Automating Radmind (back to top)

It is desirable to set up radmind on clients so that it automatically runs on a regular basis.

The two requirements of automating radmind is to prevent the user from using the computer while radmind runs and to run the radmind tools with root permissions.

This section discusses one approach using some perl scripts and the application iHook.

About iHook

iHook is an application that functions as a GUI for scripts. Specifically, it launches a script and opens a window and displays a graphic, a text field, and a progress bar. The administrator has control over which script is launched, the image displayed in the window, the contents of the text field, etc.,.

Like most command line tools, radmind does not give any feedback when running. If you configure automated maintenance to run at startup, login, or logout, users have no way of knowing that the machine is not frozen without a GUI like iHook.

iHook and radmind

To use iHook to run radmind, you need to create a script that runs radmind. This script has several requirements. It must run as a root process and it should run when a user can not login.

LoginHooks, LogoutHooks, and StartupItems will meet both needs. They launch with root permissions and wait for the script to finish before allowing the Mac to be used (in Mac OS X 10.2 for a StartupItem, an extra file need to be modified to prevent the login panel from displaying).

For more information on iHook, see the iHook README.txt and iHook Lexicon.txt that is included with the application. It can be obtained from this page.
<http://rsug.itd.umich.edu/software/ihook/>

run_radmind script

The script that runs the radmind tools can be as simple as:

```
/usr/local/bin/ktcheck -c sha1 -h your.radmind.server
/usr/local/bin/fsdiff -c sha1 -A / > /tmp/update.T
/usr/local/bin/lapply -c sha1 -h your.radmind.server /tmp/update.T
```

Or the script could perform multiple error checking, do other tasks such as rename all of the preference files in the ~/Library/ByHost folders, or create backup home folders. For more information about these tasks, please see: ByHost renamer script.
<http://www.macosxlabs.org/documentation/hard_disk_maintenance/byhost_renamer_shell/byhost_renamer_script.html>

There are example scripts provided with iHook. Please use them as your starting point. Or you can look at this run_radmind script and uncomment the iHook directives.

[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

In the scripts that follow, the run_radmind script will be refered to as $run_radmind = "/path/to/radmind". This means that this script is not intended to be a part of any other scripts and should not be dependent on them.

It is desirable to restart after running radmind, especially if there is an OS update. The scripts listed on this page always call /sbin/reboot after executing the run_radmind script.

Launching iHook

If you want your startup, login, and logout scripts to use iHook, you must use a launcher script (see the next section). The alternative is much easier: just configure run_radmind to use iHook. In all of the scripts on this page, replace the command:

$run_radmind = "/path/to/radmind"

with

$run_radmind = "/path/iHook.app/Contents/MacOS/iHook --script=/path/to/radmind"

You can not have all 4 scripts (startup, login, logout, and run_radmind) use iHook since that would launch multiple instances of iHook (startup/login/logout launches iHook and run_radmind; run_radmind launches iHook = 2 iHook instances). The reason for using iHook at startup, login, and logout is if those scripts perform time consuming tasks such as making backup home folders. Unless your scripts do this, it is recommended that only run_radmind launch iHook (simpler).

iHook Launcher scripts

By default iHook will launch a script located at /etc/logout.hook. If you want your login, logout, or startup scripts to use iHook you have to create iHook launcher scripts.

The launcher scripts launch iHook and specify the actual login, logout, or startup script as an iHook parameter. The details of setting up the LoginHook, LogoutHook, and StartupItem are discussed below, where they are used. See those sections for more information about each type of script.

Here is an example startup script that launches iHook with the correct parameter. The file /Library/StartupItems/LabStartup/LabStartup contains:

```
#!/bin/sh
/path/iHook.app/Contents/MacOS/iHook --script=/path/startupscript
```

The actual startup script would be located at "/path/startupscript" (changing "path" to the correct path).

Here is an example of using a login and logout hook. The file /etc/ttys contains:

```
console "/System/Library/CoreServices/loginwindow.app/loginwindow -LoginHook
/path/loginHook -LogoutHook /path/logoutHook"
```

The file /path/loginHook contains:

```
#!/bin/sh
/path/iHook.app/Contents/MacOS/iHook --script=/path/loginscript $1
```

The file /path/logoutHook contains:

```
#!/bin/sh
/path/iHook.app/Contents/MacOS/iHook --script=/path/logoutscript $1
```

The actual login and logout scripts would be located at "/path/loginscript" and "/path/logoutscript" (changing "path" to the correct path).

When to run radmind (back to top)

Many people believe radmind can only be run at logout. Radmind can be run at many different times or situations. For example, radmind can be run at login, logout, particular user login, by request at logout, login/logout frequency or time period, scheduled with cron, single user mode, or even remotely via ssh or a web page.

Every Logout

This is the simplest method of automating radmind, but it is probably the method most annoying to end users.

Modify the file /etc/ttys. Change:

```
console "/System/Library/CoreServices/loginwindow.app/loginwindow"
```

To:

```
console "/System/Library/CoreServices/loginwindow.app/loginwindow -LogoutHook
/path/to/radmind"
```

Or to use iHook and /etc/logout.hook:

```
console "/System/Library/CoreServices/loginwindow.app/loginwindow -LogoutHook
/path/iHook.app/Contents/MacOS/iHook"
```

To get this change to take effect, you must restart the Mac.

One method to improve performance with radmind is to leave checksumming off when running it at logout. With checksumming, a fast Mac with a fast network connection with a small loadset can take as little as 4 minutes at the best. However, a slow mac with a slow network connection and with large image, radmind can take up to 20 minutes or more.

To turn checksumming off, just remove the "-c" flag (and the checksumming parameter, i.e. "sha1" or "md5") from the fsdiff command. The fsdiff command is located in the script that executes radmind, either /etc/logout.hook, /path/to/radmind, or whatever other location you choose.

Other Logout Options

There are several ways to avoid or lessen the the delay of running radmind after every logout. The easiest is to use a script that checks the time and only run radmind during non peak times.

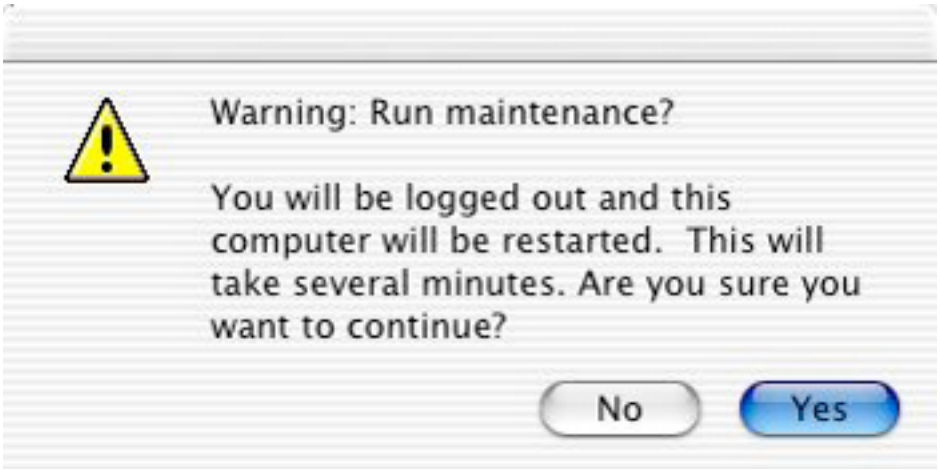
The script radmind_logout_timewindow will check the time and run radmind only at admin specified times.

[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

Instead of running radmind at logout, you could use a script to cleanup after every logout at a fraction of the time it takes to run radmind (several seconds compared to several minutes). See the guest documentation for scripts that will do this.
< http://www.macosxlabs.org/documentation/guest/intro.html>
Manually (at logout by request)

Sometimes you want to give consultants and public users the ability to manually run Radmind, but not give them access to terminal with an administrative account. To give them this ability use an AppleScript that has a double-clickable icon and gives user feedback, a "trigger file", and a logout script.

The AppleScript RunMaintenance will ask "are you sure", writes the trigger file /tmp/runRM, and logs the current user out. Be sure to change "<<" and ">>" to the one character version by pressing option-\ and option-shift-\. Save the AppleScript as an application.



[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

The person who clicks on the AppleScript must be logged in as a user with permissions to write to a location where the AppleScript saves a trigger file. One place every user should be able to write is /tmp folder. However, this folder might be cleaned up at startup, so it wont work for any of the startup scripts listed here. You will need to create your own location and change the scripts.

Using "trigger files" like this is similar to Mac OS 9's "Cleanup at startup" file. If the file exists at startup, Mac OS 9 did not shutdown properly and so it should run Disk First Aid.

Execute a logout script by modifying the /etc/ttys file and adding the LogoutHook flag.

```
console "/System/Library/CoreServices/loginwindow.app/loginwindow -LogoutHook /path/logoutHook"
```

The following logout script will check to see if the trigger file exists, and if it does, it will run radmind.

```
#!/usr/bin/perl

$force_run_file = "/tmp/runRM";
$run_radmind = "/path/to/radmind";

if ( -e $force_run_file ) {
  system $run_radmind;
  unlink $force_run_file; # remove just in case.
  system "/sbin/reboot";
}
```

Login via a particular user

If Mac OS X is not logged in, it is possible to run radmind by creating a "radmind" user and then logging in as the "radmind" user. This is a useful method of running maintenance when a Mac is at the login panel and eliminates logging in. In a loginhook script, place this code:

```
#!/usr/bin/perl

$theuser = $ARGV[0];
$run_radmind = "/path/to/radmind";

if ($theuser eq "radmind") {
  system "$run_radmind";
  system "/sbin/reboot";
}
```

Finally, execute the script at login by modifying the file /etc/ttys:

```
console "/System/Library/CoreServices/loginwindow.app/loginwindow -LoginHook
/path/loginHook"
```

Note: the difference between this /etc/ttys file and the logout version is "LoginHook" instead of "LogoutHook". It is possible to have both login and logout scripts like this:

```
console "/System/Library/CoreServices/loginwindow.app/loginwindow -LoginHook
/path/loginHook -LogoutHook /path/logoutHook"
```

Scheduled with cron

If you desire to run radmind on some schedule, like running it during off hours or to load balance the network or server, it is possible to schedule radmind with cron.

The main problem with using cron is that a user might be logged in and using the Mac when cron starts radmind. Therefore, it is desirable to check for this state and display user feedback and warn users.

In order to schedule radmind with cron, add similar code to the file /etc/crontab:

```
30 5 2 * * root /path/to/script.pl
```

To edit /etc/crontab you can use Cronnix, or the Terminal.
<http://www.versiontracker.com/moreinfo.fcgi?id=9478&db=mac>

If you desire to load ballance, you may need to create many crontab files. The following scripts creates crontab files that runs the default Mac OS X periodic scripts after 7 am, runs radmind twice a week (between midnight and 7am), and reboot the other nights. The script plethoraCrontabsClient will create the crontab files and upload them to the radmind server. The script updatePosTranscript runs on the radmind server and verifies and moves the transcripts

and files. The script plethoraCrontabsServer runs on the radmind server and calls updatePosTranscript repeatedly to verify and move the crontab files.

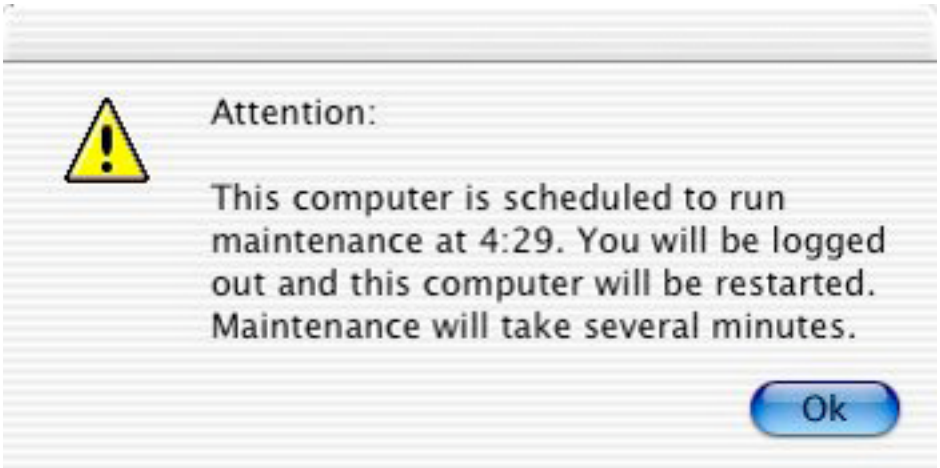
[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

To notify a user that radmind is going to run, this cronScript will check to see if a user is logged in and launch an AppleScript notifying the user.

[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

The script restarts the machine before running radmind because logout is often canceled by applications that will not quit when told, which would prevent radmind from running. To guarentee that maintenance will run this script restarts and then to checks for the radmind "trigger file" at startup and runs radmind then.

If a user is logged in, this MaintenanceNotification AppleScript (click to view in popup window), when saved as an application, will launch and notify the user of the maintenance.



[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

The startupScript checks for the trigger file and if it is present it runs radmind and restarts.

[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

To use the above script at startup, you need to create this folder and two files:

/Library/StartupItems/RunRadmindAtStartup/
/Library/StartupItems/RunRadmindAtStartup/RunRadmindAtStartup
/Library/StartupItems/RunRadmindAtStartup/StartupParameters.plist

You will probably need to create the folder "/Library/StartupItems" as well. The folder within "/Library/StartupItems" must have the same name as the startup script.

Note: if your run_radmind script is not configired to use iHook, and instead you want the above startup script to launch using iHook, the file /Library/StartupItems/RunRadmindAtStartup/RunRadmindAtStartup should be an iHook launcher and you will need a second script that contains the above code that checks for the radmind trigger file. See the sections above concerning launching iHook and iHook launcher scripts.

This is an example StartupParameters.plist (click to view in popup window):

```
{
  Description = "Runs radmind at startup if needed";
  Provides = ("Order from Chaos");
  Requires = ("Disks", "Resolver", "Network");
  OrderPreference = "Early";
}
```

In Mac OS X 10.2.x, the file "StartupParameters.plist" in the folder /System/Library/LoginWindow/ needs to be modified or else the it will autologin or allow users to login while radmind runs. Add "Order from Chaos" (or whatever your script provides) to the LoginWindow's dependencies, like this:

```
Requires = ("Disks", "DirectoryServices", "Core Graphics", "Core Services", "Order from Chaos");
```

For more information about startup items, please see Apple's documentation.
http://developer.apple.com/techpubs/macosx/Essentials/SystemOverview/BootingLogin/Customization_Techniques.html

Finding Radmind Status

Getting the Radmind status requires that you modify the run_radmind script. When the radmind tools finish they return an exit code indicating success or failure. By writing this code to a file, all your scripts can easily find the status.

To find the radmind last run date, modify the run_radmind script to "touch" a lastrun file if radmind is successful. The touch command is a simple method of creating an empty file or updating the modification date without changing the contents of the file. By getting the modification date of the file, you can easily find when radmind last ran.

The script run_radmind will run radmind and write any errors to a file. It also saves the modification date to a file (this script contains *extra* code that is commented out, so just ignore it for now).

The section "GET THE LAST RUN DATE OF RADMIND" of the script radmindStatus finds the date that radmind last ran. If the date is greater than 3 working days (Mon-Sat) then there is an error.

[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

If radmind is scheduled to run by cron, and if you use different crontabs to load balance radmind to run on different days or on different lab off hours, it is desirable to find out which day radmind is scheduled to run and report that as well. The section "GET THE DAYS CRON SHOULD RUN RADMIND" of the script radmindStatus will open the crontab file and search it for your cron radmind script and figure out what days radmind is scheduled to run.

Finally, you want to check to see if the radmind tools finished successfully the last time it ran. Just check to see if the error file exists. See the section "FIND OUT IF RADMIND TOOLS SUCCEEDED".

Reporting Radmind Status via the network

Remote reporting can be done many ways: e-mail, weblog or database, central system logger, etc.,. You could also modify the message of the day file (/etc/motd) or add a message to ~/.login for ssh reporting. This area of reporting is left up to the reader to develop.

[The latest version of this script can be found at http://www.macosxlabs.org/slides/#uef_tech_2002 and is titled "Radmind – Scripts".]

Reporting Radmind Status visually

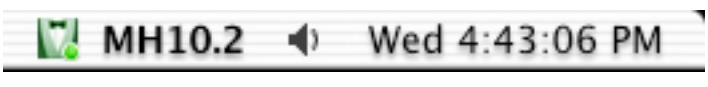
Visual alerts have several advantages over network alerts. First, in order to see visual alerts, the administrator must visit the lab. Getting in the habit of actually being physically in the lab reduces problem resolution time dramatically. It is too easy to ignore network reports or get "too busy" to respond to them.

Second, making the visual alerts obvious and simple enables lab staff or users to easily see if there is a problem and if the administrators allow manaul radmind launches (see above), then radmind can be run manually, which is especially nice just in case the adminisrators get "too busy".

Two places to leave visual messages is in the menu bar and the login panel. It is easy in Mac OS X to add a system menu. It is also easy to modify the login panel. So whether the Mac is logged out or logged in, you can easily see the status just by looking at the screen.

Reporting Radmind Status visually using Menuversum

This image shows the menubar with a green icon, indicating there were no problems. It also shows which mornings (MH = Monday, Thursday) radmind is scheduled to run. Next to that is the month and day that radmind last ran.



This image shows the menu bar with a red icon and an exclamation point, indicating that there is some sort of error.

Colored icons is the easiest way to tell a consultant that there is a problem.

Using Menuversum, it is possible to add a custom system menu with your own icon and text. Menuversum keeps track of a menu's icon and text with a plist file. By modifying this with a script, you can easily change the text or graphic that should appear in the menu.

First, you must install Menuversum. Download it from www.vercruesse.de/software.

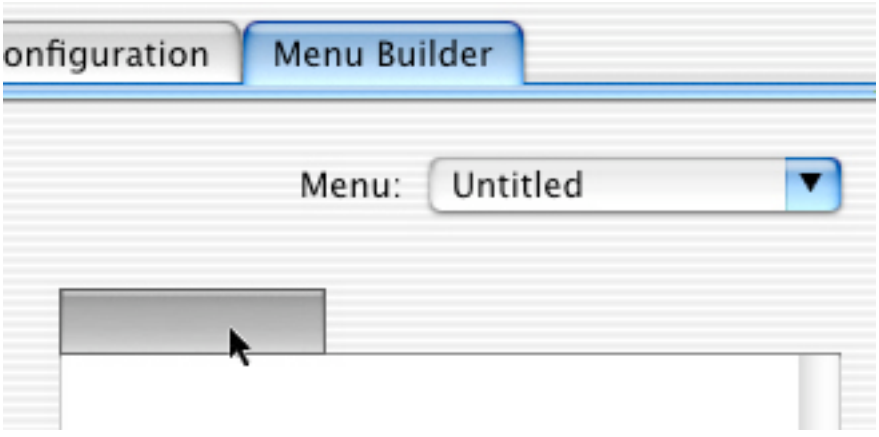
Copy the folders DeVercruesseMenuversum.framework and DeVercruesseUtilities.framework to /Libary/Frameworks/
Copy the folder Menuversum to /Library/Application Support/
Copy the bundle Menuversum.prefPane to /Library/PreferencePanes/

Note: In Mac OS X 10.2.x, the Menuversum menus will not work without also installing Menu Extra Enabler. Download Menu Extra Enabler from www.unsanity.com/download.php?product=mee.

Next, you can create your own menu. You must use Menuversum's System Preference Pane to create your own menu.

Note: As of Menuversum version 1.0b1, the Preference Pane does not work in 10.2, you must use Mac OS X 10.1.x or you can download some menus (see above).

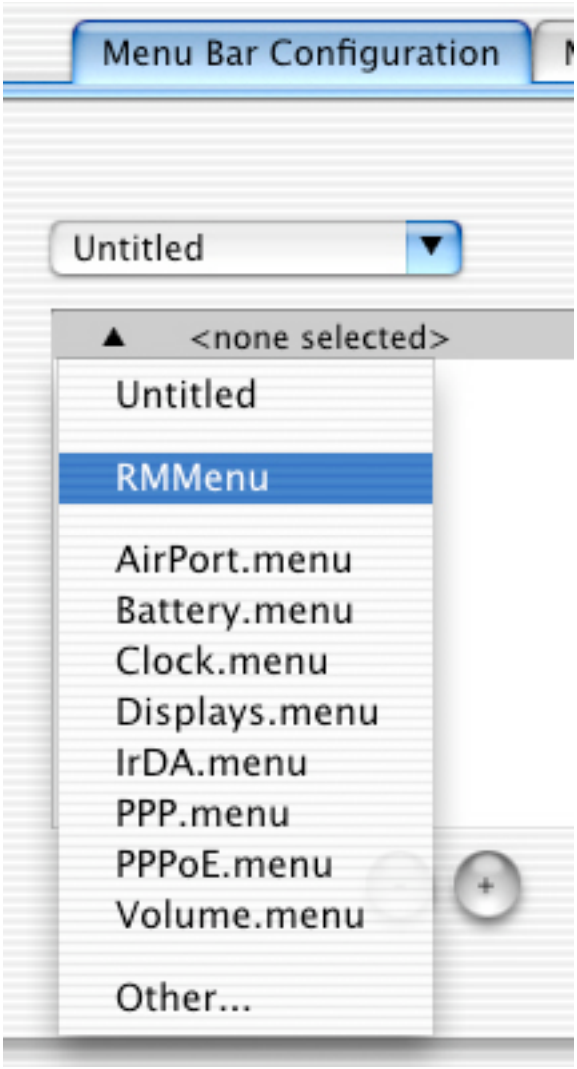
To build a menu, open System Preferences and click on the Menuversum preference pane. Click on "Menu Builder" and then click on the "Menu:" pop up button and select "New". Click on the white spot that represents the "menu".



Next, click on pop-up menu that says "Default" and select "Icon & Title". Find your icon and type some text that you would like (NOTE, both of these will be replaced by the script--these would become default values if your script should fail). Click "OK".

Click on the "Menu:" pop-up menu and select "Save". You need to pay attention to where you save it since the file needs to be modified.

Next, click on the "Menu Bar Configuration" tab and select a "New" configuration. Press the "+" button to add a menu. Click on the triangle to select what type of menu it should be.



Drag the menu to the location that you would like. Click the "Apply" button to test if your menu bar appears correctly. Finally save the menu.

In addition to the files located in /Library/Application Support/ and /Library/Frameworks/, include the following files in your Menuversum overload. These are the files that tell the OS to load the menu.

/Library/Application Support/Menuversum/Menus/RMMenu (or whatever you named your menu)
/Library/Application Support/Menuversum/Sets/RMMenu (or whatever you named your configuration)
/Library/Preferences/com.apple.systemuiserver.plist
/Library/Preferences/de.vercruesse.menuversum.plist
/Users/name/Library/Preferences/ByHost/com.apple.systemuiserver.xxxxxxxxxxxx.plist
/Users/name/Library/Preferences/com.apple.systemuiserver.plist
/Users/name/Library/Preferences/de.vercruesse.menuversum.plist

After you have made sure the menus work after running radmind, the section "MODIFY SYSTEM MENU ITEMS" of the script radmindStatus modifies the contents of the menu plist file so that it points to the appropriate image and contains the status text.

Finally, you need to copy a few more files and place them in the startup item you created. In addition to the actual script and the StartupParameters.plist, you need to put the following files into /Library/StartupItems/RadmindStatus/

/Library/Application Support/Menuversum/Menus/RMMenu
radmind_green.jpg (your "good" icon)
radmind_red.jpg (your "bad" icon)

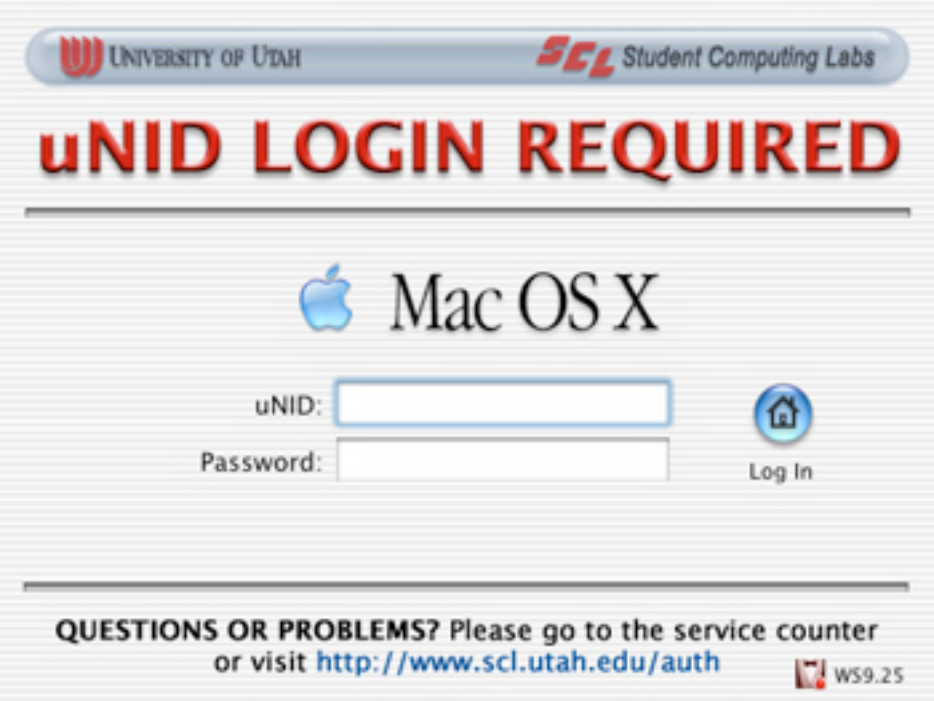
Reporting Radmind Status visually via the Login Panel

The login panel is another location that you may wish to have the radmind status shown, especially if your lab is authenticated. It is possible to add a text field using Interface Builder and then replace text in the field with a script. It is also possible to rename one of the default images so that a different image can be displayed showing status.

The following images show the login panel that has been modified by adding a text field, and changing two images. One image acts as the custom school graphic (which is transparent and has been resized to fit the window) and the other displays the a small radmind assistant icon.

This image shows the green icon indicating there were no errors.

This image shows the red icon indicating there were errors.



To modify the loginpanel, first, you need to create the custom images. Use your favorite graphic editing application or you can download some images (see above). Replace the default images with your custom images (your images must be saved as tiff format). The image files are:

In Mac OS X 10.1.x:

/System/Library/CoreServices/loginwindow.app/Contents/Resources/English.lproj/loginpanel.tiff

/System/Library/CoreServices/loginwindow.app/Resources/MacOSXart.tif

In Mac OS X 10.2.x:

/System/Library/CoreServices/SecurityAgent.app/Contents/Resources/loginpanel.tiff

/System/Library/CoreServices/SecurityAgent.app/Contents/Resources/MacOSXart.tif

Then modify the login panel using Interface Builder. In Mac OS X 10.1.x, the file you need to modify is:

/System/Library/CoreServices/loginwindow.app/Contents/Resources/English.lproj/login.nib

In Mac OS X 10.2.x, the file you need to modify is:

/System/Library/CoreServices/SecurityAgentPlugins/loginwindow.bundle/Contents/Resources/English.lproj/login.nib

They are basically the same file, but are located in different places in the different versions.

When you have the nib open, open the window object if it is not open. In the window object, click on the image wells (they will have a generic image icon with a crack through it) and drag them to move them or click on the dots surrounding them to change their size. Modify the window or other elements by clicking on them and dragging them or opening the information window (shift-command-i).

Add a text field in the corner, and put the text "xxxxxxxx" in it. The script will look for that text to replace.

To learn how to use Interface Builder, there is an Interface Builder Help Center (available by clicking the Help menu), Apple's developer pages, or online help. Note: Interface Builder is included with the developer tools, which can be obtained several ways. See Apple's webpages for more information.

<http://developer.apple.com/tools/interfacebuilder/>
<http://developer.apple.com/techpubs/macosx/DeveloperTools/InterfaceBuilder/InterfaceBuilder.help/Contents/Resources/English.lproj/>

After you have a your modified login panel, you need to copy a few files and place them in the startup item you created. In addition to the actual script and the StartupParameters.plist, you need to put the following files into /Library/StartupItems/RadmindStatus/

Mac OS X 10.1.x:

/System/Library/CoreServices/loginwindow.app/Contents/Resources/English.lproj/loginpanel.tiff

/System/Library/CoreServices/loginwindow.app/Resources/MacOSXart.tif

/System/Library/CoreServices/loginwindow.app/Contents/Resources/English.lproj/login.nib/objects.nib

Mac OS X 10.2.x:

/System/Library/CoreServices/SecurityAgent.app/Contents/Resources/loginpanel.tiff

/System/Library/CoreServices/SecurityAgent.app/Contents/Resources/MacOSXart.tif

/System/Library/CoreServices/SecurityAgentPlugins/loginwindow.bundle/Contents/Resources/English.lproj/login.nib/objects.nib

Finally:

radmind_green.jpg (your "good" icon)

radmind_red.jpg (your "bad" icon)

Imaging a Mac with radmind for the first time (back to top)

Custom Radmind Installer

It is possible to create a startup script that uses iHook to run radmind for the first time. This is especially nice when converting a large lab to use Radmind. It is nice because the administrator can leave the Macs alone without fear that someone will stop the process. Contrast this to having using Terminal, where the administrator must login as root user and leave it like that until it is finished.

To use the installer, you just take any Mac OS X box and do these things:

- Make sure TCP/IP networking is setup and working.
- Make sure that the IP or hostname of the client is listed in the config file on the radmind server or that the client has the correct certificate.
- Perhaps turn energy saver off.
- Install the Radmind Initial Installer.
- Restart.

At startup, iHook will launch a script that runs the radmind tools. This script will run at startup even if the machine is force restarted while running. It does not require you to install the radmind tools seperately. If you made a mistake and need to stop radmind from running so you can log in, just unplug the ethernet cable at startup.

Download the Radmind Initial Installer.

Note: we have successfully upgraded from 10.1.3 to 10.1.5 and from 10.1.5 to 10.2.1 (and from 10.2.1 to 10.1.5 -- opps!), but if for some reason radmind fails somewhere between the two versions, you may need to reinstall the OS to get it bootable.

You can create your own custom installer. Installers are created with PackageMaker, which is included with Apple's developer tools (located at /Developer/Applications/PackageMaker.app).

To create your own installer, you need the script InstallRadmind, the file StartupParameters.plist, and this installer script. In addition, you need to include iHook and the radmind tools ktcheck, fsdiff, and lapply. You can also create your own custom installer graphic that iHook will display while radmind runs.

Create a project folder. Two folders inside of that folder, one will be moved to /Library on each client (named StartupItems), the second will contain resources like the installer script (it can be named anything, but use "Resources"). Inside of the StartupItems folder, create another folder named InstallRadmind. Inside the InstallRadmind folder, place these items:

~/InstallRadmindProject/StartupItems/InstallRadmind/iHook.app
~/InstallRadmindProject/StartupItems/InstallRadmind/fsdiff
~/InstallRadmindProject/StartupItems/InstallRadmind/ktckeck
~/InstallRadmindProject/StartupItems/InstallRadmind/lapply
~/InstallRadmindProject/StartupItems/InstallRadmind/radmind_installer.pl

~/InstallRadmindProject/StartupItems/InstallRadmind/InstallRadmind
~/InstallRadmindProject/StartupItems/InstallRadmind/StartupParameters.plist

Inside of the Resources folder, place this file:

~/InstallRadmindProject/Resources/postflight

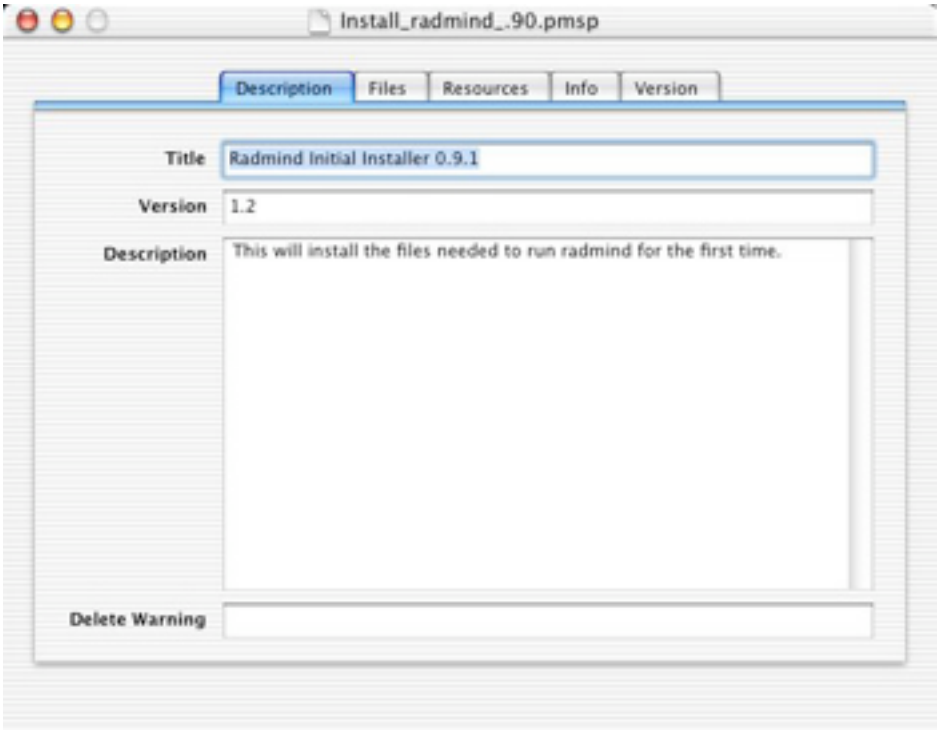
You could also add a background image to the installer and a welcome message by including files that are named "background.***" and "welcome.***". For example:

~/InstallRadmindProject/Resources/background.tif
~/InstallRadmindProject/Resources/Welcome.rtf

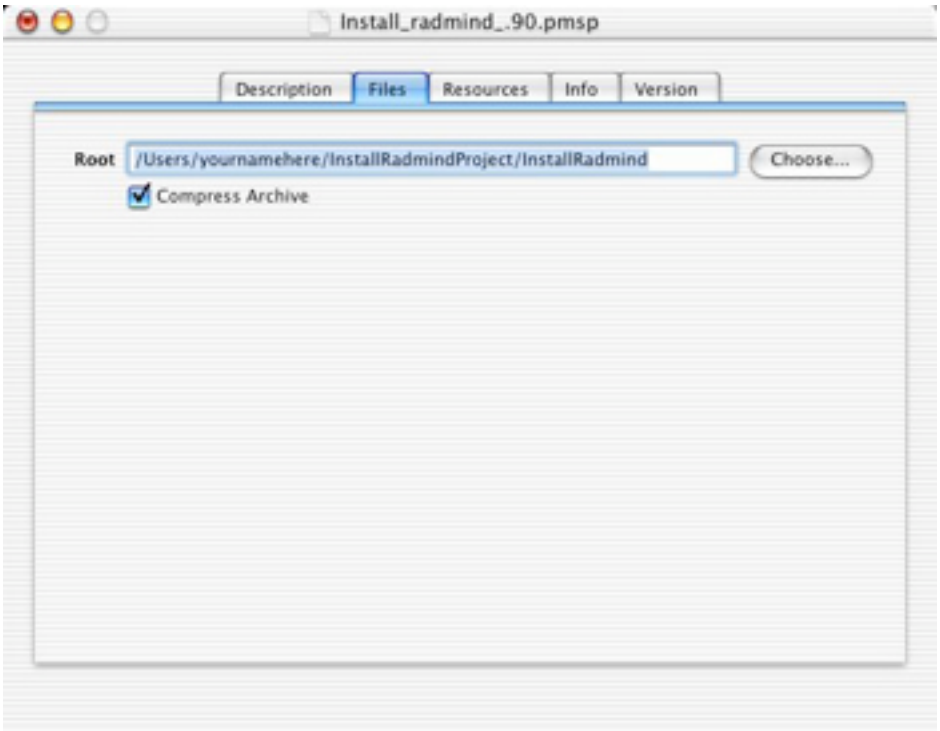
See the help files in the PackageMaker's Help menu for more information about the file formats available and other files that you can include.

Note: Mac OS X 10.1 has a different PackageMaker than Mac OS X 10.2. Packages made with the version that ship with the 10.2 developer tools will not work on Mac OS X 10.1.x. The images below are of the version that ships with Mac OS X 10.2.

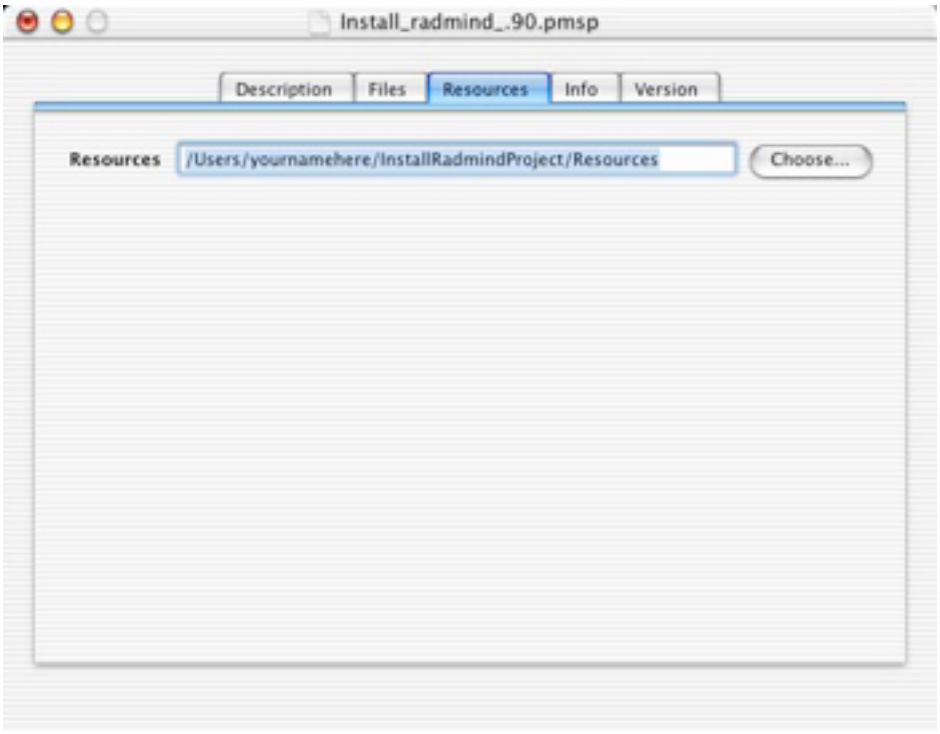
When you have your files ready to be packaged, open PackageMaker. Most of the fields are self explainitory. Just go ahead and fill them out.



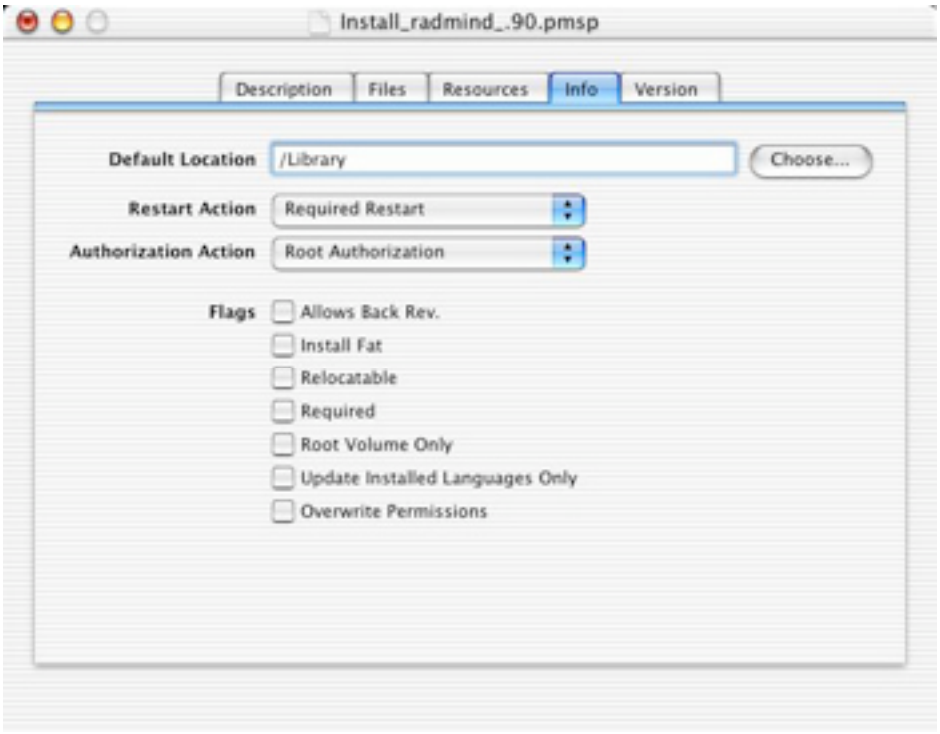
Under the files tab, set the Root Directory to "~/InstallRadmindProject/InstallRadmind". You can compress the package if you would like.



Set the resources directory to "~/InstallRadmindProject/Resources".



Set the "Default Location" to "/Library". Also select "Requires Restart" and "Root Authorization".



Then select "Create Package" from the "Tools" menu. Save it with the name you want (changing the name of the package after it is saved will cause it to not work).