Chapter 8 Hardware Key Protection

Not many people know too much about this protection scheme. But one obvious observation is that companies are making there own devices and confused companies with low marketshare are buying into there hardware protection scheme, which really is not that good to tell you the truth.

Basically what the Eve™ hardware key does is it hooks up to your ADB port and has a chip in it which can be read from. All of the packages I have seen that use the Eve™ protection key come with an Eve INIT. Basically this INIT reads one or more values from the Eve key and stores it somewhere in memory. The program that will then do a Compare to the memory to see if the values match if they do then the program will continue, however if there is a null value found the program will warn you that the Eve Key is not present and then quit. If the value is incorrect it will tell you that you either have the Eve hooked up incorrectly or that you are using an Eve from a different software package.

The easiest way around the Eve Hardware key would be to change the Branch conditions of the checks it does 1) to locate Eve and 2) to compare the values. Once you have changed these branch conditions the program will work as it would have if the Eve protection had not been implemented.

From what I have heard some software locations or other important data to the program is stored in the Eve™ that may be needed upon run time. For example, if a program is encrypted and the decryption routine is stored on the Eve™ key it would be almost impossible to deprotect the program without a valid Eve™ key to work with.

So, I would just use MacsBug the way we have been and just trace through for the checks. Eve™ will let you know when something is wrong through the use of dialogs, so tracing is pretty easy. Find those branches and change the conditions.