SECURING MAC OS X LEOPARD (10.5)

Project Reference	Corsaire Whitepaper - Securing Mac OS X Leopard 10.5 v1.0
Author	Daniel Cuthbert
Date	18 August 2008
Distribution	General release

Securing Mac OS X Leopard (10.5)

Introduction

Mac OS X Leopard (10.5) continues in the tradition of the previous operating system, Tiger, by providing one of the most secure default installations of any desktop operating system. The operating system also includes new features and additional tools to help users manage the security of their data. The install follows the accepted best practice of disabling network services unless explicitly requested by the user, and the default security settings suit the needs of most users in a workstation scenario.

While the default installation provides a relatively secure system, it may not always meet organisational security requirements.

This guide is aimed at users in environments requiring stronger security controls in their operating system, making full use of the protection features offered by Mac OS X 10.5. It may also be of use to System Administrators wishing to enforce an organisation-wide desktop security policy.

This guide is an updated version of the *Securing Mac OS X Tiger (10.4)* and includes the new security features offered by Mac OS X Leopard (10.5).

Table of Contents

Securing Mac OS X Leopard (10.5)

1. Using this Guide

This guide covers the security features of Mac OS X Leopard (10.5.4) as a multi-user networked system. Most of the console and network based security features are common between Mac OS X and Mac OS X Server, however this guide does not cover Server's additional user, directory and network based security features.

The reader should be familiar with using the UNIX command line and editing plain text configuration files. Most of the operations require administrator access and Corsaire would recommend that each file be backed up before it is edited.

In Leopard, some property list (plist) files used to store system and application preferences use a binary format by default. They can be converted to a text format using the *plutil* command or accessed using the Property List Editor application. For example, to edit the file /Library/Preferences/com.apple.loginwindow.plist the following command can be run from the terminal:

```
$ sudo open /Library/Preferences/com.apple.loginwindow.plist
```

Settings are added using key and value pairs, where the key is case sensitive[1].

While every effort has been made to test the settings specified in this guide, no guarantee can be made as to their effectiveness or suitability for individual systems.

Any changes to a users system are made at the users own risk; Corsaire accepts no liability for losses incurred.

2. New Security Features in OS X Leopard

Whilst Leopard introduces a host of new features, only a small portion of these relate to security. This section provides a summary of the features Corsaire feels offer the most value in terms of added protection. A full list of security features may be found in Apple's "Mac OS X Security[2]" document.

[1] When saving the file, the "Save as..." function should be used, since the Save function results in a permission error due to the default admin rights being read-only.

[2] http://images.apple.com/macosx/pdf/MacOSX_Leopard_Security_TB.pdf

2.1.1 Application Tagging and First Run Warning

Application tagging provides a means of identifying software downloaded from the Internet (a potentially untrustworthy location). The tags, associated as file meta-data, can be used to identify and warn users that the content may be unsafe and should be executed with caution. The user must authorise a downloaded application before the operating system executes it. Only files downloaded using certain applications (such as Safari and iChat) currently have the appropriate facilities to tag such code. This is a much needed step in the prevention of malicious content execution. It also gives the user information about when it was downloaded, the application used to download it, and which URL it was downloaded from.

2.1.2 Sandboxing

The Leopard XNU kernel now has mandatory access controls available. The benefit of this addition is that it is now possible to write security policies, which determine what any given program can and cannot do. This provides another step in the direction of limiting the impact of malicious code and software exploits. For example, a policy can be defined to limit Safari's interaction with the operating system to prevent an exploit from adding a new user, or Mail.app can be prevented from downloading and installing malicious content.

2.1.3 Application Firewall

Previous versions of Mac OS X provided a network level firewall, which was effective at stopping network level attacks but not suited to controlling access to applications. Leopard has introduced application-specific firewall functionality to resolve this issue, with the ability to create 'per application' policies to control incoming network communications. The system also signs unsigned applications that are authorised by the user, which helps ensure that the policies are only applied to signed applications (locally or by a trusted third party).

The current implementation allows users to specify access for services and applications in both directions. An example of this would be allowing Skype to have inbound connections but denying Microsoft Word from receiving inbound connections.

2.1.4 Address Space Layout Randomisation (ASLR)

Many types of attacks involve corrupting program memory. In most cases, if attackers can corrupt the targeted memory, they may be able to divert the execution of the program in question from using its own code and instead using the attackers' code (the classic buffer overflow[3], for example). Leopard now randomises[4] the functions and their locations in memory, preventing attackers from easily determining their location in advance, therefore making it more difficult for attackers to implement exploits.

Although this implementation of ASLR makes Leopard more resistant to traditional stack and heap based buffer overflow attacks, and integer attacks, it should not be considered a panacea for all exploits.

2.1.5 Input Manager Restrictions

Leopard implements stronger controls over input managers, an overdue enhancement for Mac OS X. Originally input managers were loaded from the user's home directory at ~/Library/InputManagers. This in itself was a security risk as there was no centralised control of which input managers were loaded.

Leopard only installs input managers from the /Library/InputManagers folder; any other locations are ignored. Files in the /Library/InputManager location have to be owned by the root user and the admin group. Furthermore, any process running as root, or in the wheel group, is unable to load any bundle input manager. This does not stop a user from manually moving files into this group and changing the permissions, but it does make it more difficult for malicious applications to install themselves.

2.1.6 Parental Controls

Leopard saw an upgrade to the parental controls function, which allows users control over the resources each account has access to. Whilst the name implies it is aimed at children, its use is far more powerful.

Parental controls are enforced in the kernel, which allows for the creation of restricted accounts. For example, an account can be set up to only allow access to Safari, which can be useful in a kiosk environment. Kernel level control means this account can effectively be prevented from executing arbitrary programs via flaws within the exposed applications.

---

[3] http://www.owasp.org/index.php/Buffer_Overflow

[4] There are doubts as to the approach adopted by Apple regarding their ASLR implementation, see: http://www.matasano.com/log/981/a-roundup-of-leopard-security-features/

2.1.7 Code Signing (Digital Signatures)

Leopard implements a facility to digitally sign applications. The main feature of code signing is to verify that an application has not changed since it was created. Many types of malware change programs on the system, which when invoked by users or the operating system can attempt to further subvert security. However, when using code signing, executing a modified signed application causes an alert to be generated and the user to be notified. This can help to warn users of malware infection.

3. Security Hardening Guidelines

3.1 Summary of Security Guidelines

This hardening guide provides coverage to the following areas:

- *Common Criteria Tools* – Explaining the Common Criteria ISO Standard and available Apple tools.
- *Managing Users* – How to control what users are allowed to perform on the system.
- *General System Security Settings* – General security settings and preferences.
- *Networking and Services* – Information about the available services and how to deploy them securely.
- *Logging and Auditing* – Using logging, process accounting and auditing to maintain accountability.
- *Additional Security* – Additional programs to enhance overall system security.

3.2 Common Criteria Tools

The Common Criteria for Information Security Technology is an ISO standard for computer security. The standard provides assurance that the vendor has completed a rigorous test to ensure their product has met the strict requirements specified by the Common Criteria standard.

Apple has released the Common Criteria Tools for Mac OS X 10.5 with the following statement[5]:

"By providing an independent assessment of a products ability to meet security standards, Common Criteria gives customers more confidence in the security of Information Technology products and leads to more informed decisions."

These tools can be downloaded from:
http://www.apple.com/support/downloads/commoncriteriatoolsfor105.html

3.3 Managing Users

3.3.1 Controlling Administrative Access

The default installation of Mac OS X comes with the root user disabled, which prevents the standard process of logging into or using switch user (su) to access the root account. This reduces the exposure to many common attacks traditionally aimed at UNIX operating systems.

It is possible to enable the root user but this is strongly discouraged, and use of administrative users and sudo is recommended instead.

3.3.2 Administrative Users

The default user account is an administrative account; this in itself proves a security risk as access to administrative accounts should be controlled.

The access control mechanisms of the system may be further secured by granting administrative rights to only specific users. For each administrative user, there should be two user accounts, one to perform normal user operations, and the other to perform administrative functions. For example, if the user James is a designated administrator he should have a standard system account "james" with no special privileges and an administrative account "admin_james" with administrator rights. This provides accountability where there is more than one administrator on a system. The administrative users should be restricted from logging into the system from network services using their administrative accounts. This further reduces the risk of authentication credentials being compromised.

[5] http://www.apple.com/support/downloads/commoncriteriatoolsfor105.html

3.3.3 Sudo

Since the root user is disabled, it is not possible to use the su command to obtain root privileges. Mac OS X instead makes use of the sudo program.

By default Leopard allows all administrative users access to the sudo command and it allows these users to run any program with sudo. In some circumstances, this may contravene system usage policies. In these cases, it is possible to disallow sudo access to the administrator group and instead enable it on a per user basis.

From the terminal, edit the /etc/sudoers file by typing:

```
$ sudo visudo
```

Insert a hash (#) character, in front of the line:

```
#admin ALL=(ALL) ALL
```

To allow only the user ‘bob’ access to sudo add the line:

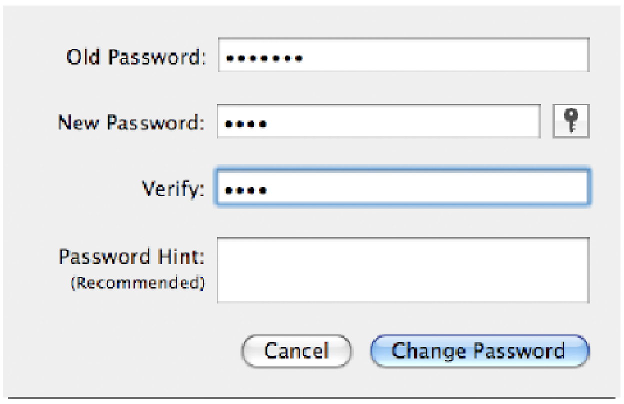
```
bob      ALL= (ALL) ALL
```

Make sure that at least one user has permissions to run sudo before saving the file! Access controls within the sudoers file can be specified minutely. For example, it is possible to grant the user James access to the file /usr/bin/kill, but only with the privileges of user Tim. See the sudo man-page for more details on tightening access controls through sudo.

3.3.4 Passwords and Password Assistant

Strong password choice is a fundamental part of a system’s security. On multi-user and networked systems it is even more important that users adhere to organisational standards for password choice and password management.

The password assistant is a user-friendly application that assists users in choosing good quality passwords. It can be accessed by clicking the key button that is present on all password choice dialogs. For example, when changing the account password:

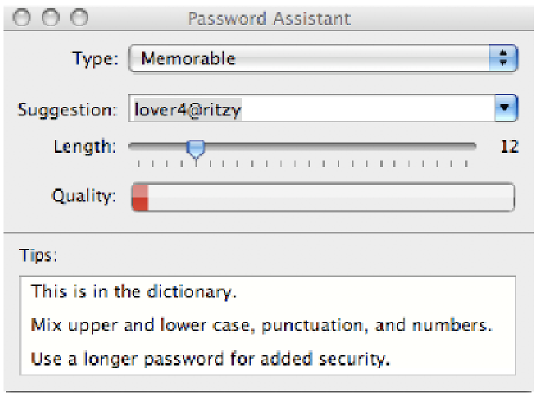


Old Password: •••••••

New Password: ••••

Verify: ••••

Password Hint: (Recommended)



Type: Memorable

Suggestion: lover4@ritzy

Length: 12

Quality:

Tips:

This is in the dictionary.
Mix upper and lower case, punctuation, and numbers.
Use a longer password for added security.

The assistant provides feedback to the user as to the quality of the chosen password in the form of a colour-coded bar, and provides tips on improving the password. The assistant also suggests strong, memorable, passwords for users as an alternative.

3.3.5 System Wide Passwords

Leopard provides the *pwpolicy* command line tool to allow administrators to set per user and global password policies, including the ability to specify:

- Password strength in the form of length and character set.
- Password expiration.
- Password re-use restrictions.
- Maximum number of failed authentication attempts.

However, it is not possible to determine whether a password is based on a dictionary word (apart from via the password assistant tool), nor can the use of special characters or mixed case be enforced.

In the absence of an organisational password policy, the following good practices are recommended:

- Users cannot reuse the last 12 passwords.
- Passwords must be at least 8 characters in length.
- Passwords must contain at least 1 alphabetic and 1 numeric character.
- Passwords must be of mixed case and contain at least 1 special character – *this cannot be enforced through pwpolicy and should be achieved through user awareness training.*
- After 5 failed authentication attempts the account is locked out.

This can partially be implemented using the following command:

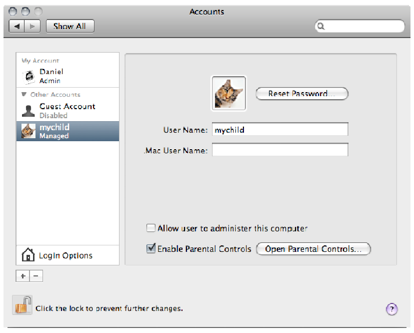
```
$ sudo pwpolicy -a adminusername -setglobalpolicy "usingHistory=12
minChars=8 requiresAlpha=1 requiresNumeric=1 maxFailedLoginAttempts=5"
```

Where *adminusername* is the name of the current administrative user, see the pwpolicy main page for further information on its use.

3.3.6 Parental Controls

Mac OS X Leopard improves the parental control function, which can be used to limit the functionality of user accounts. For example, it may be desirable to create a low privilege account that has access to a specific application set (e.g. Safari.app and Mail.app), with restrictions on how these applications can be used, and which can only be used during office hours. This may be a useful setup for a company's Reception, where access to computing resources must be controlled.

To enable parental control, select the option from *System Preferences -> Parental Control:*

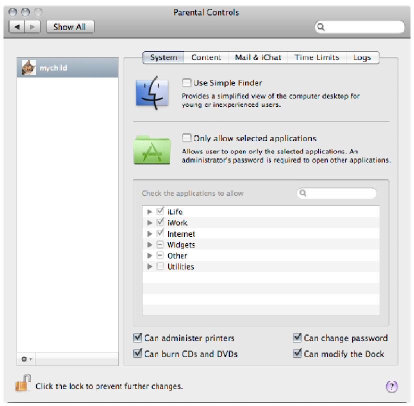

Accounts
Show All
Reset Password...
Login Options
Open Parental Controls...
Click the lock to prevent further changes.

My Account
Daniel
Admin
Other Accounts
Guest Account
Disabled
mychild
Managed

User Name: mychild
.Mac User Name:

☐ Allow user to administer this computer
☑ Enable Parental Controls

Once you have selected the user you wish to control, the control panel allows you to fine-tune the amount of access that account has:



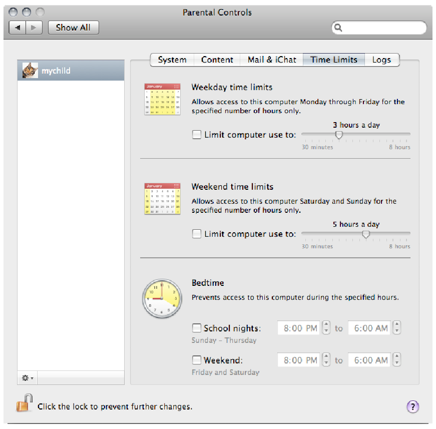
☐ Use Simple Finder
Provides a simplified view of the computer desktop for young or inexperienced users.

☐ Only allow selected applications
Allows user to open only the selected applications. An administrator's password is required to open other applications.

Check the applications to allow
- ☑ iLife
- ☑ iWork
- ☑ Internet
- ⊟ Widgets
- ⊟ Other
- ☐ Utilities

☑ Can administer printers
☑ Can burn CDs and DVDs
☑ Can change password
☑ Can modify the Dock

You can also limit when the user is allowed to use the computer:



Weekday time limits
Allows access to this computer Monday through Friday for the specified number of hours only.
☐ Limit computer use to: 3 hours a day (30 minutes – 8 hours)

Weekend time limits
Allows access to this computer Saturday and Sunday for the specified number of hours only.
☐ Limit computer use to: 5 hours a day (30 minutes – 8 hours)

Bedtime
Prevents access to this computer during the specified hours.
☐ School nights: 8:00 PM to 6:00 AM
Sunday – Thursday
☐ Weekend: 8:00 PM to 6:00 AM
Friday and Saturday

3.3.7 Guest Account

Mac OS X introduced a guest account, which removes the requirement of a temporary user having a full account on the system. Apple claims that the guest account is purged at logout, which ensures data is kept private, by restricting access to other users' data. Furthermore, any files downloaded by the guest account are removed on logout. **Corsaire has found this to be untrue, with files created by the guest account in /tmp and /Users/Shared still accessible once the guest account has been logged out.**

By default, the guest account is not enabled under Mac OS X Leopard. This is good practice, as the guest account should only be enabled if it is required, as this account is in itself quite powerful. A guest account user can:

- Install cron jobs which run even if the guest account is not logged in.
- Mount remote file systems and have the mount remain static after the account has logged out.

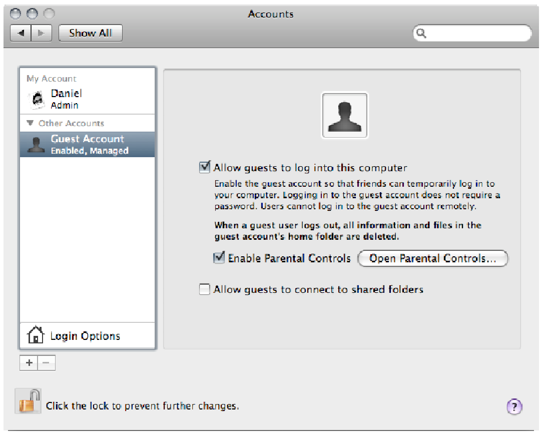
The following is still possible, even when the guest account has been logged out:

- The guest account can view other users' folders with the exception only of standard folders, such as desktop, library and documents.
- The guest account can create background processes and have them continue running even when the account is logged out.

Corsaire would recommend that if the guest account is required, appropriate steps should be taken to remove unnecessary privileges and access to the system. One way to harden the guest account is to use the previously discussed parental controls function and fine-tune what the guest account is allowed to perform.

In addition, the Umask should be changed as described in 3.4.10.1 to prevent the guest account from reading users' files.

To enable the guest account, select the option from *System Preferences -> Accounts*:



☑ Allow guests to log into this computer

Enable the guest account so that friends can temporarily log in to your computer. Logging in to the guest account does not require a password. Users cannot log in to the guest account remotely.

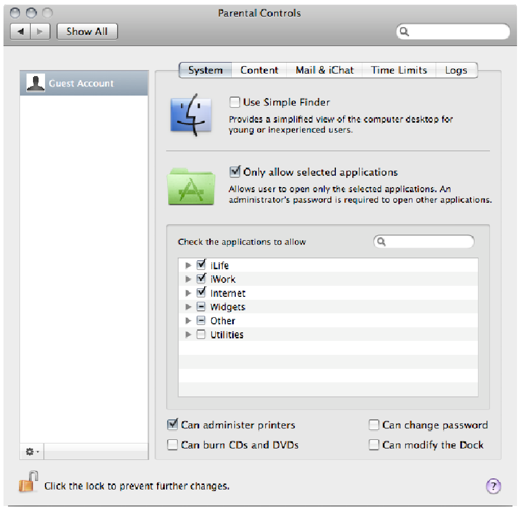
When a guest user logs out, all information and files in the guest account's home folder are deleted.

☑ Enable Parental Controls Open Parental Controls…

☐ Allow guests to connect to shared folders

Click the lock to prevent further changes.

Once the account is enabled, click on Open Parental Controls and restrict system access as appropriate:



☐ Use Simple Finder

Provides a simplified view of the computer desktop for young or inexperienced users.

☑ Only allow selected applications

Allows user to open only the selected applications. An administrator's password is required to open other applications.

Check the applications to allow

- ☑ iLife
- ☑ iWork
- ☑ Internet
- ⊟ Widgets
- ⊟ Other
- ☐ Utilities

☑ Can administer printers
☐ Can change password
☐ Can burn CDs and DVDs
☐ Can modify the Dock

Click the lock to prevent further changes.

3.4 General System Security Settings

There are a number of general system security settings which can be activated to increase the level of security beyond the default install of Apple OS X Leopard.

3.4.1 Physical Access Controls

In environments where attackers can gain physical access to the system, it is important that additional security mechanisms are in place to protect the system from unauthorised access.

Securing Mac OS X Leopard (10.5)

If an attacker gains physical access to a system, they can boot an alternate operating system and read data stored on the hard drive, or enable a firmware password that can render the system inoperable. A secure solution is to control physical access to systems by putting them under lock and key, but sometimes this isn't possible – especially for mobile users or desktop users in shared environments.

3.4.1.1 Firmware Security

There are two types of firmware currently used by Apple:

- *Open Firmware* for Apple hardware still using the PPC chipsets.
- *Intel EFI firmware* for the newer hardware utilising the Intel chipsets.

Firmware security changes not explicitly endorsed by Apple may result in permanent damage to the computers logic board.

3.4.1.2 Accessing Open Firmware (OF)

Open Firmware (OF) is the BIOS used by Apple PPC systems, and is used to provide low-level control to some parts of the hardware. OF uses a command line driven interface more akin to that used by Sun Microsystems than the menu-driven BIOS used by x86 PCs. For the purposes of securing the system, two operations need to be performed in OF: setting a password, and changing the security mode. These features are only available in OF version 4.1.7 or 4.1.8, depending on the computer (see http://docs.info.apple.com/article.html?artnum=106482).

The changes to the firmware described below are made directly from the OF command line. Apple has released a graphical tool that sets the firmware password[6].

OF password protection offers the ability to:

- Block the ability to use the "C" key to start up from an optical disc.
- Block the ability to use the "N" key to start up from a NetBoot server.
- Block the ability to use the "T" key to start up in Target Disk Mode (on computers that offer this feature).

[6] This tool can be found on the installation media under */Application/Utilities/Open Firmware Password.app* on the installation disk.

- Block the ability to start up in Verbose mode by pressing the Command-V key combination during startup.
- Block the ability to start up a system in Single-user mode by pressing the Command-S key combination during startup.
- Block a reset of Parameter RAM (PRAM) by pressing the Command-Option-P-R key combination during startup.
- Require the password to use the Startup Manager, accessed by pressing the Option key during startup.
- Require the password to enter commands after starting up in OF, which is done by pressing the Command-Option-O-F key combination during startup.
- Block the ability to start up in Safe Boot mode by pressing the Shift key during startup.

To access the OF command line, the system should be rebooted and Command-Option-O-F held down while the system boots. A screen similar to the following should be presented:

```
Apple PowerMac,4 4.4.9f1 BootRom build on 11/13/02 at 13:41:09
Copyright 1994-2002 Apple Computer, Inc.
All Rights Reserved

Welcome to Open Firmware, the system time and date is: 02:36:52 01/15/2003
Full security mode.

To continue booting, type "mac-boot" and press return.
To shut down, type "shut-down" and press return.

ok
0>
```

3.4.1.2.1 Setting a Firmware Password

From the OF command line, type:

```
password
```

When prompted, enter and re-enter the chosen password. The password should comply with the organisational security policy or generally accepted good practices:

```
0> password
Enter a new password: ************
Enter password again: ************
Password will be in place on the next boot! Ok
0>
```

Once the password is set, it is necessary to set the security mode to one of the three values: *none*, *command* or *full*, which are described in more detail below:

- **None** – This is the default setting and provides no OF security protection. Even if a password is set, it has no effect if the security mode is set to none. This mode also makes it possible to set another firmware password without first entering the old one.
- **Command** – This setting causes the system to prompt for a password when any changes to OF are attempted. It also requires a password when booting from any device besides the default boot device.
- **Full** – This mode requires that a password be entered before booting and before any changes are made to OF. A password is required before *every* reboot.

Once the appropriate security mode has been selected, it can be set by typing:

```
setenv security-mode full
```

To save the changes and reboot, type:

```
reset-all
```

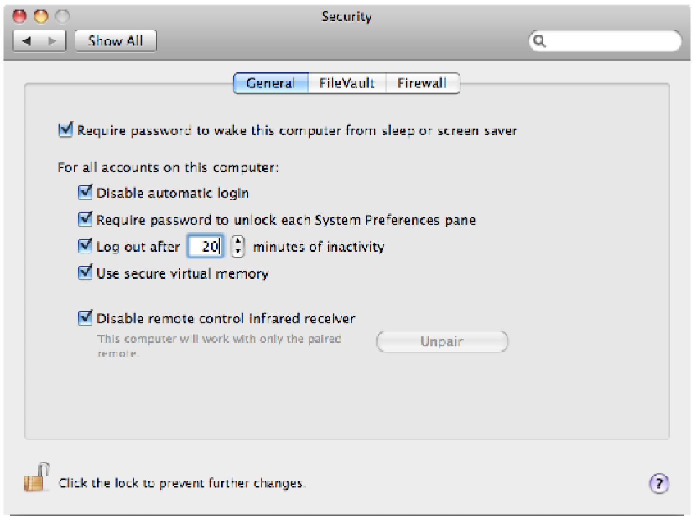
3.4.1.3 EFI Firmware

EFI Firmware was introduced for Intel Macs and offers the same degree of control as OF.

Using the EFI Firmware password application, it is possible to block the same functions as the OF with the added addition of being able to block the ability to use the "D" key to start up from the Diagnostic volume of the Install DVD.

3.4.2 Security Preferences

A number of system-wide security settings can be accessed from the *System Preferences -> Security pane*:



☑ Require password to wake this computer from sleep or screen saver

For all accounts on this computer:

☑ Disable automatic login

☑ Require password to unlock each System Preferences pane

☑ Log out after 20 minutes of inactivity

☑ Use secure virtual memory

☑ Disable remote control infrared receiver

This computer will work with only the paired remote.

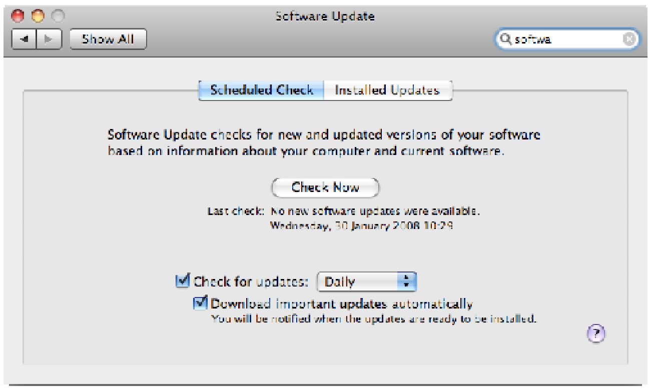
Click the lock to prevent further changes.

These security settings allow the user to:

- Set the requirement of a user having to enter a password to wake the computer from sleep mode or to unlock the screen saver.
- Disable automatic login by presenting a logon screen.
- Require the administrator password to unlock any of the System Preferences panes.
- Log the user out after being inactive for a specific amount of time.
- Make use of secure virtual memory, which stops any information in memory from being read.
- Disable the remote control function of the system which could otherwise be controlled via infrared.

3.4.3 Software Update

Mac OS X uses the software update tool to download and install system and application patches. Corsaire would recommend that it be configured to check for new updates daily, to be configured to perform downloads in the background and to notify the user when the update is ready for installation. The preferences for software update can be found under *System Preferences -> Software Update:*



Software Update checks for new and updated versions of your software based on information about your computer and current software.

Last check: No new software updates were available.
Wednesday, 30 January 2008 10:29

☑ Check for updates: Daily

☑ Download important updates automatically
You will be notified when the updates are ready to be installed.

3.4.3.1 Command Line Software Updates

Software updates can also be listed and applied through the command line tool: /usr/sbin/softwareupdate. This makes it possible to install updates using shell scripts, or to invoke the update process remotely through a remote administration facility, such as SSH. For example, the following command could be executed to automatically install all required updates and log the output to a file:

```
sudo /usr/sbin/softwareupdate -i -a 2>&1 >> /Library/Logs/auto-softwareupdate.log
```

Caution should be used when adding software update to the crontab, as some required updates need the system to be manually rebooted before taking effect.

3.4.4 Setting a Login Banner

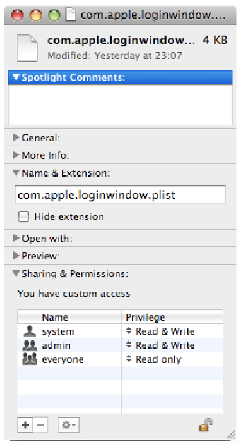
A login banner informs users accessing a system about the system's function, ownership and consequences of unauthorised access. This information should be displayed at all points of entry to the system, usually, login prompts on the desktop, shell logins and other application access prompts. An appropriate login banner should be defined, after consultation with the organisation's legal team.

An example login banner could be similar to:

```
This is a private computer system and is for authorised use only.
Any or all use of this system and all files on this system may be intercepted and monitored.
```

```
Unauthorised or improper use of this system may result in disciplinary and/or legal action.
By continuing to use this system you indicate your awareness of and consent to these terms and
conditions of use.
LOG OFF IMMEDIATELY if you are not an authorised user of this system or do not agree to the
conditions stated in this warning.
```

The first place that this banner should be displayed is at the desktop login prompt where all local users see it. To insert a login banner in the Mac OS X login window, first navigate to /Library/Preferences and 'get info' (right click or command-i) on the file *com.apple.loginwindow.plist*, and change the access control list to allow read and write access for admin users:



com.apple.loginwindow....

com.apple.loginwindow... 4 KB
Modified: Yesterday at 23:07

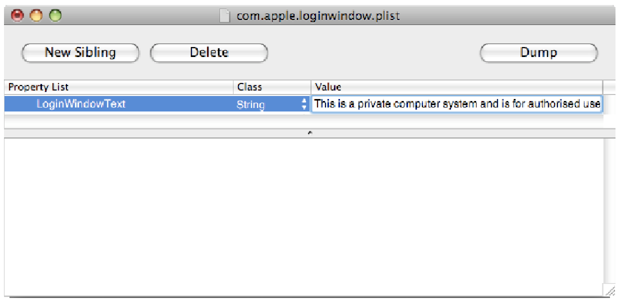
Name & Extension: com.apple.loginwindow.plist
Hide extension [ ]

Sharing & Permissions:
You have custom access

Name	Privilege
system	Read & Write
admin	Read & Write
everyone	Read only

Next, edit the file /Library/Preferences/com.apple.loginwindow.plist and insert the key named *LoginwindowText* with the value of the login banner that should be displayed. The file can be edited as follows:

```
sudo open /Library/Preferences/com.apple.loginwindow.plist
```



Property List	Class	Value
LoginWindowText	String	This is a private computer system and is for authorised use

The login window with the new text is displayed after a reboot.

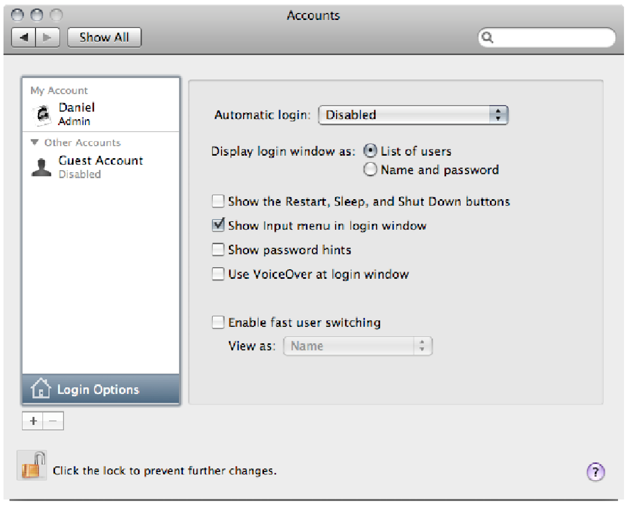
Note: This message will not be displayed when switching users using the fast user-switching feature.

A number of other services should use the same login banner; Corsaire would recommend a text file containing the banner be created as */etc/login_banner*. Since each service that displays the banner formats it according to its own protocol, the length of each line should be less than 70 characters (less than the default for many terminal applications).

3.4.5 Displaying Usernames

By default, Mac OS X displays a list of usernames with an accompanying graphic at the console login prompt. This provides potentially useful information for attackers with physical access (for example, shoulder surfing attacks) and should be disabled. Disabling this feature requires the user to enter their usernames and passwords.

Disable this setting from: *System Preferences -> Accounts -> Login Options -> Display Login Window as: Name and password*:



Automatic login: Disabled

Display login window as: [✓ List of users] Name and password

- [ ] Show the Restart, Sleep, and Shut Down buttons
- [x] Show Input menu in login window
- [ ] Show password hints
- [ ] Use VoiceOver at login window
- [ ] Enable fast user switching

View as: Name

3.4.6 Password Hints

Password hints allow users to set a reminder to help recover their forgotten passwords. While this is a helpful feature for some home users who don't login very often, it is typically not appropriate in a corporate environment, as it increases the risk of an attacker successfully guessing the password.

To disable password hints on the system, open the file /Library/Preferences/com.apple.loginwindow.plist as root, in the Property List Editor application:

```
sudo open /Library/Preferences/com.apple.loginwindow.plist
Change the RetriesUntilHint value to 0.
```

3.4.7 Restart, Sleep and Shutdown

The restart, sleep and shutdown buttons are provided on the login screen. It is possible to prevent these buttons from being displayed in the login window.

To prevent the buttons from being displayed, deselect the option from *System Preferences -> Accounts -> Login Options -> Show the Restart, Sleep and Shutdown buttons.*

3.4.8 Screensaver

A screensaver should be activated after a short period of inactivity, and should require a password to unlock the workstation. This prevents unauthorised passers-by from accessing an unattended workstation that is logged in.

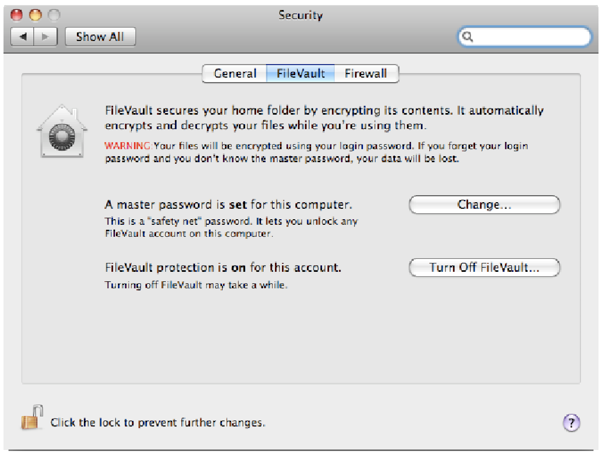
The screensaver can be enabled from *System Preferences* -> *Desktop & Screensaver*. To enable password protection on the screensaver, "*Require password to wake this computer from sleep or screensaver*" should be selected from the System Preferences -> Security Panel.

3.4.9 Data Encryption

Mac OS X provides built in data encryption features using the AES[7] algorithm with 128 or 256 bit keys. This allows users to encrypt data with military strength cryptographic functions.

3.4.9.1 File Vault

The main user encryption feature is the FileVault facility, which encrypts and decrypts a user's entire home folder, protecting the data from unauthorised physical access. Decryption is performed in real-time as needed and appears seamless to the user:



FileVault secures your home folder by encrypting its contents. It automatically encrypts and decrypts your files while you're using them.

WARNING: Your files will be encrypted using your login password. If you forget your login password and you don't know the master password, your data will be lost.

A master password is **set** for this computer.
This is a "safety net" password. It lets you unlock any FileVault account on this computer.

FileVault protection is **on** for this account.
Turning off FileVault may take a while.

Click the lock to prevent further changes.

The user's login password is used to decrypt the encrypted folder. An additional 'master password' may also be set which is able to decrypt *all* FileVault protected folders on the

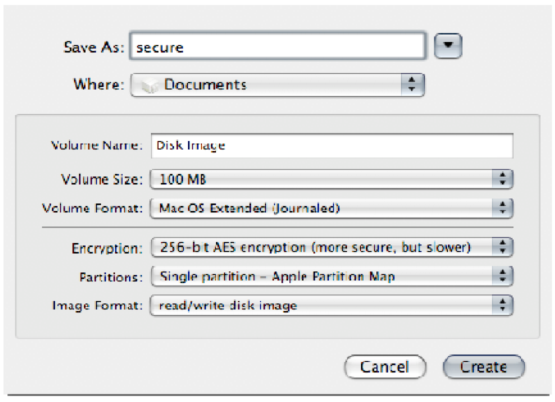
[7] http://csrc.nist.gov/CryptoToolkit/aes/

system. This provides the ability to recover a user's data should they forget their password or leave the organisation, allowing the holder access to all users' encrypted data. Organisational access control policies should dictate whether this is desirable, and how the master password should be managed. The ability to decrypt user data is currently a requirement of the UK's Regulation of Investigatory Powers (RIP) Act (http://www.homeoffice.gov.uk/crimpol/crimreduc/regulation/).

FileVault can be enabled on a per-user basis from the *System Preferences -> Security Pane*.

3.4.9.2 Disk Utility

The disk utility application (*Applications* -> *Utilities* -> *Disk Utility*) can also be used to encrypt data. When a new image is created, '256-bit AES (more secure, but slower)' should be selected as the *encryption setting*. A password is required to decrypt the image when mounted. This is especially useful for exchanging encrypted data, or for mobile users who wish to store their data on external drives. An option is provided to store the password for an encrypted volume in the user's keychain; this is useful for most users and should only be ignored by users with the utmost concern for the confidentiality of their data:



Save As: secure
Where: Documents
Volume Name: Disk Image
Volume Size: 100 MB
Volume Format: Mac OS Extended (Journaled)
Encryption: 256-bit AES encryption (more secure, but slower)
Partitions: Single partition – Apple Partition Map
Image Format: read/write disk image

3.4.10 File System Security

The initial account used to administer the system, as well as any accounts created prior to changing the umask, allowa all users read access to the files in their home folders. Corsaire would recommend that this be changed so that only owners and groups have read access to these files. For all users on the system, execute the following command (where username is the name of the user):

```
sudo chmod -R 740 /Users/username
```

This operation has to be performed every time a new user is added to the system.

This has the added effect of preventing other users from reading the contents of ~/Public and ~/Sites folders and from writing files to the ~/Public/Drop Box folder. Corsaire would recommend that the permissions on these folders be changed on a per user basis.

3.4.10.1 Default Umask

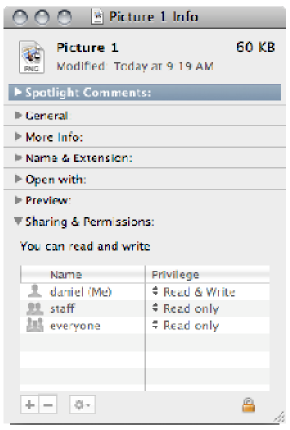
The default umask on Mac OS X systems is 022, which means that by default all users are granted read access to all newly created files. Corsaire would recommend that this be changed to 027 so that only users in the same group are permitted automatic read access to new files. Organisational security standards may dictate an alternative umask. Since the umask setting must be specified as a decimal value, the octal value 027 would be 23 in decimal notation. To change the default umask, execute the following command:

```
sudo defaults write /Library/Preferences/.GlobalPreferences NSUmask 23
```

3.4.10.2 Access Control Lists

Access Control Lists (ACLs) were introduced with Mac OS X 10.4. They allow for a more granular control of file system resources than was previously offered through standard BSD file permissions.

By default, ACL support is turned on in Leopard. As part of this integration, users can now specify basic ACLs using the finder and assign arbitrary read and write rights to be granted to any user, not just the UNIX method of user/group/everyone permissions.



Picture 1 60 KB
Modified: Today at 9:19 AM

You can read and write

Name	Privilege
daniel (Me)	Read & Write
staff	Read only
everyone	Read only

The ACLs implemented by Mac OS X 10.5 follow the POSIX standard and are entirely compatible with ACLs on Microsoft Windows systems. With Leopard, Mac OS X can finally serve files and exist as a full peer on a Microsoft Windows network.

3.4.10.3 Roles, Permissions and Resources

Access controls define the actions that a *role* can perform on a *resource.* A role is typically a user or a group of users. Permission can be an action like *read* or *write* and a resource can be a file or directory on the file system.

Standard UNIX file permissions provide three classes of 'roles' (owner, group and other) and three classes of *permission* (write, read and execute) that can be used to control access to a resource. While this provides an adequate access control mechanism in some environments, it can be cumbersome and unwieldy in many multi-user environments. In some cases, if user Bob wants to allow user Alice to read one of his files (but did not want anyone else on the system to read the file), he would have to create a new group, put Alice in the group, and then allow read access to that group – a fairly complex operation for a relatively straightforward policy. UNIX permissions simply do not offer the functionality needed to reflect the Access Control policy for a resource. For example, using UNIX permissions, it is not possible for a user to permit someone to delete a file, but not change (write) it – since there is no separate "delete" permission defined. Similarly, it is not possible to allow a user to append to a file, without also allowing them to change the existing contents.

ACLs address these issues by supporting more granular permissions and by allowing these to be assigned to roles (individuals, or groups of users), instead of only groups. The following tables list the supported permissions and the resources to which they can be applied:

All file system objects

Permission	***Description***
Delete	Delete the item. Deletion may be granted by either this permission on an object or the delete_child right on the containing directory.
Readattr	Read an object's basic attributes. This is implicitly granted if the object can be looked up and not explicitly denied.
Writeattr	Write an object's basic attributes.
Readextattr	Read extended attributes.
Writeextattr	Write extended attributes.

Readsecurity	Read an object's extended security information (ACL). When not explicitly denied, all users who can list the object can read its security attributes.
Writesecurity	Write an object's security information (ownership, mode, ACL).
Chown	Change an object's ownership.

Non-directory file system objects (e.g. files)

Permission	***Description***
read	Open for reading.
write	Open for writing.
append	Open for writing, but in a fashion that only allows writes into areas of the file not previously written.
execute	Execute the file as a script or program.

Directories

Permission	***Description***
list	List entries.
Search	Look up files by name.
add_file	Add a file.
add_subdirectory	Add a subdirectory.
delete_child	Delete a contained object.

A single ACL can be specified per resource, each ACL consists of a list of *ordered* Access Control Entries (ACEs). The semantics of an ACE follow the pattern:

Role [allow or deny] permission resource

This may be familiar to users of network based access control lists or firewall rules. Mac OS X uses the chmod command to manipulate ACLs. For example, the following command can be used to add an entry to an ACL that allows user Alice read access to a resource:

```
chmod +a "alice allow read" ./resource.txt
```

It is important to note that ACLs are acted on before the UNIX permissions and they operate on the principle that the first matching rule is the one that takes effect – in other words, the order of the list is significant. For example, consider the following file, BSD permissions and ACL:

```
-rwxrwxrwx + 1 alice  wheel    6 May  1 15:45 test.txt
 0: user:bob deny read
 1: user:bob allow read
```

The standard UNIX permissions permit all users (including Bob) read access to the file. However, entry 0 in the ACL explicitly denies read access to user Bob, while the subsequent rule allows him read access. In this case Bob is denied access, since rule 0 is the first matching rule.

3.4.10.4 Inheritance

ACLs set on directories can be inherited by subdirectories and files. The following inheritance permissions are applicable to directories:

Permission	***Description***
file_inherit	Inherit to files.
directory_inherit	Inherit to directories.
limit_inherit	This flag is only relevant to entries inherited by subdirectories. It causes the directory_inherit flag to be cleared in the entry that is inherited, preventing further nested subdirectories from also inheriting the entry.
only_inherit	The entry is inherited by created items but is not considered when evaluating the ACE for the given resource.

3.4.10.5 Manipulating ACLs

ACLs can be manipulated using the chmod command and the following modes:

Permission	***Description***
+a *entry*	Add entries to the list. Note that this mode inserts entries following the canonical form: • Local deny • Local allow • Inherited deny • Inherited allow
-a *entry*	Deletes the first matching entry.
+a# *position entry*	Inserts the entry at exactly ‘position’ in the list. Note that this does not follow the

Permission	Description
	same canonical form as the +a mode.
=a# *position entry*	The entry at ‘position’ is replaced by the entry provided.

To view ACLs, Leopard provides an additional option to the ls command: -e, typically used as: ls –le.

3.4.10.5.1 Example

Consider the following access control requirements for a folder and its contents:

1. All users are permitted to view file names and change into directories.
2. The user ‘test’ is allowed to read all files in the documents directory, but not in any subdirectories.
3. User ‘Bob’ is permitted to create new files and edit existing files, but is not permitted to create directories.
4. The group ‘admin’ is permitted to create directories but not to delete them or view their contents.
5. The user ‘Alice’ (who is not an administrator) is allowed to delete directories and files.
6. Anything not explicitly allowed in these rules, is denied.

Note: Strictly speaking, this access control policy is not enforceable on any system that implements discretionary access control, since administrative users always have the ability to change permissions or disable ACLs altogether, even if this right is not explicitly granted by the policy.

Since the UNIX permissions comes into effect if there is no matching ACL, for this example, the permissions on the directory are set to 000 i.e.

```
sudo chmod 000 documents
```

The umask also plays a role here, since newly created files use the UNIX permissions defined by the mask. An umask of 027 is assumed. The chmod command is used to manipulate the ACL and add ACEs. The +a method inserts the ACE in canonical form – this can have an adverse effect on the policy since canonical form may not express the policy exactly. Since the policy is clearly defined, the ACE is added with explicit entry numbers to ensure that it is applied in the intended fashion.

1. All users are permitted to view file names and change into directories:

```
sudo chmod +a  "everyone allow list,search,directory_inherit" documents
```

The ‘list’ right allows users to view the files in the directory (e.g. ls documents) and the ‘search’ right allows users to change into the directory. The ‘directory_inherit’ permission ensures that this ACE is also used on any subdirectories.

Note: subdirectories will only inherit the ACE if they are created after the entry was made.

2. The user ‘test’ is allowed to read all files in the documents directory, but not in any subdirectories:

```
sudo chmod +a# 1 "test allow read,file_inherit,directory_inherit,limit_inherit" documents
```

The ‘read’ attribute is only applicable to non-directory objects and is therefore only applied to contained files. The ‘limit_inherit’ permission ensures that this ACE is only applied to the documents directory itself, and not to any subdirectories.

3. User ‘Bob’ is permitted to create new files and edit existing files, but is not permitted to create directories:

```
sudo chmod +a# 2 "bob allow add_file,directory_inherit" documents
```

The ‘add_file’ permission is only applicable to directories, and only permits the role to add a file, not another directory:

```
sudo chmod +a# 3 "bob allow write,file_inherit,directory_inherit" documents
```

The write permission is only applicable to files, not directories. The ‘file_inherit’ permission is used so that the permission is inherited by files. Although Bob has write permission, he does not have delete permissions so he is not be able to delete files (although he can edit them and clear their contents).

4. The group ‘admin’ is permitted to create directories but not to delete them or view their contents:

```
sudo chmod +a# 4 "admin allow add_subdirectory,directory_inherit" documents
```

The add_subdirectory command neatly provides the required functionality. However, since subdirectories created by the admin group uses the UNIX permissions set by the umask, they are writable by the owners. And since the sticky bit is not set on the documents directory, admin users are able to delete files in the subdirectories they created. The following command explicitly denies this ability:

```
chmod +a# 0 "admin deny delete_child,directory_inherit" documents
```

Note: the rule is inserted at position 0; this follows general good practice when manipulating an ACL. Where the role and permission are specific it is preferable to place the entry towards the top of the ACL. This ensures that the entry is the first matching rule and does not get overridden by any prior 'allow' rules. In this example, the position is irrelevant, since the corresponding 'allow' rule is the UNIX permission which is always evaluated after the ACL.

5. The user 'alice' (who is not an administrator) is allowed to delete directories and files:

```
chmod +a# 5 "alice allow delete_child,directory_inherit" documents
```

The 'delete_child' permission means that all contained objects can be deleted, including directories and files.

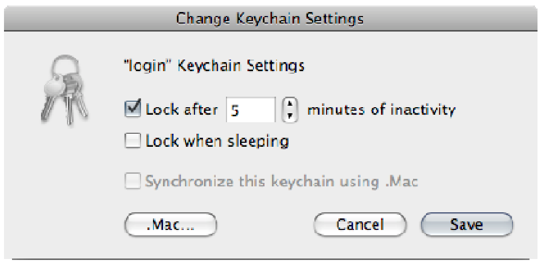
3.4.11 Keychain

The Mac OS X keychain allows users and applications to store and access authentication details in one place. It uses the familiar metaphor of a keychain to store and access private authentication credentials. Users can lock or unlock the keychain with a single password; applications can only access authentication details when the keychain is unlocked. Multiple keychains can be created to group similar authentication credentials.

By default, a keychain called "login" is used to store credentials used by most applications. The password for this keychain is the same as the login password and the keychain is automatically unlocked when a user logs in and is locked again upon logout.

The security of the "login" keychain can be further improved by changing its password to something other than the login password. This ensures that the keychain has to be explicitly unlocked before any items can be accessed and also prevents keychain items from being accessed if the login credentials are compromised. From the keychain access application in *Applications -> Utilities, choose Edit -> Change password for Keychain "login".*

A keychain should also be locked after a period of inactivity and when the system wakes from sleep. These options are accessed from the *Edit -> Change settings for Keychain "login"* menu in the keychain access application:



Change Keychain Settings

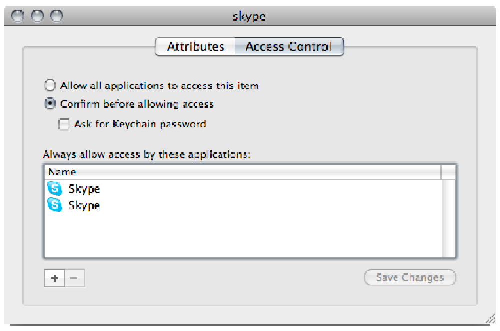
"login" Keychain Settings

[✓] Lock after 5 minutes of inactivity

[ ] Lock when sleeping

[ ] Synchronize this keychain using .Mac

The keychain access application also allows individual access controls to be placed on each key in the keychain. Where keys grant access to particularly sensitive information, Corsaire would recommend that the access control be changed to '*Ask for Keychain password*'.



skype

( ) Allow all applications to access this item

(●) Confirm before allowing access

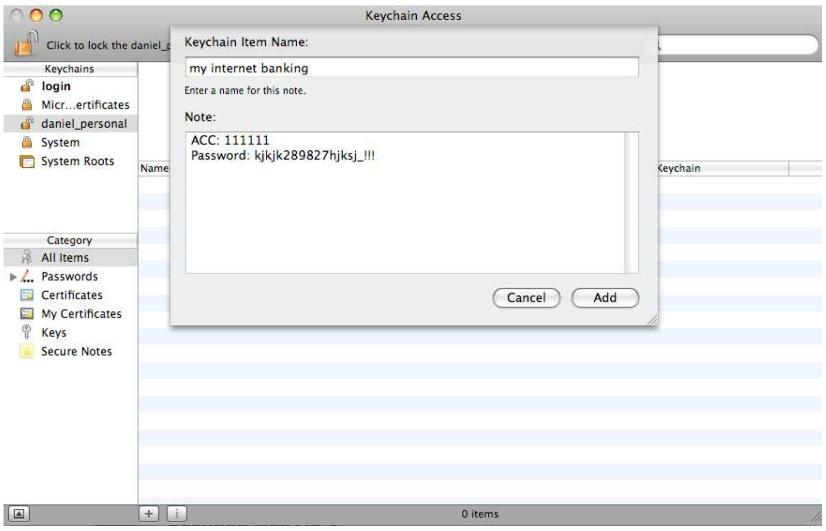
[ ] Ask for Keychain password

Always allow access by these applications:

Name
Skype
Skype

Keychains can also be used to store confidential information that does not belong to a specific application, such as bank, credit card or confidential personal information in the form of a *Secure note item*. These can be added to existing keychains from the *File -> New Secure Note Item.* Corsaire would recommend that secure notes be added to a separate keychain that does not share the same password as the login keychain. Should the confidentiality of the login password be compromised, this ensures that the secure note remains secure.

To create a new keychain and add a secure note, choose *File -> New Keychain*, once a name and password have been chosen, select the keychain and choose *File -> New Secure Note Item:*



Keychain Item Name: my internet banking

Enter a name for this note.

Note:

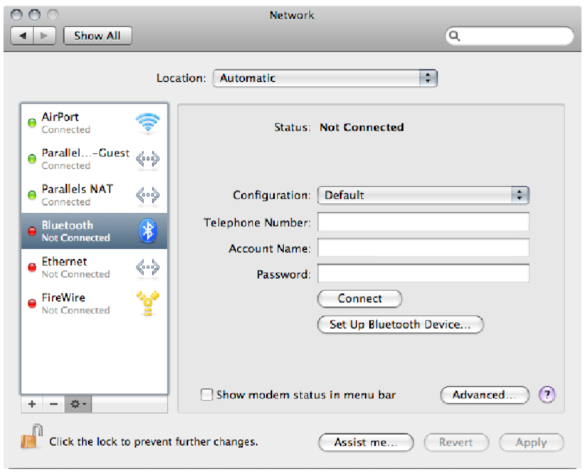
```
ACC: 111111
Password: kjkjk289827hjksj_!!!
```

From the keychain access application's preferences there is an option to *Show status in menu bar.* This allows the common keychain function to be easily accessible from the menu bar. It also allows quick access to the screen lock function.

3.5 Networking and Services

3.5.1 Securing Network Interfaces

Leopard gives the user the ability to disable unused network interfaces, which adds to the overall security of the system. As a hardening step, Corsaire would recommend disabling Bluetooth if it is not required. Go to *System Preferences → Network* and select Bluetooth, then click on the gear icon next to the minus icon and select *"Make Service Inactive":*



Location: Automatic

Status: Not Connected

Configuration: Default

Telephone Number:

Account Name:

Password:

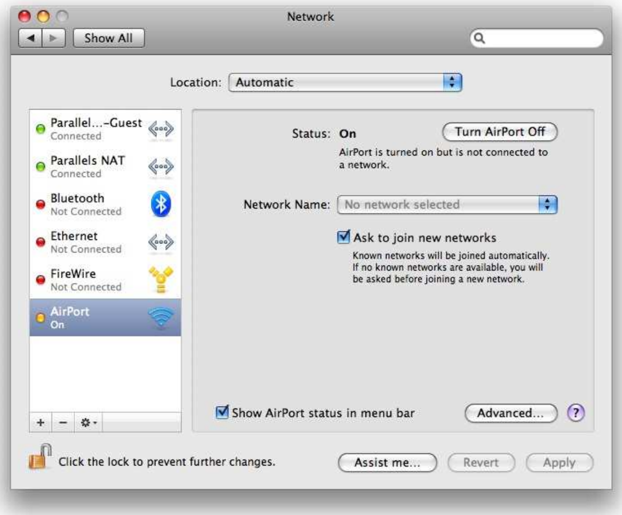
Show modem status in menu bar

The same can be achieved with any other devices that are not being used regularly.

3.5.1.1 Airport Security

Apple Airport allows you to connect to any available wireless network by default. Corsaire would recommend that the option to ask before joining networks is enabled. This can be achieved by going to:

System Preferences → Network → Airport



Location: Automatic

Parallel...-Guest	Connected
Parallels NAT	Connected
Bluetooth	Not Connected
Ethernet	Not Connected
FireWire	Not Connected
AirPort	On

Status: **On**

AirPort is turned on but is not connected to a network.

Network Name: No network selected

☑ Ask to join new networks

Known networks will be joined automatically. If no known networks are available, you will be asked before joining a new network.

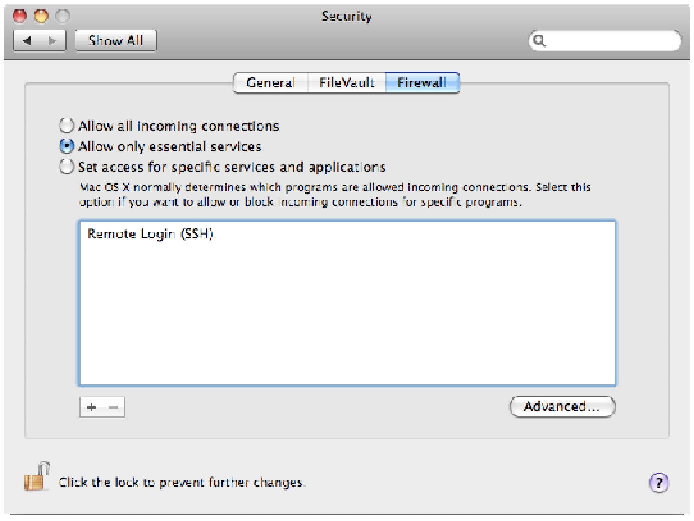
☑ Show AirPort status in menu bar

3.5.2 Firewall

There are two firewalls present in OS X Leopard: IPFW and the Application Firewall.

3.5.2.1 IPFW

Mac OS X is derived from BSD, and as such features the IPFW firewall. The major function of any firewall is to control and limit access to the host being protected. By default the firewall is disabled, allowing attackers to probe enabled network services (most network services are also disabled by default) for potentially useful information and to attempt to exploit weaknesses that may be present.



Allow all incoming connections [✓ Allow only essential services] Set access for specific services and applications

Mac OS X normally determines which programs are allowed incoming connections. Select this option if you want to allow or block incoming connections for specific programs.

Remote Login (SSH)

Click the lock to prevent further changes.

OS X Leopard also disables the firewall if it is installed as an upgrade.

Corsaire would recommend that the firewall be activated and this can be done from the Firewall tab in the *Security* pane of *System Preferences*.

The three options available are:

- Allow incoming connections.
- Allow only essential services.
- Set access for specific services and applications.

3.5.2.2 Allow Incoming Connections

The option of allowing all incoming connections is not advised. This allows attackers access to network services and provides little protection against network attacks.

3.5.2.3 Allow Only Essential Services

Allowing only essential services blocks everything except a handful that support networking, such as Bonjour. The following services are permitted as part of the “Allow Only Essential Services” rule:

- Racoon, which is the IPSec implementation.
- mDNSResponder, which implements Bonjour.

- configd, which implements DHCP and other networking services.

3.5.2.4 Creating Additional Firewall Rules

As the underlying firewall is IPFW, it is possible to configure additional rules to further enhance the security and overall functionality.

First, create the directory */Library/StartupItems/Firewall*

Next, create and open the file */Library/StartupItems/Firewall/StartupParameters.plist* in a text editor and insert the following:

```
{
    Description = "Firewall";
    OrderPreference = "None";
    Provides = ("Firewall");
    Requires = ("Network");
    Messages =
    {
      start = "Starting firewall";
      stop = "Stopping firewall";
    };
}
```

Create and open the file */Library/StartupItems/Firewall/Firewall* in a text editor and add a firewall script that defines the firewall rule-base. This is a shell script that is launched as part of the system start-up process and should contain all the actions to be performed by the firewall, including the definition of a rule-base. This should be a direct reflection of the organisation's network access policy. An example of a minimal configuration is included below:

```
#!/bin/sh

# Enable verbose logging
/usr/sbin/sysctl -w net.inet.ip.fw.verbose=1

# Here we define our system variables
# Replace xxx.xxx.xxx.xxx with the external address of your system
EXT="xxx.xxx.xxx.xxx"
FW="/sbin/ipfw"
```

```
# Here we define our network details
SSH_HOST="192.168.0.1"
# Replace yyy.yyy.yyy.yyy with the address of your primary DNS server
DNS1="yyy.yyy.yyy.yyy"

# Start of our firewall rules
# First flush the firewall rules
$FW -q flush

# Allow all traffic from the loopback interface
$FW add 100 allow all from any to any via lo0
# Loopback traffic on a 'real' interface is fake
$FW add 200 deny log logamount 1000 ip from any to 127.0.0.1/8

# Allow the system itself to initiate any connections externally
$FW add 300 allow all from $EXT to any

# Allow established connections to continue
$FW add 400 allow tcp from any to any out keep-state
$FW add 500 allow udp from any to any out keep-state
# Block replies if we do recall initiating the conversation
$FW add 600 deny log tcp from any to any established in

# Allow DNS replies back to the system
$FW add 700 allow udp from $DNS1 53 to $EXT

# Allow certain hosts to SSH in
$FW add 800 allow tcp from $SSH_HOST to $EXT 22

# iTunes Music Sharing
# $FW add 900 allow tcp from <YOUR ADDRESS HERE> to any dst-port 3689

# Deny all other traffic
$FW add 65534 deny log ip from any to any
```

A more detailed example, with service-based restrictions, can be found at:

- http://securosis.com/publications/ipfw.html

Since this file is used system wide, the ownership and permissions of the Firewall directory should be changed as follows:

```
sudo chown -R root:wheel /Library/StartupItems/Firewall
sudo chmod -R 700 /Library/StartupItems/Firewall
```

For the new firewall rules to take effect, either reboot or manually execute the /Library/StartupItems/Firewall/Firewall script with super user privileges:

```
sudo /Library/StartupItems/Firewall/Firewall
```

The state of the firewall (enabled or disabled) can be changed by directly changing a kernel parameter:

```
sysctl -w net.inet.ip.fw.enable=0
```

The above command disables the firewall, while the following enables it:

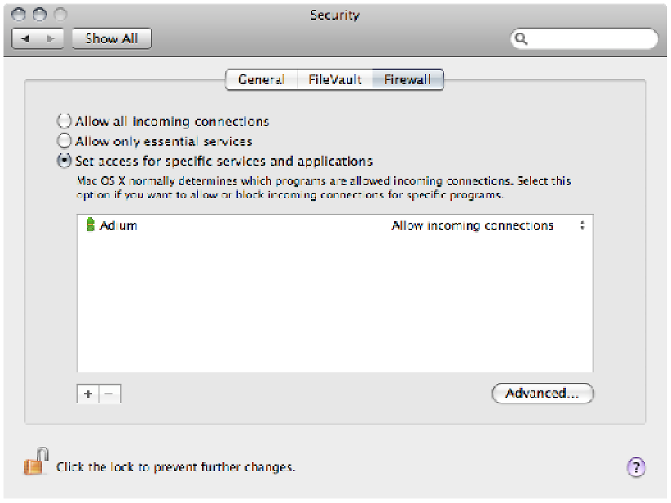
```
sysctl -w net.inet.ip.fw.enable=1
```

For those not wanting to edit files using the command line, a useful GUI application is available from the following website: http://www.hanynet.com/noobproof/index.html.

Whilst the firewall does add some protection from unwanted connections, it also fails to completely stop unwanted access to the users Mac. Even when the most secure setting, ‘Block all incoming connections’, is chosen, it still allows access to system services such as NTP and Bonjour from the Internet.

3.5.2.5 Application Firewall

Mac OS X Leopard introduced the application-based firewall, known as Application Firewall, which allows administrators to create specific access rules for services and applications. The application firewall works by applying access controls based on the application, which saves users from needing to configure ports and protocols, which may be subject to change or other dynamic behaviour:



( ) Allow all incoming connections
( ) Allow only essential services
[✓ Set access for specific services and applications]
Mac OS X normally determines which programs are allowed incoming connections. Select this option if you want to allow or block incoming connections for specific programs.

Adium	Allow incoming connections

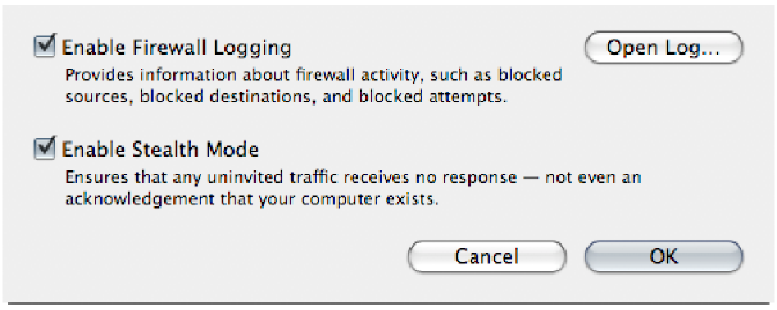
Click the lock to prevent further changes.

Whilst Corsaire would recommend this option, it is important to point out that unless users specifically tell the firewall which application to control, it is still possible to start applications and have them accessed from the internet without the firewall blocking them.

This last option makes use of another new feature in Mac OS X Leopard, namely code signing. When an application is added to the list, Mac OS X digitally signs the application. The benefit of code signing lies in the ability to detect application modifications and to warn users that the application may have been tampered with.

All applications not in the list that have been digitally signed by a certificate authority trusted by the system (for the purpose of code signing) are allowed to receive incoming connections. Every Apple application in Leopard has been signed by Apple and is allowed to receive incoming connections. If users wish to deny a digitally signed application they should first add it to the list and then explicitly deny it.

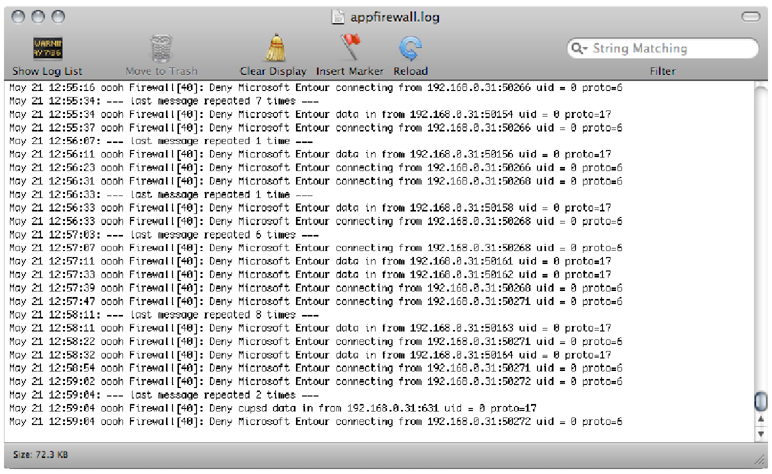
Another feature that should be enabled by default is that of stealth mode, which blocks any incoming ICMP requests. This can be enabled in the advanced option:



☑ Enable Firewall Logging
Provides information about firewall activity, such as blocked sources, blocked destinations, and blocked attempts.

☑ Enable Stealth Mode
Ensures that any uninvited traffic receives no response — not even an acknowledgement that your computer exists.

To view the firewall log files, click on *Open Log.* This opens the console program and from there it is possible to view the firewall activity. This is important as it gives users an idea of what activity the firewall is blocking, and can be used to ensure the system is secure:



```
May 21 12:55:16 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50266 uid = 0 proto=6
May 21 12:55:34: --- last message repeated 7 times ---
May 21 12:55:34 oooh Firewall[40]: Deny Microsoft Entour data in from 192.168.0.31:50154 uid = 0 proto=17
May 21 12:55:37 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50266 uid = 0 proto=6
May 21 12:56:07: --- last message repeated 1 time ---
May 21 12:56:11 oooh Firewall[40]: Deny Microsoft Entour data in from 192.168.0.31:50156 uid = 0 proto=17
May 21 12:56:23 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50266 uid = 0 proto=6
May 21 12:56:31 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50268 uid = 0 proto=6
May 21 12:56:33: --- last message repeated 1 time ---
May 21 12:56:33 oooh Firewall[40]: Deny Microsoft Entour data in from 192.168.0.31:50158 uid = 0 proto=17
May 21 12:56:33 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50268 uid = 0 proto=6
May 21 12:57:03: --- last message repeated 6 times ---
May 21 12:57:07 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50268 uid = 0 proto=6
May 21 12:57:11 oooh Firewall[40]: Deny Microsoft Entour data in from 192.168.0.31:50161 uid = 0 proto=17
May 21 12:57:33 oooh Firewall[40]: Deny Microsoft Entour data in from 192.168.0.31:50162 uid = 0 proto=17
May 21 12:57:39 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50268 uid = 0 proto=6
May 21 12:57:47 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50271 uid = 0 proto=6
May 21 12:58:11: --- last message repeated 8 times ---
May 21 12:58:11 oooh Firewall[40]: Deny Microsoft Entour data in from 192.168.0.31:50163 uid = 0 proto=17
May 21 12:58:22 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50271 uid = 0 proto=6
May 21 12:58:32 oooh Firewall[40]: Deny Microsoft Entour data in from 192.168.0.31:50164 uid = 0 proto=17
May 21 12:58:54 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50271 uid = 0 proto=6
May 21 12:59:02 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50272 uid = 0 proto=6
May 21 12:59:04: --- last message repeated 2 times ---
May 21 12:59:04 oooh Firewall[40]: Deny cupsd data in from 192.168.0.31:631 uid = 0 proto=17
May 21 12:59:04 oooh Firewall[40]: Deny Microsoft Entour connecting from 192.168.0.31:50272 uid = 0 proto=6
```

Size: 72.3 KB

3.5.3 Network Services

By default all networking services are disabled, which provides fewer opportunities for remote attackers. Enabling network services (SSH, Personal Web Sharing, FTP etc.) allows users some form of remote access to the system and should only be permitted if there is an explicit requirement for their use. Where possible, enabled services should be limited to a specific access control policy, enforced by the firewall (see above).

3.5.3.1 Launchd

Leopard uses 'launchd' to handle and control all of the network services. After the system has booted, and the kernel is running, OS X makes use of launchd to finish the system initialisation. Launchd loads the parameters for each daemon found in:

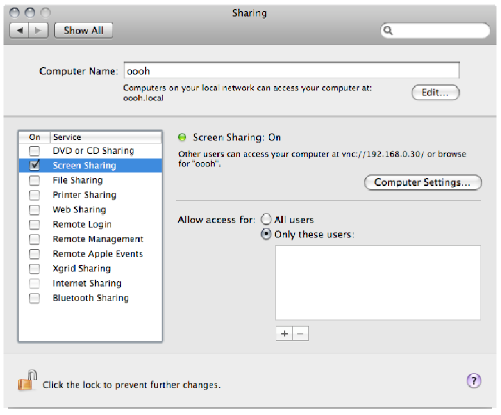
- ~/Library/LaunchAgents (Per-user agents provided by the user).
- /Library/LaunchAgents (Per-user agents provided by the administrator).
- /Library/LaunchDaemons (System wide daemons provided by the administrator).
- /System/Library/LaunchAgents (Mac OS X Per-user agents).
- /System/Library/LaunchDaemons (Mac OS X System wide daemons).

Users can determine which services are started automatically by viewing the contents of each of the above directories or by using a graphical utility like Lingon[8].

3.5.3.2 Screen Sharing

Mac OS X Leopard now comes with a built-in option of sharing and controlling remote desktops through VNC.

Screen sharing can be enabled from *System Preferences -> Sharing*. By default, Leopard does not allow anyone to connect to the system. In order to control who has access to the system, it is possible to set specific users to have access only:



Computer Name: oooh

Computers on your local network can access your computer at: oooh.local

On	Service
☐	DVD or CD Sharing
✓	Screen Sharing
☐	File Sharing
☐	Printer Sharing
☐	Web Sharing
☐	Remote Login
☐	Remote Management
☐	Remote Apple Events
☐	Xgrid Sharing
☐	Internet Sharing
☐	Bluetooth Sharing

● Screen Sharing: On

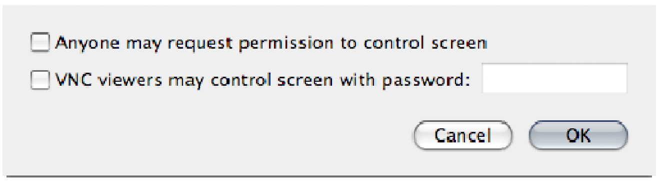
Other users can access your computer at vnc://192.168.0.30/ or browse for "oooh".

Allow access for: All users [✓ Only these users:]

Click the lock to prevent further changes.

Once the user has been chosen, click on *Computer Settings* and ensure that a password is required in order to connect:

[8] http://lingon.sourceforget.net



☐ Anyone may request permission to control screen
☐ VNC viewers may control screen with password:

By default, screen sharing is configured to encrypt passwords and keystrokes only. Corsaire would recommend that all network data be encrypted, which can be achieved by performing the following steps:

1. Open up Terminal and type in "open /System/Library/CoreServices/Screen Sharing.app".
2. Go to the Screen Sharing preferences in the top left of the screen and select preferences.
3. Ensure that "Encrypt all network data" is selected.

3.5.3.3 Back to My Mac

Back to My Mac (BTMM) allows Apple's .Mac users the ability to connect to their machine from any other Leopard based server over the Internet. BTMM makes use of IPSec to encrypt the data between machines and also uses Kerberos to eliminate the need for entering a username and password to reach a computer in your BTMM network (single sign-on).

BTMM by default uses the following network ports for communication:

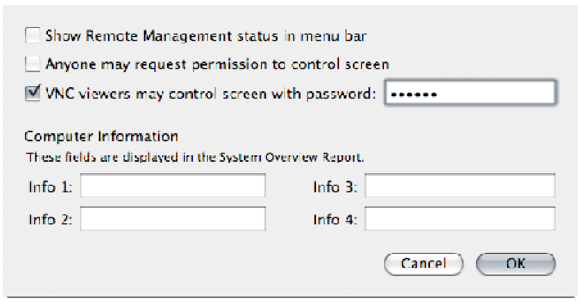
- TCP 443 for authentication
- UDP port 4500 for network connections between machines

By default, the above ports are enabled, so Corsaire would advise that they be disabled if users do not require the functionality of BTMM.

3.5.3.4 Remote Management

Remote management allows users to control Mac OS X using Apple's remote management application. This feature is similar in some way to the screen sharing function but with the added ability to control the Mac. Remote management makes use of the popular VNC server to allow users to connect to the Mac and works with other Open Source VNC clients.

It is important that a password is set before remote management is enabled; this prevents unauthorised access to the machine. This can be achieved by clicking on *Computer Settings* and entering a strong password:

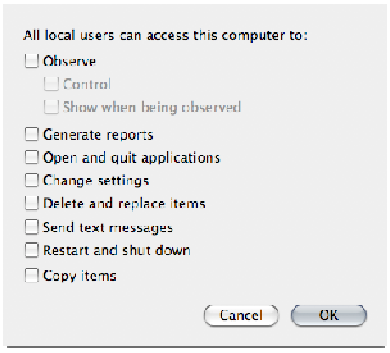


☐ Show Remote Management status in menu bar
☐ Anyone may request permission to control screen
☑ VNC viewers may control screen with password: ••••••

Computer Information
These fields are displayed in the System Overview Report.

Info 1:
Info 2:
Info 3:
Info 4:

It is possible to control what users are able to perform whilst using this function, by clicking on *options* and selecting any of the following:



All local users can access this computer to:

☐ Observe
☐ Control
☐ Show when being observed
☐ Generate reports
☐ Open and quit applications
☐ Change settings
☐ Delete and replace items
☐ Send text messages
☐ Restart and shut down
☐ Copy items

It is also possible to control the Mac by using the command line. The *networksetup* command allows you to configure a variety of settings, for example:

Set the Mac to use a specific wireless router:

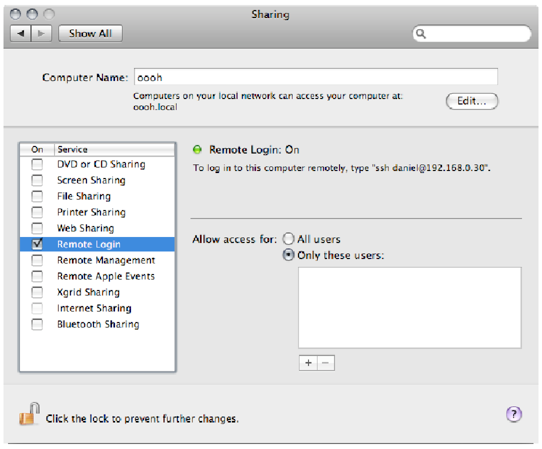
```
networksetup -setairportnetwork corsaire thiswouldbeasecurepassword
```

If users are managing a handful of Mac's, this command could be easily included into a script, which could be run to ensure all the machines were running the same configuration.

Remote management, once enabled, listens on port TCP 5900. Corsaire would recommend a firewall rule restricting access to this port be created, to restrict access to authorised IP addresses.

3.5.3.5 Remote Logon (SSH)

The default installation of OpenSSH allows only SSH version 2 connections, which is a change from previous versions. Leopard now enables users to control who has access to the computer when remote logon is enabled:



Computer Name: oooh

Computers on your local network can access your computer at: oooh.local

On	Service
☐	DVD or CD Sharing
☐	Screen Sharing
☐	File Sharing
☐	Printer Sharing
☐	Web Sharing
✓	Remote Login
☐	Remote Management
☐	Remote Apple Events
☐	Xgrid Sharing
☐	Internet Sharing
☐	Bluetooth Sharing

● Remote Login: On

To log in to this computer remotely, type "ssh daniel@192.168.0.30".

Allow access for: All users [✓ Only these users:]

Click the lock to prevent further changes.

3.5.3.5.1 Authentication

OpenSSH has a number of authentication options including username and password, challenge-response, Kerberos and public key authentication. Password and public key authentication, being the most popular implementations, are discussed below.

By default OS X 10.5 permits users to authenticate with their username and password or by creating a public-private key pair and placing their public key in the ~/.ssh/authorized_keys file. Using public key authentication provides a degree of protection from brute-force attacks, since users without the correct key are not be permitted to authenticate. However, when creating the key pair, it is possible for a user to specify a blank password for their private key which means that they can login to the host without entering any form of password when connecting from that particular client (or by using that private key). This can be considered a security risk in certain environments since it means that if the client host is compromised then the attacker is able to login to the server without authentication. If public keys are used, then users should be reminded that the password used to protect the private key should comply with organisational password standards and must not be blank. To disable public key authentication, edit the /etc/sshd_config file and add the following option:

```
PubkeyAuthentication no
```

3.5.3.5.2 Cryptographic Settings

The symmetric cipher used to encrypt the SSH session is chosen by the client from the set of ciphers made available by the server. In the default installation, these are:

- aes128-cbc
- 3des-cbc
- blowfish-cbc
- cast128-cbc
- arcfour
- aes192-cbc
- aes256-cbc
- aes128-ctr

The organisation-wide security policy may dictate which ciphers are acceptable for remote login. To change the list of available ciphers, edit the /etc/sshd_config configuration file and change the 'Ciphers' setting to list the permitted ciphers. The ciphers should be listed on the same line and comma separated. For example, to only permit the AES algorithms with 192 or higher key length use:

```
Ciphers aes192-cbc,aes256-cbc,aes192-ctr,aes256-ctr
```

SSH also uses a message authentication code for data integrity protection. By default the following MACs are provided by the server:

- hmac-md5
- hmac-sha1
- hmac-ripemd160
- hmac-sha1-96
- hmac-md5-96

To restrict this to the more secure MACs, edit the 'MACs' setting in the configuration file, for example:

MACs hmac-sha1-96,hmac-ripemd160

3.5.3.5.3 Login Banner

SSH should be configured to display a login banner before a user authenticates to the service. As discussed in section 3.4.4, a text file containing the login banner should be created in /etc/login_banner. To display the login banner to users, edit the /etc/sshd_config file and replace the line:

```
#Banner /some/path
```

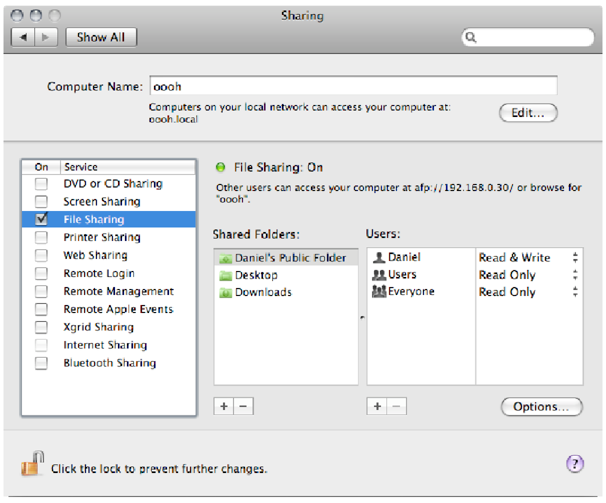
with:

```
Banner /etc/login_banner
```

The sshd service has to be restarted before the changes take effect. This banner is displayed to all users attempting to access the system through SSH.

3.5.3.6 File Sharing

OS X provides many ways to share files with the option of added security. Users can now specify what sharing protocol to use and also apply access control lists to ensure that users do not have full access to the file system:



Computer Name: oooh

Computers on your local network can access your computer at: oooh.local

On	Service
☐	DVD or CD Sharing
☐	Screen Sharing
✓	File Sharing
☐	Printer Sharing
☐	Web Sharing
☐	Remote Login
☐	Remote Management
☐	Remote Apple Events
☐	Xgrid Sharing
☐	Internet Sharing
☐	Bluetooth Sharing

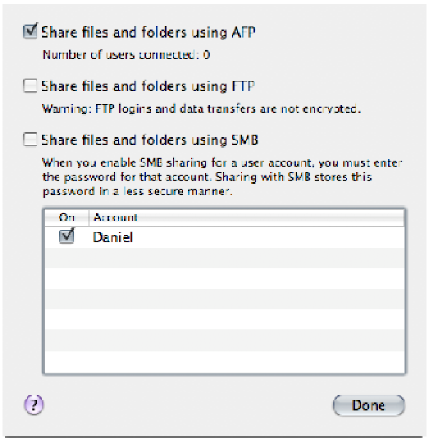
● File Sharing: On

Other users can access your computer at afp://192.168.0.30/ or browse for "oooh".

Shared Folders:
- Daniel's Public Folder
- Desktop
- Downloads

Users:	
Daniel	Read & Write
Users	Read Only
Everyone	Read Only

Click the lock to prevent further changes.



[✓] Share files and folders using AFP
Number of users connected: 0

[ ] Share files and folders using FTP
Warning: FTP logins and data transfers are not encrypted.

[ ] Share files and folders using SMB
When you enable SMB sharing for a user account, you must enter the password for that account. Sharing with SMB stores this password in a less secure manner.

On	Account
✓	Daniel

Corsaire would recommend that any file sharing be restricted to specific directories and not system-wide access.

3.5.3.7 FTP

File Transfer Protocol (FTP) is a well-established protocol for transferring data. It is a clear text protocol, which transmits both the authentication credentials and the data itself unencrypted between the client and server. For this reason it is susceptible to network sniffing attacks and its use on un-trusted network segments should be avoided if possible.

The SSH service offers file transfer mechanisms in the form of SCP (secure copy) and SFTP (secure file transfer) which encrypts all traffic between the client and server. However, encrypting data incurs a processing overhead which makes the SSH file transfer services slightly slower than FTP.

FTP can be enabled from the *Sharing* section of *System Preferences under File Sharing*.

3.5.3.7.1 FTP Login Banner

By default, the FTP server login banner is stored in a specific file. To display the login banner defined above, it is necessary to create a link to the correct file name:

```
sudo ln -s /etc/login_banner /etc/ftpwelcome
```

This banner is displayed to users who attempt to connect to the FTP server.

3.5.3.7.2 FTP User Access Controls

All users with local accounts on the system are granted FTP access by default. Corsaire would recommend that this access be further restricted so that only users with a clearly defined need for FTP are permitted access. Furthermore, the administrative user should not be permitted FTP access to the system since the clear text nature of the protocol poses too great a risk for credential theft. Other users can be restricted based on their username and/or source IP address.

To restrict access on a per user basis, create and edit the file /etc/ftpusers, an example could be:

```
stephen deny
jane allow
glyn
martin@192.168.0.1 allow
martin
```

The format of the file is: 'username@host directive'. The simplest entry is a username without a directive, where FTP assumes the user is denied access. In the example above, user Glyn is denied access, along with Stephen; Jane is permitted access. The user Martin is able to access FTP from host 192.168.0.1, but is denied access from all other hosts.

3.5.3.7.3 Chroot Jails and FTP

Users accessing the system through FTP are allowed to move about the entire file system as if connected locally. Read, write, create and delete access to files is limited by their user access privileges and the file permissions. To further protect the system and restrict access to potentially sensitive data, Corsaire would recommend that users are only permitted FTP access to their home folders. To enable this feature, create the file /etc/ftpchroot and insert all the usernames who are to be granted FTP access, but have their access limited to their home folders. The file is a simple list of usernames, such as:

```
james
bob
```

3.5.3.8 Apache (Web Sharing)

Personal web sharing is provided through the Apache web server, commonly used as a commercial and personal web-server throughout the Internet. Apache is a full featured, complex web server with a multitude of configurations and their associated security implications. Using Apache to share files is not recommended for most organisations, since more secure alternatives with better authentication mechanisms such as SSH are available. Mac OS X Leopard ships with Apache version 2.2.6.

This section details basic changes to the default installation to allow users to serve access-controlled web pages. A full secure web-server deployment is beyond the scope of this document. For a detailed treatment of the Apache web server and related configuration options, see the Apache website: http://httpd.apache.org.

3.5.3.8.1 Access Control

This configuration is suitable for local (personal) web sites where the server delivers content from the users' home directory; access to all other resources is denied by default. Edit the /private/etc/apache2/httpd.conf file and find the following section:

```
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>
```

This should be modified to the following:

```
<Directory />
    Options FollowSymLinks
    AllowOverride None
    Order deny,allow
    Deny from all
</Directory>

<Directory /Users/*/Sites>
        <Limit GET POST>
        </Limit>
        Order deny,allow
        Deny from all
</Directory>
```

Access to the URL http://your.mac.system/ is now denied for *everyone*. The second 'Directory' section explicitly denies access to all users' Sites directories, but this behaviour is still overridden by the user specific configuration files found in /private/etc/apache2/users. At the end of the general /private/etc/apache2/httpd.conf file is the line:

```
Include /private/etc/apache2/users/*.conf
```

This loads individual configuration files for each user, overriding the general access controls that have previously been specified. These user files are automatically created when a new user is added to the system. Corsaire would recommend that instead of reading the configuration files of all users, read only those of users that are authorised to use web sharing. Instead of using the * wildcard character to read all files, specify each users configuration file explicitly:

```
Include /private/etc/apache2/users/martin.conf
Include /private/etc/ apache2/users/jane.conf
Include /private/etc/ apache2/users/james.conf
```

3.5.3.8.2 Granting Access

Once access has been denied by default, and only specific users are permitted to share their ~/Sites directories, it is now possible to edit the default configuration for each user file and secure it further. The files are merely extensions to the main configuration file, and follow the same format. A default configuration is included below:

```
<Directory "/Users/martin/Sites/">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

By default, everyone is allowed access. If the source IP address(es) of the users who are accessing this website are known, then these can be specified to further restrict access, for example:

```
<Directory "/Users/martin/Sites/">
    Options Indexes MultiViews
    AllowOverride None
    Order deny,allow
    Allow from 10.0.1.1
    Allow from 192.168.0.1
</Directory>
```

This has the effect of only allowing access to the http://your.mac.system/~martin URL to users from the 10.0.1.1 and 192.168.0.1 systems.

The default configuration of Apache on Mac OS X provides unencrypted HTTP transport. However, the OpenSSL suite is included in the base OS and can be enabled to provide full encryption of web server traffic. See http://developer.apple.com/internet/serverside/modssl.html for more information.

3.5.3.8.3 Configuring Apache's Server Version

As part of a standard HTTP exchange, the server sends a header used to identify itself. Under a default installation of Mac OS X this is returned as: "Apache/2.2.6 (Unix) mod_ssl/2.2.6 OpenSSL/0.9.7l DAV/2" which reveals both the exact version of Apache installed, and the underlying operating system. Attackers using remote scanning software frequently identify vulnerable systems by their version numbers obtained from these banners. It is considered good security practice to remove such information from servers, particularly those accessible from public networks.

To change the HTTP server header, edit the /private/etc/apache2/httpd.conf file, and add the line:

```
ServerTokens prod
```

Once the server is restarted, it returns a header string of "Apache". This may be further obfuscated through the use of the mod_header Apache component, although its use is beyond the scope of this document. See http://www.apache.org for more information.

3.5.3.9 Bonjour

Bonjour, formerly known as Rendezvous is Apple's implementation of the ZeroConf protocol. It uses network broadcasts to advertise system services on the local subnet, such as printers, iTunes, iChat, SSH and FTP etc.

Advertising the services available on a Mac OS X system could provide an attacker with additional information that may be useful when conducting an attack. In cases where the local subnet is not a trusted network, such as public WiFi access points, Corsaire would recommend that the data being broadcast to the subnet be limited.

It is possible to disable support for Bonjour in each of the affected applications (e.g. iTunes, iChat, etc.) or to disable the Bonjour service as a whole. To disable Bonjour on the Mac OS X system, issue the command:

```
sudo launchctl unload -w /System/Library/LaunchDaemons/com.apple.mDNSResponder.plist
```

It should be noted that Mac OS X was not designed to operate without Bonjour and disabling the service could have a number of adverse affects on the operation of the system, such as loss of printer functionality.

3.6 Logging and Auditing

It is important that user accountability be maintained on a secure system. Accountability ensures that actions performed on the system can be traced back to a user; this is maintained through system and application logs, process accounting and auditing.

3.6.1 Logging

The syslogd daemon is used for system and application logging on Mac OS X, and its behaviour is controlled by the file: /etc/syslog.conf. Since syslog is a common UNIX logging facility, existing standards within the organisation may stipulate how it should be configured.

The following changes to the default configuration in /etc/syslog.conf are recommended:

Change the line:

```
auth.info;authpriv.*;remoteauth.crit                    /var/log/secure.log
To:
auth.info:.*;remoteauth.err;auth.err               /var/log/secure.log
```

This ensures that remote authentication error messages and other authentication error messages are logged to the /var/log/secure.log file.

Additionally, Corsaire would recommend that logs be stored on a remote system. Remote logging ensures that a backup copy of the hosts logs is maintained off-system, which is useful in the case of a full system compromise where the logs on the system itself become untrustworthy. Remote logging across the network is supported by the syslog facility, but since this is performed through an unencrypted and unauthenticated transport the risk exists that the log information could be compromised in transit or manipulated. Additionally ensure that the network used to transport log information and the remote syslog server are adequately protected.

To enable remote logging of the /var/log/secure.log and /var/log/system.log files to the host 10.0.0.1, add the following lines to the syslog.conf file:

```
auth.info;authpriv.*;remoteauth.crit                    /var/log/secure.log
authpriv.*;remoteauth.err;auth.err                @10.0.0.1
*.notice;authpriv,remoteauth,ftp,install.none;kern.debug;mail.crit     @10.0.0.1
```

3.6.1.1 Process Accounting

Mac OS X supports process accounting, which logs all commands executed by all users. This facility may be desirable in environments where very detailed operating system audit data is required. To enable this feature create the folder /var/account:

```
sudo mkdir /var/account
```

And then create the empty file /var/account/acct:

```
sudo touch /var/account/acct
```

This file is used to store all the process accounting information. To start accounting on the system issue the command:

```
sudo accton /var/account/acct
```

Process accounting is automatically started on system start-up. Since accounting has to write to the specified file, if it cannot for any reason (e.g. the file system is full), then accounting is stopped and only resumes when the problem has been corrected.

It should be noted that process accounting can record large quantities of data, and if unmanaged could lead to disk space issues. It is important to ensure that log file archival and rotation, as well as data analysis, procedures are operational to benefit from maintaining such detailed auditing.

3.7 Additional Security

3.7.1 Intrusion Detection Systems (IDS)

IDS can assist administrators in identifying successful and attempted attacks on a system by monitoring file system integrity or by analysing network traffic for dangerous payloads. The most important consideration when it comes to IDS is not the technology used, but the soundness of the process in which it is used. The software can only *detect* attacks once they have happened, and only then alert the administrator; it takes no corrective action. For this reason, the real value of IDS is only seen when there are clear and defined processes for correctly managing the alerts generated by these systems. Without these processes, the resources (financial, effort, etc) consumed by an IDS are largely wasted.

A number of open source options exist for Network and Host IDS for use with Mac OS X. Some examples are:

- Snort – the lightweight Network Intrusion Detection System, which has become one of the most popular NIDS in deployment.
- Prelude IDS – a Hybrid IDS framework used for collecting, analysing and correlating network and host IDS events from multiple sources.
- Bro IDS – a Network IDS capable of analysing high speed networks.
- Logsurfer – a system log analysis tool that can be used to generate alerts on the detection of suspicious activity and divergence from expected system behaviours; based on swatch.
- Simple Event Correlator – a flexible correlation tool for aggregating and analysing events from audit and log files, which can be used to generate alerts; rules can be easily defined using regular expressions.

3.7.1.1 File Integrity Checkers

Host based intrusion detection on OS X comes mainly in the form of file integrity checking programs such as Tripwire. These programs take a snapshot of important system files and folders, sign them, and store them in a secure database. Another useful feature is that any file or directory on the system can be covered. Making a file system hierarchy is also a useful concept as the FIC can then detect new additions (for example). Periodic checks are then performed on the system that compares the current files on the system (policy configurable) with those in the database. If any differences are noticed - in size, ownership, permissions or content, the designated administrator is notified.

Free software packages that provide this kind of functionality and have been updated for Leopard are:

- Tripwire – is the most mature and tested version with support for almost every OS platform (A commercial version of this program is also available, though not for Mac OS X).
- Radmind – a popular program with a native OS X graphical interface.
- Samhain - a tool for file integrity checking that optionally can be used as a client/server application for centralized monitoring of networked hosts.

The following software packages have been tested on a number of UNIX platforms and may work under Mac OS X:

- Osiris – supports all UNIXs.

Host based IDS provides a means of alerting administrators if an attacker has managed to gain access to a system, and modifies or installs files and software. As essential as it is, this form of alerting is often not quick enough to prevent costly damage to a system. Where systems are deployed in high threat environments, Corsaire would recommend that a network based IDS be deployed.

3.7.2 Viruses and Malware

Viruses, trojans and other malware are relatively uncommon on the Mac OS X platform, and as a result currently present a far lower risk than on Windows systems. However, it should be highlighted that their relative absence does not mean that the operating system itself is immune to malware, only that the current combination of a low Mac OS X install base, together with the operating system's security features, make the Mac OS X platform less attractive than other platforms from a virus writer's perspective.

In some organisations, security policies may mandate the use of anti-virus solutions for all desktop systems, regardless of the relative absence of viruses that specifically target Mac OS X. This could help prevent a Mac OS X system from acting as a virus transmission agent in a heterogeneous computing environment.

A number of well-known commercial anti-virus vendors now produce versions of their products for Mac Mac OS X, including:

- McAfee's Virex - http://www.mcafee.com/us/enterprise/products/anti_virus/file_servers_desktops/virex.html
- Symantec's Norton AntiVirus - http://www.symantec.com/norton/products/overview.jsp?pcid=ma&pvid=nav11mac
- Sophos' Anti-Virus - http://www.sophos.com/products/enterprise/endpoint/security-and-control/mac/
- Intego's VirusBarrier - http://www.intego.com/virusbarrier/

References

Hardening guidelines for Mac OS X 10.4

http://research.corsaire.com/whitepapers/060517-securing-mac-os-x-tiger.pdf

http://www.cisecurity.org/bench_osx.html

http://www.nsa.gov/snac/os/applemac/osx_client_final_v_1_1.pdf

Open Firmware

http://www.kernelthread.com/mac/osx/arch_boot.html

http://docs.info.apple.com/article.html?artnum=106482

Firewall

http://docs.info.apple.com/article.html?artnum=306938

ipfw man page

http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/firewalls.html

File System

chmod manual page

Explanation of OS X FileSystem

http://www.kernelthread.com/mac/osx/arch_fs.html

Network Services

http://developer.apple.com/documentation/Darwin/Reference/ManPages/man5/launchd.plist.5.html

TCP and UDP ports used by Apple Software products

http://support.apple.com/kb/TS1629?viewlocale=en_US

Acknowledgements

This Guide was updated by Daniel Cuthbert, Principal Consultant at Corsaire and is based on original work by Stephen de Vries, Principal Consultant at Corsaire. Contributions were also provided by Glyn Geoghegan and David Ryan, both Principal Consultants at Corsaire.

About The Author

Daniel heads up Corsaire's Security Training and has over nine years of industry experience. During this time he has focused on Security Assessment for some of the world's largest consultancies and financial, telecommunication and media institutions. He holds a Masters Degree from the University of Westminster in IT Security. He is a founding member of OWASP and formerly the UK Chapter Head. He has pioneered the Secure Development Lifecycle (SDLC) approach and has lectured extensively on the subject.

About Corsaire

Corsaire are experts at securing information systems, consultancy and assessment. Through our commitment to excellence we provide a range services to help organisations protect their information assets and reduce corporate risk.

Founded privately in the United Kingdom in 1997, we operate on an international basis with a presence across Europe, Africa and the Asia-Pacific rim. Our clients are diverse, ranging from government security agencies and large blue-chip FTSE, DAX, Fortune 500 profile organisations to smaller internet start-ups. Most have been drawn from banking, finance, telecommunications, insurance, legal, IT and retail sectors. They are experienced buyers, operating at the highest end of security and understand the differences between the ranges of suppliers in the current market place. For more information contact us at contact-us@corsaire.com or visit our website at http://www.corsaire.com.