AN ADAPTIVE TECHNIQUE FOR SOFTWARE LICENSING USING SMART CARD AND CRYPTO SYSTEM

G.Syam Prasad\textsuperscript{1}, Dr. G.Samuel Varaprasad Raju\textsuperscript{2}

\textsuperscript{1}Associate Professor in CSE Dept, Srinivasa Institute of Technology, Vijayawada, India, gudapati_sp@yahoo.co.in
\textsuperscript{2}Professor in School of Distance Education, AU, Vizag, India.

Abstract

Introduction of new software technologies also increased a great demand for the efficient software digital right management protection to prevent unauthorized copying and illegal distribution, according to the analysts, Software publishers lost several billions of revenue due to copying of their products. Many defense mechanisms have been proposed but still software pirates have been are in continuous look out for breakage of the vital information by cracking the systems. This paper examines the existing defense mechanism and impact of economical and technical challenges associated with the software piracy controlling process. This paper proposes an adaptive technique for controlling of software licensing.

Index Terms: Smartcard, Software publishers, Optimal utilization of software, Software piracy.

1. INTRODUCTION

The problem of software piracy is of great concern to the software publisher and software distributors. After software is developed it is subjected to users with contracted rights for use but constrained with modifying, sharing or selling the software. Violation of the said constraints leads to formidable issue of software piracy. Distributed computing systems require effective techniques to minimize the piracy impact without degrading the system performance. Software birth marking is one of the effective techniques to minimize the software piracy with guaranteed system performance. It ascertains intrinsic properties in the software program to detect copy of the software [3]

With the invention of many new software technologies there has been an increase in software piracy too. The problem of software piracy is of great concern to the software publisher and software distributors. A new software prevention scheme always comes with its counterpart. Cracking a serial key of the software can be classified as a Turing computable problem of automata and all these types of problems are solvable and computable which helps in spreading piracy. This paper presents a scheme for legal distribution of software with dual security. The dual Security consists of optimal utilization of software and selection of software restriction from the software itself. In this we use couple of directions in applying efforts to stop software piracy concept. Hardware tokens (smart cards) are integrated along with the software token (software serial keys). On another hand the optimal utilization technique of the software in the customer organization is achieved. The optimal utilization [2, 4] of the software is an ethical way to prevent software piracy. The software piracy can be prevented in an organization by using the software in a best and most optimized way. Therefore, this paper presents a scheme involving management of license keys among the user machines needing the license key for executing the software dynamically. The isolation of software and its restrictions or instructions for execution also results in minimizing the software piracy. If the software execution details for example the serial key needed for execution of the software is
embedded in the software then there is a high probability for a pirate to crack the serial key and then use this patch for making copies of the software, thus illegally spreading the software. So this paper forwards a way in which the software is devoid of the mechanism or procedure to run the software. The license keys needed for execution of the software may be present at different location and it can be isolated hardware like smart cards [5] or other plugging.

2. RESPONSIBILITIES OF THE LOGICAL DISTRIBUTOR SERVER

In this model [3] an organization tries to keep the information about the specified software on a Logical distribution server (consider to be Tomcat) and the complete management of the dynamic distribution of the software license is to be done with coordination with Logical server. In this methodology the organization cannot use the software on the number of computers, exceeding the number of license purchased but this methodology provides an ethical way for optimal uses of the software in the network of an organization. The Logical server has a counter which keeps a check on number of license keys that are reserved at a particular instant of time to various users. Besides this the coordinator machine maintains a list of active clients which are executing the software and a separate list for the waiting clients. Subsequently when any other machine requires software to execute, it broadcast same message request packet dedicated for that specific port in the network and waits for its response. The coordinator listens the request messages of these clients and sends them back a response message indicating the presence of coordinator. In this process the coordinator gets the IP address of the client machines and the client machines gets the IP address of the coordinator machine and now here after they can communicate to each other by unicasting the message packets. In this stage the client will get an inactive copy of the software which needs a runtime license key to come in the active and working state. At this time the client sends the message packet to the coordinator demanding a dynamic license key. The coordinator keeps the status of the copy of the currently active software on the various client machines. Here two cases arise:-

Case 1: If the number of machines executing the software is less than the number of license purchased –in this case a message packet containing the encrypted license key is transferred back to the client. On receiving the license key electronically the client’s software application turns into active or working state. This information is updated in the list of currently active clients available with the coordinator. It also updates the counter and decreases it by one.

Case 2: If the number of machines executing the software is greater than the number of license purchased—in this case after the client has demanded the license key from the coordinator, a message is sent to the client by the coordinator which asks the client whether to wait or quit. If the client waits for the key then it is put into the waiting list. The concept of waiting list works on the principal of FIFO. Number of request messages can be reduced to achieve better efficiency by introducing the concept of hierarchical coordinators in the different sub networks of the customer.

3. PROPOSED TECHNIQUE

In our approach assume we take two entities Software warehouse and customer. Software warehouse provides the software and the associated Smart card for sale to the customer; here smartcard stores unique License keys [6] of the software which is made available. This on Logical platform binds the customer to smartcard via software product while installation process. Each smart card can be read using the card reader at the time of software installation [5]. As the software is installed on a machine it needs a key for its execution which is obtained from the smart card.

In this process validating has two things involved customer’s client on which installation is taking on and the logical validating server with in the software warehouse

4. METHODOLOGY

The computer machine can be modified in number of ways, we assumed that mother board number is un altered and most constant hardware piece associated with the machine. At the time of installation software product validation system collects information like motherboard number, we in our propose, When the card producer produces the card then at the time of manufacturing it embeds license keys, Card OS which handles the storage of license keys and a random number - ‘x’ on it. At the time of installation of software there occurs an execution of an application which generates a random number – ‘y’ this key concatenates with the motherboard number (MK_1) during the process of installation.
We assume that software warehouse provides the licenses vary from single to multiple installations on the client machines, and software warehouse server is responsible for the validation. For this purpose [1] Diffie Hellman Key Exchange algorithm is proposed. Each card bears unique x, p, g. At the time of installation of software at the computer, it generates y and store it in a read only file at the computer. System gets the value of p, g and x from smart card. Calculate \( K_1 = g^x \mod p \) and \( K = k^y \mod p \) on machine and store the value of k in smart card. At the time of execution: x, k, p, g are retrieved from smart card. Calculate \( K_2 = g^y \mod p \) and \( K' = k^2 \mod p \) on the machine. If \( K = K' \) then the list of license keys containing a pair (public and private key) is transferred from smart card to the coordinator machine else no transfer occurs.

5. THE LOGICAL SERVER MACHINE

Now the coordinator machine say ‘B’ contains a list of public and private keys needed by the clients for the execution of their software. A client say ‘A’ wishes to establish a logical connection with B and requires a onetime session key to protect the data (license keys) transmitted over connection method for such establishment of logical connection for secure transmission of license keys using a third party as a Key Distribution Centre (KDC)

Client (A) and coordinator (B) both shares their private keys KA and KB with KDC. The following steps show the functioning of KDC.

1) Request is sent to KDC along with a nonce the request message has identity of A and B.

2) KDC sends a message to ‘A’ encrypted with private key Ka (so that only ‘A’ can read it). KS Session key EKb (KS, IDA) is sent to B as it is to establish connection and prove A’s identity.

3) Client ‘A’ stores session key KS and sends EKb (KS||IDA) to coordinator (B). Coordinator now knows session keys KS, knows other party-the client (A) and knows that information is generated by KDC (because Kb is known only to itself & KDC).

4) B sends nonce and public license key KL to A using

REFERENCES

[1] “Dynamic Software License Key Management Using Smart Cards” Vineet Kumar Sharma, Dr. S.A.M Rizvi 2010 International Conference on Advances in Computer Engineering


