

Current Anti-piracy Systems

In this article, we take a look at some of the methods which have been employed to prevent software piracy, their loopholes and security issues. We would also be addressing the "privacy issues" as there is a continuous fight between piracy and privacy laws. However, that topic would be better discussed in the legislation section.

Software Level Protection Systems :

Serial ID :

This is probably the oldest method employed even now-a-days to combat piracy.

The working as such is pretty simple. All softwares come with a unique Serial ID. If the correct serial ID is not entered, the installation procedure would stop abruptly. The serial ID need not be unique; it could even be a sequence of characters whose sequence satisfies a particular mathematical algorithm (formula).

The biggest drawback of this system is that there is nothing to prevent the user from freely distributing the copy of software along with the serial number that came along with it. He could simply copy the software onto the CD along with a serial.txt or id.txt file which contains the necessary authentication information. Once that is done, any number of copies of the software can be made which totally eliminates the usefulness of the Serial ID check employed.

Some of the examples are Windows 95,98,2000,ME Office 95,97,2000.

It is very alarming to note that almost all major softwares still depend upon this technique to prevent the piracy of their products.

Necessity to Authenticate :

This method uses the Serial ID protection at the start and then once the user enters his personal information, the software generates another serial number which may depend upon your hardware (like CPU, motherboard information), your serial ID or your personal details or a combination of the above. This new Serial ID needs to get transferred back to the company and they give you the authentication code to enable the software to run.

Before we see the flaws in the system, the disadvantages of such a system are very evident. Firstly, whenever you change your hardware, you might need to obtain new authentication code. Although this might not seem as much of a drawback assuming you don't change the hardware of your machine too often, it still means that if you are a power user, or a tweak-enthusiast who likes to try new hardware, or simply a guy who does a lot of testing and benchmarking, such a system would be ridiculous.

Now, we shall see the flaws. Firstly, it is pretty evident that since the 2nd serial number is generated using your hardware information, and this is processed to give you the new authentication code, if anyone hits upon the correct mathematical relation between the 2nd serial number and authentication code, he could generate that offline without sending any information to you.

For example, 3D Studio Max still uses the above method and suffers the above mentioned attack.

Secondly, the software simply "**checks**" the presence of the Authentication code. Such things could be patched by 'Assembly Level Programming*' and the software would run without the authentication code.

A typical example would be Windows XP Final version which was flawed in the above way.

Thirdly, suppose the software runs without the authentication information for a certain number of days (which certainly seems to be the case with almost all software which offer protection coming under this section), it could be easily fooled into thinking that the days never run out (which has been the case with Windows XP beta versions).

So, although this definitely offers better protection than the normal serial ID method, it offers very few practical benefits as it does nothing really to combat piracy.

Some of the examples are Windows XP, Maya, 3D Studio, AutoCAD etc. which have been cracked** very easily.

***Assembly Level Programming :**

Assembly level programming is nothing but programming the hardware directly without using any 3rd party software like C, Basic etc. In this method, we feed the instructions directly to the microprocessor using Hexadecimal numbers. We shall see how this works in detail later.

****Crack :**

The process of making a non-authenticated software work like authenticated software is called cracking. It may include within itself various methods, like Assembly Level Patching, Key Generators etc.

Hardware Protection Systems :

Key Disk :

This method was used to protect softwares a long time ago. This method is no longer used due to the demise of the 1.44 MB floppy disk, but still, here goes.

In this method, the software used to come with a floppy disk (called Key Disk) which you needed to insert whenever you need to run the software. The software upon execution, checks for the presence of the Floppy Disk and only if this Key Disk (the floppy) is present, the execution would continue. This method is almost similar to the Serial ID, except that here, the user doesn't enter the authentication information by himself but that the Authentication information is automatically verified by the software.

However, there are disadvantages to this system also. One could distribute the software with multiple copies of the key disk, as the key-disk is only a floppy, copying the contents should not be much work. Secondly, this system is also vulnerable to Assembly Level Patching.

Thirdly, in this system, if the floppy disk gets corrupt (whose chances are very high) then you have wait for your next key disk to come in order to use the software. Thus, this system is no longer used as it is tougher to implement on Serial ID but offers very little benefit over the Serial ID.

LPT1 or COM1 Lock :

This method although is probably the first of its kind to offer hardware level protection is used by very few software companies, mainly due to the fact that this method doesn't really offer much better protection, even though it appears to do so.

In this method, the software comes with a device which can be fitted to your COM port or your LPT1 port. Whenever you execute your software, the software checks for the presence of this device on the appropriate port and only then executes.

This method offers 2 advantages over the key disk method. Firstly, the chances of hardware device failing is very very low hence it is reliable for the user. Also, it is tougher to duplicate this hardware unlike a floppy disk.

However, as this system only "checks" for the presence of the device, it is highly vulnerable to assembly level patching. Tally, a popular accounting software, uses this system, but the flaw of this system is evident from its failure. Also, there are some programs out there which

can simulate the presence of certain devices like the printer or the mouse, even if these devices don't actually exist.

CD-Media Corruption:

This method is used by games or certain Audio CDs to prevent piracy. Once you read the working of this method, it becomes evident as to how it cannot be used for softwares.

After floppy disks, CD Media took over as the medium to distribute various softwares and games. However, this very CD Media is what probably set piracy to untraceable heights and is still the biggest threat to piracy (probably peer to peer over the internet is a bigger abode of piracy). Thus, developers thought by somehow preventing mass duplication of CD-Rom, they could prevent piracy. You shall see how this was not to be.

In this method, the contents of a CD Media are intentionally corrupted, thus your Audio CD player would ignore such errors, but if you tried to copy the contents onto a hard disk or convert them to mp3, it would not be possible. In games, usually, the last part, which could be an audio track or a file which is unnecessary, is corrupted which is required for the game to run properly but cannot be copied onto the hard disk.

The flaws of the method are as follows. First of all, the method assumes that once corruption takes place, data cannot be recopied to another CD. But this itself was a joke because there are plenty of softwares out there which allows the users to write such corrupted CDs to other blank CDs using a method called "Raw Mode".

Secondly, this method needs the CD to be present in the drive while the software is being accessed. If, by any chance, the software can be fooled into thinking that the CD is present (Assembly Level Patching), then this system becomes useless.

This method is still used for games with absolutely no success whatsoever. Also, a recent article that came across me :

'Controversial copy-protection mechanisms on CDs could be negated with something as simple as a marker pen.

According to one German geek who sent the tip to technical magazine Chip.de, a variety of copy-protection systems, including Cactus Data Shield and KeyAudio, which also stop music CDs being played in CDRom drives, can be circumvented with a felt-tip pen.'

Here, the data which was corrupted on purpose by the developer was marked by a felt-tip pen, and by doing thus, even the Cd-Roms were able to copy all the necessary data and ignore the data marked by the felt-tip pen.

Intel's Microprocessor Serial ID:

This method achieved much more criticism than success. Although, this essentially isn't a hardware protection, I have included it in this section because this Serial ID was embedded in the Intel Pentium 3 processors.

The purpose of this system was to negate piracy. Here, every Intel Pentium 3 processor came with a unique serial ID. Then, after you install your operating system and other softwares, this information along with your Serial ID information was sent without your knowledge to be evaluated. As the processor serial ID was unique and "uncrackable", this system couldn't be fooled, but it was inefficient because it still means that after receiving the information, you need to track down the culprit, which even though is not very tough, is not very easy. Also, this system infringed various anti-privacy laws and all consumers united against this.

Obviously, Intel had no choice but call off the system. They had to disable the Serial Numbers of their processors.