MacAegis v3.0

MacAegis is an easy-to-use protective shell for Macintosh programs that provides a high degree of security for your application.

This Chapter contains the following sections:

- MacAegis Overview
- MacAegis
- UniKey Inst
- File Encrypt

MacAegis Overview

MacAegis is a ready-made encryption and protection package that works in combination with a special hardware key to provide complete protection for your software application.

MacAegis takes your application program as input, encrypts the executable code, adds secure application specific interrogation routines and creates a new executable file that incorporates your encrypted application and MacAegis's interrogation routines.

The interrogation routines check for the presence of the proper MacAegis key for your application. If the correct key is found on the modem or printer port, MacAegis decrypts your application *in memory only* and executes the application. If no MacAegis key is found or the wrong one is attached to the user's computer, MacAegis will neither decrypt nor execute your application. MacAegis will, at this point put up a dialog telling the user to attach the security key to the port and halt the program until the proper key is attached.

You have the option, when you set up MacAegis for your application, to choose whether to keep MacAegis memory-resident or not. Resident mode will check for the key's presence repeatedly, not just once at start-up.

MacAegis Benefits

Protecting your application with MacAegis is a simple and straightforward process. Some of the benefits of MacAegis are listed below.

Quick and Easy
MacAegis takes only a few minutes to implement and requires no knowledge of security programming.

Transparent
The MacAegis keys are transparent to the modem and printer ports. This means that you may attach the MacAegis key to the serial port and attach a modem or printer (any serial device) to the key. The MacAegis key does not interfere with the device activity on the port; signals simply pass through the MacAegis key to the intended device.

Remark: During key interrogation all activity on the port must be

inactive.

MacAegis routines are highly optimized so that a user of your application is not aware of MacAegis's presence. Once your application is decrypted and loaded, it runs as normal. The user is free to make backup copies of your shelled application, as well.

Secure

MacAegis contains sophisticated anti-debug and encryption techniques to ensure an extremely high level of application security. The patented Software Security hardware key and proprietary encryption and decryption algorithms provide powerful protection.

Your application is decrypted as it executes into memory only(RAM), MacAegis never decrypts onto a disk or other permanent storage media.

MacAegis also provides a virus detection option that checks to ensure that your application has not been altered.

Versatile

You can use MacAegis to perform the following functions:

- Language independent; encrypt almost any 68k or Power PC native(PEF file format only) executable program, regardless of size

- Encrypt/Decrypt data files that belong to or are created by your protected application

- Protect multiple applications with a single key or with multiple keys

Efficient

- Adds only about 40K to the file size, depending on what MacAegis options you select

- Uses very little memory overhead during execution

The MacAegis Diskette

The MacAegis diskette contains the following files:

MACAEGIS Macintosh shelling program

FILE ENCRYPT A program to encrypt/decrypt data files

UNIKEY INST An executable that installs application specific information into the hardware key

MacAegis

This is a program that will allow you to protect an application by securing it with a shell-like wrapper that offers runtime decryption of your application.

MacAegis Shelling Process Restrictions

During the actual shelling of your program there are certain restrictions that must be followed in order to complete the shelling process. If any of these restrictions are not followed the shelling process may stop.

These restrictions apply only during the shelling of your application, they do not apply to the execution of your protected (shelled) application.

The shelling process restrictions are the following:

- There must be no hardware or software debuggers loaded while shelling
- Under System Software 6.0x, multi-finder must be disabled
- There must be no background communications or printing while shelling
- Run MacAegis shelling program on a 680[2,3,4]0, or

Power Macintosh only.

Important Note The *protected* program will **not** have these restrictions

Getting Started with MacAegis

Shelling Recommendations

We suggest the following when shelling your application:

- Restart machine holding down the shift key, before shelling (this will turn all extensions off, disabling any debuggers)
- Do not try to shell a program that already uses a self-checksum, it will cause the program to fail

Preparation

In order to prepare your application for the shelling process, do the following:

- Follow the restrictions and recommendations described earlier
- Create a folder on your hard disk and name it something like *Protected*
- Copy the MacAegis program and the SavedParam files from the MacAegis diskette to this new folder
- Make a copy of your application to be shelled and move it to this folder, rename it something like *filename.prot* (just to be able to distinguish it from the original, because MacAegis will overwrite this file during the shelling process with the protected version)

Now you are ready to run the MacAegis program.

Running MacAegis

1. Double click on the MacAegis icon

At this point there will be a noticeable delay of several seconds before the MacAegis program starts(approx.7 seconds on a Quadra 800). This delay is shorter on

Power Macs.

Then a screen will appear prompting you for a unique identification string (Application Key). This is the `Define the Ap-Key` dialog from the `Shelling` menu.

`Define the Ap-Key`
`(⌘D)`

This dialog prompts you for information about the application to be shelled. Choose either **Get** to retrieve information on an already existing application or **New** to create the necessary application information needed by the shell.

Choosing **New** will prompt you for some application specific information. Enter the name of your application and then some comments. Then enter an 16-byte application key. The 16-byte application key can be either a text or hexadecimal string . Then enter a 1-byte application ID.

2. Click OK to continue with shelling

3. Go to `Shelling` menu and select the **Tune up the** `parameters` menu item

This brings up the **Tune up the** `parameters` dialog from the `Shelling` menu

`Tune up the parameters (⌘T)`

This dialog lets the user select the parameters he wants to shell his application with. The options are as follows and some check boxes and radio buttons allow you to set these:

Shelling Method
The Shelling Method radio buttons allow you to select the shelling method most appropriate to your application. You may choose one of the three following choices:

1. JumpTable - shell/encrypt just the program's jump table

2. First executing code segment (according to jump table) - shell/encrypt just the 1st exec. code seg.

3. Join jump table and 1st exec. code segment - shell/encrypt both jump table and 1st exec. code segment, by first incorporating the 1st exec. code segment into the jump table and then encrypting the "modified" jump table. This method may not work in some cases, but may be worth trying. MacAegis can detect some possible incompat-ibilities with your application and this method. If

your program behaves abnormally please re-protect with this option disabled.

Data File Encryption

The File Encryption option allows you to have runtime self encryption/decryption of data files that belong to

your application or are created by your application.

Choose the various options that best support your particular application.

There are two types of file encryption/decryption supported, they are for files created by your application, and files used by you application (existing already).

For files used by your application use our **FileEncrypt** program provided (see section on **File Encrypt**). These files will automatically be decrypted by the protected application when used. Remember to use the **File Encrypt** program to encrypt these data files with the same encryption key used during the shelling process.

For files created by your protected application follow the instructions below.

If the **Data File Encryption** check box is checked two additional check boxes will become enabled.

The first box if checked will allow you to encrypt/decrypt files created by your protected application containing a certain substring, which you can specify, in their filename. There is an edit text field available to enter this string.

The second box if checked, allows you to specify a list of files to be encrypted/decrypted by your application. There is an edit text field to hold the list of filenames. Be sure to separate filenames in this list by a semicolon. A wildcard character (*) can be entered in this edit text field to specify all files created by your application to be encrypted/decrypted.

If the **Data File Encryption** check box is unchecked there will be no data file encryption at all.

Selftest

The Selftest option, if checked will include a self-test in the system. This test is useful for virus-detection.

The self-test uses a checksum to determine whether the application code has been modified. If modification exists, the protected program will terminate with an error message.

Initially try this option, if it causes problems in execution re-Shell with this option unchecked.

4. Set the parameters that you desire for your protected application, Click **OK** and go to the `Shelling` menu and choose the **Misc. options** menu item. Check the **Resident Mode** check box if you'd like to repeatedly check for the hardware key every so many seconds. This time period can be set by editing the available edit text field. If you would like to look for the key only upon startup, do not check this **Resident Mode** check box. Then Click **OK** to proceed.

5. Go to the `Shelling` menu and choose the `Open a file to shell` menu item

This will bring up the `Open a file to shell` dialog *(there may be a 1-2 second delay at this point)*

`Open a file to shell (⌘O)`

This dialog lets the user select the application he wants to protect with the MacAegis from the standard `Open` dialog

6. Locate your application with this dialog and click on the `Open` button, to shell your application

If successful, a dialog will alert you that this process has been completed.

7. Choose the `Quit` option from the `Quit` menu to exit

8. Reboot your computer before shelling another program or using either **UniKey Inst** or **FileEncrypt**

You have now protected your application with MacAegis.

UniKey Inst

This is a program that will allow you to install application information in your hardware key so it will operate with your Shelled application.

Running UniKey Inst

1. Double click on the **UniKey Inst** icon

Then a screen will appear prompting you for a unique identification string (Application Key).

This is the `Define`

Ap-Key dialog from the MacUniKey menu.

Define Ap-Key
(⌘D)

This dialog prompts you for information about the application to be shelled. Choose either **Get** to retrieve information on an already existing application or **New** to create the necessary application information needed by the shell.

Choosing **New** will prompt you for some application specific information. Enter the name of your application and then some comments. Then enter an 16-byte application key. The 16-byte application key can be either a text or hexadecimal string . Then enter a 1-byte application ID.

2. Click **OK** to continue

3. Go to the **MacUniKey** menu and select the **Erase the UniKeys** option

A dialog will appear with a list of keys found, double click on the key to erase. After you have erased all the keys in this manner, click **Exit** to continue.

Note: when receiving new hardware keys from SSI, always erase the keys before programming them, otherwise programming will fail.

4. Go to the **MacUniKey** menu and select the **Write the UniKeys** option

A dialog will appear with a list of keys found, double click on the key to program. After you have programmed all the keys in this manner, click **Exit** to continue.

5. Go to the **MacUniKey** menu and select the **Test the UniKeys** option

A dialog will appear with a list of keys found, double click on the key to test. After you have tested all the keys in this manner, click **Exit** to continue.

6. In the case of any improper operation check the MacUniKey's connection and/or reset serial port(s) using `(⌘S)` key combination or by choosing the **Reset the serial ports** option.

7. Choose the `Quit` option from the `Quit` menu to exit

8. Reboot your computer

File Encrypt

This is a program that will allow you to encrypt/decrypt data files that go along with your shelled application. This feature offers runtime encryption/decryption of your application's data files.

Running File Encrypt

1. Double click on the File Encrypt icon

Then a screen will appear prompting you for a unique identification string (Application Key). This is the `Get the appl. Key` dialog from the `File` menu.

`Get the appl. Key (⌘K)`

This dialog prompts you for information about the application to be shelled. Choose either **Get** to retrieve information on an already existing application or **New** to create the necessary application information needed by the shell.

Choosing **New** will prompt you for some application specific information. Enter the name of your application and then some comments. Then enter an 16-byte application key. The 16-byte application key can be either a text or hexadecimal string . Then enter a 1-byte application ID.

2. Click **OK** to continue

3. Go to **File** menu and choose the **Encrypt a file** menu item to encrypt a data file (or **Decrypt a file** to decrypt a data file).

4. Then choose a file to be encrypted/decrypted

5. Continue this process until all data files that you want encrypted have been.

6. Reboot your computer

Remember to use the same encryption key, to encrypt the data files, that was

used earlier in the shelling process, otherwise files will not be self-decrypted successfully during runtime.