Herding Cats: Managing a Mobile UNIX Platform

Maarten Thibaut and Wout Mertens – Cisco Systems

ABSTRACT

Laptops running UNIX operating systems are gaining market share. This leads to more sysadmins being asked to support these systems.

This paper describes the major technical decisions reached to facilitate the pilot roll out of a mobile Mac OS X platform. We explain which choices we made and why. We also list the main problems we encountered and how they were tackled.

We show why in the environment of a customer support organization at Cisco, a file management tool with tripwire-like capabilities is needed. Radmind appears to be the only software base meeting these requirements. We show how the radmind suite can be extended and integrated to suit mobile clients in an enterprise environment.

We provide solutions to automate asset tracking and the maintenance of encrypted home directory disk images. We draw conclusions and list the lessons learnt.

Introduction

Mobile UNIX Platforms

People have been using UNIX for more than 25 years. For much of that time computers running UNIX were large, clunky machines that could only be called mobile if you happened to own a truck. The physical location of the computer and its network address remained the same for much of the machine's lifetime.

Today we are faced with mass-produced UNIX laptops that move around the world as fast as their users do.[1]

Solutions specific to UNIX machines need to be put in place to handle UNIX laptop deployments in the enterprise.

About Us

Cisco's [2] Technical Assistance Center (TAC) is a worldwide technical support organization. The TAC provides technical support to Cisco customers. We, the authors of this paper are members of the IT support group for the technical support engineers of the TAC.

All development, testing and debugging of networking products at Cisco is done on UNIX platforms – mostly Suns [5] running Solaris. Engineers need UNIX because of the large amount of utilities (some of them internal) available only for UNIX platforms.

In the last few years some of our users had been given Windows laptops, many of them had installed Cisco's version of Linux [3] on them in order to get access to the UNIX tools they needed. Recently our support group noticed a shift towards Apple [1] PowerBooks. It became clear that a supported mobile platform with UNIX capabilities was something our user base would greatly appreciate.

[1] Some laptops move faster than their users due to incorrect baggage handling or theft.

Additional Material

You can find additional material at http://radmind.org/contrib/LISA05 . There you'll find:

- The radmind extensions discussed in this paper (distributed as patches)
- The scripts called by cron to update the system (radmind-update) and some scripts that toggle radmind-update features
- The login scripts enforcing the use of FileVault
- The asset database code (client and server)
- Various tidbits and scripts that could be of use to other system administrators
- MCX configuration examples

Choices

Constraints

The following constraints are imposed on us by the department using our services:

- Our environment needs to be highly available (even the desktops). We can meet this requirement by keeping spare desktops around, clustering our servers and distributing our services.
- Users need access to UNIX tools
- Users need a mobile platform so they can work from home or while visiting customer sites
- Customer data must be protected (encrypted) when carried off-site

Practises

All Solaris client computers in our environment are identical. Users do not get privileges to install or delete software outside their home directories. While this may seem unnecessarily restrictive it allows our team to scale: identical systems mean identical problems and identical solutions; this enables us to make sure the desktop is highly available.

We decided to apply the same principle for the pilot of our new platform. This allowed us to manage the software loadsets installed during the pilot. However, some software couldn't be installed on all systems so some kind of modular setup looks more feasible for a full rollout.

Mac OS X Laptops

In order to choose the right platform a list of requirements was made and a technical analysis was performed. The platform scoring the highest in the analysis was piloted. If no show stoppers were found a wider beta rollout would be initiated to find any remaining issues before migrating the entire organization.

The method used to analyze the various platform choices was a weighted-score analysis. Each item in the list of requirements is given a weight factor. A high weight factor means the requirement is important. Each platform is then given a score for that requirement, higher scores mean a better result.

Fixed desktops scored low because of the lack of mobility. Thin clients have higher mobility but aren't truly portable. Windows [4] scored low on manageability and UNIX compatibility. Linux laptops seemed unfeasible due to the lack of support from laptop vendors.[2] Mac OS X scored high for user installable apps, hotplugging sound devices and screens and for its overall user interface.

So the results clearly suggested an Apple PowerBook platform, mainly because of the fact that it is the only viable portable UNIX platform in the list; portability and UNIX capabilities had high weights attached in the analysis.

Radmind

No matter what client platform is chosen, we need to manage its disk image. In the case of Apple the package management system doesn't meet our needs because:

- Most Mac OS X software is distributed as an application inside a compressed disk image. To install the software the user simply drags the application onto the /Applications folder. There is no package information stored in this case.
- Apple's package management system has no uninstall capability (even if the software is distributed as a package).
- There is no tripwire-like functionality to warn us about changes to the filesystem.

Many open source tools exist to create and distribute packages. They lack a tripwire [13] functionality. Radmind [12] was the only toolset that either fulfilled the requirements or could be easily altered to fulfill the requirements.

Radmind's tripwire capabilities [10, 16] trap changes to the filesystem caused by package installs on the Mac OS X machine. This allows us to use any method to install a package and record the changes made to the system in a transcript.

FileVault

All customer data must be encrypted when carried off-site. There are two ways this can be tackled:

1. Encrypt the entire hard disk. This makes sure no hard disk data is accidentally revealed but it also means that once the system has booted, all data is available. Note that this causes an increase in load times as you're encrypting the entire operating system.
2. Encrypt user data only. Obviously, there is less overhead in this scenario. It can be argued that such a modular approach is inherently more secure: as soon as the user logs out their data becomes unavailable.

No vendor-supported out-of-the-box solution was found for Linux. Apple's FileVault feature uses an AES-128 encrypted autosizing disk image to store the user's homedir. While it wasn't designed with double digit gigabyte sizes in mind it scales reasonably well in our experience. The only challenges we had were:

- There is no out-of-the-box way to enforce its use. We ended up scripting the forced creation and maintenance at login time.
- Passwords mysteriously change. This is easily fixed, provided you've installed a "master password"[3] that can unlock any filevault on the computer.
- The disk image needs to be compacted every once in a while, which takes a *long* time and can only be done at logout.

All in all we are quite happy with this solution.

Asset Tracking

There are good reasons to track a system and its users:

- the computer needs to be returned to the lease company when the lease ends
- knowing who to ask for the system when you need it back
- if the machine was used in some illegal act it helps if you know where it was and who was using it
- your organization requires recording of all user logins

The at the end of this section shows the attributes we wanted to know about.

Somehow we'll need to get mechanisms in place that give us the information we need.

At the time we couldn't find any off the shelf solution to manage our systems. We implemented our own software, AssetInterface. AssetInterface is a network client that stores asset information in a central database. The information is stored locally until the

[2] Note that Tadpole Computer [6] now offers both Linux and Solaris laptops. Sun also sells Solaris laptops.

[3] This can be configured in the Security preference pane in Mac OS X's System Preferences.

database server becomes available – i.e., when the client is on the intranet.

The information in the asset database is also useful for support; it provides us with a history of DHCP addresses for the machine. We can login to a user's machine without having to ask for the IP address.

Attribute	Meaning
primary user	The person normally using this machine and responsible for returning it
login user	The person actually using this machine
location	Where is the machine?
system image release	What is the version of the OS? Are all updates installed?
serial number	The hardware serial number of the computer
ip address	Current IP addresses of the computer (wireless, ethernet, . . .)
mac address	Current MAC addresses of the computer (wireless, ethernet, . . .)

Directory Services

Mac OS X can be configured to use a wide range of directory services such as NIS [7], LDAP [15] and Active Directory [8]. Since we had an existing LDAP infrastructure for our Solaris systems we used it for our Powerbooks too.

One noteworthy point is that we duplicated the LDAP tree layout of Apple's Open Directory http://www.apple.com/server/macosx/features/opendirectory.html product and pushed it into our existing LDAP server as a separate subtree. The regular tree contains the identification and authentication data. The Open Directory subtree contains authorization data encoded as MCX records.

This setup allowed us to use Apple's Workgroup Manager application to configure the MCX features, discussed further. This requires some schema changes on your LDAP server.[4] You can bind to any unmodified third party LDAP server but this leaves you without the MCX feature set. You can also setup your own OpenDirectory server just for MCX management, leaving authentication to, e.g., Active Directory.

Managing the System State

There's more to managing a system than just controlling the contents of the harddisk. Strictly speaking, the system state consists of:

- the system hardware
- the state of all bits in memory and on the harddisk
- the CPU and hardware state

Luckily, we can control most aspects of the system though its harddisk image. For example,

- scripts run at boot time can change the PROM password or update firmware
- pre-apply scripts can unload kernel extensions before upgrading a kernel driver

File System Management

Radmind was conceived to help system administrators manage lab systems, it's typically run at logout or system boot. Scripts chain together the radmind tools to download loadsets, check for differences, apply changes, etc.

It works well on laptops because all network connections originate from the client.

In what follows we outline some of the changes we had to make to suit radmind to our needs.

Scripting

The clients run a script from cron to check for updates and install them. We divert from common practice in several ways:

- We run this script while the user is active – this allows updates to happen over VPN links. Contrary to popular belief, this approach caused very few problems.[5]
- Once updates are ready to install the user is notified of the pending update. They can then choose whether to allow the update to take place at that time. The notification to the user includes details such as the expected download size and whether the update requires a reboot. This allows the user to make an informed decision – which turned out to be very important:
  1. users need to feel empowered
  2. there may not be enough time for the update to install
  3. the user might be in the middle of something important and doesn't want to be disturbed
- Pre-apply and post-apply scripts are downloaded before any other loadsets and before the pre-apply scripts are run.
- When an update requiring a pre-apply script is pushed out, the pre-apply script can be bundled with the update. This is not the case with radmind which requires a three step scenario to accomplish the same task:
  1. push out the pre-apply script
  2. wait for all machines to receive the update
  3. push out the update that needs the pre-apply script

  Laptops can be out of touch for weeks on end so it's not practical to wait for step 2 to complete.
- The example script provided with radmind always runs a file system check. Ours only does

[4] LDAP administrators don't like making schema changes – make sure you ask nicely.

[5] Note that on Mac OS X, Software Update also installs software while the user is logged in.

so when there are new loadsets downloaded from the server – keeping clients' system load low.
- The process checking the file system is run at low priority to reduce user impact.
- We automatically create unique SSL certificates for each client machine. The server uses the common name (CN) inside the client's SSL certificate to decide which command file to present to the client. We can use this later on to setup test environments or to specify different system images for different platforms.[6] This also makes it easy to read and grep through syslog files (many SSL services log the CN of the client certificate). Automating the creation and installation of the unique SSL key is included in the asset tracking solution.
- Each application has its own transcript (and negative) so that:
  - a modular approach can be introduced later on (allowing the user to select which optional transcripts to install)
  - single applications can be easily added or removed without touching existing transcripts
- Config files for applications and the OS are kept in separate transcripts so that we don't lose the config files when we upgrade to a new release of the transcripted software.

Prebinding

Apple chose prebinding as a way to improve application launch times.[7]

On Mac OS X a typical binary uses functions from 10 or more different libraries or "frameworks," each of which typically binds to many more frameworks. The net effect is a slow system because of all the run-time dynamic binding that has to be done by the dynamic loader each time an application is launched.

One way to get around this is to do most of this work at installation time rather than at run time. The bindings are performed and the result is saved inside the binary. The next time the binary runs the bindings only have to be redone if the target of the bind has changed.

This clashes with a radmind-controlled installation because of the install-time prebinding: when you install an OS update that updates the C library, 80% of the executables change with it. When you upload these changes as a new transcript you end up overruling a large portion of the base loadset – though all that changed was the prebinding information inside those binaries.

We had to change Radmind so that it would calculate a file checksum that remains constant throughout prebinding changes. The patch is partly based on ctool[8] and is pending inclusion in the standard radmind distribution.

Apple will eventually move away from the current prebind method: in the future, prebind information will be stored in a separate directory maintained by the OS.

NetInfo

One thing you can't manage directly through radmind is NetInfo. It's a database that contains overriding values for all name services going from accounts to automatic mounts. It has the highest precedence and cannot be disabled. We ended up making a post-apply script that keeps the NetInfo database in sync with a template. This way we can use radmind to make changes to the system accounts and groups.

User Configuration Management

User Preferences

Most Mac OS X applications use the standard preferences storage system.[9]

This is a very clean system that allows the sysadmin to setup default values for any preference a program uses, using a series of overriding locations and a standard preference file layout. For example: an application called Bar developed by the Foo company stores its preferences in ~/Library/Preferences/com.foo.bar.plist . You can provide default settings by storing a file of the same name in /Network/Library/Preferences, /System/Library/Preferences or /Library/Preferences (the latter override the former).

If you want to override settings instead of just provide defaults you'll need to use the MCX infrastructure discussed below.

Unfortunately, not all applications use this preference system. These applications need to be managed in a more ad-hoc manner, if at all.

MCX

MCX (Managed Clients for OS X) allows control over many settings on OS X. For instance, you can specify which preference panes a user is allowed to open or override any preference for any application that uses Mac OS X's native preference framework.
- MCX also controls credential caching. You have to use MCX to make OS X store a local copy of a users' credentials, known as a Mobile User account.
- Portable Homedirectories also need to be setup with MCX.

You can find MCX configuration examples on our website. More information can be found at Macenterprise.org: http://macenterprise.org/content/view/61/42/ .

[6] The radmind config file allows wildcards for the hostname comparison. We chose a common name that looks like "ppc7450.PowerBook15.W654398KQZ4.johndoe" . For example, we could give a special image to *.johndoe or ppc7450.*.

[7] Several Linux distributions do something similar. In the Linux world, it's called "prelinking."

[8] Ctool calculates hashes of binaries. The checksum doesn't change when the prebinding on the binary is redone. See the website http://ctool.darwinports.com/ .

[9] See Apple's developer website http://developer.apple.com/documentation/CoreFoundation/Conceptual/CF-Preferences/ for details.

Supporting Infrastructure

Global radmind Deployment

For the server side of radmind, standard practice is to have one server that runs one radmind process on one port.

- In order to scale globally we have one radmind server per main site. They each serve files off a volume that is mirrored across all sites.
- The master volume is kept on a separate host that doesn't serve files. This makes sure we can make changes to the volume without interfering with the radmind server processes.
- We run three radmind instances per server: the stable, testing and staging releases. This allows us to test command files and radmind configurations before we change the stable release. Some of radmind's directories on the server can be shared between instances where appropriate.
- We created scripts for distributing changes from the master that make sure that all files are consistent before sending them to the slave servers.
- Changes to radmind's code can alter the layout and contents of the transcripts. In such cases we push the new radmind to the stable clients. In the same update we change the port number of the server where the new transcripts can be found. When all systems have upgraded in this hop-skip way, we upgrade the stable release and point the clients there again.

Backups

All relevant user data is regularly synchronized to the Solaris NFS server using a home grown rsync [14] front-end. The server data is periodically backed up to tape removing the need for a separate, native backup solution on Mac OS X. This also puts the data within easy reach on the NFS server where the users can get to it from their Solaris desktops. In our OS X 10.4 release we plan to use the built-in Portable Homedirectories instead of the rsync front-end.

Reporting

Some errors trigger an email report program (surprisingly named ''mail-report'') that automatically mails the admins with the error output. This email includes network configuration information allowing us to login to the system remotely and fix the problem before the user even notices it. We changed the mail configuration so it delivers the mail as soon as the user connects to the intranet. /etc/postfix/main.cf (postfix's main config file) is auto-generated from a template, this way we can fill in the hostname in the config file using a script.

Open Issues

Radmind Atomicity and Bandwith Use

Radmind's lapply phase downloads and installs files directly onto the live file system. It would be better if it downloaded all changes first and then applied them. This way updates don't fail somewhere in the middle when the user removes the network cable.

Radmind conserves bandwidth by only downloading changed files. The network connection to the server is not compressed, however. We are working on zlib compression for the data channel.

"Don't Care" Option

Radmind can provide defaults for a file. If the file isn't present on the system it is downloaded. If it *is* present its content is not checked – only its permissions are. We wanted a more versatile ignore command in radmind so some files would be ignored completely. A patch is available on our website.

Software Install

Some software needs to be installed on the root partition. Even if the user is given permissions to install applications there, the files will simply be removed at the next radmind-update. One possible solution is to record such installs using radmind. The loadset is added to those downloaded from the server, allowing local overrides to the base image. We hope to have this solution by the start of the next project phase.

Configuration Management

Full configuration management wasn't a priority during the pilot. We'll need to implement such a system later to cope with the many different setups and configurations that are possible. There are many projects that attempt to address issues like this, e.g., LCFGng [9] and CFengine [11]. LCFGng uses RPM packages to install and maintain the system. It can likely be modified to use radmind transcripts instead, adding a powerful security layer that can detect and fix issues automatically.

Lessons Learnt

- Most package management tools are not written with fault tolerance in mind. They assume that they know the state of the system without checking it. Even if they include a ''package checks'' tool, they don't allow you to restore missing or altered files.
- Tripwire-like tools operate on the filesystem – not on an assumed state of the filesystem. Because of this they are inherently fault tolerant. Other package tools should consider implementing a fast way to:
  - verify the filesystem contents versus the package descriptions
  - fix inconsistencies and compromised components
- OS X is a great platform if you don't want to give your users administrator privileges. Many tools and frameworks are in place (and stable) that give users control over their systems while

keeping the reigns firmly in sysadmin hands. On top of that, most third party applications can be installed without privileges by simply dragging and dropping them on the user account. In our case it was possible to give our users limited sudo capabilities only. This didn't seriously limit their ability to do their jobs.

- Radmind gave us the ability to see what horrible things some third party drivers did to our precious image and allowed us to compensate accordingly.
- Keeping transcripts clean and modular can be difficult but pays off when you upgrade or disable applications.
- Giving users the ability to select which software they want installed on their systems is important for a full roll out. However, a pilot project can afford not to bother with that and can provide the same system image to all clients instead: the needs of the many outweigh the needs of the the few – or the one.

Conclusion

A mobile Mac OS X platform can be adequately managed using radmind. Leased laptops require some form of asset tracking software – a solution had to be written in-house.

Author Information

Maarten Thibaut earned his license in electrical engineering from KIHO, Gent (Belgium) in 1995 and joined Cisco Systems in 1998, where he has been a lab and system administrator for Cisco's Technical Assistance Center. He has a variable amount of cats. Reach him electronically at mthibaut@cisco.com .

Wout Mertens earned a Master's degree in computer engineering from the Universiteit Gent (Belgium) in 2000 and upon graduation he joined Cisco Systems as a system administrator in the same group as Maarten. In his spare time he tinkers with computers, sings, beatboxes and cooks – usually all at once. You can reach him at wmertens@cisco.com .

Acknowledgements

Many thanks to Lee Damon and Bart Lauwers for reviewing and proofreading this paper.

Bibliography

[1] Apple computer, inc., http://www.apple.com/ .

[2] Cisco systems, inc., http:/www.cisco.com/ .

[3] Gnu/linux, http://www.linux.org/ .

[4] Microsoft Corporation, http://www.microsoft.com/ .

[5] Sun Microsystems, Inc., http://www.sun.com/ .

[6] Tadpole Computer, http://www.tadpolecomputer.com/ .

[7] *System and Network Administration 1990*, Sun Microsystems, Inc., March, 1990.

[8] *Windows Server 2003: Active Directory Infrastructure*, Microsoft Press, 2003.

[9] Anderson, P. and A. Scobie, *Lcfg – The Next Generation*, UKUUG Winter Conference, http://www.lcfg.org/doc/ukuug2002.pdf, 2002.

[10] Arnold, Edward R., *The Trouble with Tripwire*, Technical report, SecurityFocus, http://www.securityfocus.com/infocus/1398 , 2001.

[11] Burgess, M., and R. Ralston, "Distributed resource administration using cfengine," *Software: Practice and Experience*, Num. 27, 1997.

[12] Craig, Wesley D., and Patrick M. McNeal, "Radmind: The Integration of Filesystem Integrity Checking with Filesystem Management," *Proceedings of The 17th Annual Large Installation Systems Administration Conference (LISA 2003)*, San Diego, California, http://www.usenix.org/events/lisa03/tech/full_papers/craig/craig.pdf , October, 2003.

[13] Kim, G. and E. Spafford, "Experiences with Tripwire: Using Integrity Checkers for Intrusion Detection," *Proceedings System Administration, Networking, and Security, III*, 1994.

[14] Tridgell, A. and P. Mackerras, *The rsync Algorithm*, June, 1996.

[15] Wahl, M., T. Howes, and S. Kille, *RFC 2251 – Lightweight Directory Access Protocol (V3)*, Technical report, The Internet Society, 1997.

[16] Wilson, G. Samuel, Jr., *Solaris 10 Filesystem Integrity Protection Using radmind*, http://www.sans.org/rr/whitepapers/detection/1617.php , Technical report, SANS Institute, 2005.