韓國 OA學會 論文誌
第6卷 第4號, 2001. 12.

2001-6-4-3-7

인터넷을 통한 소프트웨어 불법사용 방지시스템 설계

이 상 렬*

A Design of Illegal Usage Protection System of Software through Internet

Sang-Ryul Lee*

요 약

본 논문에서는 인터넷에서 암호화 방법을 이용하여 소프트웨어의 불법사용을 방지할 수 있는 시스템을 설계하였다. 현재의 대부분 소프트웨어들은 저작권 보호를 위하여 Lock Key등과 같은 물리적인 장치를 사용하지 않고 사용상의 편의성 때문에 설치 암호를 입력하는 방식을 이용하고 있다. 따라서 암호만 알면 누구나 소프트웨어를 자유로이 이용할 수 있기 때문에 정품 소프트웨어를 구입한 사용자가 의도적으로 암호를 노출시킬 경우 라이센스 수량 이상으로 불법사용 될 수 있다. 본 논문에서 제안한 시스템을 이용할 경우 소프트웨어 사용자는 라이센스 수량만큼만 동시에 사용할 수 있음으로 소프트웨어의 불법사용을 원천적으로 봉쇄할 수 있다. 또한 소프트웨어 개발자는 판매 수량을 정확히 파악할 수 있음으로 경영에 도움을 받을 수 있고 소프트웨어 사용자는 개발자로부터 소프트웨어의 유지보수를 신속히 받을 수 있다.

Abstract

In this paper we designed the software copyright protection system using encryption technology. Recently most of the softwares have used a password instead of physical devices such as lock key for the software copyright protection. In this case if the password be disclosed, somebody will can use the software illegally. Therefore if the legal user disclose the password intentionally, somebody will can use more than license numbers. If we use the system suggested in this paper it will not be permitted to use concurrently more than license numbers. As a result nobody can use any softwares illegally. All of the software developers can check a sale amount correctly. And the users can get the service of software maintenance quickly.

* 상지영서대학 인터넷정보과 조교수

Ⅰ. 서론

정보화 사회가 발전할수록 소프트웨어의 중요성은 점점 증대되고 있다. 하지만 그 동안 소프트웨어의 불법사용으로 수많은 소프트웨어 개발회사들이 경영악화로 도산하였으며 이는 국가 정보산업 발전에 막대한 지장을 초래했다. 또한 정부의 무차별적인 소프트웨어 불법사용 단속으로 합법적인 사용자들마저 많은 불편을 겪게 됐다. 최근 인터넷의 활성화로 사회 각 분야에서 인터넷을 이용한 응용분야가 점차 확대되고 있다. 소프트웨어 유통을 포함한 모든 상거래가 인터넷을 통해 이루질 날이 멀지 않았다. 그렇다면 소프트웨어가 인터넷을 통해 유통될 경우를 대비하여 소프트웨어 불법사용 방지 방법에 대한 연구가 필요하리라 본다. 다행히 최근 인터넷에서 정보보호를 위하여 다양한 암호화 기술을 이용하고 있는데[1] 이 기술을 잘 활용한다면 소프트웨어의 불법사용을 방지할 수 있는 좋은 방법을 찾을 수 있으리라 본다.

그 동안 업계에서 소프트웨어의 불법사용을 방지하기 위하여 사용해 왔던 방식 중 몇 가지를 살펴보면 다음과 같다.

- Lock Key 사용 : 프로그램 실행시 주기적으로 컴퓨터의 프린터 포트를 검사하여 Lock Key가 꽂혀 있는지를 체크한다. 만일 이 장치가 없다면 프로그램 실행을 중지한다. 이 방식은 사용자가 여러 종류의 소프트웨어를 이용한다면 프린터 포트에 줄줄이 이 장치를 꽂아야하기 때문에 사용하기 불편하고 상호간 충돌의 문제점도 있다.
- 특정 플로피 디스켓 체크 방식 : 프로그램 실행시 주기적으로 컴퓨터의 플로피 디스켓 드라이브를 검사하여 플로피 디스켓이 꽂혀 있는지를 체크한다. 물론 이 플로피 디스켓은 복사를 할 수 없게 특수 제작된 것이다. 만일 이 플로피 디스켓이 없다면 프로그램 실행을 중지한다. 따라서 여러 종류의 소프트웨어를 동시에 사용할 수 없다. 그리고 플로피 디스켓은 손상되기 쉬운 단점이 있다.
- 설치 수량 제한 : 설치 프로그램 내부에 설치 횟수를 제한함으로써 일정 횟수 이상은 설치할 수 없게 만드는 방식이다. 설치 프로그램은 복사가 불가능하게 특수 제작된 플로피 디스켓에 있으며 이 디스켓이 손상되었을 때는 설치 불가능하다.
- 특정 시스템 체크 방식 : 소프트웨어를 특정 시스템에서만 실행 가능하게 만드는 방식이다. 일반적으로 대형 컴퓨터 제조회사가 시스템에 포함하여 제공하는 번들용 소프트웨어에 많이 이용되는 방식이다.
- 프로그램 설치 암호 사용 : 암호를 알아야만 프로그램을 설치하여 실행할 수 있게 만든 방식이다. 적법한 사용자가 암호를 의도적으로 공개할 경우에는 불법사용을 막을 수 없다. 최근에 가장 많이 이용되고 있는 방식이다.

대부분의 방식은 일시적이고 부분적인 방지 효과는 있으나 근본적으로 불법사용을 방지할 수 없으며 사용상에도 불편이 크다. 결국 사용자가 스스로 불법사용을 자제하지 않는 한 불법사용은 얼마든지 가능하다.

본 논문에서는 암호화 기술을 이용하여 소프트웨어를 불법적으로 사용하는 것을 원천적으로 봉쇄할 수 있는 방법을 연구하고자 한다. 이를 위해 대칭키 암호화 방식[2]과 공개키 암호화 방식[3] 중에서 정보 보호 기능 뿐 아니라 상호 신분 확인[4] 그리고 수신 거부 방지 기능이 있는 공개키 암호화 방식을 이용하고자 한다.

2장에서는 소프트웨어 라이센스의 분류 및 소프트웨어 불법사용의 유형을 살펴보고 3장에서는 본 논문에서 이용할 암호화 기술들에 대해 살펴본다. 4장에서는 본 논문에서 제안한 시스템의 구성과 운영 절차를 살펴보고 5장에서 제안한 시스템을 분석하고 6장에서 결론을 맺는다.

Ⅱ. 소프트웨어 라이센스의 분류 및 소프트웨어 불법사용의 유형

한국소프트웨어저작권협회(SPC)[5]에서는 다음과 같이 소프트웨어 라이센스를 분류하고 있다.

■ 기계별 라이센스

구매자는 사용 기계별로 소프트웨어를 구입해야 한다. 예를 들어 95대의 PC를 보유하고 있다면 95개의 소프트웨어를 구입하여야 한다. 시스템 운영체제(O/S)가 대표적인 기계별 라이센스 소프트웨어이다. 그 밖에 이 유형의 라이센스에 속하는 예로는 화면보호기(screen saver), 유틸리티(utility) 및 기타 PC 하드웨어에 수반되는 프로그램들이 있다.

■ 슈트(SUITES-몇 가지 상이한 프로그램이 패키지화된 상품)

슈트는 응용 프로그램을 그룹화 하여 하나의 패키지에 포함시킨 형태이다. 슈트에는 여러 응용 프로그램들이 포함되지만, 단일한 개인이 사용하는 것을 목적으로 하고 있으며 포함된 프로그램에 대해 하나의 라이센스만이 허가된다. 즉, 슈트에 포함된 프로그램들을 쪼개어 서로 다른 사람이 사용할 수는 없다. (예 : 한글과컴퓨터의 한글오피스, MS-OFFICE 등)

■ 네트워크 라이센스

이는 제한된 수의 사용자들이 동시에 하나의 소프트웨어에 연결하여 사용하도록 허용되는 것을 말한다. 이러한 형태는 네트워크 환경에서만 사용되며, 따라서 네트워크의 확산에 따라 점점 보편화되고 있다. 예를 들어 25명의 네트워크 사용자가 있지만 그 중 10명의 사용자만이 동시에 사용가능 하다면 해당 프로그램을 10카피만 구입하면 된다. 다만 25명의 PC에 소프트웨어를 모두 설치하는 것은 해석상 허용되지 않는다. 어느 경우에도 반드시 정품수량과 설치수량이 일치해야 한다.

■ 사이트 라이센스

사이트 라이센스를 이해하기 위해서는 사이트 개념을 공간상의 개념으로서 이해하여야 한다. 즉 해당 사이트를 벗어나는 곳에서는 사이트 라이센스의 이익을 주장할 수 없다. 사이트 라이센스는 여러 카피의 소프트웨어에 대한 할인을 포함할 수도 있고, 또는 하나의 디스크에 대한 무제한 카피를 허용하는 것일 수도 있다. 판매업자는 네트워크의 노드(node) 수에 근거해 가격을 조절할 수 있고, 또는 파일서버의 숫자에 따라 조정할 수도 있다. 일반적으로 사이트별 라이센스를 취득하는 것이 개인별 카피를 구입하는 것보다 저렴하다.

그리고 SPC에서 소프트웨어를 불법사용 하는 것으로 간주하는 유형은 다음과 같다.

■ 라이센스된 소프트웨어 한 카피(copy)를 구입한 다음, 라이센스 약정서의 조건을 위반하여 여러 대의 기계에 복제하는 행위

■ 소프트웨어를 친구나 동료와 함께 돌려가며 사용하는 행위

■ 약정서에 허용되지 않은 컴퓨터나 노트북컴퓨터에 설치하는 행위

■ 집에서 사용중인 정품을 회사에서 사용하는 행위 등

이상을 종합하면 소프트웨어의 적법한 사용을 위해서는 설치 수량 제한, 설치 장소 제한 그리고 사용자 제한을 철저히 해야 한다.

Ⅲ. 암호화 기술

정보보호를 위해 사용되는 기술에는 물리적으로 보호하는 방법도 있지만 컴퓨터 통신과 같이 누구에게나 공개되는 상황에서는 정보를 수신하더라도 해독할 수 없게 암호화하는 기술이 절대적으로 필요하다. 현재 인터넷에서 주로 사용되는 웹기반 프로토콜로는 1994년 미국의 넷스케이프사에 의해 최초로 개발된 SSL(Secure Socket Layer)[6]이다. 1995년 미국의 EIT(Enterprise Integration Technology)사에서 S-HTTP(Secure-Hypertext Transport Protocol)를 개발하였으나 최근 거의 사용되지 못하고 있다. SSL에서는 대칭형 암호 알고리즘으로 RC2, RC4, Triple-DES[7], SKIPJACK을, 비대칭 암호 알고리즘으로 RSA[8]를, 해시 알고리즘으로 MD5[9], SHA-1을 사용한다. 비대칭 알고리즘은 주로 전자서명을 위해서, 해시 알고리즘은 메시지 인증을 위해서 사용되고 일반 통신 내용은 대칭 알고리즘을 사용한다.

일반적으로 사용자 A가 사용자 B에게 공개된 채널을 통하여 정보 전달을 하고 있을 때 그 정보가 안전할 수 있으려면 다음과 같은 위험을 방지할 수 있어야 한다.

■ 도청 : 타인 C(도청자)가 전송내용을 중간에서 엿듣는 행위

■ 변조 : 타인 C(통신 방해자)가 전송내용을 중간에서

가로채어 그 내용의 일부를 변조하여 사용자 B에게 보내는 행위

- 위장 : 타인 C(위장자)가 사용자 A인 것처럼 사용자 B에게 나타나는 행위
- 송신부인 : 송신자 A가 자신이 전송한 내용을 향후에 부인하는 행위
- 수신부인 : 수신자 B가 자신이 수신한 내용을 향후에 부인하는 행위

지금부터 본 논문에서 이용될 몇 가지 암호화 방법에 대해 살펴본다. 평문(Plain Text) P를 키(Key) Ke를 이용하여 암호문(Cipher Text) C를 만드는 과정을 암호화(Encryption)라하고, 암호문 C를 키 Kd를 이용하여 원래의 평문 P를 복구해 내는 과정을 복호화(Decryption)라 한다.

암호화 : $C = \{P\}_{Ke}$

복호화 : $P = \{C\}_{Kd}$

여기서 암호키 Ke와 복호키 Kd가 같을 경우 대칭 암호시스템, 다를 경우 비대칭 암호시스템이라 한다.

대칭 암호시스템의 대칭키는 어느 한 쪽에서 임의로 정하면 되는 것임으로 생성은 누구나 쉽게 할 수 있다. 그러나 생성한 키를 상대방에게 어떤 방법으로 안전하게 전달하느냐가 문제다. 이 암호기법의 예로 1977년 미국 표준국에서 미 연방 정보처리표준으로 채택된 DES(Data Encryption Standard)가 많이 사용되고 있다.

비대칭 암호시스템을 이용하는 모든 사용자는 암호키와 복호키 쌍을 갖고 있다. 암호키는 누구나 볼 수 있게 공개하는 키로서 공개키(PK : Public Key)라 하며 복호키는 자신만이 알 수 있게 비밀리에 보관하는 키로서 비밀키(SK : Secret Key)라 한다. 이러한 비대칭 암호시스템을 사용하여 정보의 비밀유지는 물론 사용자 인증을 할 수 있는 암호시스템을 공개키 기반구조 시스템(Public Key Infrastructure)[10][11][12]이라 한다. 비대칭 암호시스템의 암호키는 다음과 같은 조건을 만족시켜야 한다.

- $\{\{P\}_{PK}\}_{SK} = P$
- $\{\{P\}_{SK}\}_{PK} = P$
- PK로부터 SK를 연역해 낼 수 없어야 한다.
- 암호키 쌍을 쉽게 많이 발생시킬 수 있어야 한다.

일반적으로 비대칭 암호기법은 대칭 암호기법보다 암호화 알고리즘이 복잡하기 때문에 암호화 속도가 느리다. 그러나 위의 두 번째 조건 $\{\{P\}_{SK}\}_{PK} = P$ 이 가능하기 때문에 대칭 암호시스템에서는 불가능한 디지털 서명을 할 수 있다. 비대칭 암호기법의 예로는 ElGamal 암호, Lucas 암호, 타원 곡선 암호, 양자 암호 등 많은 방식이 있으나 1977년 MIT의 R. Rivest, A. Shamir, L. Adleman이 개발한 RSA 암호가 가장 많이 이용되고 있다.

Ⅳ. 시스템 구성 및 운영 절차

인터넷을 통해 소프트웨어의 불법사용을 방지할 수 있는 소프트웨어 유통 시스템의 구성을 먼저 살펴보고 소프트웨어의 공급, 설치, 사용이 합법적으로 이루어 질 수 있는 방법을 제시하고자 한다.

4.1 소프트웨어 유통 시스템 구성

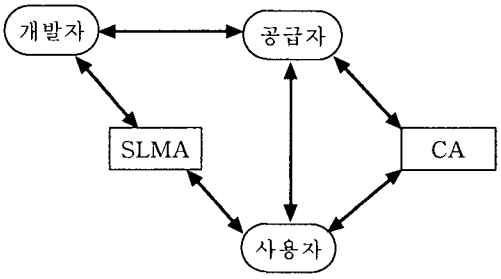

그림 1. 소프트웨어 유통 시스템 구성

소프트웨어 유통 시스템은 그림 1과 같이 개발자, 공급자, 사용자 외 2개의 관리 서버로 구성되며 모두 인터넷으로 연결되어 있다. 구성 요소 각각의 역할은 다음과 같다.

- 개발자 : 소프트웨어를 직접 개발한 자이며 실제 판매된 수량을 정확히 파악하고 싶어한다.
- SLMA(Software License Management Agent) : 소프트웨어 사용의 적법 여부를 판단해 주는 기능을 수행한다.

- 공급자 : 소프트웨어 판매를 개발자로부터 위임받은 자이며 판매 소프트웨어가 불법사용되지 않기를 바란다.
- CA(Certificate Authority) : 개인의 공개키를 포함한 인증서를 발급하는 공인된 기관으로서 상호간의 신분을 확인할 수 있다.
- 사용자 : 소프트웨어를 공급자로부터 정상적으로 구입하여 합법적으로 사용할 수 있는 자로서 소프트웨어의 신속한 유지보수를 원한다.

4.2 소프트웨어 공급 절차

그림 2는 사용자가 인터넷을 통해 안전하게 소프트웨어를 주문하여 공급받는 과정을 보여준다.

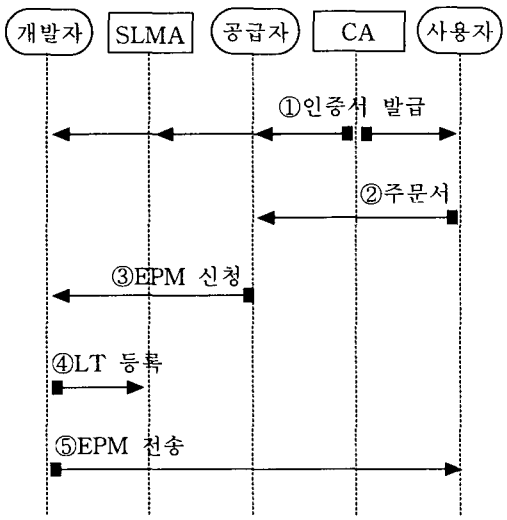

그림 2. 소프트웨어 공급 절차

① 사용자, 공급자, 개발자 그리고 SLMA는 각각 CA로부터 자신의 인증서를 발급 받아 자신의 저장장치에 보관해 둔다. X.509 v3 인증서의 양식은 표 1과 같다.

표 1. X.509 v3 인증서 양식

Version Number
Serial Number
Signature Information
Issuer
Validity Period
Subject
Public Key Information
Issuer UID (optional)
Subject UID (optional)
Extentions (optional)
Certificate Signature

사용자가 공급자의 쇼핑몰에 접속할 때 공급자는 자신의 인증서를 사용자에게 전송함으로서 사용자에게 공급자의 신분을 확인시켜줄 수 있으며 또한 자신의 공개키를 알려줌으로서 향후 사용자가 암호문을 작성할 수 있도록 한다. 인증서는 CA가 서명한 문서로서 이를 해독하려면 CA의 공개키가 있어야 한다. CA의 공개키를 PKc 비밀키를 SKc라 할 때 인증서를 암호화하고 해독하는 과정은 다음과 같다.

■ 인증서 신청 단계 :

인증서 신청자는 공개키와 비밀키 쌍을 스스로 비밀리에 생성한다.

신청자는 CA에 자신의 공개키를 포함하여 인증서 생성을 신청한다.

■ 인증서 생성 단계 :

CA는 표 1과 같은 인증서 양식에 맞춰 문안 M을 작성한다.

CA는 인증서 문안 M을 자신이 비밀키 SKc로 암호화하여 인증서 $\{M\}_{SKc}$를 생성한다.

■ 인증서를 이용한 신원 확인 단계 :

상대방에게 본인의 인증서를 제시한다.

상대방은 CA의 공개키 PKc로 수신한 인증서 $\{M\}_{SKc}$를 해독하여 $M=\{\{M\}_{SKc}\}_{PKc}$을 찾아낸다. 즉 M의 내용은 CA가 서명한 내용임으로 인증서 소유자의 신원을 확인할 수 있다.

이후 사용자 인증서를 Certi-U, 공급자 인증서를

Certi-P, 개발자 인증서를 Certi-D, SLMA 인증서 Certi-S 로 표기한다.

② 사용자는 공급자의 공개키 PKp로 주문서를 암호화하여 자신의 인증서와 함께 공급자에게 보낸다.

$\{\{주문내용\}_{PKp}\}_{SKu}$, Certi-U,

공급자는 주문서를 해독하여 주문 상품명 및 수량 등을 체크하고 사용자 인증서로부터 주문자의 신원을 확인한다. 또한 사용자의 서명 기능이 있음으로 주문 사실을 부인할 수 없기 때문에 허위 주문을 방지할 수 있다.

③ 공급자는 주문 상품명과 수량 등의 정보를 포함한 Mo를 개발자에게 통보한다.

$\{Mo\}_{SKp}$, Certi-S

이를 받은 개발자는 Certi-S로부터 공급자의 신원을 파악하고 PKp를 찾아내어 주문 상품명과 수량 등의 정보를 해독한다.

$Mo \equiv \{\{Mo\}_{SKp}\}_{PKp}$

④ 개발자는 사용자 ID, 라이센스 수량(설치 횟수) 등의 사용권한 내역 정보를 포함한 LT(License Token)을 만든 후 SLMA에 이를 등록하고 공급할 소프트웨어를 암호화하여 사용자에게 전송한다. LT를 SLMA에 전송할 때 그 내용이 변조되는 것을 방지하기 위하여 $\{LT\}_{PKs}$ 로 암호화하여 보낸다. SLMA는 이 내용을 자신의 비밀키 SKs로 해독하여 $LT \equiv \{\{LT\}_{PKs}\}_{SKs}$를 찾아내고 그 내용을 보관해 둔다.

⑤ 소프트웨어 암호화는 소프트웨어 전체를 암호화하지 않고 일부만 암호화함으로서 암호화/복호화 시간을 절약할 수 있다. 따라서 소프트웨어를 S1과 S2로 나누어 S2는 암호화가 되지 않은 PPM(Plain Program Module)로 만들고 S1은 사용자의 공개키 PKu로 암호화 $\{S1\}_{PKu}$ 하여 EPM(Encrypted Program Module)으로 만든다. PPM부분은 언제 누구라도 온라인 등의 방법을 통해 받아볼 수 있도록 공개하며 EPM은 개발자가 사용자에게 온라인으로 직접 전달한다.

이로써 사용자는 EPM을 자신의 비밀키 SKu로 복호화하여 $S1 \equiv \{\{S1\}_{PKu}\}_{SKu}$를 얻을 수 있음으로 자신이 의도적으로 SKu를 누설하지 않는다면 자신만이 이 소프트웨어를 사용할 수 있다.

4.3 소프트웨어 사용권 확인 절차

그림 3은 사용자가 소프트웨어를 사용할 자격이 있는지를 확인하는 과정을 보여준다.

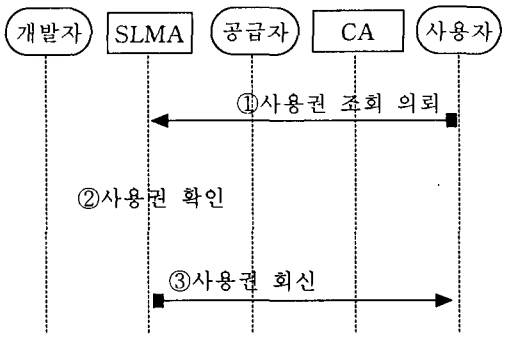

그림 3. 소프트웨어 사용권 확인 절차

실행 프로그램 내부에 사용권을 확인하는 모듈을 넣어 둠으로서 프로그램이 실행중에 SLMA와 사용권 확인을 하여 사용여부를 결정한다. 사용권을 확인하는 모듈이 한두 군데만 있을 경우 디버거를 이용하여 그 부분을 바이패스 시키는 방법으로 불법사용을 할 수 있음으로 프로그램 개발자는 사용권을 확인하는 모듈을 여러 곳에 두어야 함을 유의해야 할 것이다. 그리고 실행 중에 체크하는 방식임으로 그 모듈을 최적화하여 실행 속도에 지장을 주지 않도록 해야할 것이다. 사용권 조회 의뢰 및 사용권 회신 내용은 중간에서 도청 및 변조를 할 수 없도록 암호화하여 송수신한다.

① 우선 사용자측 소프트웨어의 사용권 확인 모듈에서 다음과 같은 사용권 조회 의뢰 내용을 SLMA의 공개키로 암호화하여 SLMA에 보낸다.

$\{사용자ID, 소프프웨어ID, 랜덤번호\}_{PKs}$

② SLMA는 자신의 SKs로 암호문을 해독하고 사용 가능 여부를 검사한다. SLMA의 LT에 필수적으로 기록되어 있어야 할 내용은 다음과 같다.

(사용자ID, 소프트웨어ID, 라이센스 수량, 검사주기, 이전 검사시각, 사용중인 수, 사용중인 IP목록)

SLMA는 일정 시간(모듈이 사용권 조회를 의뢰하는 시간 간격)동안 사용권 조회 의뢰 건수를 검사할 경우 동시에 사용할 수 있는 사용자 수를 라이센스 수량만큼으로 제한할 수 있다. 예를 들어 검사주기를 5분으로 할 경우 이전 검사시각이 6:00이고 현재 시각이 6:05이면 이전 검사시각을 현재시각으로 변경하고 사용중인 수를 0으로 그리고 사용중인 IP목록을 지운다. 현재 시각이 6:00~6:05 사이이면 현재의 IP가 사용중인 IP목록에 없을 경우 사용중인 IP목록에 현재의 IP를 기록하고 사용중인 수를 1 증가시킨 후 그 수가 라이센스 수량보다 크면 -Rsp를 만든다. 여기서 사용하는 시각은 SLMA 시스템 시각이다.

③ SLMA는 자신의 비밀키를 이용하여 아래와 같은 암호화된 응답문을 작성하여 사용자에게 보낸다. 랜덤번호를 사용하는 이유는 매번 암호문을 다르게 생성함으로써 중간에서 도청자가 암호문 전체를 복사하여 재사용 하는 것을 방지하기 위함이다.

{사용자ID, 소프트웨어ID, 랜덤번호, ±Rsp}$_{SKs}$

이를 받은 사용자측 소프트웨어는 SLMA의 공개키를 이용하여 암호문을 해독하여 +Rsp를 받았을 때만 실행을 계속하고 -Rsp를 받았을 때는 프로그램 사용권한이 없음으로 실행을 종료한다. 사용자ID, 소프트웨어ID, 랜덤번호를 비교함으로써 직전에 사용권 조회 의뢰한 것에 대한 회신인지를 확인할 수 있다.

V. 시스템 분석

현재의 불법 소프트웨어 단속은 설치 수량 위주로만 이루어지고 있지만 라이센스 종류에 따라 설치 장소 및 사용자까지 사실상 제한하고 있기 때문에 불법 소프트웨어의 사용을 막을 수 있는 시스템이 되려면 다음 세 가지 조건을 만족해야 한다.

- 소프트웨어 설치 수량을 라이센스 수량 이하로 제한할 수 있어야 한다.
- 소프트웨어 설치 장소를 제한할 수 있어야 한다.
- 소프트웨어를 사용하는 사용자를 제한할 수 있어야 한다.

소프트웨어에는 설치 후 사용하는 것과 설치 과정 없이 파일만 복사하여 바로 이용할 수 있는 것으로 분류할 수 있다. 전자의 경우는 소프트웨어 공급, 설치 그리고 실행 3단계 과정을 거치며 후자의 경우는 설치단계가 필요 없다. 각 단계에서 시스템의 구성 요소들간의 온라인 또는 오프라인 상태에 따라 그 특성을 분석해 본다.

표 2는 소프트웨어 공급단계에서 사용자와 공급자간 그리고 설치/실행단계에서 사용자와 SLMA간에 온라인/오프라인 상태를 나타낸다. 〈경우1~4〉는 소프트웨어를 설치한 후 실행하는 경우이며 〈경우5~6〉은 설치과정 없이 복사 후 즉시 실행하는 경우이다.

표 2. 소프트웨어 처리 단계별 온라인/오프라인 상태

	공급단계	설치단계	실행단계
〈경우1〉	On	Off	Off
〈경우2〉	On	On	Off
〈경우3〉	On	Off	On
〈경우4〉	On	On	On
〈경우5〉	On		Off
〈경우6〉	On		On

모든 경우 공급자는 소프트웨어를 암호화하여 전송함으로 안전하게 적법한 사용자에게 전달할 수 있다. 그밖

에 각 경우의 특성을 분석해 보면 다음과 같다.

- 〈경우1〉, 〈경우5〉 : 적법한 사용자라면 라이센스 수량 이상으로 여러 곳에 설치하여 사용할 수 있다. 다만 실행단계에서 비밀키를 요구할 경우 사용자의 제한은 가능하다. 하지만 적법한 사용자가 자신의 비밀키를 의도적으로 누설할 경우에는 라이센스 수량 이상의 적법하지 않은 사용자가 사용할 수 있다. SLMA가 별도로 필요하지 않은 경우이다.
- 〈경우2〉 : SLMA를 이용하여 라이센스 수량만큼만 설치할 수 있도록 제한할 수 있다. 따라서 실행시 비밀키를 별도로 요구하지 않더라도 라이센스 수량 이상으로 사용할 수 없다. 하지만 적법하지 않은 사용자가 소프트웨어를 실행시킬 수 있다.
- 〈경우3〉, 〈경우6〉 : 적법한 사용자라면 라이센스 수량 이상으로 여러 곳에 설치하여 사용할 수 있다. 그러나 실행시 SLMA를 이용하여 라이센스 수량만큼만 실행할 수 있도록 제한할 수 있다. 따라서 적법한 사용자가 자신의 비밀키를 의도적으로 누설하더라도 라이센스 수량 이상의 사용자가 동시에 사용할 수 없다. 실행시 비밀키를 요구한다면 적법한 사용자만 사용 가능하게 할 수 있다.
- 〈경우4〉 : SLMA를 이용하여 라이센스 수량만큼만 설치할 수 있도록 제한할 수 있다. 실행시 SLMA를 이용하여 라이센스 수량만큼만 실행할 수 있도록 제한할 수 있다. 따라서 적법한 사용자가 자신의 비밀키를 의도적으로 누설하더라도 라이센스 수량 이상의 사용자가 동시에 사용할 수 없다. 실행시 비밀키를 요구한다면 적법한 사용자만 사용 가능하게 할 수 있다.

〈경우2〉, 〈경우4〉와 같이 설치 횟수를 제한하더라도 설치된 하드디스크 전체를 복사할 경우 설치 횟수를 제한할 수 없는 문제와 하드디스크가 파손되었을 때 재설치를 해주어야 하는 문제 등이 있기 때문에 SLMA를 운영하더라도 설치 횟수를 제한하기가 쉽지 않을 것이다. 따라서 본 논문에서는 동시에 이용할 수 있는 사용자 수를 라이센스 수량 이하로 제한할 수 있는 〈경우3〉, 〈경우6〉을 선택하여 설치수량 제한 조건을 간접적으로 만족시킬 수 있었다.

SLMA와 온라인 상태를 유지할 경우 인터넷 IP주소를 이용하면 사용자의 위치 파악이 가능함으로 설치 장소를 제한할 수 있다. 또한 사용자의 비밀번호를 관리할 수 있음으로 사용자 제한까지 할 수 있다. 따라서 본 논문에서 제안한 시스템은 소프트웨어의 불법사용을 막을 수 있는 시스템이 되기 위한 세 가지 조건을 모두 만족시킨다. 나아가 소프트웨어의 사용 기간을 제한할 수도 있음으로 소프트웨어 대여도 가능할 것이다. 그리고 온라인을 통해 소프트웨어 버전 업그레이드도 신속히 이루어질 수 있을 것이다.

Ⅵ. 결론

인터넷 이용자가 폭발적으로 증대함으로써 전자상거래가 활성화되고 있다. 따라서 소프트웨어의 유통도 온라인으로 변화되리라 본다. 따라서 본 논문에서는 온라인으로 소프트웨어를 안전하게 전달하고 불법적인 사용을 방지할 수 있는 방법을 제안했다.

사용한 암호시스템은 대칭키 암호화 기술을 이용한 공개키 암호시스템이다. 이 암호시스템은 정보의 비밀성 보호는 물론 공인된 기관인 CA로부터 사용자 인증을 받을 수 있음으로 소프트웨어 사용자는 정상적인 공급자로부터 소프트웨어를 구입할 수 있으며 공급자 역시 사용자로부터 받은 주문을 의심할 필요가 없다. 그리고 소프트웨어 사용권을 검사해 주는 SLMA를 설치함으로써 라이센스 수량만큼만 사용자가 이용할 수 있게 하였으며 소프트웨어 개발자는 판매 수량을 직접 관리하기 때문에 소프트웨어 판매를 담당하는 공급자와의 마찰을 없앨 수 있다. 또한 소프트웨어 유지보수는 개발자가 직접 담당하게 함으로써 소프트웨어 공급자는 판매활동에만 전념할 수 있고 사용자는 신속한 유지보수를 받을 수 있을 것이다. SLMA를 좀 더 잘 활용한다면 라이센스 수량 관리는 물론 사용 기간 제한, 버전 업그레이드 용이, 사용 장소 제한, 사용자 제한 등을 할 수 있을 것이다.

현재 대부분의 소프트웨어들은 비밀번호를 이용하여 설치 제한을 해 왔으나 이런 방법으로는 근본적으로 소프트웨어의 불법사용을 막을 수 없다. 다만 사용자가 스스로가 불법사용을 자제해 주길 바랄 뿐이다. 본 논문에서는 소프트웨어의 불법사용을 원천적으로 봉쇄할 수 있는

방법을 제안함으로써 소프트웨어 불법사용 단속 자체를 불필요하게 만들었다. 향후에는 물리적인 방법을 이용하지 않고 소프트웨어의 설치 횟수를 제한하는 방법을 연구할 필요가 있다.

참고문헌

[1] 한국정보보호진흥원, http://www.kisa.or.kr/

[2] G.J. Simmons, "Symmetric and Asymmetric encryption", Comput. Surv. 11, 4, Dec. 1979.

[3] W. Diffie and M.E. Hellman, "New Directions in Cryptography", IEEE Trans. on IT, Vol. 1 IT-22, No.6, Nov. 1976.

[4] A. Cain, "Security, Authentication, and Privacy on the Web", Fourth International WWW Conference, 1995.

[5] 한국소프트웨어 저작권 협회, http://www.spc.or.kr/

[6] A.O. Freier, P. Karton, P.C. Kocher, The SSL Protocol V3.0, Transport Layer Security Working Group, Nov. 1996

[7] Data Encryption Standard, Federal Information Processing Standard (FIPS) Publication 46, National Bureau of Standard, U.S. Department of Commerce, Washington DC, Jan. 1977.

[8] R. Rivest, A. Shamir, L. Adleman, "A method for obtaining digital signature and public-key cryptosystem", Commun. ACM 21, 2, Feb. 1978.

[9] R. Rivest, The MD5 Message-Digest Algorithm, MIT Laboratory for Computer Science and RSA Data Security Inc., Apr. 1992.

[10] IETF PKIX RFC2459, "Internet X.509 Public Key Infrastructure Certificate and CRL Profile", Jan. 1999

[11] C. Adams, S. Farrell, "Internet X.509 Public Key Infrastructure Certificate Management Protocols", IETF draft, Feb. 2001

[12] 한국정보보호센터, "공개키 기반구축에 관한 연구", Dec. 1997

저 자 소 개

이 상 렬

1981년 한양대학교 전자공학과 공학사

1983년 한양대학교 전자공학과 공학석사

1983년~1993년 삼성전자 컴퓨터연구실 과장

1993년~1997년 경인여대 사무자동화과 조교수

1998년 한양대학교 전자공학과 공학박사 수료

1997년~현재 상지영서대학 인터넷정보과 조교수