Enterprise Mac Administrator's Guide

Charles S. Edge Jr.,
Beau Hunter,
Zack Smith

Apress®

Enterprise Mac Administrator's Guide

Copyright © 2009 by Charles S. Edge Jr., Beau Hunter, and Zack Smith

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13 (pbk): 978-1-4302-2443-3

ISBN-13 (electronic): 978-1-4302-2444-0

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

Lead Editor: Michelle Lowman
Technical Reviewers: Joe Kissell, Dee-Ann LeBlanc, and Brad Lees
Editorial Board: Clay Andres, Steve Anglin, Mark Beckner, Ewan Buckingham, Tony Campbell, Gary Cornell, Jonathan Gennick, Michelle Lowman, Matthew Moodie, Jeffrey Pepper, Frank Pohlmann, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh
Project Manager: Debra Kelly
Copy Editors: Katie Stence and Sharon Terdeman
Composition: ContentWorks, Inc.
Indexer: Ann Rogers/Ron Strauss
Artist: April Milne

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail orders-ny@springer-sbm.com, or visit http://www.springeronline.com.

For information on translations, please e-mail info@apress.com, or visit http://www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales–eBook Licensing web page at http://www.apress.com/info/bulksales.

The information in this book is distributed on an "as is" basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

To Lisa & Emerald, with love
—Charles S. Edge Jr.

Dedicated to my wife, Monica, who, despite completely losing me to the world of bits and bytes for the last six months, has been a source of constant support.
—Beau Hunter

Contents at a Glance

Contents

About the Authors

Charles S. Edge Jr. is the Director of Technology at 318, which is based in Santa Monica, California, and is the largest Mac consultancy in the United States. At 318, Charles leads a team of more than 40 engineers and has worked with network architecture, security, and storage for various vertical and horizontal markets. Charles maintains the 318 corporate blog at 318.com/techjournal as well as a personal site at krypted.com.

Charles is the author of a number of titles on Mac OS X Server and systems administration topics, including three titles from Apress for Mac OS X 10.6. He has spoken at a variety of conferences including DefCon, Black Hat, LinuxWorld, Macworld, MacSysAdmin, and the Apple WorldWide Developers Conference. Charles is the developer of the SANS course on Mac OS X Security and coauthor of its best practices guide to securing Mac OS X as well. Charles now lives in Minneapolis, Minnesota, with his wife, Lisa, and sweet little daughter, Emerald.

Beau Hunter has worked professionally with Apple technologies since 1999 and has supported businesses running the Mac OS for more than 10 years. Throughout this time, he has developed a strong skill set supporting and securing Apple OS X Server in multiple capacities: clustered web and database solutions, cross-platform integration, high-performance SANs, high-capacity backup systems, automation, and cross-platform mass deployment and integration.

Beau has spoken at numerous events, including Macworld 2009, and has been confirmed to speak at Macworld 2010. In his free time he can be found writing Python and PHP, playing PC games, and rooting for the Seattle Seahawks. In November 2009, Beau and his wife, Monica, will be returning to their true home— Seattle, Washington.

Zack Smith has worked as an IT consultant his entire adult life. He has consulted for insurance companies, entertainment companies, medical organizations, and governmental agencies. Zack is an Apple Certified Trainer and as such has taught Apple's Security Best Practices and many other Apple Certified System Administrator–level classes, such Mac OS X Deployment and Mac OS X Directory Services, at Apple and various market centers in Boston, Virginia, Los Angeles, and Cupertino. . Zack has spoken at Macworld San Francisco and at smaller venues as well, such as IT user groups. Zack is the author of a set of open source IT administration software and scripts and has long-term plans to be a full-time Objective-C developer. When not attending IT and security conferences or traveling for work at 318, Zack can be found in Portland, Oregon, with his partner in crime, Anna, and dog, Watson.

About the Technical Reviewers

■**Joe Kissell** is Senior Editor of *TidBITS*, a Web site and e-mail newsletter about the Mac and the Internet, and the author of numerous print and electronic books about Mac software, including *Take Control of Mac OS X Backups* and *Take Control of Upgrading to Snow Leopard*. He is also a Senior Contributor to *Macworld* and was the winner of a 2009 Neal award for Best How-To Article.

Joe has worked in the Mac software industry since the early 1990s and previously managed software development for Nisus Software and Kensington Technology Group. He was named one of MacTech's 25 most influential people in the Mac community for 2007. When not writing about Macs, Joe likes to cook, travel, watch movies, and practice tai chi. He also runs a number of Web sites, including JoeKissell.com and the popular *Interesting Thing of the Day* (itotd.com). Joe lives in Paris with his wife, Morgen Jahnke, and their cat, Zora.

■**Dee-Ann LeBlanc** has been into computers since she first got her hands on one and shortly after had her first Apple computer. Since then she's done help desk work, technical consulting, computer books and articles, and technology journalism covering a variety of platforms. Her specialties include Linux, open source, OS X, and content management systems.

■**Brad Lees** has more than 12 years of experience in application development and server management. He has specialized in creating and initiating software programs in real estate development systems and financial institutions.

His professional career has been highlighted by his positions as Information Systems Manager at The Lyle Anderson Company of Scottsdale, Arizona; Product Development Manager for Smarsh; Vice President of Product Development for iNation; and Information Technology Manager at The Orcutt/Winslow Partnership, the largest architectural firm in Arizona, based in Phoenix.

A graduate of Arizona State University, Tempe, Brad and his wife, Natalie, reside in Phoenix with their four children.

Acknowledgments

I'd like to first and foremost thank the Mac OS X community. This includes everyone from the people that design the black box to the people that dissect it and finally the people that help others learn to dissect it. We truly stand on the shoulders of giants. Of those at Apple that need to be thanked specifically: Schoun Regan, Joel Rennich, Greg Smith, JD Mankovsky, David Winter, Stale Bjorndal, Cawan Starks, Eric Senf, Jennifer Jones, and of course the one and only Josh Wisenbaker. A special thanks to Randy Saeks for his contribution to the title and to Michael Bartosh without whom any of the directory services content in this title likely would not have been possible. Also, thanks to the crew at 318 for their hard work. Without you guys I would never have been able to take the time to complete this book!

Finally, a special thanks to the fine staff at Apress for tuning this book to be a well-oiled machine of prose and code. This especially includes Clay Andres for getting the book kick-started and, of course, Debra Kelly, the best whip cracker I have had the joy of working with to date. Thanks also to my coauthors, Beau and Zack, for tirelessly working with me to meet our deadlines — it was a fun ride!

Charles S. Edge Jr.

Introduction

In the beginning was the command line. You can automate anything and everything in Mac OS X, but knowledge of the command line will be required to fully automate your deployment and integrate Mac OS X in the enterprise while maintaining a low total cost of ownership. This isn't to say you can’t integrate Mac OS X into a large organization en masse without using the command line — you can.However, from automation to troubleshooting, opening up a terminal window will be key to keeping your sanity, if only from time to time. But don’t fear the terminal, and know that the fundamental tasks required and the fundamental methodologies with Windows deployments are the same as with Mac OS X.

If you are reading this book, then you are likely charged with integrating Macs into your environment, whether kicking and screaming (which we hope this book will change) or as the sponsor. The message that you take away from this book is hopefully that you can do anything you want to with Mac OS X, from deploying 10,000 machines overnight to building a petabyte worth of storage to house all sorts of data for your Macs, provided you are not averse to learning a little bit of command line to achieve your goals. The power and flexibility of Mac OS X along with the best of the open source community is right at your fingertips to help along the way.

The first question many in IT ask when told about the need to use the command line is, "**B**ut isn’t Mac OS X supposed to be easy to use." It is. But we're not talking about just using the Mac. We're talking about building and managing a complicated IT infrastructure, which at some point requires staff that is tooled with the mastery of the internals of each platform for which they are tasked as the steward. As such, the more you learn about internals, the more you learn about the basics, the more you can automate, the more you learn about what goes on under the hood, the more you can master management en masse, and, ultimately, the more appropriately you will be able to address issues and concerns on an enterprise-wide scale as they arise. To take this a step further, the more you learn about managing a second platform (no matter what the platform is), the better you will be at managing others. But drastic reduction in Total Cost of Ownership is possible with OS X compared to other platforms for a variety of reasons. And since users are typically happier on a Mac, who wouldn’t want a happier user base combined with lower recurring costs.

Paradigm Shifts

Just as when enterprise computing was young, you will need to rethink some of your strategies to accommodate for a wider variety of platforms, resulting in a paradigm shift of sorts. But luckily you are not alone, and the jump is not as bad as many seem to think. There are a number of resources to help you through the process. From web sites to books, from Apple engineers to third-party providers/channel partners, from e-mail lists to user groups, you are not on an island. And while it is not fully open source, the Mac platform is a largely community-driven affair. One of our contributions to that community is this book, where we take on the lofty task of bridging the gap between your enterprise and your Mac.

The fundamentals of designing a Mac-based enterprise are the same as with any other platform — the specifics are not. In any enterprise organization you will need to perform a mass deployment, whether all at once or a refresh cycle performed on an ongoing basis. Every enterprise will also need centralized servers that provide a number of services to hosts on the network, including directory services, shared storage, groupware, and application servers. But the software that provides the needs of an enterprise is often different with the Mac than with other platforms. This isn't to say that the functionality of solutions already in use in many organizations cannot be extended to cover Mac OS X. But in some cases it is going to garner a higher return on investment to prop up an entire infrastructure to support the Mac while in others you are best to leave your existing solutions in place and extend them to the Mac.

Mac OS X is a standards-compliant operating system — to a point. Given the support of a number of standards, Mac OS X can be integrated into a primarily Microsoft environment. This includes support for Active Directory, Exchange support (either through Entourage or natively with ActiveSync), DFS, SMB/CIFS, and NFS. Many Microsoft-centric solutions will work out of the box. But when compared to the features available to Windows-based users, you may find yourself frustrated with integrating systems on a large scale. Users may also be frustrated with certain features that are missing when moving from Mac to Windows. Ultimately some of these features can even result in needing to purchase a third-party solution, deploying a thin client-based solution, or using virtualization solutions to ease the pain of integration, be it temporarily or permanently.

None of these obstacles are insurmountable. Through each release of Mac OS X, the system has become more and more enterprise friendly. And with each subsequent release you can expect that trend to continue. But don't expect to be able to do business as usual; expect to slightly alter your way of thinking to a more open model of computing. That shift toward openness, once you get right down to it, will make the process far easier and far more rewarding and in the end will lead you to a new paradigm in how you deal with enterprise computing.

Measure Twice, Cut Once

This likely goes without saying, but here goes: Before you deploy and integrate on a large scale, test. Before you test, plan. The more you plan, the less work you will ultimately have to do. What do you need to plan for? In our experience, it all starts with directory services. This is why the very first chapter of the book jumps into directory services, and from there we cover further integration in the same order that most organizations build out that infrastructure. It varies between environments, but if you go through each chapter and take into account the technologies introduced, then you will be able to plan more holistically.

Mac OS X is a great platform and suitable for a bevy of uses, but not the right fit for providing a number of network services. Therefore, throughout the book you will find information for integrating with existing infrastructure that may or may not be more suitable given your shift in platforms (however extensive that shift may be). Aside from infrastructure, the Mac systems you are planning to deploy and support require users to be productive on them, something they may not be able to do within the confines of Mac OS X. The book ends with virtualization and thin client solutions that can be leveraged to provide services that otherwise would not be available to the Mac platform.

Application Availability

While the book covers virtualization, the best deployments are going to be those that don’t require any applications to be virtualized. If your organization has invested in leveraging a consumer model — a mixture of using cloud services and migrating client-based software into intranets — then the Mac is more likely going to be able to take on your software with ease. But if you are using a number of proprietary products that do not come with a Mac OS X client, then you may need to use some form of virtualization to bridge the gap.

Long term, though, you need a plan to migrate to applications that are cross platform in order to keep the costs for your Mac OS X clients at a minimum. There are a number of sites available to help you find software for the Mac, most notably versiontracker.com. But there will be times when the Mac software is not as advanced or well kept as the Windows versions. This can lead to frustration from end users who possibly once championed the platform. In this case you may have to virtualize the software or an entire operating system in order to achieve parity. But this is where testing on a per-group basis will become key to planning your deployment.

When testing, make sure each user in your pilot thoroughly tests each piece of software. Find the biggest power users in a group and ask them to be your testers. Their voices will often be heard the loudest when things don't go well. But if you can keep them involved in the process and communicate with them along the way, once you achieve success you will often have the best proponent you could ask for.

How This Book Is Organized

Sandwiched between chapters on directory services and virtualization there are a variety of other topics that have been near and dear to organizations big and small as they grapple with integrating Mac OS X. These topics have been broken down into a number of chapters, each playing a critical role and requiring specialized planning. A summary of the chapters, aimed at guiding your planning and deployment:

Chapter 1 - **Directory Services** is a critical aspect of Mac OS X integration. In this chapter we cover how to set up a directory services environment using Open Directory, Apple's own directory service solution. Whether you are an Active Directory environment, eDirectory, or some other variant of a supported directory service, you will need to become acquainted with the fundamentals of implementing Open Directory. Additionally, Open Directory can be leveraged to work with Active Directory, providing a compelling framework for policy management.

Chapter 2 - **Directory Services Clients** are as critical as directory services themselves. In this chapter, the focus is on how to configure the directory services client from the command line, allowing you to deploy complex and automated binding scripts. The script examples provided with Chapter 2 will, at a minimum, help to get any mass deployment of Mac OS X in motion, saving a considerable amount of time and giving a glance into best practices that can be applied to further automation topics that will arise throughout the book.

Chapter 3 - **Active Directory** deserves a dedicated chapter. Why? The binding process, while part of the directory services framework, is considerably different than that of the other directory services modules. The third-party solutions, requirements, roadblocks to a successful integration, and the methodology are just that different from the other directory services modules. These differences should show the considerable amount of development taken on by Apple in order to provide such a feature-rich Active Directory solution.

Chapter 4 - **Storage** is a requirement for any business. Sure, some pundits say that eventually storage will all be in the cloud, but it's not yet. And you need to automatically mount, log into, and configure storage in such a way that your Mac clients can connect to it, use it for home directories, synchronize it, and even share it out themselves if need be.

Chapter 5 - **Messaging and Groupware** mean productivity. In this chapter we look at the options for typing your Mac OS X clients into shared groupware services hosted on Microsoft Exchange and Mac OS X Server. We also look into implementing groupware-oriented policies in the environment and automatically configuring groupware applications as part of your deployment process.

Chapter 6 - **Mass Deployment**. Whether it's imaging, deploying the image, or automating the tasks that enable you to be closer and closer to the one-touch image, this chapter is all about providing a step-by-step process to accomplishing these tasks. However, over the past few years a number of solutions have emerged to make mass deployment infinitely easier for administrators. Therefore, of the tasks we follow through the steps, we will use a different solution for each, allowing you to see a spectrum of options.

Chapter 7 - Mac OS X has a rich **Client Management** framework. In this chapter we look at local and directory services–based deployments of policies and explore the options for extending existing solutions to cover client management.

Chapter 8 - By **Automating Administrative Tasks**, you as an IT professional (or the manager of an IT professional) will be freed up to take on enhancing how your business interacts with technology (or you'll learn to fish, sleep nights, etc.). In this chapter we take a deep look into scripting and other forms of automation. This is where mastery of the command can become absolutely critical.

Chapter 9 - **iPhones** are cool. They're popular and gaining a considerable footprint in the enterprise space, given the penchant for synchronizing with Microsoft Exchange and the robust Objective-C development platform. But how do you deploy and manage thousands of the things? And while you're doing that, how do you use the features for connecting to standard enterprise application sets? In this chapter we help you get there and introduce you to some tools and techniques to ease the burden.

Chapter 10 - **Virtualization**. You just can't do everything on the Mac that you can do in Windows XP, Windows 7, Linux, or any other operating systems you can think of. Therefore, we give you a whole chapter of virtualization and thin client best practices and deployment techniques to ease the burden of your now doubled operating system footprint if you embark on this convoluted journey.

Chaos Theory

There is no magic bullet for your deployment. Most environments are going to be different in some way, shape, or form from every other environment out there. But provided there is industry-standard infrastructure (and most vendors have long since moved into providing industry standards) then rest assured that there is some way to make your Mac clients integrate fairly seamlessly into the enterprise. Therefore, while we don't have a magic bullet to offer, we do have a plethora of options for a given situation, options you can use to cut costs, reduce required human capital, and free up IT staff for creating value to businesses rather than living in the IT cost center.

Chapter 1

Directory Services

A **directory service** is the software that stores, organizes, and provides access to information in a directory. In the context that we will use the term throughout this book, we mean a database of users, groups, computers, and network devices such as printers. The directory service supplies that database to client computers. In most enterprise, educational, and larger institutions, common directory service implementations range from Microsoft's Active Directory (AD) to Novell's eDirectory, as well as the open source Open LDAP. Most modern directory services are based on standards developed in the public forum.

The most common standard architectural guidelines are defined in the X.500 model "The Directory: Overview of concepts, models and services." While the concepts and roots of most directories are complex, by their very nature they share the simple goal of unified user management, authentication, and authorization. Directory servers with different origins thus find many commonalities in their structure and accessibility. The Lightweight Directory Access Protocol (LDAP), which is utilized by nearly every major directory service system, is a testament to this need for accessibility, as we will discuss later in this chapter. Put simply, any system engineered for large-scale centralized authentication must inherently allow disparate clients to participate, otherwise it is doomed to a finite growth potential.

In Mac OS X, there are a number of plug-ins that allow you to leverage a variety of different directory services. Each computer must at least contain a local directory service database to establish a baseline of system-critical data, such as users, groups, and even some configuration data. If every Mac OS X computer sold required an enterprise directory service just to login, Apple stores would not be popping up like Starbucks in cities around the United States. Local authentication is a cornerstone of all modern operating systems, and often the gateway for small and medium businesses to grow into larger directory systems over time. A common misconception is that Apple's *Open Directory* terminology is applied only to its enterprise-class authentication services. In reality, the same term refers to those local or client standards implemented in local accounts. In fact, in previous operating systems, Apple even had the same technology running on Open Directory masters, such as 10.2 `netinfod` and 10.3 Password Server. This concept of architecting what amounts to miniature directory servers into the base operating system allows for later migration to larger directory service systems without much reeducation of entry-level system administrators. The best example of this is Apple's parental controls system that, at its base, leverages the same technology used to manage thousands of Mac OS X in enterprise environments every day. Due to such forethought, clients can also be configured out of the box to utilize a variety of other external directory services; support for several network-based directory service systems is provided without the installation of any additional software.

This chapter starts with an explanation of how the local directory service works. Once we have explained how local users can be managed, we will move on to discuss LDAP, the industry-standard directory database used to supply directory services. Next, we will cover various types of binding to directory servers from Mac OS X that let end users log into their computers using a centralized username and password. Finally, we will look at building external accounts and show how to build a directory service based on Apple's Open Directory.

Local Accounts

In Mac OS X, System Preferences are similar to Control Panel in Windows, and they allow you to configure a wide range of settings. The information you set in these panes is stored in files throughout the operating system. Local directory service configuration is accessed through the Accounts preference pane, which provides the ability to add local user and group accounts. Accounts can also be added to groups, assigned a type, and a few other options can be set.

To access a System Preferences pane, click on the Apple in the top left corner of the screen and then on System Preferences, or launch the application directly from the /Applications folder. You will then be shown all of the System Preferences available. Next, click on Accounts and you'll see the list of Accounts on the left side of the screen. As you click through each one, you will see the options for that account on the right side of the screen. To make changes in this area, you must first authenticate to System Preferences by clicking on the lock in the lower left corner of the Preferences window. For the authentication to succeed, the user must be a member of the local directory service's admin group.

> **TIP:** The `/etc/authorization` file is used to determine which users are able to attain elevated privileges for a variety of operations. In a standard OS X environment, the admin group will be able to obtain escalation for all authorization rights. However, this file can be modified to provide very granular administrative access to users. For instance, to manage users via the System Preference pane, a non-admin group could be specified under system.preferences.accounts, which would then give its members administrative access solely to the Accounts pane of System Preferences.

Creating Accounts

To add an account, first click on the lock icon in the Accounts System Preferences pane, then click on the plus sign to create an account. In the Account: field you'll see the five options shown in Figure 1-1, which indicate the basic account types for Mac OS X. These include:

- Administrator: Administrative accounts, accounts with elevated privileges; can open System Preference panes and perform most tasks.
- Standard: Standard User accounts; cannot open System Preference panes and cannot perform administrative tasks.
- Managed With Parental Controls: Standard User accounts with policies applied to them.
- Sharing Only: Accounts that cannot log onto the local system but can access resources via file sharing protocols.
- Group: A group of user accounts.

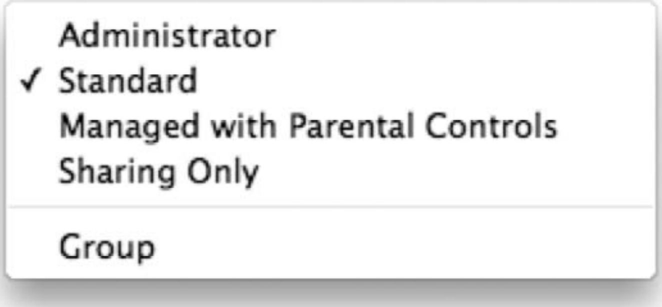


Figure 1-1. *Contextual menu for account types*

Once you have selected an account type, enter a full name in the Name: field and a short name in the Short Name: field. For example, the full name might be John Doe and the short name jdoe. By default, the short name is generated from the full name in lower case with spaces removed. The full name is primarily used for display purposes and can be changed at will. The short name has additional system-level functions. Notably, it is used to name a user's home directory when first created, though that directory can be changed to a different location that does not correspond to the short name (such as a "mystuff" folder on a external drive).

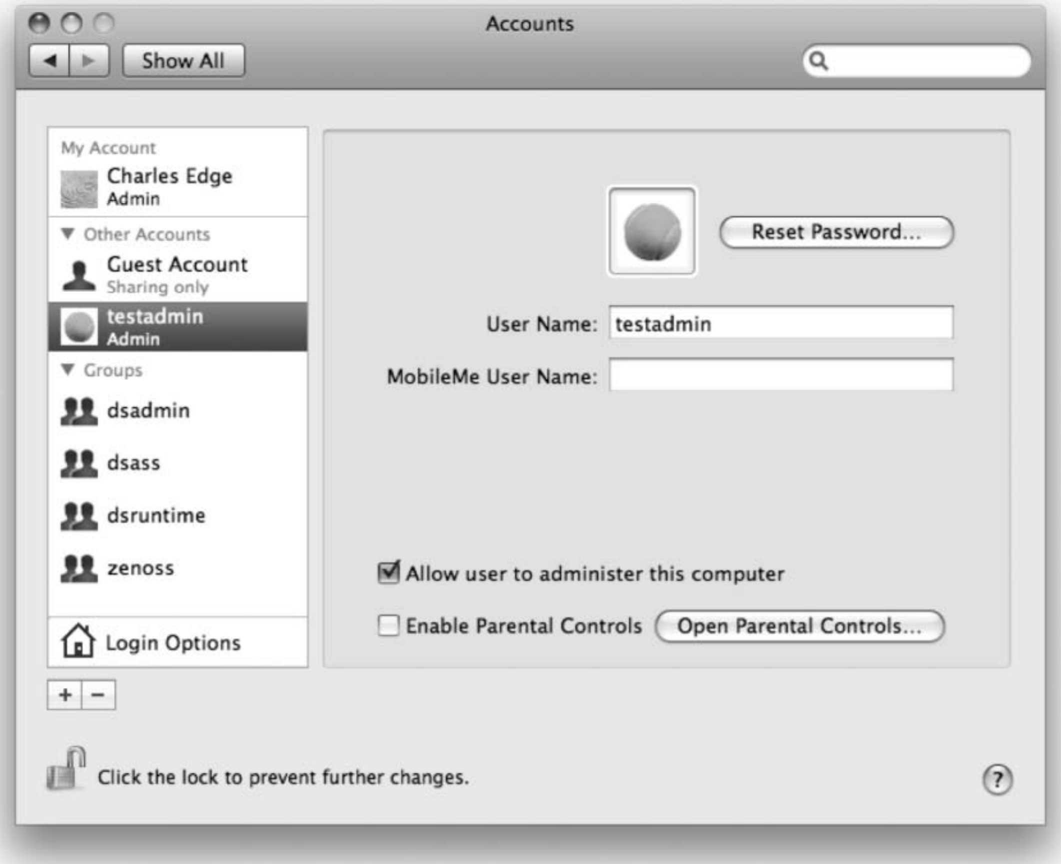
The short name is used for other purposes as well, such as establishing a primary email mailbox for the user or for linking scheduled items through cron. Because of this, setting the initial short name demands some consideration. It's also worth noting that the short name cannot easily be edited in the prominent user interface, and though right-clicking on a user account and choosing Advanced Options allows you to edit this name (as seen in Figure 1-3), doing so has other repercussions, such as loss of group membership (such as admin); possible loss of preference data if an application stores configuration data based on the short name; or disassociation of the user's home folder. In most cases when you plan to modify a user's short name, you will also want to rename his home directory to coincide. This is merely for cosmetic reasons and is not a necessity. You can change short name jdoe to psherman and still utilize the original home directory stored at /Users/jdoe. If you do change the home directory to /Users/psherman, you should make sure you rename the user's home folder on the file system to match the new path specified (in this case, from the original home directory value /Users/jdoe to /Users/psherman).

Next, enter the password the user will use in the Password: field and then enter it again into the Verify: field. The small key icon in this dialog box will reveal the Password Assistant, an interface that assists users with choosing strong passwords by supplying them with visual feedback. This functionality is available as a stand-alone program using third party applications available on the Internet, and can also be accessed via the Keychain Access application when you create a new password item. Optionally, you can enter a hint as to what the password is in the Password Hint: field. If a password hint is set for a user, it will be displayed when the user fails to authenticate when logging in. Here you can also check the box to enable FileVault, which encrypts the contents of the user's profile or home folder.

When you are satisfied with your settings, click on the Create Account button. You have now created your first Mac OS X user. If you are done making changes, you should close the lock options available in the Security system preference pane, which will cause the System Preference to forget your previous authentication each time the application is reopened during your timed session. Alternatively, if you forget to close the lock, the elevated privileges will time out.

Granting Administrative Privileges

As noted earlier, you can choose to make a user an administrator of the local computer when you create an account. To elevate an existing account to an administrative account, you can simply check the Allow user to administer this computer checkbox, as shown in Figure 1-2. To set up basic policies for an account, you can click on the Open Parental Controls button for any non-administrator account and enable them. (We will cover more in-depth policies on local and network directory services accounts further in Chapter 7, Client Management.)



User Name: testadmin

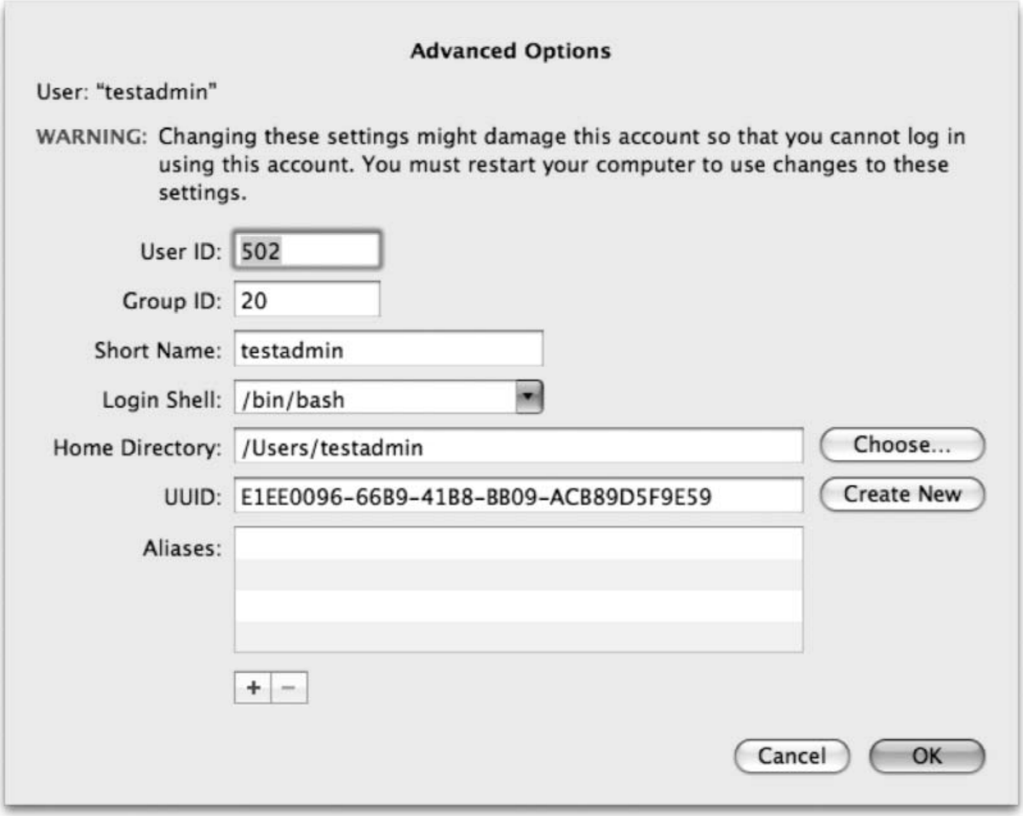
MobileMe User Name:

☑ Allow user to administer this computer

□ Enable Parental Controls

Figure 1-2. *Making a user an administrator*

As mentioned previously, you can also edit some slightly more advanced settings from within the Accounts System Preference pane. These settings are accessible by control-clicking on the account name and then clicking on Advanced Options, which brings up a screen similar to the one in Figure 1-3. This screen lets you change the values for various attributes of the accounts, including Short Name, User ID, default group, path to the home folder, default shell and the generated ID for the account. You can also add aliases using the plus sign; this allows the same account to authenticate using multiple names in the authentication dialogs throughout the operating system. We will discuss these attributes later in the chapter.



Advanced Options

User: "testadmin"

WARNING: Changing these settings might damage this account so that you cannot log in using this account. You must restart your computer to use changes to these settings.

User ID: 502

Group ID: 20

Short Name: testadmin

Login Shell: /bin/bash

Home Directory: /Users/testadmin

UUID: E1EE0096-66B9-41B8-BB09-ACB89D5F9E59

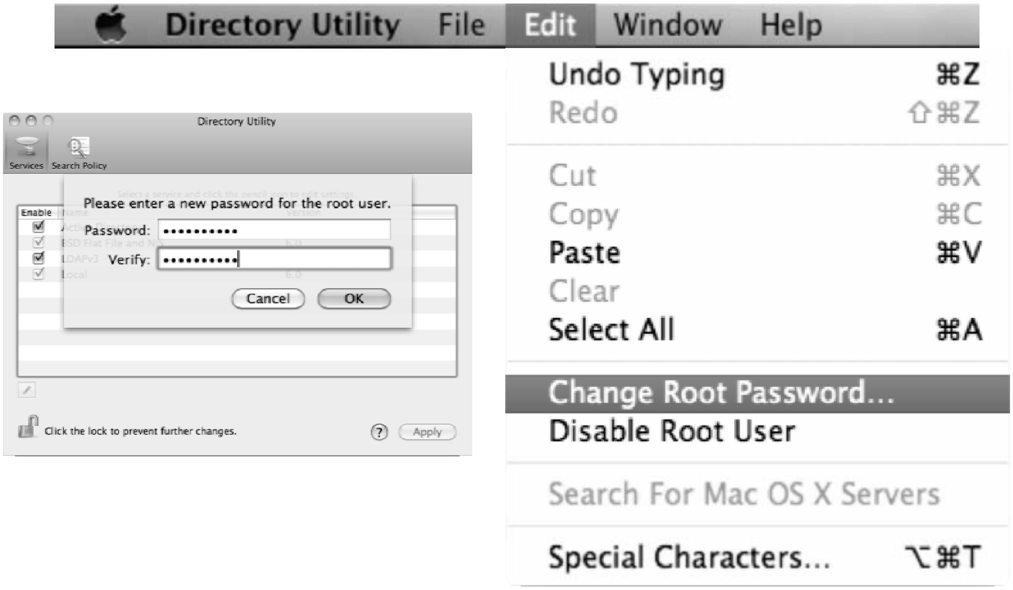
Aliases:

Figure 1-3. *Advanced account options*

The Root Account

In a Unix, BSD, or other *nix environments, the root account can do things that even standard administrators typically can't do. A root account can be a security risk, which is why Apple has disabled root by default, but it is an account you may find you need to enable from time to time. If you are new to administering Mac OS X from the command line, you may wish to enable the root account for certain GUI operations that would otherwise use the command line, such as renaming a home folder or editing a configuration file owned by root.

To enable the root account, open the Directory Utility application found on the Accounts pane of System Preferences (version 10.6), or in the /Applications/Utilities folder (version 10.5). As with most secure operations in Mac OS X, you will need to authenticate to perform this action using the lock in the corner of this window. Then click on the Edit menu and select Enable Root User, which will display the screen shown in Figure 1-4. Next, enter the password that will be assigned to the root user and click on OK.



Please enter a new password for the root user.

Password: ••••••••••

Verify: ••••••••••

Click the lock to prevent further changes.

Figure 1-4. *Enabling Root in Directory Utility*

You can also enable the root account using the command line. The `dsenableroot` command can be used to enable the root user and assign it a password. To enable root, enter:

```
dsenableroot
```

First you will be prompted for the current user password; this user must be an administrative account. You will then be prompted twice, first for a password to assign the root account and then to verify the password. On success you'll see the following success code:

```
dsenableroot:: ***Successfully enabled root user.
```

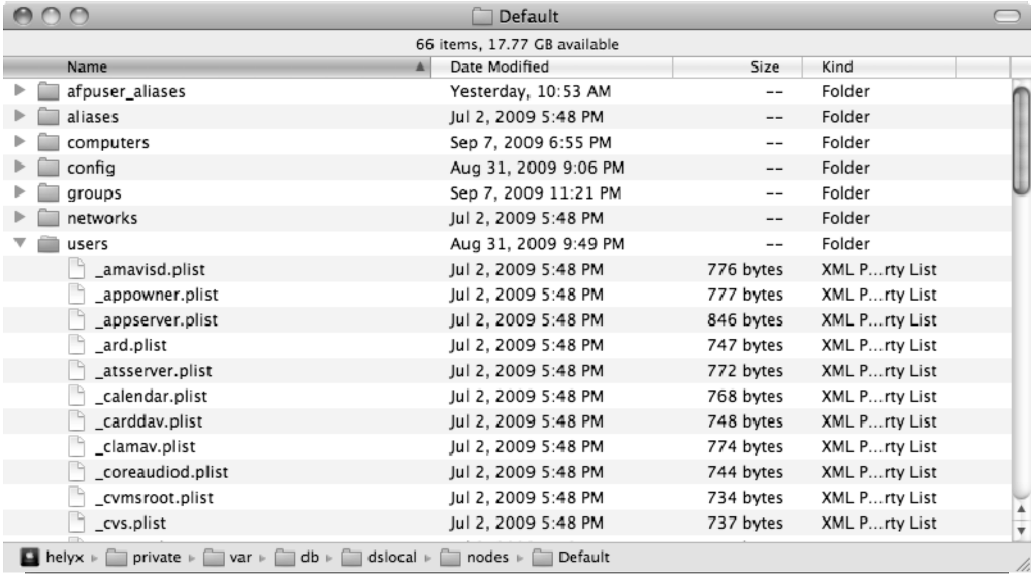
To disable the root account, enter:

```
dsenableroot -d
```

TIP: It is best to leave the root account disabled when you do not need it. If you do enable it, do so only temporarily.

How the Local Directory Service Works

The local directory data resides primarily in the folder found at /private/var/db/dslocal. This folder, which will require elevated privileges to access, contains numerous files pertaining to the computer's directory service configuration. For instance, accounts for Users and Groups are stored in flat property list (.plist) files nested in the /private/var/db/dslocal/nodes/Default directory. Users are stored in /private/var/db/dslocal/nodes/Default/users while groups are stored in /private/var/db/dslocal/nodes/Default/groups. Every local user and group account has a corresponding .plist file found in these directories, as seen in Figure 1-5, which shows the contents of /private/var/db/dslocal/nodes/Default/.



Default

66 items, 17.77 GB available

Name	Date Modified	Size	Kind
afpuser_aliases	Yesterday, 10:53 AM	--	Folder
aliases	Jul 2, 2009 5:48 PM	--	Folder
computers	Sep 7, 2009 6:55 PM	--	Folder
config	Aug 31, 2009 9:06 PM	--	Folder
groups	Sep 7, 2009 11:21 PM	--	Folder
networks	Jul 2, 2009 5:48 PM	--	Folder
users	Aug 31, 2009 9:49 PM	--	Folder
_amavisd.plist	Jul 2, 2009 5:48 PM	776 bytes	XML P...rty List
_appowner.plist	Jul 2, 2009 5:48 PM	777 bytes	XML P...rty List
_appserver.plist	Jul 2, 2009 5:48 PM	846 bytes	XML P...rty List
_ard.plist	Jul 2, 2009 5:48 PM	747 bytes	XML P...rty List
_atsserver.plist	Jul 2, 2009 5:48 PM	772 bytes	XML P...rty List
_calendar.plist	Jul 2, 2009 5:48 PM	768 bytes	XML P...rty List
_carddav.plist	Jul 2, 2009 5:48 PM	748 bytes	XML P...rty List
_clamav.plist	Jul 2, 2009 5:48 PM	774 bytes	XML P...rty List
_coreaudiod.plist	Jul 2, 2009 5:48 PM	744 bytes	XML P...rty List
_cvmsroot.plist	Jul 2, 2009 5:48 PM	734 bytes	XML P...rty List
_cvs.plist	Jul 2, 2009 5:48 PM	737 bytes	XML P...rty List

Figure 1-5. *Contents of a dslocal node*

The above output is trimmed, but each folder will contain a plist file for each respective user, computer, or group in the local directory. Accounts that begin with an underscore (_) are hidden service users and groups. For example, the web server uses the_www account, which obtains user settings from the _www.plist file. The _www user can't log in because the account has no shell or password. If you created a new user in the above section, look in the /private/var/db/dslocal/nodes/Default/users directory and you should see a .plist file with a name that corresponds to the new user's short name.

Inside a .plist file there are a number of attributes containing data about a given user or group. Looking at local users and groups from a Microsoft Windows perspective, files in the local directory node resemble registry keys for local accounts. Examine the .plist file for the user created earlier and look for the key called authentication_authority.

```
<key>authentication_authority</key>
<array>
                <string>;ShadowHash;</string>
</array>
```

This key specifies the service that will be utilized to authenticate the user. Notice that it says ShadowHash, which indicates that the system will use a local file called a hash file to authenticate the user. Mac OS X password hash files contain copies of a user's password in multiple formats; this Rosetta Stone allows for different services to authenticate a user with their own native password encryption type. If this were not the case, the password would need to be stored in a much less secure reversible hash in order to support the various authentication schemes out there. It also should be noted that for ShadowHash users, any network service that does not support SHA-1 (Secure Hash Algorithm 1) or NTLM (NT LAN Manager) authentication will require cleartext authentication; SSL is highly recommended in these scenarios.

In the user's plist file, you will also see a `generateduid` key, which is used to track the user account even if the short name is changed. GeneratedUIDs are based on a standard called the Universally Unique IDentifier (UUID), which is a complex, programmatically generated string of characters that will never be duplicated in our lifetime. A UUID is unique across time and space for every user.

If you look in the /private/var/db/shadow/hash directory, you will find a file that is named using the value of this key. This means that even if a user account's username is changed, the password will still be tied to that account. Moreover, it prevents stale password files from collecting, which would happen if passwords were based on the short name. In 10.4 and later, the password hash file will contain at least a SHA-1 salted hash for the user, which is a secure, unrecoverable password type. If Windows file-sharing services are enabled for the user, it will also contain the respective NTLM hash for that user, which is used by our Windows file-sharing components. Apple has struggled to implement the best balance of security and functionality in regard to password hashes. While hashes for Windows file sharing require NTLM, the NTLM hash type is more susceptible to common password attacks, which makes its recoverability more feasible. Apple only enables the NTLM hash when Windows file-sharing users are specifically configured for SMB/Windows sharing access in the System Preferences Sharing pane. Storing passwords in a hash file allows for a consistent password file location, with flexible extensibility for other password hashes such as NTLM. In the above example, the authentication_authority record, which has a value of `;ShadowHash;`, tells the local directory service to consult the user's local hash file when the user attempts to authenticate.

The data from the account property lists can be managed by modifying the text files directly. For example, if you want to change a user's picture, you could alter the picture key. However, editing property lists directly can be pretty cumbersome, so Apple has provided a host of commands that can be used to manage and query data from the local directory node and other directory services plug-ins without having to read raw XML-style property list data. Some commands have GUI equivalents while others do not. Here are some of the commands:

- `dirt`: used to test authentication in 10.4 and 10.5, tests authentication, for example `dirt -u zsmith -p 'd0gc4t'`. The only GUI equivalent would be the login window or an authentication screen. As of 10.6, the dirt utility is no more; the dscl utility now performs this role.
- `dscacheutil`: looks up information stored in the Directory Services cache and flush various caches
- `dscl`: used to edit and browse directory services settings, such as user accounts, group accounts, and search policies (the order in which Mac OS X looks up account information in each directory service). The closest GUI equivalents would be the Accounts System Preference pane and the Directory Utility. This command is covered in more depth in the next section.
- `dseditgroup`: used to edit, create, and delete groups or to add or remove group members.
- `dsenableroot`: manages the root user account (enable, disable, and reset the root password). The GUI equivalents are the Change Root Password and Enable Root User or Disable Root User options in the Edit menu of Directory Utility.
- `dserr`: prints a description of Directory Services-related errors, example `dserr 14090`. Once you have the error code, you can use the man page for DirectoryService to look up the meaning of each error (or Google for more information on the specific errors, but quote errors if there is a – in front of the number).
- `dsexport`: exports directory services data. Similar functionality is available using the Export feature of Workgroup Manager, a tool distributed as part of Mac OS X Server.
- `dsimport`: imports directory services data. Similar functionality is available using the Import feature of Workgroup Manager.
- `dsmemberutil`: looks up UUIDs and group information and flush group cache, for example `dsmemberutil flushcache`.
- `dsperfmonitor`: run performance monitors of the directory services plugin, useful with debugging operations, for example `dsperfmonitor -dump`.
- `id`: look up a user identity, including group memberships, for example `id zsmith`.