Rights and Rules Demystified

Session M255

Macworld San Francisco 2005

David Pugh & Jan Stewart

University of Michigan

Apple Certified System Admistrators 10.3

Apple Certifications for 10.3.x

Course Names	Certification Level	Length (days)
Mac OS X Help Desk Essentials	ACHDS	3
Mac OS X Server Essentials	ACTC (with Help Desk Essentials)	4
System Administration of Mac OS X Clients	Apple Certified System Administrator	5 days each course
System Administration using Mac OS X Server		

train.apple.com • 800-848-6398

You have the right to tweak your system. Apple made it a rule.

What We'll Cover

- Authentication vs. Authorization
- Changing Authorization Policies
- Property Lists (plists)
- Examples

Why You Need To Know This

- You'll learn how the authorization system works
- You'll learn how to modify plists, opening up numerous hidden customization possibilities
- You'll learn how to set up finer grained access controls

Where this is useful

- Computer lab machines
- Shared office machines
- Desktop machines requiring higher security
- Home machines used by a wide age range of children

Authentication and Authorization

Authentication

- Are you authentic? = Are you who you claim to be?
- Examples:
  - Local password
  - Smart Card
  - Biometrics

Authorization

- Are you authorized? = What are you allowed to do?
- Examples:
  - Ownership
  - Unix group membership

Authentication & Authorization

- A Passport is AUTHENTICATION - it proves that you are who you say you are, but does not AUTHORIZE you to work in the U.S.
- A Social Security Card AUTHORIZES you to work in the U.S., but is insufficient without proper AUTHENTICATION

Authentication Policies

- Set in Directory Access
  - LDAP
  - Active Directory
- PAM

DEMO

Setting Authentication Policies In Directory Access

http://www.umich.edu/~jhstew/umldapv3/

Authorization Policies

- Set in /etc/authorization
- Contains two main sections
  - Rights
  - Rules

Generally speaking...

- To open up access:

  Modify /etc/authorization

- To restrict existing access:

  Modify MCX settings through the "Limitations" tab on the Accounts preference pane, or with the "Preferences" tab in Workgroup Manager *(see session M235)*

Rights

- What are you attempting to do? (Actions)
- Examples:

 Log in

 Unlock a preference pane

 Install an application

Rules

- A rule is just a predefined way to check if certain conditions are met
- Think of it like a macro
- Examples:
  - Is this the user currently logged in?
  - Is this user an admin?

The Password Box

The Password Box

/etc/authorization

rights

system.login.screensaver
system.login.console
system.preferences
system.restart
system.install.admin.user
⋮

rules

authenticate-session-owner
authenticate-admin
is-admin
is-root
⋮

Rights

- If the exact right isn't found in /etc/authorization, partial matches will be used
- Example: `config.remove.system.foo` will look for the following rights in /etc/authorization until one is found:
  - `config.remove.system.foo`
  - `config.remove.system.`
  - `config.remove.`
  - `config.`
  - *<EMPTY>*

Property Lists (plists)

- XML formatted preference file
- Two ways to edit it:
  1. Text editor (vi, emacs, pico -w, etc.)
  2.

Property Lists (plists)

```
<key>rights</key>
<dict>
        <key>com.apple.appserver.privilege.admin</key>
        <dict>
            <key>class</key>
            <string>rule</string>
            <key>comment</key>
            <string>Used for administrative access</string>
            <key>rule</key>
            <string>appserver-admin</string>
        </dict>
</dict>
```

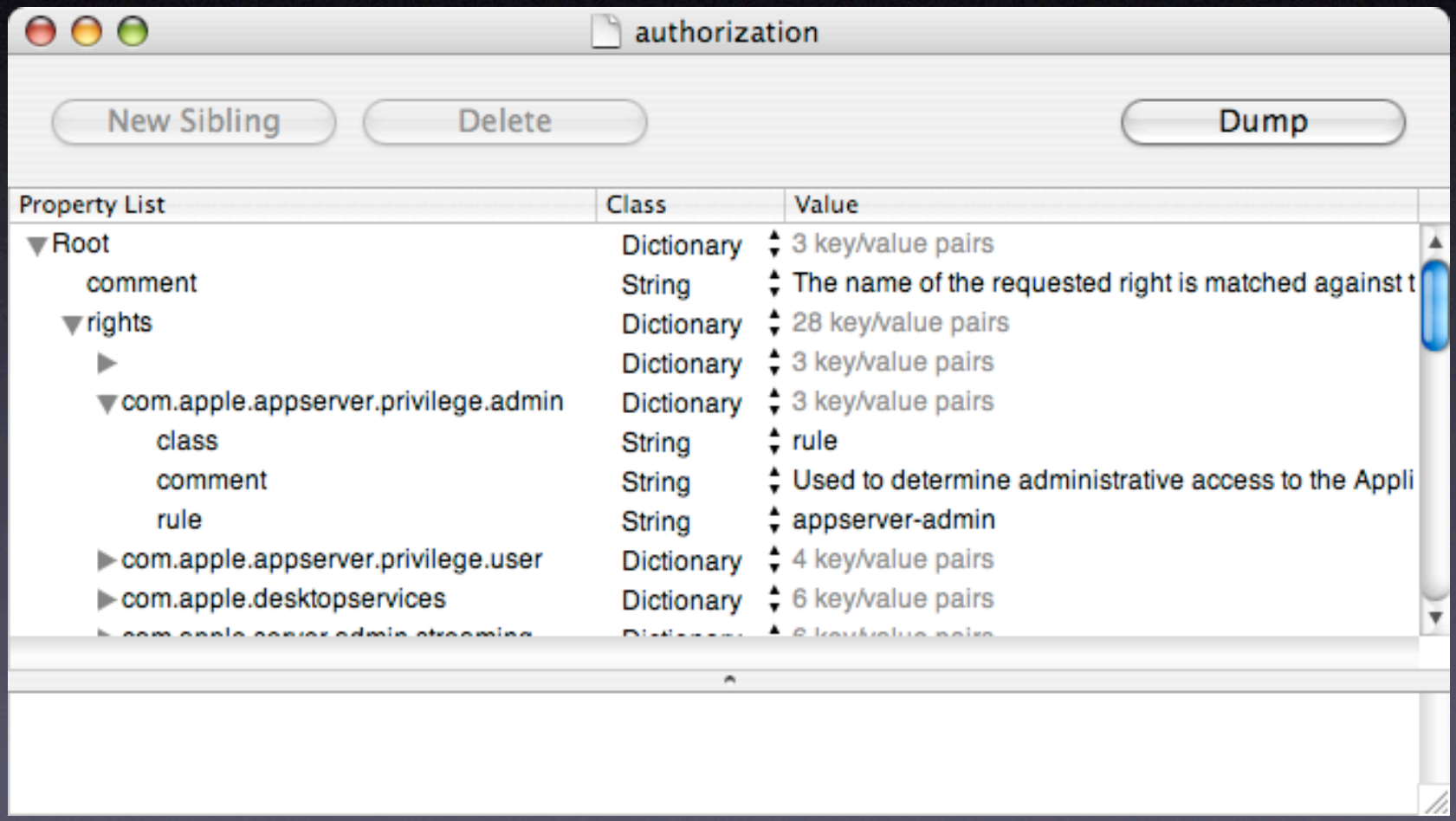
Property Lists (plists)

```
<key>rights</key>
<dict>
        <key>com.apple.appserver.privilege.admin</key>
        <dict>
            <key>class</key>
            <string>rule</string>
            <key>comment</key>
            <string>Used for administrative access</string>
            <key>rule</key>
            <string>appserver-admin</string>
        </dict>
</dict>
```

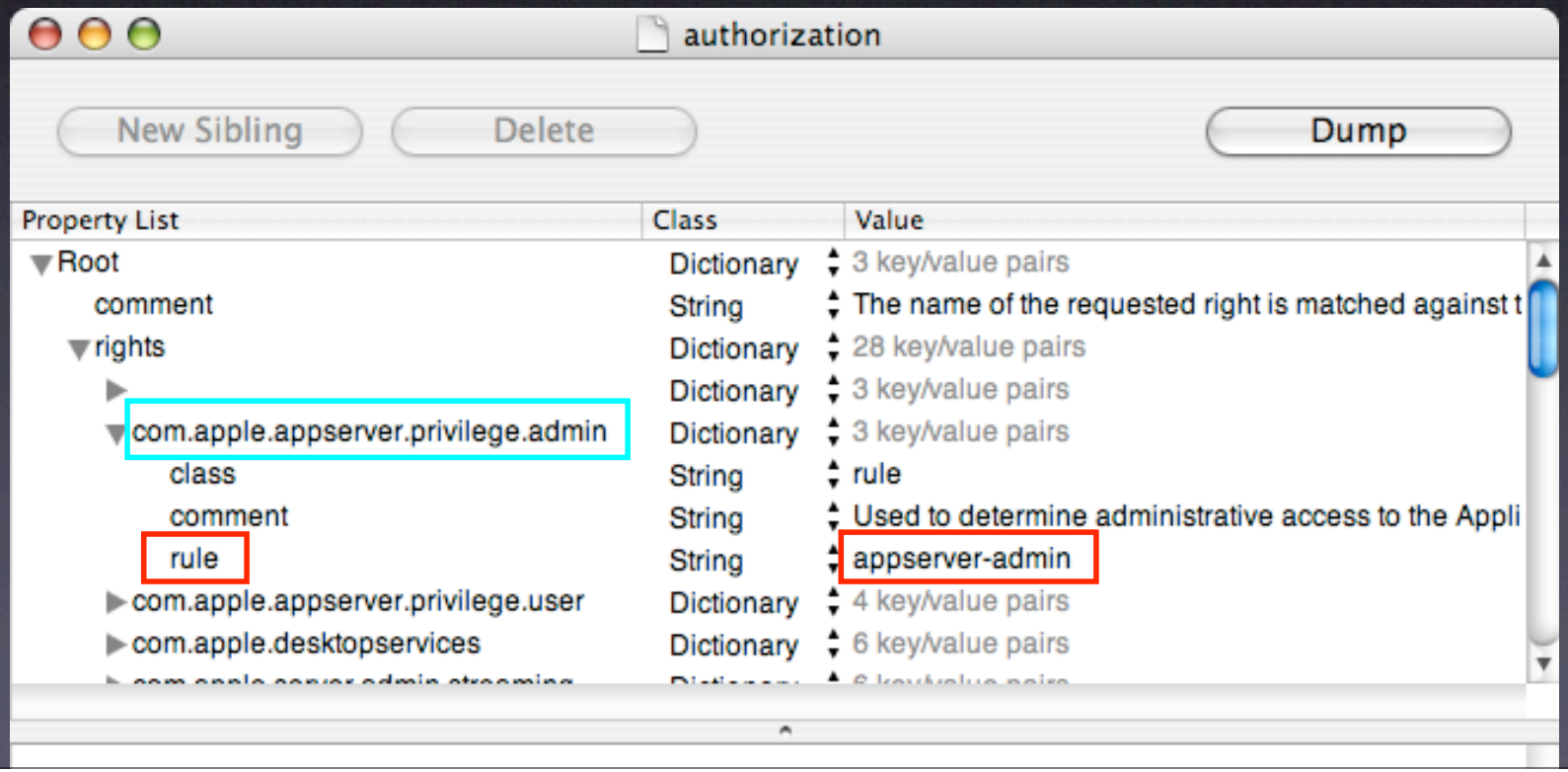
Property Lists (plists)

- XML formatted preference file
- Two ways to edit it:
  1. Text editor (vi, emacs, pico -w, etc.)
  2. Property List Editor (installed with Xcode Developer Tools)

Property Lists (plists)



```
<key>rights</key>
<dict>
        <key>com.apple.appserver.privilege.admin</key>
        <dict>
            <key>class</key>
            <string>rule</string>
            <key>comment</key>
            <string>Used for administrative access</string>
            <key>rule</key>
            <string>appserver-admin</string>
        </dict>
</dict>
```



The Catch...

- The permissions on /etc/authorization are:

```
-rw-r—r— 1 root admin /etc/authorization
```

- If you use a GUI app, it must either be able to authenticate when saving, or you must run the app as root:

```
sudo /Developer/Applications/Utilities/Property\ List\
Editor.app/Contents/MacOS/Property\ List\ Editor
```

Caution!

- Think carefully about the implications of changing /etc/authorization
- “So it’s easier” is not a good reason
- “So it’s possible” is a good reason
- Would you leave your home unlocked just so you don’t have to get out your keys?

Caution!

- It's easy to make a mistake in /etc/authorization that would render your OS unbootable
- Make backups before you change a system file

```
cp /etc/authorization /etc/authorization.backup
```

- Verify integrity of /etc/authorization after any changes:

```
plutil -lint /etc/authorization
```

plutil

```
myhost% plutil -lint /etc/authorization
/etc/authorization: OK
myhost%
```

(then I deleted a <dict> line in the file)

```
myhost% plutil -lint /etc/authorization
/etc/authorization:
XML parser error:
        Found non-key inside <dict> at line 27
Old-style plist parser error:
        Expected terminating '>' for data at line 1
myhost%
```

Caution!

- Apple may overwrite /etc/authorization without warning in a software update
- Make backups

```
cp /etc/authorization /etc/authorization.backup
```

- Verify functionality periodically

When Things Go Wrong...

- All is not lost!
- Two easy recovery methods

Target Disk Mode

- Hold ‘t’ during startup of the sad Mac until a moving firewire logo appears
- Connect a firewire cable between this and another Mac
- The sad Mac’s harddrive will show up as an external disk on the happy Mac
- On the happy Mac,

```
cd /Volumes/SadMac/private/etc
cp authorization.backup authorization
```

- Reboot the sad Mac - now it’ll be happy

Single-User Mode

- Hold ⌘s during startup

```
Singleuser boot — fsck not done
Root device is mounted read-only
If you want to make modifications to files,
run ‘/sbin/fsck -y’ first and then ‘/sbin/mount -uw /’
localhost:/ root# /sbin/fsck -y
** /dev/rdisk0s3
** Root file system
** Checking HFS Plust volume.
fsck_hfs: Volume is journaled. No checking performed.
fsck_hfs: Use the -f option to force checking.
localhost:/ root# /sbin/mount -uw /
localhost:/ root# cp /etc/authorization.backup /etc/authorization
```

- Reboot

Inside a RIGHT:

```
<key>com.apple.desktopservices</key>
<dict>
    <key>class</key>
    <string>user</string>
    <key>comment</key>
    <string>privileged file operations from the finder</string>
    <key>group</key>
    <string>admin</string>
    <key>mechanisms</key>
    <array>
        <string>builtin:authenticate</string>
    </array>
    <key>shared</key>
    <false/>
    <key>timeout</key>
    <integer>0</integer>
</dict>
```

Inside a RIGHT:

```
<key>com.apple.desktopservices</key>
<dict>
    <key>class</key>
    <string>user</string>
    <key>comment</key>
    <string>privileged file operations from the finder</string>
    <key>group</key>
    <string>admin</string>
    <key>mechanisms</key>
    <array>
        <string>builtin:authenticate</string>
    </array>
    <key>shared</key>
    <false/>
    <key>timeout</key>
    <integer>0</integer>
</dict>
```

CLASS:
allow, deny, user, rule,
evaluate-mechanisms

Inside a RIGHT:

```
<key>com.apple.desktopservices</key>
<dict>
    <key>class</key>
    <string>user</string>
    <key>comment</key>
    <string>privileged file operations from the finder</string>
    <key>group</key>
    <string>admin</string>
    <key>mechanisms</key>
    <array>
        <string>builtin:authenticate</string>
    </array>
    <key>shared</key>
    <false/>
    <key>timeout</key>
    <integer>0</integer>
</dict>
```

Inside a RIGHT:

```
<key>com.apple.desktopservices</key>
<dict>
    <key>class</key>
    <string>user</string>
    <key>comment</key>
    <string>privileged file operations from the finder</string>
    <key>group</key>
    <string>admin</string>
    <key>mechanisms</key>
    <array>
        <string>builtin:authenticate</string>
    </array>
    <key>shared</key>
    <false/>
    <key>timeout</key>
    <integer>0</integer>
</dict>
```

GROUP:
must be a member of this unix group

Inside a RIGHT:

```
<key>com.apple.desktopservices</key>
<dict>
    <key>class</key>
    <string>user</string>
    <key>comment</key>
    <string>privileged file operations from the finder</string>
    <key>group</key>
    <string>admin</string>
    <key>mechanisms</key>
    <array>
        <string>builtin:authenticate</string>
    </array>
    <key>shared</key>
    <false/>
    <key>timeout</key>
    <integer>0</integer>
</dict>
```

Inside a RIGHT:

```
<key>com.apple.desktopservices</key>
<dict>
    <key>class</key>
    <string>user</string>
    <key>comment</key>
    <string>privileged file operations from the finder</string>
    <key>group</key>
    <string>admin</string>
    <key>mechanisms</key>
    <array>
        <string>builtin:authenticate</string>
    </array>
    <key>shared</key>
    <false/>
    <key>timeout</key>
    <integer>0</integer>
</dict>
```

SHARED:
authentication credentials
shared with other rights?

Inside a RIGHT:

```
<key>com.apple.desktopservices</key>
<dict>
    <key>class</key>
    <string>user</string>
    <key>comment</key>
    <string>privileged file operations from the finder</string>
    <key>group</key>
    <string>admin</string>
    <key>mechanisms</key>
    <array>
        <string>builtin:authenticate</string>
    </array>
    <key>shared</key>
    <false/>
    <key>timeout</key>
    <integer>0</integer>
</dict>
```

TIMEOUT:
Seconds before application will “relock”

Inside a RULE:

```
<key>authenticate-session-owner-or-admin</key>
      <dict>
      <key>allow-root</key>
      <false/>
      <key>class</key>
      <string>user</string>
      <key>comment</key>
      <string>the owner or any admin can authorize</string>
      <key>group</key>
      <string>admin</string>
      <key>mechanisms</key>
      <array>
                <string>builtin:authenticate</string>
      </array>
      <key>session-owner</key>
      <true/>
      <key>shared</key>
      <false/>
</dict>
```

Rights & Rules
Some other values

```
<key>default-prompt</key>
<dict>
   <key></key>
   <string>This is a custom prompt that is shown</string>
   <key>en</key>
   <string>This is the prompt shown to english locales</string>
</dict>



<key>allow-root</key>
</true>



<key>session-owner</key>
</true>
```

So what can we do with all of this?

Example

- Scenerio: Change who can set the DVD player region
- Why: Allow non-admins to set the region
- Why not: DVD region can only be changed five times - you don't want it to end up locked in the wrong region

Region Control

Region Control

Region Control

Region Control

Region Control

- Two /etc/authorization keys:
  - system.device.dvd.setregion.initial
  - system.device.dvd.setregion.change
- It might be desirable to allow anyone to set the region the first time, but require an admin (or deputy) for subsequent changes

Region Control

Here's what that portion of /etc/authorization looks like after the change:

```
<key>system.device.dvd.setregion.initial</key>
<dict>
   <key>class</key>
   <string>allow</string>
   <key>comment</key>
   <string>Used by the dvd player to set the regioncode the first time.</string>
</dict>
<key>system.device.dvd.setregion.change</key>
<dict>
   <key>class</key>
   <string>user</string>
   <key>comment</key>
   <string>Used by the dvd player to change the regioncode the 2nd-5th time</string>
   <key>group</key>
   <string>jradmin</string>
   <key>mechanisms</key>
   <array>
        <string>builtin:authenticate</string>
   </array>
   <key>shared</key>
   <false/>
   <key>timeout</key>
   <integer>0</integer>
</dict>
```

Region Control

Region Control

The drive region code was successfully changed.

OK

Example

- Scenerio: Allow anyone, not just admins, to install software
- Why: Allow users to install any software anywhere without giving them the keys to the kingdom
- Why not: Security risk - Any user could install software that would give them full access or mess up their machine

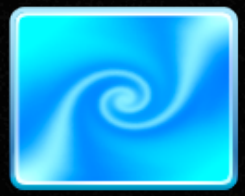
DEMO

Anyone Can Install

- Modify every system.install.*.* right:
  - Change class to ‘allow’
  - Delete group, mechanisms, shared, timeout
  - Example:

```
<key>system.install.root.user</key>
<dict>
        <key>class</key>
        <string>allow</string>
        <key>comment</key>
    <string>Used by installer tool: user installling in root domain (/System)</string>
</dict>
```

Example



- Scenerio: Allow a temp to unlock people's screens without giving them full admin access
- Why: Lab staff unlock and logout a screen locked machine when the user has left without requiring an administrator
- Why not: Security risk: non-admins could unlock anyone's screen anytime

DEMO

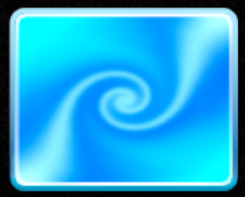
Creating a New Group

```
sudo nicl . -create /groups/jradmin gid 20500

sudo nicl . -create /groups/jradmin passwd “*”

sudo nicl . -create /groups/jradmin RealName “Junior Admins”
```

(optional)

```
sudo nicl . -merge /groups/jradmin users mike
```

Screen Saver Unlocker Group



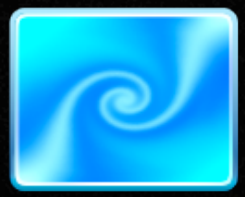
Before:

```
<key>system.login.screensaver</key>
<dict>
	<key>class</key>
	<string>rule</string>
	<key>comment</key>
	<string>the owner as well as any admin can unlock the sc
reensaver;modify the group key to change this.</string>
	<key>rule</key>
	<string>authenticate-session-owner-or-admin</string>
</dict>
```

After:

```
<key>system.login.screensaver</key>
<dict>
	<key>class</key>
	<string>rule</string>
	<key>comment</key>
	<string>the owner as well as any admin can unlock the sc
reensaver;modify the group key to change this.</string>
	<key>rule</key>
	<string>authenticate-session-owner-or-jradmin</string>
</dict>
```

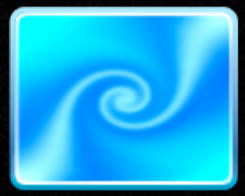
Screen Saver Unlocker Group



Copy the `authenticate-session-owner-or-admin` RULE and rename things appropriately:

```
<key>authenticate-session-owner-or-jradmin</key>
     <dict>
     <key>allow-root</key>
     <false/>
     <key>class</key>
     <string>user</string>
     <key>comment</key>
     <string>the owner as well as any jradmin can authorize</string>
     <key>group</key>
     <string>jradmin</string>
     <key>mechanisms</key>
     <array>
              <string>builtin:authenticate</string>
     </array>
     <key>session-owner</key>
     <true/>
     <key>shared</key>
     <false/>
</dict>
```

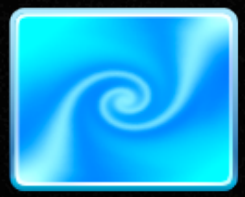
Example



- Scenerio: Remove the ability for admins to unlock anyone else's screensaver
- Why: To satisfy paranoid users
- Why not: No safety net for forgotten passwords. Admin can't access the machine without the user present for safe shutdown.

DEMO

More Secure Screen Saver



Here's what that portion of /etc/authorization looks like after the change:

```
<key>system.login.screensaver</key>
<dict>
	<key>class</key>
	<string>user</string>
	<key>comment</key>
	<string>only the owner can unlock the screensaver</string>
	<key>mechanisms</key>
	<array>
		<string>builtin:authenticate</string>
	</array>
	<key>session-owner</key>
	<true/>
	<key>group</key>
	<string>someinvalidgroup</string>
</dict>
```

Simply changing the rule to authenticate-session-owner did not work...

Example

- Scenerio: Allow Kerberos passwords for everything
- Why: Utilize existing passwords; Single Sign-On
- Why not: If machine is bound to an LDAP directory, could potentially give too many people access to the machine

DEMO

Kerberized Console

- Add a "`builtin:krb5authnoverify`" string to the mechanisms array for the following ***rights***:
  - system.login.console
  - system.preferences
  - system.install.root.user
  - system.install.admin.user
  - system.install.root.admin
  - com.apple.desktopservices
  - system.privilege.admin

Kerberized Console

- Add a "`builtin:krb5authnoverify`" string to the mechanisms array for the following ***rules***:
  - authenticate-session-owner-or-admin
  - authenticate-admin
  - authenticate-session-owner

Kerberized Console

Here's what one portion of /etc/authorization looks like after the change:

```
<key>com.apple.desktopservices</key>
<dict>
   <key>class</key>
   <string>user</string>
   <key>comment</key>
   <string>authorize privileged file operations from the finder</string>
   <key>group</key>
   <string>admin</string>
   <key>mechanisms</key>
   <array>
        <string>builtin:authenticate</string>
        <string>builtin:krb5authnoverify</string>
   </array>
   <key>shared</key>
   <false/>
   <key>timeout</key>
   <integer>0</integer>
</dict>
```

Example

- Scenerio: Require certain group membership to burn CD-Rs
- Why: In a lab, restrict who can copy things off the computer
- Why not: You may want to allow people to copy things off the computer

DEMO

Restricted Burning

Here's what that portion of /etc/authorization looks like after the change:

```
<key>system.burn</key>
<dict>
   <key>class</key>
   <string>user</string>
   <key>comment</key>
   <string>authorization to burn media</string>
   <key>group</key>
   <string>jradmin</string>
   <key>mechanisms</key>
   <array>
        <string>builtin:authenticate</string>
   </array>
   <key>shared</key>
   <false/>
   <key>timeout</key>
   <integer>0</integer>
</dict>
```

Synopsis

- Modifying /etc/authorization is a powerful way to tune your system policies to your specific situation
- As with most things on the Mac, you can use the command-line or the GUI to make your changes
- You must be careful anytime you edit system files
- If you take the proper precautions (make backups), it's easy to recover if you make a mistake

Links

- http://developer.apple.com/technotes/tn2002/tn2095.html
- http://developer.apple.com/documentation/Security/Conceptual/authorization_concepts
- http://www.macenterprise.org/
- http://www.afp548.com/
- http://www.macosxhints.com/

Thank You!
Rights and Rules Demystified

Session M255

Macworld San Francisco 2005

David Pugh & Jan Stewart

University of Michigan

Apple Certified System Admistrators 10.3