Server Side Algorithms for WHLK (Watermark, Hardware Parameters and License Key) framework

Nishant Gupta, Shubhnandan S. Jamwal

Software piracy is the most significant and burning issue in the age of the internet. Software piracy has been a direct threat for software vendors in terms of revenue and, therefore, a number of effective and efficient techniques have been employed for the detection and prevention of software piracy. One very important technique is software watermarking and using registration keys. This paper proposes the Server-side algorithms for registration and embedding the watermark into the software using the WHLK (Watermark, Hardware Parameters and License Key) approach. We have tested the algorithms and the analysis of the proposed algorithms proves that these registration algorithms are more reliable and efficient in comparison to other techniques.

SERVER-SIDE ALGORITHMS FOR WHLK FRAMEWORK

Nishant Gupta[1], Shubhnandan S. Jamwal[2]

[1]Research Scholar, Brampton, Ontario, Canada
[2]Department of Computer Science & IT, University of Jammu, Jammu, J&K, India

{[1]nis.decent, [2]jamwalsnj}@gmail.com

ABSTRACT. Software piracy is the most significant and burning issue in the age of the internet. Software piracy has been a direct threat for software vendors in terms of revenue and, therefore, a number of effective and efficient techniques were employed for detection and prevention of software piracy. One of very important technique is software watermarking and using registration keys. This paper proposes the Server-side algorithms for registration and embedding the watermark into the software. We have tested the algorithms and the analysis of the proposed algorithms proves that these registration algorithms are more reliable and efficient in comparison to other techniques.

Keywords: Software watermarking, algorithms, software piracy, server-side registration.

1 INTRODUCTION

In today's global marketplace, the software industry has made progress by leaps and bounds. The fast evolution of software industry not only leads neophytes to new opportunities, but also posing threats to software vendors. The curse of software piracy hampers software industry in software development process. Initially the main focus of software companies is to develop new and intuitive software. Afterwards, the time, money and resources are being spent to protect their intellectual property through proper researching and developing techniques. It has become important for software companies to know what can be done to stop the average user from attempting to use the pirated software due to the increasing amount of attention directed towards prevention of software piracy.

According to Business Software Alliance (BSA), the amount of financial loss caused by software piracy in 2011 was more than 63,456 million U.S. dollars [1]. In addition, the rate of loss is growing day by day with the expansion of market size. BSA estimated that the worldwide software piracy rate went up to 42 percent in 2011. It was found that total losses due to software piracy in India stood at a staggering figure of about INR 13,783 crores in 2011 showing about 63 per cent piracy rate in India [2]. Thus, many studies have been promoted to protect software from being

pirated. These approaches which have emerged are hardware-based and software-based. Some of the common software protection measures are:

Intellectual property protection: The main objective in this approach is to link the software with information of author by using watermarking techniques [3].

Protection against function analysis: In this approach, the objective is to prevent a malicious host from discovering the function which is computed by a software element. Code obfuscation and function hiding are used in this control measure which is complemented by the use of hardware tokens.

Software use-control: This control measure is aimed at guaranteeing only the authorized users to the software based on some contractual conditions.

Our paper implemented new algorithms comprising of integration of software watermarking and use of registration code.

2 LITERATURE REVIEW

In the past few years, numerous studies have been conducted related to global software piracy.

In 2005, William Zhu et. al. [3] gives a brief overview of software watermarking. In this research, the taxonomy of software watermarking and its algorithms were discussed.

Qiang Liu [4] in his paper focused on some special protective approaches using some exterior components (such as smart cards, hardware security unit and dongles) to make more difficult for unauthorized use of software. The paper also compared these approaches and offer some insight into the different approaches adopted. But it needed to bind the software with exterior component (hardware). The cost of exterior component has to be added to the cost of software which seems to be unreasonable.

Yawei Zhang et.al [5] proposed a software-splitting technique which put the split contents to the client instead of the remote trust server. In this technique, the extracted contents of the software were encrypted using a key relating to the hardware characteristics. These encrypted contents were decrypted dynamically during the main program running. Even though it is a useful technique, it is not stimulating for Visual Basic and JAVA languages as it is almost impossible to directly manipulate in the memory.

Petar Djekic and Claudia Loebbecke [6] in their paper studied the influence of technical copy protections on application software piracy. Empirical investigation has been done through scientific research as to what extent the technical copy protections avoids illegal copying. These research findings cannot confirm to be also applicable to graphics application software.

Vineet Kumar Sharma et. al. [7] identifies that there were a number of flaws in the existing defense mechanisms. The research examined the social and technical challenges associated with prevention of software piracy. In this paper, the currently used static licensing structure in software distribution was characterized with its drawbacks. Hence a dynamic software licensing scheme was proposed comprising of software serial keys integrated with hardware token to provide an ethical optimal utilization of software to the customer organization resulting in stopping software piracy.

Ibrahim Kamel, Qutaiba Albluwi [8] proposed a technique in which the watermarks were hidden in various data structures used by the code, e.g., R-trees, Bþ-trees and linked lists to protect software copyright. In this paper, a detailed security analysis and performance evaluation has been done to show that these embedded watermarks are robust and can withstand various types of attacks. This approach can also be applied to other data structures which do not put restrictions on the order of the data; however, it is not suitable for data structures that are sensitive to the order of the data like Bþ-trees, Quad-trees, k-d tree, etc. in addition to memory-resident data structures like stacks, linked lists, arrays, binary trees, etc.

Aniket Kulkarni, Sachin Lodha [9] examined software protection through code obfuscation technique which resists reverse engineering attacks. Although the scheme requires high development efforts to construct obfuscated code, it can be useful to obfuscate code which implements secret functionality such as license checking mechanism.

Numbers of strategies have been employed to make it more difficult for unauthorized user to use and duplicate the software. Ireneusz J. Jozwiak and Krzysztof Marczak [10] introduced such a strategy to provide a hardware key which is typically installed in the parallel port or USB on the computer to provide a software interlock. The software will not execute if the key is not in place. This technique is relatively expensive and cumbersome for the developer and authorized user respectively, while remaining vulnerable to the theft conducting piracy by duplication of the hardware key. Another technique requires the user to enter a customer identification number or serial number during the process of software installation. This will prevent the software installation if the registration information is missing or invalid. But this approach is easily defeated by transferring the serial number or customer identification number to one or more unauthorized user. Another approach under the study registers the software with the manufacturer or distributor to obtain an operational code or password which is necessary for software installation. Once the operational code or password is obtained, it may be perpetually transferred along with pirated copies to numerous users which were unauthorized.

Ajay Nehra et. al. [11] proposed a SMS gateway based technique which checks the authenticity of the software at every fixed time interval. It will be advantageous to identify fake unauthorized users. Software companies were saved from huge loss by blocking such installed software. As a manual response for each software on the client machine will be a tedious task, there is a need to find an automation process.

3 OBSERVATION

WHLK Model [12] is developed and implemented on the basis of analysis of findings from a survey conducted on a number of stakeholders involved in academic as well as industrial sector. The results of this exploratory study clearly show that users are well aware of software piracy. Even though the pirated software does not fulfill requirements of the user, but the involvement in software piracy is highly influential. A very high percentage of users is acquiring software through their respective vendors who

load the pirated copies of software in their machines. This hard disk loading type of piracy shows the highest percentage among all types of piracy. Even the core factors like risk, market, morality and education are not able to affect the software pirates.

The findings clearly show that most of the software is acquired through piracy instead of purchasing them. The queries generated in the questionnaire related to software section resulted in a very high percentage of software acquiring through different types of software piracy. Most of the software under different categories, i.e. operating system, computer languages, office, databases, utilities, and educational are acquired by the users through one type of piracy or the other. Results show that 98% of software is pirated which give a different picture of piracy rate in India as compared to BSA reports of 63% piracy rates in 2011.

In contrast to other techniques, our paper shows the implementation of the server-side algorithms during the purchase and registration of software.

4 WHLK MODEL

The implementation and testing of WHLK Model [13] have been proposed to reduce unauthorized use of software. This model introduced an integration of Software Watermarking, Hardware Parameters and License Key. We incorporated the new processes in the software registration process after noting the weaknesses in existing approaches of software registration. Our approach is to force the software to be used only in an authorized running environment instead of preventing it from illicit copying. To accomplish this task, the user is required to submit his identification details. These details applied through algorithms, thereby generating a random unique key (string). This key is embedded as a watermark in the software program. Then an automated process fetches the serial numbers of CPU, the main board, processor and other hardware information. These hardware characteristics and License key of software are integrated to be used as parameters. An encryption algorithm is applied to these parameters. It resulted in a random unique number called as Registration Code (RGCN) which is being acknowledged to the software user. WHLK model provides an integration of the privacy of users, security of information and license key to control software piracy. As this model enjoys a modular design, it can be implemented by any machine with flexible configurations and windows X operating systems. In addition to these features of model, it also allows flexible registration information definition. WHLK Model prevents illegal uses on software copy and also makes it harder to create an additional available copy based on diversity.

5 SERVER SIDE REGISTRATION ALGORITHMS

(a) SOFTWARE WATERMARKING ALGORITHM

Software Watermarking Algorithm 1.0 is implemented on the vendor's side when a user purchases the software. The personal information of user viz. name, affiliation and identification number are input as parameters for generating a unique key. This

unique key is used to embed a watermark in the code of software. CT (Collberg Thomborson) algorithm is applied on this unique key and embeds the watermark in the class file. After successfully applying this algorithm, another algorithm, namely StringEncodeObfuscator is applied to avoid reverse engineering. The watermark embedded in the class file is secure and cannot be altered or tampered due to obfuscated file.

```
Algorithm 1.0: (Software Watermark Algorithm) Input
               the User name N, affiliation A and
               identification number I as string vari-
               ables, this algorithm generates a
               unique key UK and embed watermark in
               the software using UK.

Step 1. Read: N, A, I from the user.

Step 2. Generate UK.

Step 3. Embed watermark using CT algorithm.

Step 4. String Encode Obfuscator algorithm is applied.

Step 5. Exit.
```

(b) NEW REGISTRATION ALGORITHM

Registration Algorithm is designed for user to register the software first time on a new machine. While registering the software on computer machine, the New Registration Algorithm 1.1 fetches the hardware characteristics of the client machine. These characteristics include the harddisk ID, CPU ID, Processor Serial Number and MAC ID. These values are concatenated and stored in a string. This string variable is verified by the server in the database. If this variable exists in the database, Algorithm 1.4 is applied else RNGCryptoServiceProvider is applied for generating unique key. A unique key is generated by the server. Then the user inputs his E-Mail ID and License key which are being submitted to the server. This license key is concatenated with a unique key and the result is stored in another string variable. The concatenated string is the registration code. Then the system date has been generated. The registration code is stored in the database and also acknowledged to the e-mail id of the user. Lastly, the database has been updated and new registration process completes its cycle.

```
Algorithm 1.1: (NewRegistration Algorithm) Fetch the CPU ID
               (CID), Harddisk ID (HDID), MAC ID (MID) and
               Processor Serial Number (PSN) of the machine,
               this algorithm generates the unique key UK and
               concatenates it with License Key (LK) to update
               registration code database RGCNDB and send to
               client.

Step 1. Read CID, HDID, MID, PSN from the machine.

Step 2. HWD : = Concatenate (CID,HDID,MID,PSN).

Step 3. Submit HWD to server for verification.

Step 4. If HWD exists, then:
          Apply Re-registration algorithm 1.4.
          Go To Step 5.
            Else:
              RNGCrptoServiceProvider is applied.
              Server generates UK.
              Input EID and LK.
              RGCN : = Concatenate (UK,LK).
              Read: SysDate : = DateTime().
              Update RGCNDB.
        ENDIF.

Step 5.   Exit.
```

(c) RE-REGISTRATION ALGORITHM IF HARDWARE CHANGES

Another algorithm is designed to manage the changes in the hardware of the machine of the user. If the user has changed the hardware of the machine, he needs to re-register the software and in this case Re-registration Algorithm 1.2 is applied. Three parameters, user name, affiliation and ID are concatenated and stored in a string variable. This string variable is verified by the server in the database. If the variable does not exist, NewRegistration algorithm 1.1 is applied otherwise an algorithm is applied, which fetches the hardware characteristics of the client machine. These characteristics include the harddisk ID, CPU ID, Processor Serial Number and MAC ID. These values are concatenated and stored in a string variable. This string variable is verified by the server in the database. If this variable exists in the specified field of the database, Algorithm 1.4 is applied else RNGCryptoServiceProvider is applied for generating unique key. A unique key is generated by the server. Then the user inputs his E-Mail ID and License key which are being submitted to the server. This license key is con-

catenated with a unique key and the result is stored in another string variable. The concatenated string is the registration code. Then the system date has been generated. The registration code is stored in the database and also acknowledged to the e-mail id of the user. Lastly, the database has been updated and re-registration process completes its cycle.

```
Algorithm 1.2: (Re-registration Algorithm if machine hardware
               changes)
             Input the User name N, affiliation A and identifica-
             tion number I as string variables, this algorithm
             fetches the CPU ID (CID), Harddisk ID (HDID), MAC ID
             (MID) and Processor Serial Number (PSN) of the machine
             generating the unique key UK and concatenates it with
             License Key (LK) to update registration code database
             RGCNDB and send to the client.

Step 1.   Read N, A, I from the user.

Step 2.   Set X : = Concatenate (N,A,I).

Step 3.   Submit X to server for verification.

Step 4.   If X do not exists, then:
              Apply NewRegistration algorithm 1.1.
              Go To Step 5.
            Else:
                Read CID, HDID, MID, PSN from the machine.
                HWD= Concatenate (CID,HDID,MID,PSN).
                Submit HWD to server for verification.
              If HWD exists, then:
                  Go To Step 5.
                Else:
                  RNGCrptoServiceProvider is applied on HWD.
                  Server generates UK.
                  Input EID and LK.
                  RGCN : = Concatenate (UK,LK).
                  Read SysDate : = DateTime().
                  Update RGCNDB.
              ENDIF.
            ENDIF.

Step 5.     Exit.
```

(d) RE-REGISTRATION ALGORITHM FOR SAME MACHINE IF NO DETAILS AVAILABLE

This algorithm is designed to manage the re-registration process of software on the same machine in case user does not have any details. If the user does not have any of details required and needs to re-register the software on same machine again, Re-registration Algorithm 1.3 is applied. The three parameters, user name, affiliation and ID are concatenated and stored in a string variable. This string variable is verified by the server in the database. If the variable does not exist, NewRegistration algorithm 1.1 is applied else, this algorithm fetches the hardware characteristics of the client machine. These characteristics include the hard disk ID, CPU ID, Processor Serial Number and MAC ID. These values are concatenated and stored in a string variable. This string variable is verified by the server in the database. If this variable exists in the specified field of database, Algorithm 1.4 is applied else RNGCryptoServiceProvider is applied for generating unique key. A unique key is generated by the server. Then the user inputs his E-Mail ID and License key which are being submitted to the server. This license key is concatenated with unique key and the result is stored in another string variable. The concatenated string is the registration code. Then the system date has been generated. The registration code is stored in the database and also acknowledged to the e-mail id of the user. Lastly, the database has been updated and re-registration process completes its cycle.

(e) RE-REGISTRATION ALGORITHM FOR SAME MACHINE IF DETAILS AVAILABLE

This algorithm is designed to manage the re-registration process of software on the same machine if details are available to the user. In case of availability of registration code with the user while re-registering the software, Re-registration algorithm 1.4 is applied. The user inputs the registration code and server verifies the code for its existence. If the code does not exist Re-registration Algorithm 1.3 is applied else a System Date is generated. Database is updated and re-registration process completes its cycle.

(f) TIME-FRAME ALGORITHM

The time-frame algorithm is designed to block the software on the basis of the time span to secure the software from being used elsewhere. The time-frame algorithm 1.5 is applied for blocking the software if time span between the system date of registration SysDate and current system date DBDate is more than 30. This is done by calculating the time span between SysDate and DBDate. If the time span is showing the positive value less than 30, it means that the system date is correct and the user has still days to update the software. In case, the time span is showing the negative value, meaning that the system date is incorrect and needs to be corrected. If it is correct, user can update the software, but if the time span is still showing the negative value, the process generated kills the process itself after some stipulated time meaning that software terminates itself. Software does not run, but will be blocked till the user re-registers again with the correct details.

```
Algorithm 1.3: (Re-registration Algorithm for same ma-
               chine in case user does not have de-
               tails)
               Input the User name N, affiliation A and
               identification number I as string varia-
               bles, this algorithm fetches the CPU ID
               (CID), Harddisk ID (HDID), MAC ID (MID)
               and Processor Serial Number (PSN) of the
               machine generating the unique key UK and
               concatenates it with License Key (LK) to
               update registration code database RGCNDB
               and send to client.

Step 1. Read: N, A, I from the user.

Step 2.     Set X : = Concatenate (N,A,I).

Step 3. Submit X to server for verification.

Step 4. If X do not exists, then:
          Apply NewRegistration algorithm 1.1.
          Go To Step 5.
            Else:
            Read: CID, HDID, MID, PSN from the machine.
            HWD= Concatenate (CID,HDID,MID,PSN).
            Submit HWD to server for verification.
            If HWD do not exists, then:
              Apply algorithm 1.4.
              Go To Step 5.
              Else:
                RNGCrptoServiceProvider is applied.
                Server generates UK.
                Input EID and LK.
                RGCN : = Concatenate (UK,LK).
                Read SysDate : = DateTime().
                Update RGCNDB in the database.
            ENDIF.
        ENDIF.

Step 5.   Exit.
```

```
Algorithm 1.4: (Re-registration Algorithm for same ma-
               chine in case user has details)
               This algorithm inputs the registration
               code RGCN and re-register the software
               till next update.

Step 1. Input RGCN.

Step 2. Submit RGCN to server for verification.

Step 3. If RGCN do not exists, Then:
          Apply Algorithm 1.3
          Go To Step 4.
          Else:
            Read: SysDate : = DateTime().
            Update RGCNDB.
        ENDIF.

Step 4. Exit.
```

```
Algorithm 1.5: (Time-frame Algorithm)
              This algorithm reads current System Date
              DBDate  and finds the re-verification time
              by finding difference DE between SysDate
              and DBDate.

Step 1. Read: SysDate from RGCNDB.

Step 2. Read: DBDate=DateTime().

Step 3. TimeSpan ts= DBDate - SysDate.

Step 4. Int DE = ts.Days.

Step 5. String dt = DE.ToString().

Step 6. If dt.StartsWith("-"), Then:
            Process.kill().
            Go To Step 7.
            Else
              Submit update.
              DE=DE + 30.
        ENDIF.

Step 5.     Exit.
```

6 CONCLUSION

WHLK Model has been implemented on machines with variant configurations. Client-side and server-side phases of this model have been put to test on these machines and results observed are justifying the objectives of our research. WHLK Model prevents illegal uses on software copy and also makes it harder to create an additional available copy based on diversity.

The Server-side verification algorithms were designed and implemented on various machines. It was observed that the newly designed algorithm at server side woks accurately and the chances of pirating the software copy are reduced to nil.

REFERENCES

1. Business Software Alliance, 2012. Ninth Annual BSA Global Survey of PC User Attitudes. [Online]. Available: (http://globalstudy.bsa.org/2011/downloads/study_pdf/2011_BSA_Piracy_Study-Standard.pdf)

2. Business Software Alliance, 2012. Ninth Annual BSA Global Survey of PC User Attitudes. [Online]. Available: (http://globalstudy.bsa.org/2011/downloads/study_pdf/pr_india_en.pdf)

3. Zhu, W., Thomborson, C., Wang, F.: A Survey of Software Watermarking. In Proceedings of IEEE International Conference on Intelligence and Security Informatics. LNCS 3495, pp. 454-458. Springer-Verlag Berlin Heidelberg, Germany (2005)

4. Liu, Q.: Techniques Using Exterior Component against Software Piracy. Technical Report, Department of Computer Science, University of Auckland, New Zealand

5. Zhang, Y., Jin, L., Ye, X., Chen, D.: Software Piracy Prevention: Splitting on Client. In Proceedings of International Conference on Security Technology. pp. 62-65. IEEE Computer Society, Washington, DC, USA (2008)

6. Djekic, P., Loebbecke, C.: Preventing application software piracy: An empirical investigation of technical copy protections. Journal of Strategic Information Systems, Elsevier. 16, 173-186 (2007)

7. Sharma, V.K., Rizvi, S.A.M., Hussain, S.Z., Chauhan, A.S.: Dynamic Software License Key Management Using Smart Cards. In Proceedings of International Conference on Advances in Computer Engineering. pp. 244-246, IEEE Computer Society (2010)

8. Kamel, I., Albluwi, Q.: A robust software watermarking for copyright protection. Computers and Security, Elsevier. 28, 6, 395-409 (2009)

9. Kulkarni, A., Lodha, S.: Software Protection through Code Obfuscation. Project Report, Tata Research Development and Design Centre, Pune (2012)

10. Jozwiak, I.J., Marczak, K.: A Hardware-Based Software Protection Systems – Analysis of Security Dongles with Time Meters. In Proceedings of Second International Conference on Dependability of Computer Systems. pp. 254-261, IEEE Computer Society Washington, DC USA (2007)

11. Nehra, A., Meena, R., Sohu, D., Rishi, O.P.: A Robust Approach to Prevent Software Piracy, In Proceedings of Students Conference on Engineering and Systems. pp.1-3, IEEE (March 2012)

12. Gupta, N., Jamwal, S.S., Padha, D.: Watermark, Hardware Parameters and License Key: An Integrated Approach of Software Protection. International Journal of Advanced Research in Computer Science and Software Engineering, 3, 5, 1285-1289 (May 2013)

13. Gupta, N., Jamwal, S.S., Padha, D.: WHLK: Framework for Software Authentication and Protection. IEEE African Journal of Computing & ICT, IEEE, Nigeria. 7, 1, 69-80 (2014)