International Journal of Information Technology and Knowledge Management
July-December 2008, Volume 1, No. 2, pp. 379-383

TRAINED SOFTWARE

Parvinder Singh & Darshana Hooda

Abstract

Software piracy is major problem in all over the world. A new model is proposed using trained software to make the software/application trained for a particular system to avoid this practice. Trained software can be proved very powerful solution to meet the acute problem of software piracy which causes unbelievable loss to this promising industry. This model uses system specific information to train the software to provide 100% security against piracy.

Introduction

Types of Software Piracy

- Soft lifting: purchasing a single licensed copy of software and loading it onto several computers contrary to the license terms. For example, sharing software with friends, co-workers and others.
- Uploading and downloading: making unauthorized copies of copyrighted software available to end users connected by modem to online service providers and/or the Internet.
- Software counterfeiting: illegally duplicating and selling copyrighted software in a form designed to make it appear legitimate.
- OEM unbundling: selling standalone software that was intended to be bundled with specific accompanying hardware.
- Hard disk loading: installing unauthorized copies of software onto the hard disks of personal computers, often as an incentive for the end user to buy the hardware from that particular hardware dealer.
- Renting: unauthorized selling of software for temporary use, like you would a video.

Two mechanisms are employed to keep the privacy of software to minimum level. First mechanism uses the hardware locks to make it impossible or at least very difficult to make the illegal copy of software. Second mechanism uses the passwords, fingerprinting to discourage the unwanted duplication and distribution [1].

With fingerprinting we mean the act of embedding a unique identifier in software of some kind, in such a way that it will be difficult for others to find, remove or destroy the identifier. The purpose of fingerprinting is to make a number of otherwise identical copies unique, so that if an illegally made copy is found; it is possible to trace the person or persons who made it. In order to be able to do this, the identifier used must be chosen carefully, generally a code. For this to be of practical interest, the embedding technique must be such that the document is not degraded so much that the fingerprinted copies of the document are unusable.

Various codes such as binary codes, reed-Solomon codes, random codes etc. [2] are used for fingerprinting purpose. These codes can be broken, if some clever pirates (Users involved in illegal redistributions) make strategies to fool the seller [3] [4]. So are here by proposing making software intelligent using concept of training.

TRAINED SOFTWARE: STATE OF ART APPROACH

By preventing software piracy we can contribute apparently in economic growth of software industry. In this paper we are proposing a fresh and innovative way to prevent the piracy, which is much simpler and secure than existing methods as well as economical.

Proposed solution makes use of two programs- Builder Program and Trainer Program by the software seller. Under proposed solution, we pass software to a builder program. This builder program creates new modified copy with some additional placeholder statement for taking system information later on.

Seller will pass this modified code as well as system specific information (system information of the system on which we want to install our software) obtained from the client to the trainer program.

MAJOR COMPONENTS OF PROPOSED SOLUTION

Proposed solution package consists of following two modules to stop software piracy-

(1) **Builder program**: It accepts source code as input and makes necessary modification in the source files by inserting own modules/codes. These codes are placeholder statements which will be required later on to take information from the client.

(2) **Trainer program**: It requires two types of Inputs

Modified Source Code: Output generated by Builder program.

System Specific Information: Provided by the client about the system on which application will be installed.

Additional work is required to find all the system specific information of static nature and to put that information at single place. So one additional module will be developed to gather system specific information from client. This information will be used by the software seller to bind the software to that particular system. This additional module keeps client isolated from the any type of overhead of searching the system specific static information.

WORKING OF PROPOSED SOLUTION

Figure 1 shows the life cycle of Trained Software. Initially software developer required to pass original source code to the builder program as input. Builder program rebuild the source code by placing some addition statement termed as placeholder statements in the original code. Only once it is required to pass the original code to the Builder Program, no matters how many copies we sell.

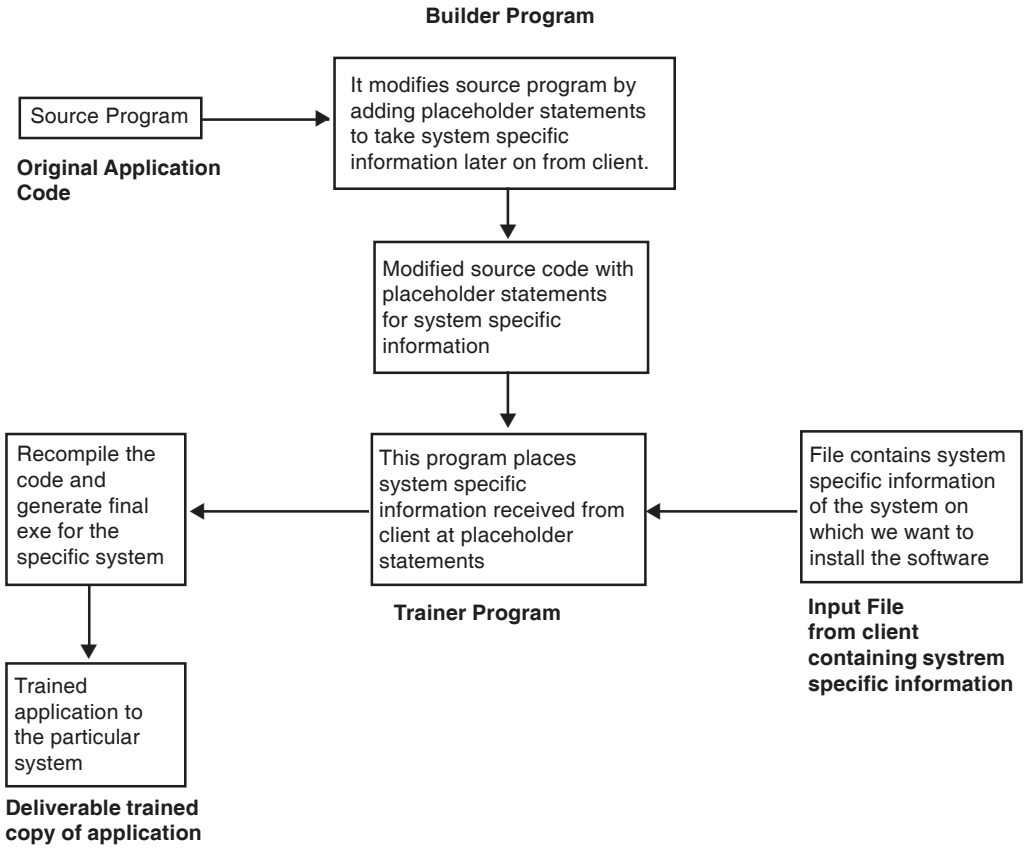

Figure 1: Life Cycle of the Trained Software

When ever client send request to buy/purchase the software, he/she is asked to provide system specific information so peculiar copy of original software can be generated for particular system of the client.

Input information received from client and output generated by Builder program then become input for the Trainer program. Trainer program places input information received from the client at the placeholders. Each time whenever client asks to buy software same step is repeated i.e. each time we have to train the software for the system of the client by using Trainer program.

Output produced by trainer program will be the Trained software for the specific system. Now recompile the trained application and make executable ready to install on that specific system.

After training of the software (output generated by Trainer program) recompilation is required. Then an executable file of it is generated. This is system specific executable file meant only for one particular system for which it is trained.

System parameters with weights according to the failure and their replacement history will be developed. So there is also scope if one or two parameter fails (according to their weights) even then application will work on that system. This is highly desirable in terms of efficiency, time and cost.

This model does not require any type of involvement from application developer. Applications developers are required to purchase said Package and pass their application to that software.

Problem with this approach is distribution of the application. Application developers can create necessary infrastructure if they want to prevent big financial loss caused by the piracy. Problem of distribution of application/software will be resolved very easily and effectively if one sells software through Internet.

CONCLUSION

In this paper, we have presented a state of art model to discourage the illegal copying of software using the concept of trained software. This approach is much more secure and cost effective than any other piracy preventing mechanism as it requires no additional hardware. Further as software is trained according to system specific information on which it runs. It is not possible to run software on any other system.

REFERENCES

[1] Wagner, N. R. "Fingerprinting", *Proceedings of the 1983 Symposium on Security and Privacy*, *IEEE Computer Society*, 18–22.

[2] J. Löfvenberg, Random Codes for Digital Fingerprinting, Linköping Studies in Science and Technology, Thesis No. 749, LIU-TEK-LIC-1999:07, ISBN 91-7219-430-8.

[3] J. Brassil, S. Low, N. Maxemchuk, L. O'Gorman, ''Electronic Marking and Identification Techniques to Discourage Document Copying,'' *Proceedings of IEEE INFOCOM'94*, Toronto, **3**, (June, 1994), 1278–1287.

[4] J. Löfvenberg, T. Lindkvist, "A General Description of Pirate Strategies in a Binary Fingerprinting System", Report LiTH-ISY-R-2259, ISSN 1400-3902.

[5] Bruce Schneir, Applied *Cryptography, Second Edition,* John Wiley & Sons, 2002

[6] L.Qian and K. Nahrstedt, "Watermarking Schemes and Protocols for Protecting Rightful Ownership and Customer Rights", *Journal of Visual Communication and Image Representation*, **9**, (1998), 194–210.

[7] R. Gopal, A Gupta, "Trading Higher Software Piracy for Higher Profits: The Case of Phantom Piracy" *IEEE International Conference on System Sciences*, (Jan 2002), 2491–2498.

[8] M. Limayem, M. Khalifa, W. W. Chin, "Factors Motivating Software Piracy: A Longitudinal Study", *IEEE Transactions on Engineering Management*, (Nov. 2004), 414–425.

[9] Tim Maude, Derwent Maude, "Hardware Protection against Software Piracy", *Communications of ACM*, **27**, (9), (Sep 1984), 950–959.

Parvinder Singh
Reader & Chairman Computer Science & Engineering

Darshana Hooda
Lecturer Computer Science & Engineering
DCR University of Sc. & Tech.
Murthal, Sonepat