Software Authorization Systems

Paul A. Suhler, Nader, Miroslaw and Neil Iscoe
Presented by Lagisetty Venkata Raghavendra Kumar

Introduction

This paper starts with the statement “For every piece of business software sold, at least one illegal copy exists”. The paper describes and classifies various software protection methods. Software piracy presents a major problem to software vendors and distributors as it reduces the profits of vendors who spend their money and time in developing software. The problem of software piracy is not so prominent in the mainframe and mini computer world because software for these computers requires constant support and updates from the vendor. But this is not so in the world of micro computers where there is a large market for the software leading to substantial amount of theft.

The battle between the software vendors and pirates continues, each one trying to beat the other. Software vendors protect themselves from the pirates by using software protection schemes to prevent unauthorized use. And the pirates are continuously trying to break the authorizing systems, armed with tools like debuggers, dis-assemblers etc.

The goals of an authorization system should be to prevent unauthorized copying or execution of illegally sold software, or at least make the life of the pirate as difficult as writing the software from scratch. A software authorization system should meet the following criteria:

- it must be inexpensive and easy to use,
- it must be compatible with existing software programs,
- it must be easy for the software vendor to incorporate in his products and distribution system.
In the next section, we shall look at various authorization techniques and ways of implementing those techniques.

**Authorization Approaches**

There are three basic authorization techniques and several ways of implementing those techniques. The three techniques are:

- copy protection,
- validation,
- encryption.

The implementations are based on hardware devices: floppy disks, special micro-processors, and devices attached to the system bus or an I/O port. The subsequent sections will give an overview of these techniques and implementation methods.

**Authorization Techniques**

Authorization can be achieved by either making it impossible for an unauthorized person to obtain a copy of the original software or by preventing an unauthorized person from executing an illegal copy of the software. The former can be achieved by the *copy protection* technique and the latter can be achieved by *validation* and *encryption* techniques.

*copy protection:* In this technique, we try to make the master disk on which the legal software exists uncopyable by using micro computer operating system utilities. This can be done by inserting dummy file segments having erroneous CRCs or invalid control codes in the program file. These erroneous segments will be ignored by the copy utility if we try to make a copy of the master disk. When run, the copied version of the program will look for the missing information and abort. A limitation to this approach is that the pirate can use a bit copier instead of the standard copy utility which would copy the entire disk bit by bit without interpreting any CRCs or control codes.
Validation: In this technique, the protected software can check for the user’s right to execute it. Blocks of validation code are embedded in the program to locate a unique key in the system. If the key is not found, the program is assumed to be on an unlicensed system. The key can be stored:

1. on the program disk (in a location not normally copied),
2. in a read-only register in a hardware device attached to the system externally,
3. or in a detached device the user must query externally in response to a program prompt.

The first method of storing the key has the limitation that it can be defeated by bit-copying. The second method might use up I/O ports and it requires the user to attach the device to the system each time he needs to execute the program. Even the last method causes similar kind of inconvenience to the user.

Encryption: The program can be encrypted so that before being executed it must be decrypted according to a unique key available only on the licensed systems. Decryption can be done at load time by CPU or an additional device or it can be performed at run time by a crypto processor.

The limitations of these systems are that decryption increases the CPI (cost per instruction) and non-encrypted copy of the program can be found if we can determine the decryption key.

Implementation Techniques

Implementation techniques are categorized according to the kind of hardware involved in the process. The categories are:

- Disks,
- Detached devices,
- I/O port devices,
- Crypto processors.
Disks: The difficulty of copying the disk itself is intended to prevent the unauthorized copying. As said before, it’s easy to defeat this by using a bit copier. Validation and encryption techniques can use the disks to store the keys at some unusual locations. It requires the user to insert the program disk each time even though the program itself is on the hard disk.

Detached Devices: Validation techniques can use detached external devices which store the keys.

I/O Port Devices: Key devices are attached to the computer though an I/O port. The user need not be aware of the validation process in this case. Encryption systems can use these devices but the low I/O bandwidth makes it a time consuming process.

Crypto Processors: Encryption techniques can use these crypto processors. They decrypt the instructions using on an internal key, and execute them. In such systems, the code would never appear in plain form and hence is unreadable. The limitations of using these processors are:

- time consuming process of decryption
- if we need to replace the CPU, then the new CPU should have the same internal key or all the existing software should be re-encrypted according to the new key.
- the security of encryption/decryption depends on the length of the key used.

Other Issues

- If one copy of the program is purchased, can it be executed on every computer in a network?
- customized validation code for each customer which would recognize his key.

Conclusion

There are 3 basic authorization techniques which can be implemented using disks, detached devices, I/O ports or crypto processors. To satisfy the
vendors and distributors, the software authorization system should be inexpensive and secure against attacks. To satisfy the customers, it should be inexpensive, easy to use and should not degrade the performance of the program. This paper provides a frame work for understanding and comparing various software authorization systems.