Software Protection by Combining Hash Function with Hardware Identifications

Dr. Ayad A. AbdulSalam
Falah Mahdi Abduallh
Fatin S. Kazaal

المستخلص :

تعرض هذه الورقة البحثية طريقة مهجنة لحماية البرمجيات من الاستنساخ، حيث يمكن تطبيقها لمنع الاستنساخ غير المشروع من خلال توليد مفتاح تشغيل مرخص، وهذا المفتاح وحيد وسهل التوليد. استثمر هذا العمل التعريف الوحيد للقرص الصلب والذي يمكن استرجاعه بواسطة البرمجيات لتوليد مفتاح التشغيل بعد معاملته بدالة Hash ذات الاتجاه الواحد.

تم تقييم الطريقة المقترحة بواسطة مقياس التعقيد، ووقت التنفيذ، حيث بامكان دالة Hash ان تضمن تعقيد عال لمنع المتطفلين من ايجاد نسخ غير مرخصة، وتم تنفيذ عدة تجارب باستخدام تطبيقات ذوات احجام مختلفة لقياس وقت التنفيذ، وقد بينت هذه المقاييس التعقيد العالي ووقت التنفيذ القصير للطريقة المقترحة.

Abstract:

This paper presents a hybrid software copy protection scheme, the scheme is applied to prevent illegal copying of software by produce a license key which is unique and easy to generate. This work employs the uniqueness of identification of hard disk in personal computer which can get by software to create a license key after treated with SHA-1 one way hash function. Two mean measures are used to evaluate the proposed method, complexity and processing time, SHA-1 can insure the high complexity to deny the hackers for produce unauthorized copies, many experiments have been executed

using different sizes of software to calculate the consuming time. The measures show high complexity and short execution time for proposed combining method.

Keywords

Software protection, copy protection, copy prevention, copy control, digital right management, hash function, protection research, one way functions, SHA_1, complexity, brute force attack, verification function, license key.

1. Introduction:

Software piracy has troubled the computer industry, producing millions of dollars of losses, and raising numerous scientific and technical problems of interest in computer security. Software is hardly sold, but it is typically licensed according to policies defined by software license owners. Licensed software usually executed within the licensed customers' computers and expected to be run according to license policy. For example, the license may establish that only users from an authorized IP address can use it, or that it can only run on a specific computer, or establishes an expiration date. However, the license owner does not have any technical warranties to enforce his policy, unless he uses a secure software protection system [1].

Literature have always used the term copy protection, but others favor restriction technologies methods and they think Copy prevention and copy control may be more neutral terms. "Copy protection" is a misnomer for some systems, because any number of copies can be made from an original and all of these copies will work, but only in one computer, or only with one dongle, or only with another device

that cannot be easily copied. The term is also often related with the concept of digital rights management. Digital rights management is a more general term because it includes all sorts of management of works, including copy restrictions. Copy protection may include measures that are not digital. A more likely description to this is "technical protection measures" (TPM), which is often defined as the use of technological tools in order to restrict the use and access to a work [1].

2. Methods and Techniques:

There are several techniques used in software protection methods by exploit hardware and software, Self Modifying Code (SMC) is a well known technique which changes the software in real-time such that the dynamic code is different from the static code, and hence provides an effective way to defeat static disassembler . However, if a monitoring program identifies the target address of the SMC codes and replaces the bytes in the target addresses with specific bytes, it will produce a corrected static code which is identical to dynamic code [2]. Another novel of software protection in which program analysis by a malicious user (attacker) is made difficult by camouflaging (hiding) a large number of instructions contained in the program. In an arbitrary instruction (target) in the program is camouflaged by a different instruction. Using the self-modification mechanism in the program, the original instruction is restored only in a certain period during execution [3].

Other software copy protection techniques include:

- A dongle, a piece of hardware containing an electronic serial number that must be plugged into the computer to run the software. This adds extra cost for the software publisher.
- Bus encryption and encrypted code for use in secure crypto processors. This prevents copying and tampering of programs used in high security environments such as Automatic Trailer Machines (ATMs).
- A registration key, a series of letters and numbers that is asked for when running the program. Many computer games use registration keys. The software will refuse to run if the registration key is not typed in correctly, and multiplayer games will refuse to run if another user is online who has used the same registration key.
- Name and Serial number that is given to the user at the time the software is purchased, and is required to install it.
- Key file, which requires the user to have a key file in the same directory as the program is installed to run it [3].

These schemes have all been criticized for causing problems for validly licensed users who upgrade to a new machine, or have to reinstall the software after reinitializing their hard disk. Some Internet product activation products can allow replacement copies to be issued to registered users or multiple copies to the same license.

Like all software, copy-protection software sometimes contains bugs, whose effect may be to deny access to validly licensed users. Most copy protection schemes are easy to crack, and the resulting cracked software is then more valuable than the uncracked version, because users can make additional copies [4].

Electronic Software Distribution (ESD) using an intelligent software protection by a CRYPTO-BOX key which is the great software protection solution in particular for modularly designed products where each customer obtains only those components of the application that they need [5].

3. One-way Functions

A one-way function is a mathematical function that is significantly easier to compute in one direction (the forward direction), but the opposite direction (the inverse direction) it might be very difficult, for example, to compute the function in the forward direction in seconds but to compute its inverse could take months or years, if at all possible [6]. That is, given x it is easy to compute f(x), but given f(x) it is hard to compute x. In this context, "hard" is defined as something like: It would take years to compute x from f(x) even if all the computers in the world were assigned to solve the problem [7]. The notion of a one-way function is central to public-key cryptography. All public key algorithms are based on one way function, which depend on (hard to reverse) fact [8].

Example:-

$2^x \bmod 127 \equiv 31$

x=?

3.1. One way Hash functions

It is a function mapping or translating one sequence of bits into another, generally smaller and it is set the hash result. Hash function maps an input of arbitrary finite bit length, to an output of fixed bit length. An input sequence (message) yields the same hash result every time the algorithm is executed using the same message as input which

is known as pre-image. Given a hash function and an input, then output easy to compute. It is computationally infeasible that a message can be derived or reconstituted from the hash result produced by the algorithm, at the same time it is computationally infeasible that two messages can be found that produce the same hash result using the algorithm. The output of a hash function is sometimes called a "message digest", this is commonly referred to as finding a collision for the hash function [9].

There are several types of hash functions such as MD4, MD5 and Secure hash algorithm SHA-1, SHA-256, SHA-384, and SHA-512 [10].

Secure hash algorithm SHA-1, SHA-256, SHA-384, SHA-512

Hash algorithm can be described in two stages: preprocessing and hash computation.

Preprocessing involves padding a message, parsing the padded message into *m*-bit blocks, and setting initialization values to be used in the hash computation.

The hash computation generates a **message schedule** from the padded message and uses that schedule, along with functions, constants, and word operations to iteratively generate a series of hash values. The final hash value generated by the hash computation is used to determine the message digest. [11]

The four algorithms differ most significantly in the number of bits of security that are provided for the data being hashed – this is directly related to the message digest length. When a secure hash algorithm is used in conjunction with another algorithm, there may be requirements specified elsewhere that require the use of a secure hash algorithm with a certain number of bits of security.

Additionally, the four algorithms differ in terms of the size of the blocks and words of data that are used during hashing [10]. Table (1) presents the basic properties of all four secure hash algorithms.

Table (1) Secure Hash Algorithm Properties

Algorithm	Message Size (bits)	Block Size (bits)	Word Size (bits)	Message Digest Size (bits)
SHA-1	$<2^{64}$	512	32	160
SHA-256	$<2^{64}$	512	32	256
SHA-384	$<2^{128}$	1024	64	384
SHA-512	$<2^{128}$	1024	64	512

4. Proposed Protection System

The proposed system uses one way SHA_1 algorithm and identification (ID) of hard disk to generate a unique key for each computer, ties the installed software to a specific machine by involving some unique feature of the machine. Some machines have a serial number in ROM, while others do not, and so some other metric, such as the date and time (to the second) of initialization of the hard disk can be used. On machines with Ethernet cards, the MAC address, which is unique and factory-assigned, is a popular surrogate for a machine serial number; however, this address is programmable on modern cards. We work to exploit the (ID) of hard disk which can get electronically as an input to SHA-1 hash function to produce a unique

key for specific computer, the key works as a license of software or application.

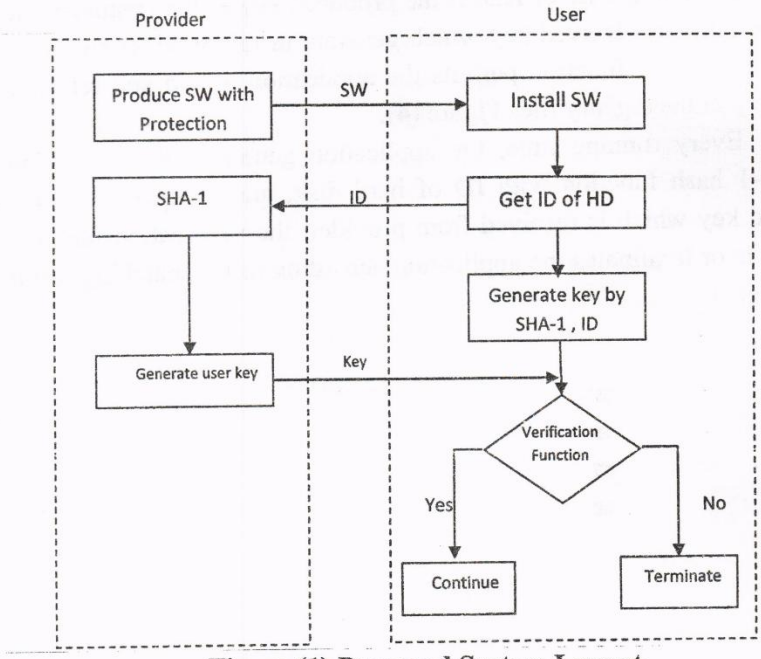
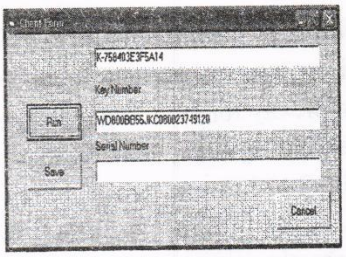


Figure (1) Proposed System Layout

In the first installing time of application in the computer, Figure (1) illustrates the layout of actions in both sides (provider, and user), the software gets the ID of the user hard disk, and ask the user to send gotten ID to provider by using any channel (phone, fax, message, e-mail, ...etc) Figure (2). When the provider receives the ID of the user HD, generates a key by running SHA-1 application using ID as an input, and sends a produced key to costumer Figure (3).

In the costumer's side, the user supplies a received key to SW, the protection procedure generates a key by using the same SHA-1 hash function and ID of HD. If the produced key in the costumer site is the same as received key which generate in the provider site, then the verification function permits the application to run and registers the key in the registry files Figure (4).

Every running time, the application generates the key using SHA-1 hash function with ID of hard disk, and compare with the stored key which is received from provider, the verification function permits or terminates the application according to the matching result Figure(5).



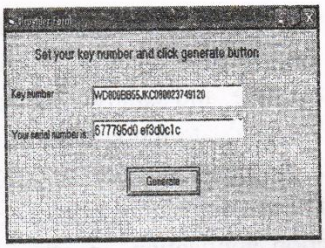
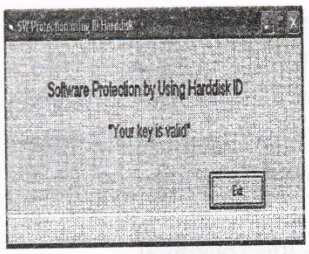


Figure (2) getting ID **Figure (3) License Key generation**



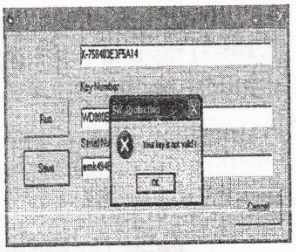


Figure (4) Verification Function (Pass) Figure (5) Verification Function (Fail)

The application continues to running while the media still is the same HD, when illegal copy the SW to other HD the two keys will differ, because the user application generates the new key for new HD, which is not match with the stored key, then the running will terminated.

5. Experimental results

Two measures are depended on to evaluate the proposed method, time and complexity, the protections have to execute in as short as possible time, and satisfy a big complexity for avoiding the intrusion.

We evaluate the system by executing many experiments on PC with 2.39 GHz Processor and 1.GB RAM to compute the consuming time for execution in millisecond. Many file sizes are tested without protection application, in other case the same files are tested after combining with proposed protection system. The time differences between two cases of all experiments were very small, ranging between (**0.1408**) ms and (**0.2968**) ms in worst case.

Figure (6) shows the increasing of size will increase the execution time in linear manner, that means the proposed system does not work as a main factor of increasing processing time when combining with any application.

Table (2) Results of experiments

Size(KB)	T-with protection (ms)	T- without protection (ms)	T- difference (ms)
24	0.235000000000582	0.00000000000326	0.234999999997322
32	0.281999999997904	0.00000000000326	0.281999999994644
36	0.203874999999243	0.06299999999924	0.140875000000003
40	0.296999999998661	0.04700000000058	0.249999999998081
40	0.26600000000326	0.00000000000326	0.266
44	0.390625	0.09375	0.296875

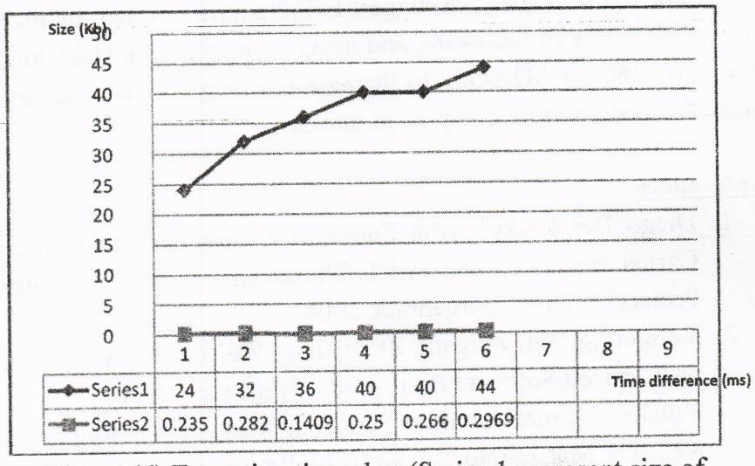


Figure (6) Execution time char (Series 1 represent size of program, series2 represent time deference)

While the SHA-1 hash function generates 16 character key, using the range of 62 different characters (A…Z, a…z, 0…9), that means the complexity of finding the key by brute force attack method will be $(62)^{16}$ probable, it is from the exponential complexity with order of $O(N)^m$, when N and m are variables.

6. Conclusions and Future Work

According to experimental results that reached in early section the following remarks were derived:

1. The complexity of braking system is so high, because the using of one way SHA_1 to generate 16 character key takes several years to break.
2. System takes short execution time and little memory space.

For future works, we suggest to build a new system by merging two or more IDs of hardware, and using more complex Hash function like SHA-256, or SHA-512 to increase the length of license key and make the system more complex to intrusion.

References

1. Diego Bendersky, Ariel Futoransky, Luciano Notarfrancesco, Carlos Sarraute, and Ariel Waissbein, "Advanced Software Protection Now", Argentina, 2004.
2. Yongdong Wu, Zhigang Zhao, Tian Wei Chui," An Attack on SMC-based Software Protection", Singapore, 2006.
3. Yuichiro Kanzaki, Akito Monden, Masahide Nakamura, and Ken'ichi Matsumoto, "A Software Protection Method Based on Instruction Camouflage",Japan, 2006.
4. Solveig Singleton," Copy Protection and Games: Lessons for DRM Debates and Development ", The progress & Fredom Foundation, 14.2 February 2007
5. Philipp Marx," Software Protection System (Crypto-Box)",Marx software security, March 1, 2004.
6. RSA Laboratories," Crypto FAQ, Simple Application of cryptography", 2007, viewed on January 12, 2009 at WWW:http://www.rsa.com/rsalabs/nod.asp
7. ALAN G. KONHEIM," Computer Security and Cryptography" , John Wiley & Sons, Inc,2007.
8. Bruce Schneier ,"Applied Cryptography", 1995.
9. Wikipedia, "Cryptographic hash function", viewed on February 10,2008atWWW:http://www.en.wikipedia.org/wiki/cryptographic_hash_function_74k

10. Nick Carruthers,"Digital Signature Schemes", 1997, viewed on February12,2009, at www.unc.edu/^carrot/Nick_carruthers_thesis.pdf

11. Brain Gladman, Carl Ellison and Nicholas Bham, "Digital Signature, Certificates and electronic Commerce", 1999, viewed on May 05, 2009 at www:http//www.jya.com /bg/ digsig.pdf .