

OS X: A Ten-Step Program

Crystal R. Miller
Oberlin College
148 West College Street
Oberlin, OH 44074
1-440-775-8775

Crystal.Miller@oberlin.edu

Henry R. Bent
Oberlin College
148 West College Street
Oberlin, OH 44074
1-440-775-8219

Henry.Bent@oberlin.edu

ABSTRACT

Updating Oberlin College's eight Macintosh computer labs from the conventional OS 9 of our youth to Apple's newest, sexy, Unix-based OS X has been the embodiment of our institution's historic motto, "Learning and Labor". New challenges, incompatibilities, and excitement have greeted our IT staff most every step of the way. Gone are the days of brightly colored computers, RevRdist, Pcounter, and happy Macs. Today our labs are filled with sleek flat-panel iMacs and Pharos Release Stations while Radmind keeps our hard drives up-to-date, clean, and secure.

The challenges encountered at Oberlin College are similar to those at most institutions migrating labs from Mac OS 9 to OS X. This paper will include our experiences with network interoperability, application compatibility, printing, efficient system maintenance and upgrades, system security, hardware compatibility, vendor apathy, and creating a pleasantly functional yet stable lab environment for our 2,800 users.

Categories and Subject Descriptors

K.6.3 [Management of Computing and Information Systems]: Software Management – *software maintenance, software process, software selection.*

General Terms: Management, Documentation, Design, Reliability, Security, Human Factors.

Keywords: OS X, Macintosh, Public Laboratories, Radmind, Printing.

STEP 1: GETTING STARTED

OS X had been on the minds and hard drives of much of our staff since its initial release March 24, 2001. Traditionally very cautious about any and all changes to our lab images, we waited until the release of OS 10.1 September 20, 2001 before seriously considering a lab implementation of OS X. Although this approach kept all of our labs at OS 9 when many individual Apple enthusiasts were already deep in the wonders of Aqua, Cocoa, and Quartz, the benefits of having fully functional labs equipped with tried-and-true hardware and software installations were worth sacrificing sheer GUI sex appeal. By January 2003 the big step to OS X was

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGUCCS'04, October 10–13, 2004, Baltimore, Maryland, USA.
Copyright 2004 ACM 1-58113-869-5/04/0010...\$5.00.

inevitable because Apple had stopped the sale of hardware capable of booting into OS 9, and most of our computers were due for upgrades in the summer of 2003 as part of our four-year replacement cycle. Fortunately we felt fluent using and supporting OS X on single-user desktops. All of the Mac users in our department had migrated to OS X and many faculty and staff members had also jumped on the OS X bandwagon and were relatively pleased with the results.

STEP 2: IDENTIFY ICEBERGS AND NAVIGATE AROUND THEM

Because of OS X's radical departure from the traditional Macintosh operating system, many familiar and favorite OS 9 applications were slow to be re-developed or became abandoned in the dustbin of computer history when OS X became the only operating system available on Mac hardware. Furthermore the Unix underpinnings of OS X presented a new set of security concerns previously unknown to Macintosh computer labs. Additionally, many of the known tools and techniques for distributing, protecting, and maintaining Macintosh lab images were also thrown out the window.

We were patient enough to wait until the major applications necessary for all of our labs were OS X native before attempting to upgrade any of our labs. However our patience did not resolve all of our software problems. The inability of OS X to play nicely with NetWare Print Queues and Sassafras KeyServer presented two of our largest obstacles. Furthermore, we knew that the traditional tools used in OS 9 were no longer going to be part of our lab management toolkit.

STEP 3: ACCEPT A PAPER-FILLED SOCIETY

Our current print monitoring system Pcounter [1] for NetWare was not planned for development for OS X anytime in the near future. Oberlin College had been using Pcounter to track a system of print quotas for 4 years. Each student was allotted 1000 pages each academic year and the option existed to allow additional pages to be purchased for a trivial amount of money. The printing situation in our labs was quickly getting out of control due to the booming growth of ERES and Blackboard more and more; ever-lengthening PDFs were being posted in electronic format for students to print out. The combination of cheap printing, admittedly a bad idea on our behalf, and the growing number of printable electronic resources was causing huge amounts of paper, toner, and money to be wasted in our labs.

In spite of the official lack of OS X software support and product development for Pcounter, our intrepid network dudes, Cal Frye and Scott Trimmer, were not discouraged. Over the course of several

months they experimented with Pcounter and NetWare Print Queues and they were able to pair the username of the lab computer with NetWare LDAP and deduct pages from student print quotas. This scheme might have been able to squeeze additional years out of the Pcounter software by giving students lower quotas and increasing the cost for additional pages. The Pcounter system also involved a lot of man hours billing student term bills for additional pages purchased and adding new pages to individual print quotas. Furthermore, we were not quite ready to have each user uniquely authenticating to our OS X lab computers. Instead we chose to look for a product that would work smoothly with OS X and automatically facilitate the purchase of additional pages. Thus the introduction of Oberlin College to Pharos Uniprint system which integrates with the Blackboard OneCard system to track and purchase printing in our public computer labs.

STEP 4: UPDATE CURRENT TOOLS

Sassafras KeyServer had been used in our labs for many years to properly maintain our license agreements and to avoid software piracy. Microsoft Office for X and Adobe Photoshop 7.0 were both keyed using Sassafras Keyserver. We also tried to key the Macromedia products at that time, but we discovered that it was not possible due to the version of the Keyserver software we were using at the time. Not until the Keyserver was upgraded to version 6.0 were we able to properly key the Macromedia software purchased for the OS X Test Lab.

STEP 5: SAY GOODBYE TO OLD FRIENDS

Lab management tools seem to come in three flavors: distributors, protectors, and maintainers. Some tools can only accomplish one task while others can do all three. The structural differences between OS 9 and OS X required many lab managers to radically rethink their approach to lab management. At Oberlin College, our OS 9 lab management was rather free love. Users were pretty much allowed to install additional applications as necessary and RevRdist [6] was used to refresh and maintain the lab images on a daily basis. Unfortunately RevRdist would not work with OS X. It was necessary for other tools to be investigated.

Foolproof, by Riverdeep Software [2], was a nice alternative for locking down OS 9 environments but it not being developed for OS X. Apple Remote Desktop was another product considered for use as a lab management tool. It allows easy distribution of software, can control lab machines in a teaching environment, and will do certain amounts of hardware and software inventory. However Apple Remote Desktop cannot be used to distribute system resources and it did not provide the kind of software and system refreshing and maintenance capabilities we were used to having with RevRdist. Apple Software Restore was also an option for creating lab images, but it seemed to take a lot of work to create one image and it did not offer the capability to update individual pieces of software. NetBoot, a lab imaging option for both OS 9 and OS X, was also considered as an option, but limitations of our network's capabilities precluded that option.

STEP 6: TEST ENVIRONMENT OR TEST DUMMIES?

In January 2003 we started the New Year designating our 22 unit teaching lab as an "OS X Test Lab". The Test Lab consisted of

twenty-two 733 MHz G4 towers with 256 MB of RAM and 40 GB hard drives, 1 Proxima projector, and an HP 5si printer.

The first Test Lab image included Microsoft Office for X, Adobe Reader 6.0, Adobe Photoshop 7.0, Macromedia Dreamweaver MX, Virex 7, Mulberry 2.2.1, HP Printer Descriptions, Fugu, KeyAccess Client for OS X, and Prosoft Netware Client for OS X. The addition of iMovie, iDVD, and iPhoto to the lab was also very much appreciated by our users.

In addition to the incompatibility problems with the Keyserver, many of the smaller applications that had previously been keyed and placed on a shared Netware volume, known as Macserv, were no longer compatible with OS X. Filemaker 5.0 and Mathematica 4.1 were both major sources of frustration for our faculty trying to use the test lab for teaching purposes. Although these items still worked and were accessible from the many other OS 9 labs on campus, it was difficult to explain to users that they were no longer functional from the OS X Test Lab. Many of our early adopters of OS X also had the Classic environment installed on their computers and were able to use OS 9 resources with very little difficulty. Because of the security concerns created by Classic, we made a decision very early on not to include Classic in our OS X lab environment.

Initially we believed it would be possible to protect the OS X operating system and applications by using the Limitations feature in the Accounts System Preference to limit the lab user to specific applications, System Preferences, and other computer capabilities. Our lab setup contained two users. The first user, Lab Manager, was setup with admin privileges. This user was used to initially install and update the OS, install and launch applications, create printers, and create the standard lab user. This lab image design flaw was our first of many mistakes.

To deploy the first Test Lab Image we used Retrospect 5.0 to copy the image off of the setup machine onto a Firewire drive. We then installed Open Firmware onto all of the lab machines and set a secure password. Then booting the computer into target disk mode, we used Retrospect to copy the image onto each computer. It took about two days to image the lab the first time.

After the image was installed on the computers it took about two days to find some of the problems that would amuse and annoy us for weeks to come. Printing was the first and most irksome issue to develop in the OS X Test lab. First, and most confusing to our users, it was quite easy to delete the lab printer from the Print Center. We knew this was a problem before ever rolling out the setup, but for some inexplicable reason we didn't estimate how amusing some users might find this activity. Second, we had grossly underestimated the annoyance factor of the nifty new feature called Rendezvous. So, printing in the labs functioned something like this – someone deletes the printer, leaves, another person finds the printer is deleted, they try to create a new printer using Rendezvous, Rendezvous seeks out some random printer in another building, the user sets it up on the computer, and in the end they still don't know where their term paper went. It was a lot of fun. Fortunately, we were quick to do two things. The permissions on Print Center were changed to allow the Lab Manager access to the file. Second, the Rendezvous capability was turned off on all campus printers by our gracious and cool networking folks.

Additionally, Microsoft Office stopped working, but nobody really wanted to type papers in our labs anyways. Although our idea to use OS X to manage User permissions seemed like it would work, it didn't compensate for the fact that although Microsoft Office

appears to contain only three different executables (Word, Excel, and PowerPoint) it is actually a conglomeration of many different executables that like to read files and write to directories that the OS X permissions scheme was not precise enough to handle. In our initial test the Office applications appeared to launch correctly. But after a dozen or so launches in the lab environment Word, Excel, and PowerPoint stopped bouncing for joy in the Dock. At first our super-sleuthing led us to think that the Keyserver was the culprit. But, our keyed Photoshop appeared to keep launching just fine for hours on end.

After a few more trial images in our OS X Test Lab, we concluded that the permissions in OS X were not finely tuned enough to give our users the access that they needed to use the computers successfully or protect the operating system carefully enough to keep it up and running. More enlightened than before, we returned to the drawing board to find other OS X lab management tools.

STEP 7: TEST LAB – TAKE II

Like monks on the quest for enlightenment, we went near and far searching for answers to the new challenges unearthed in our first test lab environment. It was apparent that trying to manage and protect the system using the built in permission limiting capabilities was not going to work. We also discovered that installing and customizing all of the necessary applications, printers, and other settings as the student user with admin privileges seemed to make certain settings stick better. After the image was tweaked the way we wanted it an additional Lab Manager user with admin privileges was created and the student User was reset to have only user privileges.

New approaches were also used for image distribution and maintenance. The utilities developed by Michael Bombich [3] appeared to be capable of providing the image distribution solutions we needed. We used Carbon Copy Cloner to create a disk image of the setup machine on a Firewire drive. Later we used Bombich's NetRestore utility to image the lab machines. This process was a bit faster than the Retrospect model we were previously using, but it still required someone to visit every computer in order to re-image a lab. We also implemented a login script that untarred copy of the User's home directory when the computer was restarted. Granted this approach still needed to be refined as it resulted in the deletion of any documents saved to the local hard drive.

Printing was also new and improved in our second rendition of the Test Lab. Pcounter and NetWare queues were removed and the Pharos Pop-Up Client was now used to send jobs to our lab printers.

STEP 8: THE SOFTWARE DANCE

What's there to do when vendors just don't understand lab environments? Through the course of moving our labs from OS 9 to OS X, we ran into many of the same hurdles that have plagued everyone in that position: software incompatibilities. Of course, the well known brand-name applications are usually fine, but specialized software from smaller firms can cause headaches. For instance many of the applications in use by our chemistry department are only available for the classic environment, which we were very reluctant to place in public labs due to security issues. Perhaps a year or two ago we might have been able to say, "We'll stick with OS 9 for another year," but the time has come when this is no longer a feasible solution; we cannot afford to keep older versions of brand-name productivity applications for the sake of a few smaller stragglers. Of course, with the classic environment in

OS X one encounters not only the quirks and flaws of OS 9, but also a whole new set of potential problems and little incompatibilities stemming from the interaction between the two operating systems. Fortunately we have had success so far with most of our legacy applications. While the safety compromise of having classic on OS X machines was not a decision we enjoyed making, users' needs must always come first.

Classic applications were not the only ones to cause trouble. We ran into a number of OS X native applications that had a hard time with unprivileged users. Clearly we need to have restricted users in a public, open environment like our labs. However, we ran into applications that wanted to read from files that couldn't be read, write to files that couldn't be written to, and more. Some problems involved things that had never caused issues before, like mounting disks. Some programs failed silently, while others were at least able to provide some information to make troubleshooting easier. Generally we were able to get around these issues by giving our unprivileged user explicit access to certain files and locations. While this again raises some security questions, in the end we determined that no matter how much damage a user tried to do (at least from a software standpoint), there was nothing that couldn't be fixed by a standard refresh of the machine

By far one of the most enduring and frustrating problems was integration with our campus Netware infrastructure. Oberlin has student and faculty storage space on Netware volumes, as well as a volume with shared applications. Student use of Netware is significant, so it became a top priority to get it running on OS X. Apple's lack of native support for Netware has always been frustrating, and the lack of good third party clients made this an even more trying endeavor. The solution we had been using with OS 9 was Prosoft Engineering's [7] client. Eventually they produced an updated version, which worked acceptably but not remarkably. More serious problems, however, cropped up when the labs were upgraded to 10.3. Suddenly the client became balky, sometimes refusing to connect, and it couldn't unmount drives. In addition, our unprivileged user became unable to use the client at all, as it would somehow forget its license information between reboots. Just when all seemed hopeless, some changes in our Netware backend enabled connections via AppleTalk. Initial testing showed both failure and success: volumes were conveniently accessible via a dock icon, but passwords would send in cleartext, and would only send the first eight characters. Testing and a few backend changes, with the help of our friendly Netware administrators, were able to smooth out these last few bumps. After many years it seems that we have finally been able to implement an easy to use Netware solution that doesn't require any third-party software.

STEP 9: RADMIND FOR RAD PEOPLE

Implementing Radmin [4] was a procedure that took a fair amount of time and frustration to implement, but the rewards have been well worth the initial trials and tribulations. At the time we began serious testing of Radmin, there was not thorough documentation about large scale implementation, nor about program internals. In addition, for security reasons we decided not to use the graphical user interface, so everything was being created and run through the Terminal. Now, however, there are several excellent documents covering Radmin [5], so I will not spend time here discussing its fundamentals.

Initial attempts to set up Radmin used the RevRDist mentality we were used to: large, separate images for each of our different labs.

However this meant that any time a change was made to any of the images, they had to be re-created on the Radmin server from scratch. Depending on the size of the image, this could easily take most of a morning, even for trivial changes! In addition, there were a large number of files that OS X would modify constantly, and tracking them down was an ultimately successful but rather tedious process. Eventually, though, we were able to convert our “testing” lab to use this image, and it was actually rather successful given what it was.

We had an opportunity to see a presentation about Radmin by two of its developers at the Ohio Mac Managers Support Group. This was immensely helpful because we were able to see Radmin put into practice the way it was intended to be used, as a modular rather than monolithic system. Once we had created an image of the OS and applications that were common to all labs, we could create specialized pieces on top of it rather than updating the entire image. This was especially helpful for printing. We have a number of labs that are virtually identical except for printer setup, and being able to easily create and modify just those settings saved significant amounts of time.

In light of our documentation troubles at the beginning of our Radmin endeavor, it was decided that having some in-house documentation would be helpful. A basic overview of our campus Radmin setup was committed to paper, written specifically with the non-technical user in mind, a significant detail that was missing from the original brief setup guides. Now, the documentation has proven to be very helpful, as we have two independently maintained departmental labs that will be using their own versions of our basic setup. Perhaps now our documentation may duplicate other efforts, but it has still been a useful reference for details specific to our campus.

Network considerations with Radmin were part of our initial jitters. Since the clients are set up to refresh using the cron system, all the machines run simultaneously at 3:15AM. Anticipating a heavy load when all 250 or so machines were moved to Radmin, we decided to use an Xserve to handle the load. Happily, slowness has not been a significant issue; it turns out that Radmin does as much work as it can on the local machine rather than spending network bandwidth for file comparison. Obviously storing applications for all of our different labs takes a non-trivial amount of disk space, but the ability to share the same base OS load between all the different machines saves over two gigabytes on the server for each setup.

Refreshing of machines can be slow, especially if they have a significant amount to add or remove. However the fact that this is done overnight means that we only have to worry about user impact once a semester for a few days when one of our labs is open 24 hours. User response to the changeover to Radmin has been generally nonexistent, which is a good thing. Most people seem to be unaware or unconcerned with its presence in our labs, and end-user transparency after an upgrade is perhaps the truest mark of its success.

Radmin is a learning experience, and we would be foolish to assert that we have a completed setup. Every new load set, no matter how large or small, has been an excellent opportunity to see the internals of OS X. For instance when printer settings are changed, we can

look at the Radmin logs to see which files were updated. Seemingly trivial changes can affect dozens of files. Over time, having developed a sense of how Radmin and the operating system interact, it becomes easier to create more precise and more efficient load sets. At this point we have had three distinct production images, each a little more stable and a little more tuned than the last. With 10.4 on the horizon, we are anticipating a fourth image already.

STEP 10: SECURITY, SECURITY, SECURITY

Securing not only Radmin but our labs in general proved to be a more daunting task than had originally been anticipated. Efforts such as password-protecting Open Firmware can and do go along way to protecting our machines from casual or moderately experienced users, but we weren't willing to stop there. We turned off almost all services that were not absolutely essential. For instance, we don't have SSH or Apple Remote Desktop access to our lab machines. While this does create more leg work when troubleshooting and fixing our computers, it's a small price to pay for not having to constantly keep abreast of security threats and patches.

There are always security holes in any computing environment; it is one of the unfortunate facts of life. Problems with worms and viruses affecting our Windows users have not gone unnoticed, and it is presumably only a matter of time before something will hit the Macintosh. Steps such as restricting the applications our users can run have helped stop rogue software installations – chat clients and file sharing programs were common finds on our public machines. In that case we were forced to be reactive to an existing problem, but the solution turned out to be simple. In another case, using SSL for connections to and from our Radmin server has helped ensure that casual users can't bypass its built-in protection. This step, though, was not only a purely preemptory action but was significantly more difficult to implement than some other changes. Both measures have worked exactly as they are supposed to, and after setup require little extra maintenance. While solving a problem after it has surfaced may save time, working through potential issues before they surface can prevent embarrassment and make everyone sleep a little easier at night.

REFERENCES

- [1] A.N.D. Technologies, <http://www.andtechnologies.com> (July 6, 2004)
- [2] SmartStuff, <http://www.smartstuff.com> (July 6, 2004)
- [3] Mike's OSX Management Software and Tips, <http://www.bombich.com> (March 19, 2004)
- [4] Radmin, <http://rsug.itd.umich.edu/software/radmin> (November 15, 2003)
- [5] Radmin Documentation, <http://rsug.itd.umich.edu/software/radmin/documentation.html> (November 15, 2003)
- [6] RevRDist, <http://www.purdue.edu/revrdist> (August 21, 2001)
- [7] Prosoft Engineering, <http://www.prosofteng.com> (July 6, 2004)