Global Information Assurance Certification Paper

Securing Macintosh OS X sites that primarily use off-site support
Geordie Korper
January 8, 2003
SANS GSEC Practical, Version 1.4b, Option 1

Abstract
This paper discusses how to safely and inexpensively support a small Mac OS X site where the primary support is from an off-site location. Mac OS X is by default a reasonably secure and an easily administered platform, but as with any operating system, ongoing administration and maintenance is necessary. In small offices where dedicated on-site staff cannot be justified, methods to remotely administer the network must be implemented in a secure and cost effective manner. This paper explores several possible ways to balance the often-conflicting goals of security and cost. Topics covered include: setup of the network infrastructure, secure alternatives for remote administration, and specific remote administration strategies for both server and client computers.

Introduction
Remote administration creates additional vulnerabilities that are not present on a standard server or client desktop. In addition, limited budgets that are associated with smaller sites require compromises that can negatively affect the overall security. The most common vulnerability is having a large number of services running from a single server that is serving both Local Area Network (LAN) and Internet clients. A proper defensive strategy requires that different services be distributed across multiple servers so that security vulnerabilities in one service cannot be combined with weaknesses in other services, reducing the total security of the system. In addition a proper defense in depth strategy would be significantly less deep if there is no division between an external and internal device.

In an ideal situation, every service would have its own dedicated server. However, in a small office environment with only a few clients, following that rule might require having more servers than clients. There are not many organizations of this size where such a solution is viable. Because every organization has different needs, this paper will present a range of options and recommendations that reflect the best balance of security and cost for the intended audience. Obviously, no one solution will fit every installation and you will need to decide what level of risk you are willing to accept, based upon the specifics of your organization.

For the purpose of this paper, the organizational system will be divided into four areas. These areas are: Wide Area Network/Internet, Local Area Network, servers and clients. Because we are looking at the infrastructure as a whole, the model used will be a hybrid of the network model of core, distribution, and access, combined with the system administrator's model of servers and clients. Depending upon the situation, we will use whichever model best serves our purposes. A particular device may operate in several of the areas, but the basic steps that need to be taken to ensure that the network, as a whole, is secure, can be broken down into controlling the flow of data within and

between these areas. Using these divisions allows us to determine where authentication, authorization, and access decisions will take place within that information flow.

Although controlling the flow of information within a local area network is an important part of any security plan, in smaller groups such as the ones discussed here, the primary concern from a security standpoint is unauthorized access from the outside. Since the Internet/Wide Area Network is the area of greatest threat the analysis will begin with that area. Although the site may have Internet accessible servers that will need tighter security than the Local Area Network servers, securing those will not be discussed in this document. The basic methods that would be used are similar to the methods for the Local Area Network servers, which are covered later in this paper, and those methods could be used as a starting point for how set up any Internet accessible servers. Since the main threat is coming from the outside, we will begin by focusing on the network infrastructure.

Network Infrastructure

The first line of defense against attackers is at the junction between the external network connection and the Local Area Network. Depending on the size of the organization, the network connection may range in capabilities from a dedicated Integrated Services Digital Network (ISDN) or Digital Subscriber Line (DSL) to a T1 (remote support using a dial-on-demand network is outside the scope of this paper). Regardless of the type of connection employed, some kind of division needs to be made between the internal and external networks. This is usually accomplished with a dedicated routing device. This could be anything from consumer level equipment such as the Airport Base Station to a Cisco router or similar equipment.

Some extremely small sites may have initially decided to use the built-in routing capabilities of Mac OS X to route traffic from a broadband connection to the Local Area Network, but this is a false cost savings. The initial costs may be low if you already had a Mac OS X computer that was internet connected, but security issues and the ongoing support needed in such a setup make it a very bad idea.

Since the cost of a basic router is small compared to other equipment in the infrastructure, and since it has significant security advantages, the question becomes what features will an organization need beyond the router's basic network segregation capabilities. With entry-level routers you may be limited to defining one external IP address and a single Network Address Translation (NAT) pool on the inside. The entry-level router may also have little if any configuration available; whereas, higher end routers may allow for a Demilitarized Zone (DMZ), Virtual Private Networks (VPN), or even complex filtering similar to what a firewall would offer.

Some or all of the router features could also be implemented using a dedicated firewall, which if used in combination with a router, would also add additional layers to the defense in depth strategy. In fact, an ISP may maintain the router in many small business installations. If that is the case, adding a firewall will be the only option if you

want to implement a truly secure environment. Network Address Translation and simple packet filters are a good first step towards increasing network security, but a good firewall with a built-in virtual private network server will greatly facilitate secure remote management of the Local Area Network computers. If the budget does not allow for the purchase of a firewall with VPN support, then you can use a basic firewall and configure it to only allow secure connections. The basic firewall will not be able to secure the connections itself; you will have to configure another device inside of the network to provide secure connections.

Simply inserting a firewall (or router that is setup to filter traffic) between the external and internal networks is not sufficient to create a secure network. It is crucial that you fine-tune the network configuration to meet the specific needs of your organization. Fortunately, nearly all devices that bill themselves as firewalls will have a default set of rules which will allow all outbound TCP/IP traffic, but only allow inbound responses to established TCP/IP connections. Things are a little different with Uniform Datagram Packets (UDP) because it is a connectionless protocol. UDP requires slightly more complicated rules and tends to vary more from one implementation to another; however the rules for UDP are still fairly restrictive by default.

With the default NAT turned on, there will also be no direct mapping between external IP addresses and internal IP addresses. If you are running any services that need to be Internet accessible which are on a LAN computer, setting up mappings between IP addresses and/or ports is a bad idea. Except for ssh on port 22, you should not create any port mappings (as discussed later). A basic rule of thumb is that if you don't need the service then keep it off. The fewer ways there are to get into your network the safer the computers on your network will be.

One of the common issues in smaller networks that have consumer routers is that the person who set up the router has used the router's built in 4 port hub to connect several LAN computers (or to connect two switches that have LAN computers on them). In addition to the poor performance that a larger collision domain creates, it is also a security risk to have LAN data going through a device that has an interface that is connected to the Internet. A basic 4 or 8 port switch is inexpensive and will make sure that only traffic that is destined for the Internet (and broadcast traffic) will be sent to the port the router is connected to.

VPN Setup

Assuming that you have properly set up the router and/or firewall and that it is properly connected to the LAN, you will then need to develop a method to bypass those defenses so that you can administer the computers on the LAN. Any secure access method, by definition, must also make it difficult for someone else to circumvent the security. Although another method of creating a secure connection between the remote computer and a local computer will be discussed later in this paper, a virtual private network (VPN) is the simplest and most secure way to gain access to local computers from the outside. The VPN will allow you to use specialized software to intercept traffic that is being sent between your computer and the remote LAN and encrypt it so that it

can safely traverse the insecure network in between, but will not unduly reduce the security of the network. Keep in mind though that once it gets to the remote VPN server it is then transferred over the LAN normally in its unencrypted form.

Mac OS X natively supports both Point to Point Tunneling Protocol (PPTP) and Internet Protocol Security Protocol (IPSec) virtual private networks. While you could use either connection method, IPSec is more secure and is rapidly becoming the VPN standard.

Unfortunately, as of this writing, Apple has not included an application for configuring or using the underlying support that MAC OS X has for IPSec VPNs. Although it is possible to set up a Virtual Private Network connection using the command line tools, there are several clients available that put a Graphical User Interface (GUI) on the underlying commands. For most people's needs, the freeware VaporSec from AFP548.com may be the best balance between cost and functionality. Its support for controlling multiple Virtual Private Network tunnels is somewhat limited, but for many people it will do everything they need. Another VPN client application that has many more features and is somewhat easier to use is VPNTracker. In addition to being a commercial product, that you will have to pay for, the latest versions do not use the same configuration files as the built-in command line tools. This may be a downside for some people who need the ability to easily start a VPN when they do not have access to the graphical user interface.

If you do not need to use a GUI, the underlying software that Mac OS X uses to connect to a VPN server is fairly straightforward. There are 3 steps to setting up a VPN using the command line: define the shared secret to use, set up the routes between the networks, and negotiate with the IPsec server to enable the secure tunnel.

Step one only has to be done once. The file /etc/raccoon/psk.txt needs to have a line listing the shared secret for the firewall in the format "Y.Y.Y.Y sharedsecret". That file is only readable by root, so you will have to use sudo to edit it.

```
sudo sh –c 'echo Y.Y.Y.Y sharedsecret >> /etc/raccoon/psk.txt'
```

In order to add the routes between the local and the remote site, you will need 3 pieces of information: local IP address, remote external IP address, and a remote internal IP range. X.X.X.X will represent the local address, Y.Y.Y.Y the firewall, and Z.Z.Z.Z/ZZ the remote network and subnet mask. If you are not used to seeing a network mask in the "slash" format, it is just the number of bits in the subnet mask written as a decimal number instead of as 4 sets of numbers separated by dots. If that sounds too complicated, just memorize that a subnet of 255.255.255.0 is /24 and for most networks that will be fine. The following command with the appropriate substitutions would create the routes:

```
sudo setkey -c << EOF
spdadd Z.Z.Z.Z/ZZ X.X.X.X/32 any -P \
      in ipsec esp/tunnel/Y.Y.Y.Y-X.X.X.X/require;
spdadd X.X.X.X/32 Z.Z.Z.Z/ZZ any -P \
      out ipsec esp/tunnel/X.X.X.X-Y.Y.Y.Y/require;
EOF
```

This creates a route from network Z to the local computer X using the gateway Y and then sets up a route in the reverse direction. Since the encryption has to happen both ways, two symmetrical routes must be created.

The last step is to tell the VPN client to negotiate the tunnel.

```
sudo racoon -f /etc/racoon/racoon.conf
```

This assumes that racoon.conf contains a configuration that will work with the settings on your VPN server. That assumption is often true, but if it is not, you will have to edit the file to create settings that match your VPN server. This usually entails changing either the encryption type or the hashing algorithm for the Phase 1 negotiation.

Once you have the VPN set up using it is very easy, but the initial process of configuring a VPN can be somewhat frustrating. If the settings on the client and the server are not exactly the same, the connection will fail, providing very little information about what the problem might be. In the Phase 1 portion of the negotiation, a connection has not yet been established and there is no way for the two sides to communicate which encryption and hashing methods to use. The key is to make sure that you use the exact same settings on both sides. The best policy is often to start by using the defaults, which on many of the firewalls are faster but provide lower encryption levels. Once you have the VPN working change the settings on both the client and server one setting at a time. The Phase 2 stage of negotiation is much simpler to configure; at that point, the two sides have the ability to truly negotiate what methods are mutually acceptable.

When you are finished using the VPN and you want to disable it, you must remove the Security Association Database (SAD) and Security Policy Database (SPD) entries you created.

```
sudo setkey –DF; sudo setkey –F
```

A common problem that you will need to be aware of is the local network having the same range of IP addresses as the remote network. The VPN software will not be able to determine whether traffic is meant for the local or remote network. Since the default NAT address on many routers and firewalls is often 10.0.1.1 or 192.168.0.1, this is a fairly common problem. Some system administrators may need to remotely administer machines on one network while on another network. A VPN would be perfect for this, but if both groups decided to use the same network range or have equipment that has limited configurability, then another method must be found to connect to the remote network.

SSH Tunneling Setup

If you do not have a virtual private network you can still encrypt traffic from your computer to the remote network, as long as you can connect to a computer on the local area network with Secure Shell (SSH). Using SSH to encrypt traffic that would normally be sent using another protocol is called SSH tunneling. AFP548.com has created a nice

GUI for the one that Apple did not include in OS X. If you prefer a GUI to the command line, AFP548.com's application, Vapor, is very good. If you decide to user Vapor, however, most people find it is much easier if they understand how the underlying SSH architecture is used to create a tunnel.

Let us assume that you have a simple router such as an Apple AirPort Base Station performing Network Address Translation. The easiest way to configure SSH tunneling in this scenario is to set up a port forward for port 22 to an internal computer running the SSH server (sshd). In many configurations this might be the Local Area Network file server; however, for better security you should make it another computer on the network. To turn on SSH access on the computer that will be acting as a gateway, go to the Sharing Pane in System Preferences and turn on remote login.

Once the port that SSH is going to use is forwarded through the NAT and once SSH access has been enabled on the computer that the port is forwarded to, you will be able to securely create a connection from a remote computer to a local computer. Although other methods are available, and you may eventually want to enable them, by default SSH allows you to connect using a username and password. From the Terminal you can use a command in the format

```
ssh -L local_port:remote_ip:remote_port username@sshserver
```

If, for example, you wanted to telnet to a router to configure it, but the router only allows configuration from computers on the remote Local Area Network you could use the command

```
ssh -L 9923:10.0.1.1:23 user@remotesite.com
telnet localhost 9923
```

The local telnet connection on port 9923 would then get transported using SSH to remotesite.com, which would open a telnet connection on port 23 to 10.0.1.1. Although this example uses telnet, any protocol would work. If in Server Settings you turn on "Enable secure connections" in the Access pane of Configure Apple File Service, you are actually using an SSH tunnel for the AFP connection. SSH tunneling is also one method that can be used when securely supporting Local Area Network clients with VNC remote control software. This is just a small part of what can be done with ssh and it is worth learning about its other features, such as using private keys instead of passwords and using ssh-agent so that you only have to authenticate once per Mac OS X login.

Servers

The best way for most small sites to secure services that need to be publicly accessible is not to run them on-site. Outsourcing the web and mail server is almost always a better option than trying to secure them on a limited budget. If there are reasons that you must locally run these services, then you can use some of the ideas in this paper as guidelines, but you will need to take additional precautions as well. Since Mac OS X server includes a web and mail server, as well as applications for configuring them you

could use the concepts in this paper to administer those functions in a secure manner. However, the overall security of your network may still be significantly weakened by the publicly available services themselves.

When speaking of a server in most small offices, it is usually a server dedicated to file sharing. You might think that you can simply turn on secure AppleShare connections on the client and the server. Unfortunately, in my testing this actually requires more setup than merely selecting the appropriate checkboxes. Doing that will not automatically cause it to use the secure connections and Apple does not provide more detailed documentation for how to enable it. Of course from our discussion of ssh tunnels, you know it can easily be done with Vapor or the command line, but that is not an end user solution. More to the point, any client connections to the file server should only be done locally or through a VPN where the data in the clear is not considered a risk inside the switched LAN environment. In the case where maintenance needs to be done to the files on the server remotely, using Vapor or the command line to set up the tunnel is probably not a significant burden. Since AppleShare uses port 548 setting up a tunnel is as simple as executing

```
ssh -L 10548:localhost:548 user@remotesite.com
```

and then in the Mac OS X "connect to server" window using localhost:10548 as the computer to connect to.

Mac OS X Server has several administration tools included with it and their use over an unsecure network is a very bad idea. In fact if you connect to the port 311 you will see that it is actually has a web interface of sorts (it does require a login and runs over https). When using these administration tools, it is best to use some additional encryption on them, if for no other reason than it hides what is actually running on the network. If all connections into the network are over a single port it makes it much harder for a person with bad intentions to figure out how to attack your computers.

The Vapor application has an option for forwarding port 660, which is the server admin port, but remember that there are several different ports that the administration programs use. Usually it is best to tunnel all the ports that you might use before you start administering the server. A list of the ports that are commonly used on the Macintosh is available from Apple. For the Mac OS X admin tools the important ports can be found in the following table:

Port	Application
311	Server Monitor, Server Status, Workgroup Manager
548	AFP (AppleShare, File Sharing, Apple File Service), iDisk
625	Open Directory Assistant, Workgroup Manager
660	Server Admin, Server Settings
687	Server Monitor, Server Status

You may need to tunnel several of those ports if you are going to do any administration of the server. A few of the admin tools will launch one or more of the other admin tools. Since those ports are not going to be open on the firewall non-tunneled traffic will fail to work, as opposed to sending traffic in the clear, but it is easier to just establish the tunnels are in advance. Using ssh you can put multiple port forwards into one command. If you run the command as a super user you can setup tunnels to the well-known ports (ports below 1024 cannot be opened by non-administrator users). A command such as the one below would allow you to administer nearly all aspects of the server even if the firewall only has port 22 open.

```
sudo ssh -L 548:server:548 -L 311:server:311 -L 625:server:625 -L
660:server:660  -L 687:server:687 user@remotesite.com
```

Your local computer would act as if it was the server and any connection that you made to localhost on those ports would actually go to the server over the encrypted link. Of course you may not want to use the standard ports if you already have services such as file sharing running on your local computer.

Until now only SSH's ability to forward ports has been discussed, but that was not what SSH was originally designed for. As its name implies its main purpose is to provide a secure shell environment. For that reason, it is perfectly suited for doing remote administration of the server where you do not need a graphical user interface. Connecting to a remote computer with "`ssh remotecomputer`" will give you the same command line you get in a local Terminal.app window.

Although many of the standard Unix methods for administering a computer do not work with Mac OS X server, Apple has provided alternative methods to do so from a command line. For example, Mac OS X maintains much of its configuration information in netinfo and various XML property lists instead of the traditional text files in the /etc directory that other Unix-like operating systems use.

Many administrators end up installing third-party remote control software such as Netopia's Timbuktu™ on the server so that they can change the server settings that are not available through the Mac OS X Server Admin Tools. Although remote control software may be necessary on a Mac OS X Client, Apple has provided command line utilities for configuring most aspects of Mac OS X Server.

There are several useful command line utilities that you can only use on Mac OS X Server. One of these is the `networksetup` command. This command will let you change the same settings as you can in the Network pane of System Preferences. Similarly the `systemsetup` command can be used to query or set other settings such as the time or the disk from which to start up. Not needing a GUI to manage most aspects of the server has both pros and cons. It not only makes it easier for you to administer the server but also means that if anyone does compromise the server they will not need to have any specialized tools beyond a shell to reconfigure anything they want.

One of the most useful command line tools from a security standpoint is the `softwareupdate` command, which will inform you of any security patches (and other Apple released software) that has not been installed and then let you install it completely from the shell. I have `softwareupdate` run as a weekly cron job on some of my servers and the results are then mailed to me so that I can see what servers still need security updates applied to them. Unlike many of the other command line tools this runs on both the server and client versions of the operating system.

If you use any of the above commands, checksumming software such as Brian Hill's CheckMate, will usually find some files to be changed and issue a warning. One of the advantages of CheckMate is that it has no way of updating the checksums from the command line. You may not want to receive security warnings every time a software update is applied though. Also if there is a local office manager who has to occasionally administer the server they may panic if they see a message on the screen that critical system files have been modified.

In addition to all the steps above you should also setup software firewalls on the servers so only the needed ports are open. Even though the LAN environment is considered safe, you should still take the basic precautions of limiting access. However, as Apple says in Security: Mac OS X and UNIX:

> … building firewall rulesets can cause your network connection to stop working, so be sure to invoke ruleset additions and changes directly on your machine and not from a remote location.

That is a good general rule for any of the security steps that you take. Before you deploy them make sure that they will still allow the required services to operate. The deny by default methodology is the correct one from a security perspective, but when the site is several hundred miles away you have to be very careful. By over-securing the network you can actually compromise the long-term security. Often the solution that most institutions implement after security policies get in the way of doing business is to open up the network far beyond what it is actually required.

In addition to local software firewalls, basic security such as complex passwords, and anti-virus software should be used on the server. Even though the LAN is considered safe and the only way to administer the servers is through secure processes standard best practices of safe system administration cannot be ignored.

Clients

Although the built-in Mac OS X functionality for administering client computers is less sophisticated than that for servers, there is also a lot less to go wrong on a client computer and the impact of any problems that occur is usually significantly smaller. Although the command line can be used to do some tasks, most remote management of client computers is best accomplished with replication of the GUI environment itself and that brings up a slightly different set of security issues than the administration programs that are used with Mac OS X Server.

The most common options for remote desktops on the Macintosh are Timbuktu, Apple Remote Desktop (ARD), and VNC. Each of these has similar security issues, but may require different solutions. Because VNC is free and cross platform many system administrators, particularly those that come from a Unix background, will choose to use it for any remote administration that requires a GUI. VNC will be used in the example, but both VNC and Timbuktu can be secured using similar techniques. If you are interested in using Timbuktu the method is identical except that Timbuktu uses port 407 instead of in the 5900 range.

There are several methods for securing VNC and Timbuktu. The best method would be to use a Virtual Private Network and the authentication within the applications (the Local Area Network is considered to be safe enough to run VNC between the LAN machines, for this example). If the site is not using a VPN, then the next best method is again an ssh tunnel. Assuming that you have sshd running at the remote site in some Internet accessible way such as the method mentioned earlier, you will forward a port from that computer to the client to which you want to connect. A sample basic command to do this is:

```
ssh -L 5911:10.0.1.15:5900 admin@host.example.com
```

Then using a VNC client such as VNCThing, you would then be able to connect to localhost:5911 and be connected to display zero of the VNC server running on 10.0.1.15. It is also a good idea to only run the VNC server when you are actively using it. OSXvnc, which is the most common VNC server, can easily be launched from the command line. Instructions for how to launch the OSXvnc from the command line can be found in the OSXvnc 1.11 Frequently Asked Questions

> Q: How can I start up OSXvnc from a command line?
> A: To startup OSXvnc from command line, go inside the Application directory (OSXvnc.app) and launch the OSXvnc-server process. To change parameters you will need to give it arguments (-rfbport to set port, -vncauth to specify a password file, etc). For usage run the command with -help.

Launching the VNC server only when you need it would add the additional step of needing to connect to the client with a regular ssh connection to start the server from the command line, but that is small trade off compared to the benefit of only having services running that are actually needed for day to day use of the computer.

Apple Remote Desktop is different from VNC and Timbuktu in that it cannot be run through SSH tunnels. Apple Remote Desktop uses UDP packets and since ssh tunnels do not work with UDP, that method will not work for securing it. If you have a VPN then you should be able to use ARD over the VPN. Even with a VPN, Apple Remote Desktop can have problems when used remotely because it tends to use large packets that may get fragmented. For security reasons fragmented packets are sometimes not allowed through firewalls. If you do not have a VPN or if ARD is not compatible with your VPN, then perhaps VNC or Timbuktu are better options for client management.

Conclusion

Remote administration of Mac OS X can been implemented in a fairly secure manner using only the built in software if needed. With the addition of a few freely available tools a graphical user interface can be put on to the security features. Using the information in this paper you should now be able to both design a reasonably secure small Internet connected network and securely administer it with either an IPsec VPN or SSH tunnels. There are many other aspects to a secure network design beyond what has been discussed here. The resources listed in the references section have details that are not covered here and it is worthwhile to consult them to learn more about securing Mac OS X.

References

Apple Computer, "Well Known" TCP and UDP Ports Used By Apple Software Products", 19 December 2002 ,
URL:http://docs.info.apple.com/article.html?artnum=106439 (1 March 2003)

"Using VNC with SSH",
URL: http://www.uk.research.att.com/vnc/sshvnc.html (31 March 2003)

"OSXvnc - Redstone Software - VNC Server for OS X",
URL: http://www.redstonesoftware.com/osxvnc/ (3 April 2003)

"OSXvnc 1.11 Frequently Asked Questions", 13 March 2003
URL: http://www.redstonesoftware.com/osxvnc/OSXvnc.html (21 March 2003)

"Security: Mac OS X and UNIX." Web Development & Mac OS X,
URL: http://developer.apple.com/internet/macosx/securitycompare.html (21 March 2003)

"AFP548.com: Vapor",
URL: http://afp548.com/Software/Vapor/index.html (31 March 2003)

"AFP548.com: Vapor", (title does not reflect the content of the page as of 4/3/03).
URL: http://afp548.com/Software/VaporSec/index.html (31 March 2003)

"equinux Inc. - Products: VPN Tracker",
URL: http://www.equinux.com/us/products/vpntracker/ (3 April 2003)

Rennich , Joel ,"Flying Racoons: IPSec, Mac OS X Server 10.2, and you", 22 August 2002
URL: http://afp548.com/Articles/Jaguar/ipsec-1.html (31 March 2003)

Rennich , Joe, "Flying Racoons: Clients? We don't need no stinking clients",
30 October 2002
URL: http://afp548.com/Articles/Jaguar/ipsec-3.html (31 March 2003)

"OpenSSH",
URL: http://www.openssh.com/ (3 April 2003)

"A possible fix for Apple Remote Desktop WAN/VPN issues",
URL: http://www.macosxhints.com/article.php?story=20030328171852761 (3 April 2003)

"Simple Setup for ssh-agent on Mac OS X 10.2",
URL: http://www.withay.com/macosx/ssh-agent-jag.html (3 April 2003)

Fred Kaye, Marc Epard and Kim White, "Netopia - - TPM_031:Firewall Issues", Timbuktu Pro for Macintosh, 28 January 2000
URL: http://www.netopia.com/en-us/support/technotes/software/tb2mac/TPM_031.html (31 March 2003)

Hill, Brian, "Mac OS X Software",
URL: http://personalpages.tds.net/~brian_hill/checkmate.html (31 March 2003)