# USB를 활용한 보안 시스템 기능개발에 관한 연구

박지예*, 이덕규*
*서원대학교 정보보안학과
wldp8206@naver.com, deokgyulee@seowon.ac.kr

# A Study on the Function Development of Security System Using USB

Ji ye Bak *, Deok Gyu Lee*
*Dept. of Information Security, SeoWon University

요 약

나날이 증가하고 있는 개인정보 유출 및 사이버 위험으로부터 피해를 최소화하기 위해 보안성 향상을 목적으로 기존의 인증 방법과 결합해 사용할 수 있는 새로운 인증시스템 방안을 고안하였다. 손쉽게 활용할 수 있는 USB를 이용하여 자체적으로 제작한 보안 모듈을 탑재함으로써 등록된 USB를 소유하고 있는 개인만이 자유롭게 정보를 사용할 수 있도록 구현하여 한층 폐쇄적이고 극단적인 인증 방법을 구축하였다.

in order to minimize damage from increasing personal information leakage and cyber risks, a new authentication system plan was devised that can be used in combination with existing authentication methods for the purpose of improving security. A more closed and extreme authentication method was established by installing a self-made security module using USB that can be easily utilized, enabling only individuals with registered USB to freely use information.

## 1. 서론

최근 코로나 19와 4차 산업혁명 등의 영향으로 빅데이터, AI 기술(Artificial Intelligence), ICT 신기술이 빠르게 발전되고 생활 속으로 밀접하게 침투하면서 해킹 사고, 랜섬웨어 감염, 개인정보 유출 등 각종 사이버 위협이 높아지는 상황이다.

(그림 1. 광고사진)

그로 인해 기업 및 개인의 정보보호의 중요성이 날로 높아지는 추세이며 해결책으로 백신, 침입 탐지 시스템(IDS : Intrusion Detect System), 방화벽(firewall), Active Directory 등이 존재하며 효과적인 보안시스템을 개발하기 위해 여러 분야에서 많은 연구가 진행되고 있다.

현재 온라인상에서는 주로 ID/PW와 PIN 인증 방법을 채택하고 있으며 유추 가능성, 개인부주의에 의한 노출 가능성, 일정한 패턴 유추 등의 문제점으로 많은 유출피해와 취약점이 나타나고 있다. 이에 여러 개의 인증 방법을 접목시킨 다중 인증 방법의 필요성이 대두되고 있으며 다중 인증 방법의 방안으로써 본 논문은 ID/PW 인증 방법과 함께 사용할 수 있도록 시스템상의 폐쇄적 잠금 기능(후킹 기법 활용)을 구현하였고 잠금 기능을 해제하는 방법으로 일상생활에서 쉽게 사용할 수 있는 USB를 활용하여 보안모듈을 탑재한 USB 인증 방법을 구현하였다. 이에 대해 자세히 살펴보고자 한다.

## 2. 잠금기능 구현

### (2.1) 잠금방법

① 잠금을 실행하면 첫 번째로 하는 일은 현재 연결된 USB 이름, 시리얼넘버, 고유날짜를 별도로 만들어놓은 usbinfo.cs 폼에 usbDetec 객체 함수로 string을 사용하여 저장한다.

② 현재 화면을 캡처한 후 화면에 출력하여 일시 정

지한 효과를 준다.

③ “인증된 USB를 연결하세요”라는 폼을 나타나게 했으며 확인 버튼을 누를 시 동작을 멈추게 했다.


(그림 2. 잠금 기능 실행 시 나타나는 폼)

④ 후킹 기법을 이용하여 키보드, 마우스 클릭(움직임은 가능), 시스템 조합키 사용을 정지한다.

```
namespace usbDetec
{
    참조 5개
    public partial class Form_lock : Form
    {
        //키보드 후킹
        private const int WH_KEYBOARD_LL = 13;
        private const int WM_KEYDOWN = 0x0100;
        private const int WM_KEYUP = 0x0101;
        private const int WM_SYSKEYDOWN = 0x0104;
        private const int WM_SYSKEYUP = 0x0105;
        private static LowLevelKeyboardProc _proc = HookCallback;
        private static int _hookID = 0;

        private delegate int LowLevelKeyboardProc(int nCode, int wParam, ref KBDLLHOOKSTRUCT lParam);
        참조 1개
        private static int HookCallback(int nCode, int wParam, ref KBDLLHOOKSTRUCT lParam)
        {
            bool bReturn = false;
            switch (wParam)
            {
                case WM_KEYDOWN:
                case WM_KEYUP:
                case WM_SYSKEYDOWN:
                case WM_SYSKEYUP:

                    bReturn = ((lParam.vkCode == 0x09) && (lParam.flags == 0x20)) ||    //Alt + Tab
                        ((lParam.vkCode == 0x1B) && (lParam.flags == 0x20)) ||  //Alt + Esc
                        ((lParam.vkCode == 0x1B) && (lParam.flags == 0x00)) ||  //Ctrl + Esc
                        ((lParam.vkCode == 0x5B) && (lParam.flags == 0x01)) ||  //Left Windows Key
                        ((lParam.vkCode == 0x5C) && (lParam.flags == 0x01)) ||  //Right Windows Key
                        ((lParam.vkCode == 0x73) && (lParam.flags == 0x20));    //Alt + F4

                    break;
            }

            if (bReturn == true)
                return 1;
            else
                return CallNextHookEx(0, nCode, wParam, ref lParam);
        }
```

(그림 3, 후킹기법을 사용한 Lock기능 설계코드)

### (2.2) 화면 및 키보드 잠금

잠금 기능의 핵심효과로 잠금 모듈을 실행하면 현재 사용 중이던 화면을 그대로 캡처한 후 정지시킨 뒤 캡처 화면을 출력하도록 설계하였다. 인증용 USB를 소유하지 않거나 인증되지 않은 사용자에게 일시적인 컴퓨터 정체가 발생했다고 혼란을 줄 수 있도록 마우스 커서가 움직이긴 하지만 클릭이나 키보드가 동작하지 못하도록 구현하였다.

### (2.3) 시스템 키 잠금 (ctrl, shift, alt)

Ctrl, shift, alt를 여러 가지 키와 결합하여 사용하는 시스템 키를 후킹 기법을 이용해 미리 지정해놓고 작업관리자, 화면이동, 시스템 종료 등의 기능을 사용하지 못하도록 설정하였다.

## 3. USB 인식

① 연결된 장치가 USB인지 판단할 수 있는 변수를 선언하고 현재 장치의 연결상태를 확인한다.

② 현재 연결된 USB의 정보를 변수를 선언하여 저장한다.

③ 등록된 USB정보와 연결된 USB 정보가 일치하는지 식별, 관리할 수 있는 변수를 선언하고 잠금 모듈 실행 시 따로 저장해놓은 인증용 USB의 이름, 시리얼 넘버, 고유날짜를 읽어온다.

④ 연결된 USB와 등록된 USB 정보가 일치하는지 확인한 뒤 Switch – case-if 문을 사용 처리한다.

(④.1) 일치할 시 인데스 번호 +1, 시간값 추가, 리스뷰 아이템을 리스트뷰에 추가한 후 리스트뷰 인덱스 추가를 실행한다.

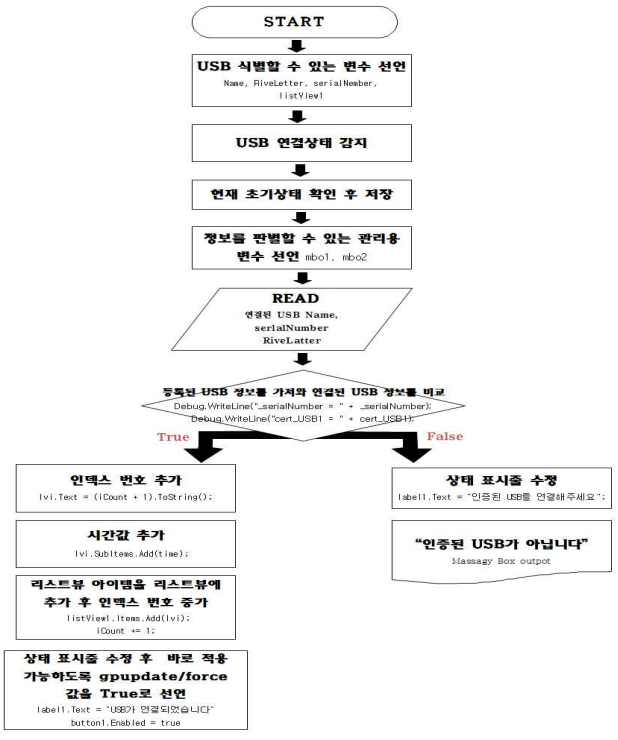
(④.2) gpupdate/force를 True로 설정하여 바로 적용 가능할 수 있도록 설정한 후 "정상ygcgv 연결 완료" 상태표시줄 변경 후 “USB가 연결되었습니다” 메시지 박스 출력

⑤ 연결된 정보와 일치하지 않을 시엔 "인증된 USB를 연결해주세요" 상태표시줄 변경 후 "인증된 USB가 아닙니다." 메시지 박스 출력

⑥ USB 연결이 종료되었을 시 인덱스 번호의 1추가, 시간값 추가, 리스뷰 아이템을 리스트뷰에 추가한 후 리스트뷰 인덱스 추가한다

(⑥.1) "정상ygcgv 해제 완료" 상태표시줄 변경 후 "USB가 해제되었습니다. USB를 연결해주세요." 메시지 박스 출력

(⑥.2) gpupdate/force를 False 값으로 설정하여 바로 적용하지 못하도록 설정




(그림 4, USB인증 모듈 순서도)

```
if (mbo2["InterfaceType"].ToString() == "USB")
{
    //if (mbo2["Caption"].ToString() == strUSBDriveName)
    // cert_USB = 등록된 정보 , _serialNumber = 연결된 정보

    //    if (cert_USB1.Equals(_serialNumber) || cert_USB2.Equals(_serialNumber) || cert_USB3.Equals(_serialNumber) || cert_USB4.Equals(_serialNumber))
    {
        string time = DateTime.Now.ToString("HH:mm:ss");    // 현재 시간 저장

        ListViewItem lvi = new ListViewItem();
        lvi.Text = (iCount + 1).ToString();         //인덱스 추가
        //lvi.SubItems.Add("USB가 해제되었습니다."); // 내용 추가
        lvi.SubItems.Add("정상 ygcgv 해제 완료");
        lvi.SubItems.Add(time);                     //시간값 추가
        listView1.Items.Add(lvi);                   //리스트뷰 아이템을 리스트뷰에 추가
        iCount += 1;                                // 리스트뷰의 인덱스 번호 증가

        //연결이 끊긴 뒤 상태표시줄 수정
        label1.Text = "USB가 해제되었습니다. USB를 연결해주세요.";
        button1.Enabled = false;        // 연결끊기면 gpupdate/force 를 못하게

        dlg = new Form_lock();
        dlg.Show();
```

(그림 5. USB 연결 끊겼을 때 코드)


(그림 6. 포스트코로나 관련 광고사진)

## 4. 결론

USB 보안 모듈과 후킹 기법을 이용한 잠금 기능 구현함으로써 기존의 인증 방법과 결합하여 이중으로 사용할 시 나타나는 보안적 측면의 효율성 향상과 쉽게 구할 수 있는 USB를 보안 모듈로 사용함으로써 얻을 수 있는 편리성을 추구할 수 있었다. 또한 원격을 통하여 방화벽이나 IDS 등 기존의 보안 시스템을 뚫고 제3자가 침투하였을 때 극단적 잠금 기능을 실행시킨 상태였다면 혼란을 주기 때문에 피해를 최소화할 수 있다고 느낄 수 있었다.

## 5. 효율적인 방안 및 모듈의 문제점

보안의 취약점이 많이 발생하는 분야와 결합하여 사용한다면 보다 안전한 보안성을 보장할 수 있다. 예를 들어 “포스트 코로나” 시대를 예측하는 요즘, 여러 기업이 선두적으로 시범운영하고 있는 재택근무 등의 큰 허점으로 기업의 중요 문서 유출 피해를 우려하는 목소리가 나오고 있다. 현재는 기업의 인트라넷의 들어가려면 직원용 ID/PW를 사용하여 접속하고 있으며 이처럼 보안 모듈이 탑재된 USB를 ID/PW 방법과 결합하여 사용한다면 취약점을 해결하는 대책마련이 될 수 있다고 생각한다.

자체적으로 개발한 USB 보안 모듈을 실전에 사용하였을 시 나타나는 취약점으로 두 가지가 있다.

1. USB를 분실하는 경우 정보 유출 가능성이 있다.
2. 극단적 잠금 기능이 실행하였을 때 컴퓨터를 재부팅할 시 잠금이 해제된다.

### 참고문헌

1) 김영진 외 “USB를 이용한 보안 토큰 시스템 구현” 한국전자통신연구원 생체인식기술연구팀, 대한전자공학회 학술대회, 2002.06 참고

2)https://terms.naver.com/entry.nhn?docId=3432459&cid=58445&categoryId=58445 정보보호의 중요성,네이버 지식백과 사전 인용

3)김평화 기자, 2020.06, IT조선 뉴스 참고 http://it.chosun.com/site/data/html_dir/2020/06/15/2020061504420.html, “포스트 코로나 시대 보안 문제 '규제 완화' 우선돼야”

4) 저자 백현우, “처음배우는 C# 프로그래밍”, 2020.05, 위키북스 참고