Securing User Accounts

Over the years, Apple has gone to great lengths to make OS X strong by hardening its user security. *Hardening* means strengthening a component in a system to make it more secure. User accounts can be hardened in OS X by limiting the resources users can access. You can accomplish this two ways: by using the built-in GUI tools, or by using the command line. In this chapter, we will go deeper into securing the Mac by focusing first on restricting user access and then on more advanced command-line security that can be used to harden user accounts.

Introducing Identification, Authentication, and Authorization

*Identification* is determining who a user is, or what something is. *Authentication* is an operating system’s attempt to verify the digital identity of someone or something attempting to communicate with the computer, such as a request to log in to the system. The sender attempting to be authenticated might be an actual person using a computer, a computer itself attempting to authenticate, or a computer program looking to run software on the machine. For example, if you were attempting to install software, authentication is the act of your Mac asking for your username and password and verifying that you are who you say you are.

**NOTE:** Because authentication is the act of establishing or confirming something as authentic, a key security component of authentication is to protect authentication attempts, such as passwords or key pairs in transit.

*Authorization* is a process that determines whether a user has access to a given resource. All modern multiuser operating systems include an authorization process. This gives the operating system the ability to identify users and then verify whether they have the appropriate credentials to access a resource. Permissions are generally defined by a system administrator, either on the computer itself or in a networked environment. On
Mac OS X, the authorization process starts with the /etc/authorization file, which contains rules used when authorizing users.

Authorization can also be extensible using pluggable authentication modules (PAMs), which were initially developed by Sun Microsystems. A PAM allows multiple authentication schemes to be integrated with new software so that programs that rely on authentication can be written independently of the underlying authentication scheme.

Verifying that the person sending you information is really who they say they are is obviously desirable all the time. Authentication, however, is time-consuming, and can inconvenience users while resulting in overhead on the authentication services. To make situations like this more convenient and efficient, many systems use a method of Identification, a procedure that verifies the person or entity authenticating is the same one it communicated with when they last authenticated. The means of identification can be established through the use of a ticket or token issued when authentication is complete. This saves the user from being required to authenticate on each communication with the server.

**WHEN IS SECURE TOO SECURE?**

One example of a situation where security and usability often collide is in the deployment of one-time passwords. A one-time password is a password that must be changed each time you log in to your computer. This effectively makes any intercepted password good for only the brief interval of time before the legitimate user logs in the next time. This way, if someone intercepts a password, it would probably already be expired or be on the verge of expiration within a matter of hours.

For nearly every situation, this is too much security and impacts the ability of users to remember passwords. In our experience, when one-time passwords are deployed, it typically means users are writing passwords down. Anytime your password policy causes users to have to write down their passwords, it is a good idea to review whether you are being too strict with password policies. For example, if passwords are too complicated to remember, people will invariably write them down. This is how the kid in the movie *War Games* got into his school computer, thus sparking an entire generation of hackers.

**Managing User Accounts**

Mac OS X is a multiuser operating system, therefore every file on the system is owned by a user, and every process is run by a user. One great way to analyze the processes in use on your system, along with the name of the users running them, is to view them using the Activity Monitor, accessible via the Utilities folder on the machine. As you can see in Figure 3–1, a variety of accounts are listed in the User column, each running separate processes. Activity Monitor has permissions to manage processes that the user can access. Therefore, processes are restricted from accessing data they should not be able to access. This security extends to files and other system resources on your computer.