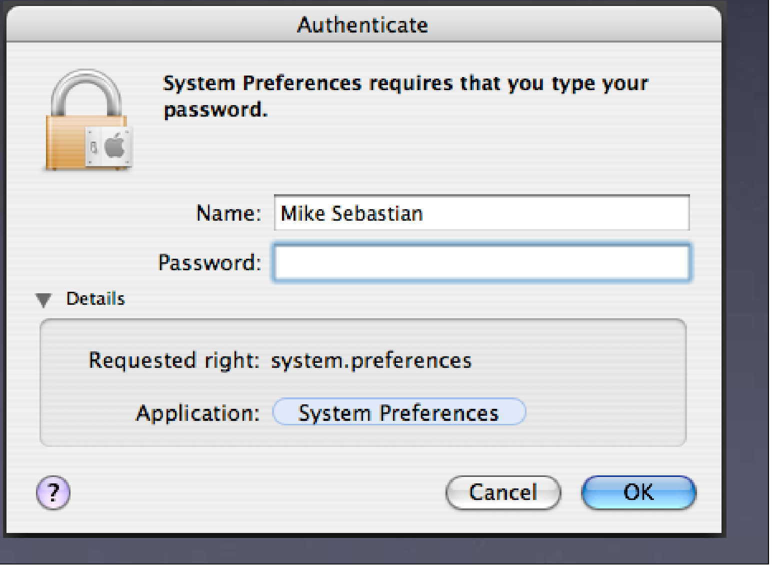
- com.apple.activitymonitor.kill
- com.apple.builtin.confirm-access
- com.apple.builtin.confirm-access-password
- com.apple.builtin.generic-new-passphrase
- com.apple.builtin.generic-unlock
- com.apple.Safari.parental-controls
- system.preferences.accessibility
- system.preferences.accounts
- system.services.directory.configure

Key additions:

- com.apple.activitymonitor.kill
  - Pro: Great for remote troubleshooting so the user can kill out-of-control processes
  - Con: Great for killing ARD and VNC daemons
- system.preferences.accounts
  - Only affects Accounts preference pane
  - It’s a start– now how about the other panes?
- system.services.directory.configure
  - Need this right AND system.preferences right to change Directory Access configurations

Upgrading users

- To expand users, first identify the capabilities needed
- The Authenticate dialog box hides that information under Details; hit the disclosure triangle to see
  - For instance, to unlock most preference panes the requested right is system.preferences



Authenticate

System Preferences requires that you type your password.

Name: Mike Sebastian

Password:

Details

Requested right: system.preferences

Application: System Preferences

Demo III

- Using NetInfo Manager
  - Duplicate the admin group
  - Change the name from “admin copy” to “powerusers” and the GID to any unused GID <500
  - Add the users you wish to enhance

Demo III

- Make `/etc/authorization` editable
- Open `/etc/authorization` with Property List Editor
- Find system.preferences and change the group value from “admin” to “powerusers”
- Save changes and set `/etc/authorization` permissions back to root:admin rw-r--r--

Demo III

- Log back in as enhanced user to verify access to… all system panes?
  - Except for Accounts– that’s another right
  - But including Startup Disk, so you can boot off another drive…

“Danger, Will Robinson!”

Re-restricting users

- With the system.preferences right an all-or-nothing change, we need another way to lock the user out of some preference panes
  - Here’s where ACLs come in handy
    - Or you could use `chmod`
    - But with `chmod`, running Repair Permissions will undo all this work…
- Dangerous panes: Classic, Energy Saver, Security, Sharing and Startup Disk

Access Control Lists

- Tradition UNIX permissions: read, write, execute
- Only three user classes: User, Group, Other
- What if I want to give two groups different levels of access?
  - Nesting folders within folders
  - Not particularly flexible

Access Control Lists

- ACLs are composed of Access Control Entries
  - ACEs are applied in order; the first matching rule is applied and later rules are ignored
  - ACEs can allow or deny specified actions by users or by groups
  - ACLs can apply to files or folders

Access Control Lists

- Tiger Server supports ACLs by default
  - Workgroup Manager gives GUI interface to set ACLs
- Tiger Client has ACLs turned off
  - Activate ACLs on volume using `fsaclctl` command
  - Add/edit ACLs from Terminal using `chmod`

Demo IV

- Using Workgroup Manager
  - In Server Admin Tools; download from Apple or find on CD
  - Select View Directories from Server menu
  - Create a new group named “uppity”
  - Add your power users but NOT administrators

Demo IV

- Launch Terminal and type

  ```
  sudo /usr/sbin/fsaclctl -p / -e
  ```

  to enable ACLs on the boot volume

-
  ```
  cd /System/Library/PreferencePanes
  ```

-
  ```
  chmod -R +a “uppity deny
  read,execute” SharingPref.prefPane/
  ```

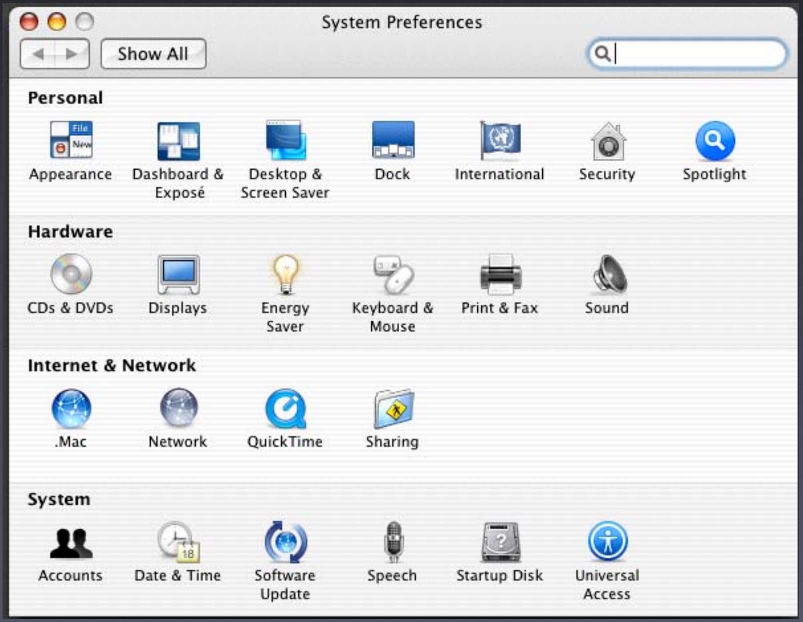
  to keep members of the uppity group from launching that prefpane

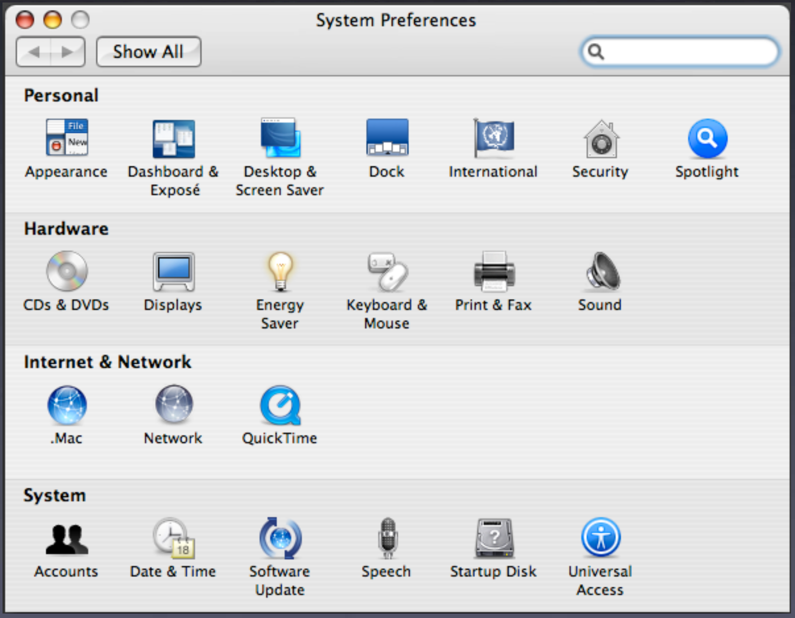
Demo IV

- Type `ls -l` to see that each file/folder with ACLs attached is marked by a “+”
- `ls -le SharingPref.prefPane` will show the contents of the attached access list
- Unlike normal changes using chmod, ACL changes are not affected by Disk Utility’s Repair Permissions option

Demo IV

- The changed pane is visible to admins…
  …but not to our power users





Permissions Tweaks

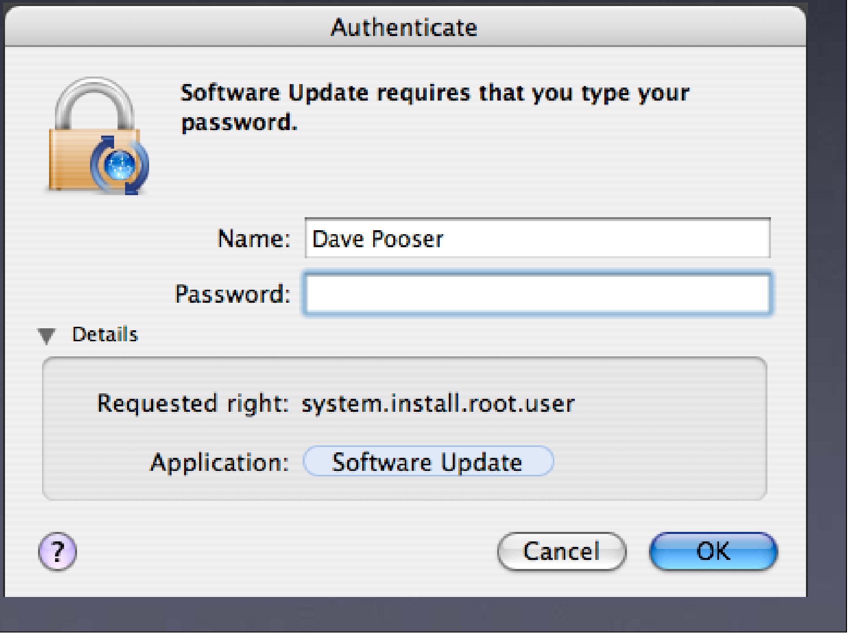
- By default, /Applications admin-writeable
  - Use ACLs to give powerusers group `add_file` and `add_subdirectory` permissions for drag-and-drop installs
    - (Microsoft Office, OmniWeb…)
    - Why not ~/Applications? Possible version conflicts
    - Licensing compliance may be problem

Editing `/etc/sudoers`

- /etc/sudoers is a list of users and groups allowed to run commands as root
  - Can allow some users to run any command (by default the admin group)
  - Can also allow users to run a specific list of commands…

Editing `/etc/sudoers`

- Example: You want power users to be able to run Software Update
- The Software Update GUI requires admin privileges
  - Specifically the *system.install.root.user* right…



Authenticate

Software Update requires that you type your password.

Name: Dave Pooser

Password:

Details

Requested right: system.install.root.user

Application: Software Update

Editing `/etc/sudoers`

- So why not edit `/etc/authorization` to give powerusers access to that right?
  - Because then they can install any package
    - As root
    - Including pre/postflight scripts

In other words, they could run any script they chose *as root*.

Editing `/etc/sudoers`

- Instead, edit `/etc/sudoers` to give the powerusers group permission to run `softwareupdate`
  - But be careful!
    - Use full path: `/usr/bin/softwareupdate`
    - Make sure the parent directory and the binary are only writeable by root

Demo V

- Use `sudo visudo` to edit `/etc/sudoers`
    - Feel free to change your editor first: `export EDITOR=/usr/bin/pico`
- Add a line as follows:

```
%powerusers  ALL=NOPASSWD: /usr/sbin/softwareupdate
```

- Translation: Members of the group powerusers can `sudo` to run `/usr/sbin/softwareupdate` as root without a password

Demo V

- Log out and log back in as the enhanced user
- Open Terminal and type:

  ```
  sudo /usr/sbin/softwareupdate -i -a
  ```

  - Translation: Run Software Update and install all updates
- Can also be created as a one-line script; make it a .command file to have a double-clickable option for Terminal-phobic users

Synopsis

- Secure the system– Open Firmware is key
- Create a powerusers group as admin-lite
- Give powerusers rights as needed
- Restrict dangerous prefpanes with ACLs
- Use `/etc/sudoers` for specific functions

Resources

Latest Presentation
iDisk: msebastian
Go > iDisk > Other User’s Public Folder

Dave Pooser
geekboy@pooserville.com

Mike Sebastian
mike@splashofcolor.com

Thank You!

Creating Neutered Administrators

Session P223

Macworld Boston 2005

Dave Pooser
Alford Media Services
ACSA 10.3

Mike Sebastian
Splash of Color
ACSA 10.3