2014년 정보보호학과 졸업작품 보고서

Secure Line 보안 USB 개발

팀명 : Secure Line

지도 교수 : 양정모 교수님

조원 : 손민규

송민섭

송무건

2014.05

중부대학교 정보보호학과

요 약 문

1. 연구제목

보안영역 탑재 USB 개발

2. 연구 목적 및 필요성

휴대용 데이터 저장장치(USB 메모리)는 휴대와 사용이 간편하기 때문에 직장인, 개인이 데이터를 저장하여 휴대해서 가지고 다니는 경우가 보편적이다. 이런 USB를 부주의로 인해 분실했을 시 발생하는 피해를 최소화 하기 위해 보안영역과 일반영역으로 나누어 인증시에만 보안영역에 접근을 허용하는 시스템을 구축하여 USB저장 메모리의 보안을 강화하기 위해 기획 및 연구를 하게 되었다.

3. 연구 내용

보안영역 탑재 USB는 USB메모리를 PC에 연결 후 인증 프로그램 미구동시 기존 제품처럼 사용할 수 있는 일반영역과 가상 하드디스크를 이용하여 보안영역으로 구분지어 사용할 수 있도록 개발하였다. 이로 인해 제한적 접근을 함으로써 사용자 이외에는 보안영역에 담긴 파일을 볼 수 없으므로, 분실시 발생하는 피해를 1차예방을 목적으로 한다.

4. 연구 결과

이번 연구에서는 보안 영역의 구현을 위하여 가상 하드디스크를 이용하였고, 하드웨어 인증에는 MAC주소 인증과, 디스크 시리얼 넘버를, 사용자 인증에는 AES 암호알고리즘을 적용하여 구현하였지만, 개발한 보안영역 탑재 USB는 암·복호화 과정이 USB메모리의 읽고, 쓰기 성능에 따라 좌우되는 단점이 존재한다. 이를 개선하여 사용자의 편리함과 처리속도 상승효과를 목적으로 앞으로 지속적인 연구를 할 것이다.

목 차

1. 서 론

1.1 연구의 목적

정보가 방대해지고 많아짐에 따라, 필요성에 따라서 개인, 기업, 단체 등에서 이동식 저장 디스크(USB)에 필요한 정보나 문서 등을 담아서 가지고 다니는 경우가 많다. 기업이나 단체가 USB에 저장하고 다니는 정보는 보통 고객정보나, 기업 내 중요문서, 자료 등을 포함해서 다니는 경우가 많은데, 이 이동식 디스크 분실로 인한 피해가 각종 매체에서 심심찮게 보도되는 경우가 있다. 이런 분실로 인한 피해는 USB매체에 저장되어 있는 정보가 암호화 되어있지 않거나, 보안용 USB가 아닐 경우가 해당되는데, 이러한 피해를 줄이고, 정보의 보안의 중요성을 경각시키고자 USB의 일반영역과 보안영역을 나누어 제한적 접근이 가능하게 하는 프로그램을 개발함으로써 암호화에 대한 이해와 프로그래밍 능력을 배양 하기 위해 주제를 정하였다.

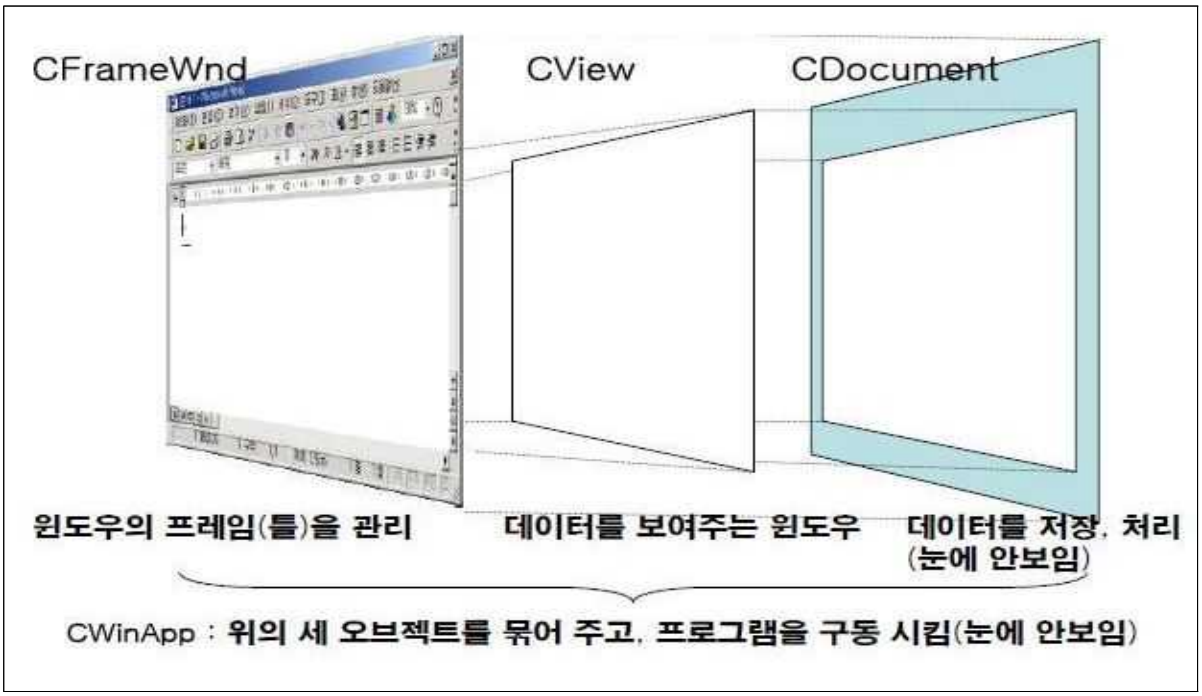
2. 이론적 배경

2.1 MFC(Microsoft Foundation Class)

MFC는 마이크로소프트사의 윈도우 응용 프로그램 개발용 클래스 라이브러리로써, Visual C++에 포함되어 있고, Win32 프로그래밍에 사용된다.

MFC는 기본적으로 700개가 넘는 방대한 Win32 API를 기반으로 윈도우즈 응용 프로그램을 제작할 때 자주 사용되는 API들을 클래스화 하여 묶어 놓고, 마이크로소프트사의 새로운 기술들이 소개 될 때 마다 해당 API를 클래스화 하여 윈도우즈 응용 프로그램을 제작하는 프로그래머에게 제공되는 아주 유용한 클래스 라이브러리이다. 또한 윈도우즈 응용 프로그램을 개발 할 때 Win32 SDK(API)를 사용하는 것보다 MFC를 사용하면 많은 이점을 누릴 수 있다.

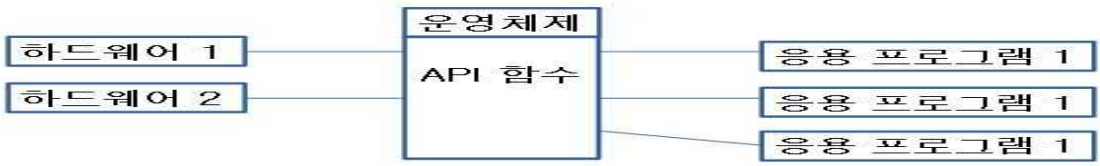
이점을 보자면 객체지향을 이용하여 C++ 프로그래밍을 하면서도 Win32 SDK 함수를 그대로 사용할 수 있으며, Win32 SDK 함수를 이용하여 윈도우즈 응용 프로그램을 개발하는 것에 비하여 MFC를 사용하면 덜 복잡하다. C++에서 제공하는 강한 타입 체크, 예외처리, 객체의 생성자와 소멸자를 이용한 메모리의 할당과 해지 등등에 대한 이점을 누릴 수 있다. 안전한 동적 메모리 관리 그리고 할당 영역 검증, 디버깅 등의 기능을 활용하여 버그가 적은 즉 안전한 프로그램을 작성 할 수 있다.



<그림 1> MFC의 기본 구조도

2.2 API(Application Programming Interface)

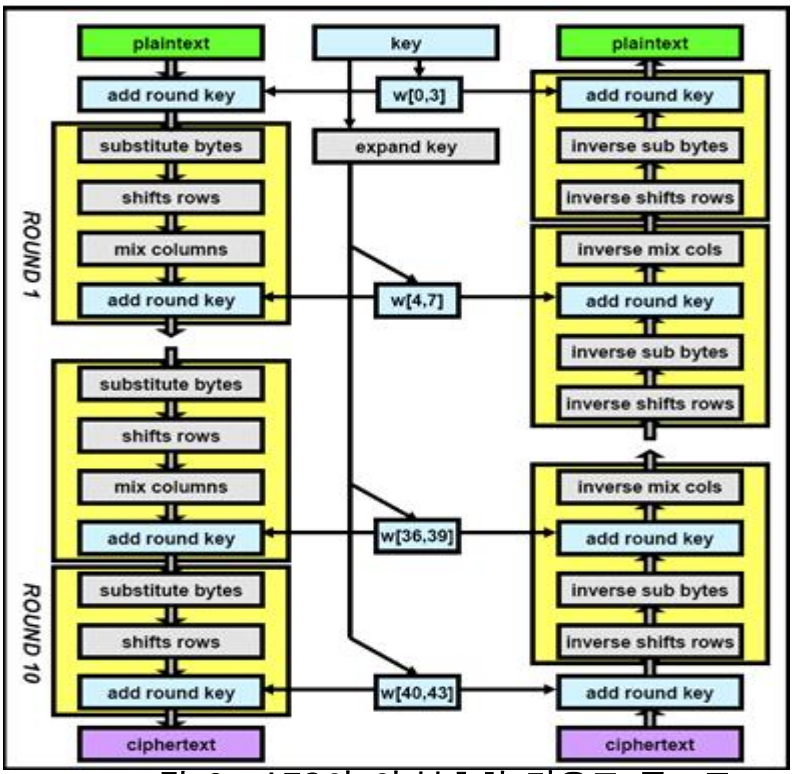
프로그램 또는 애플리케이션이 운영 체제에 어떤 처리를 위해서 호출할 수 있는 서브루틴 또는 함수의 집단이다. 윈도우즈는 응용 프로그램을 위한 함수 집합을 제공하는데 이를 API라고 하며 좀 더 쉽게 표현하면 윈도우 운영체제를 실행하고 제어하는 역할과 윈도우 자체를 만들고 운영할 수 있게 해주는 프로그래밍을 말한다. 또한 프로그램과 프로그램을 연결시켜주는 통로 역할도 담당한다.
API는 또한 자율성이 뛰어나기 때문에 GUI(Graphic User Interface)도 만들 수 있으며, 운영체제를 포함한 여러 가지 환경 조작이 용이하다.



<그림 2> API의 구성도

2.3 AES(Advanced Encryption Standard)

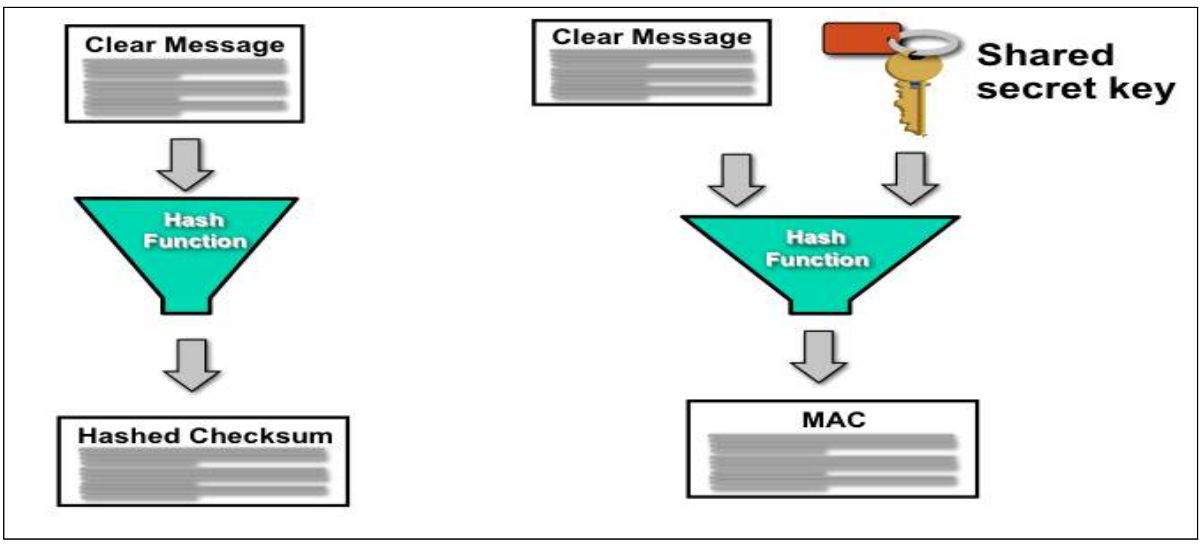
1977년도에 미국 표준으로 제정된 DES(Data Encryption Standard)는 지금까지 큰 허점이 발견되지 않았지만 키 길이가 56비트 밖에 되지 않아 현재의 컴퓨팅 기술로는 쉽게 전사공격을 하여 암호해독을 할 수 있는 문제점을 지니고 있다. 따라서 1997년에 새 표준에 대한 작업을 시작하여 2000년 10월에 AES라는 새 표준을 채택 하였다. 새 표준에 대한 제안에 의하면 '새 암호 알고리즘의 블록 크기는 128비트 이어야 하며, 알고리즘에 대한 변경 없이 128비트, 196비트, 256비트 길이의 키를 지원해야 한다'라고 제안 하였으며, 제출된 여러 제안 중 1999년에 이 중에 벨기에 암호학자인 Daemen과 Rijmen이 제안한 Rijndael 암호 알고리즘이 AES로 채택되었다. AES 암호는 키 확장과 라운드 키 선택으로 구성이 되며, 라운드 키들은 확장키로부터 취해진다. AES의 강점으로는 간단한 설계, 코드 압축 및 다양한 종류의 플랫폼, 컴퓨터 환경에서도 우수한 성능을 보여주고 메모리를 적게 차지해서 메모리 용량이 적은장치(USB, 스마트 카드 등)에도 손쉽게 쓸수 있고, 아직까지 알려진 보안공격이 없을 뿐 만 아니라 여러 가지 공격에 대해서 대응이 가능하다. 다양한 블록크기인 128, 192, 256 비트 등 독립적으로 선택 가능한 키 길이를 가지고, 기본적으로 128, 192, 256비트의 키 값을 가지나, 이론적으로는 키의 크기에 제한이 없다. 또한 키의 길이에 따라서 10회, 12회, 14회의 반복 라운드 연산을 수행하게 된다.



<그림 3> AES의 암·복호화 라운드 구조도

2.4 해쉬 함수(Hash Function)

해쉬 함수는 임의의 길이를 갖는 메시지를 입력 받아 고정된 길이의 해쉬값을 출력하는 함수이다. 암호 알고리즘에는 키가 사용되지만, 해쉬 함수는 키를 사용하지 않으므로 같은 입력에 대해서는 항상 같은 출력이 나오게 된다. 이러한 함수를 사용하는 목적은 입력 메시지에 대한 변경할 수 없는 증거값을 뽑아냄으로서 메시지의 오류나 변조를 탐지할 수 있는 무결성을 제공하는 목적으로 주로 사용된다. 해쉬 함수는 다양한 가변 길이의 입력에 적용 될 수 있어야 하며 안전한 해쉬 함수로 사용될 수 있기 위해서는 충돌을 찾아내기 어렵다는 특성을 가져야 한다. 대표적인 해쉬 함수로는 SHA(Secure Hash Algorithm), HAS-160 등이 있다. 최초의 해쉬 알고리즘은 최조의 알고리즘은 1993년 미국 국가 안전 보장국(NSA)이 설계하였으며, 미국 표준 기술 연구소(NIST)에 의해 SHS(Secure Hash Standard, FIPS PUB 180)으로 출판되었으며, 다른 함수들과 구별하기 위해 보통 SHA-0으로 불린다. 2년 후 SHA-0의 압축 함수에 비트 회전 연산을 하나 추가한 SHA-1(FIPS PUB 180-1)이 발표되었으며, 그 후에 4종류의 변형, 즉 SHA-224, SHA-256, SHA-384, SHA-512(FIPS PUB 180-2)가 추가로 발표되었다. 이들을 통칭해서 SHA-2라고 하기도 한다. SHA-1은 SHA 함수들 중 가장 많이 쓰이며, TLS, SSL, PGP, SSH, IPSec 등 많은 보안 프로토콜과 프로그램에서 사용되고 있다. 하지만 최근 SHA-0과 SHA-1에 대한 분석 결과가 발표됨에 따라 SHA-2를 사용할 것이 권장되고 있다. 또한 국내에서 개발된 대표적인 해쉬함수인 HAS-160[28]의 설계원리는 SHA-1의 설계 사상이 유사하지만, 메시지 확장 과정이 기존의 MD계열 해쉬 함수와는 차이가 있어, 최근 제안된 다양한 해쉬 함수 분석 기법에 대하여 아직까지는 안전성을 갖고 있다.



<그림 4> 해쉬 함수의 기본 구조

2.5 VHD(Virtual Hard Disk)

VHD란, 가상 하드 드라이브의 약자로, 파일 자체가 가상의 하드디스크 역할을 한다. 여기서 의미하는 가상 하드 디스크란 디스크 이미지 파일을 실제 물리 디스크처럼 사용하는 것을 말한다.

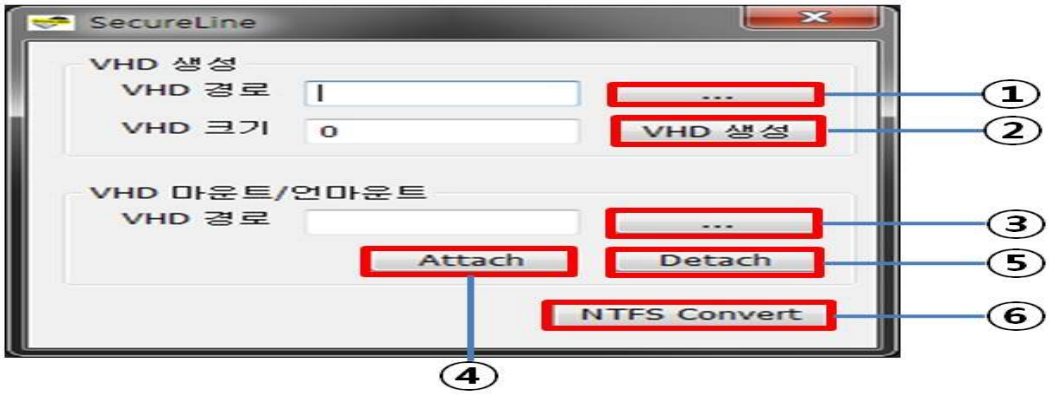
실제로는 존재하지 않는 디스크를 디스크 이미지 파일을 통하여 논리적으로 마치 실제 물리 디스크를 사용하는 것과 같은 효과를 낼 수 있다.

Windows 7 이후 버전에서 VHD를 만들고, 사용 할 수 있으며 일부 버전에서는 VHD 파일로 물리적 컴퓨터를 부팅 할 수 있다.

VHD를 이용하면 단순히 자료를 저장하는 것뿐만 아니라 운영체제 구동까지 가능하므로, VHD를 이용해 다양한 운영체제를 설치해 쓸 수 있고, 백업 및 복구도 간편하여 문제가 생겼을 때 대처도 쉽게 할 수 있다.

3. 연구 내용

3.1 VHD 생성 및 마운트 언마운트



<그림 5> VHD 생성 및 마운트 언마운트 UI

① VHD를 생성할 경로를 지정한다.

② VHD를 생성할 크기(MB)를 입력하고 생성버튼을 누른다.

③ VHD가 생성된 경로를 선택 한다.

④ Attach = VHD 파일을 마운트 시킨다.

⑤ Detach = VHD 파일을 언마운트 시킨다.

⑥ NTFS Convert = FAT32를 NTFS로 변환시켜 준다.

3.1.1 test-dlgDlg.cpp (VHD 생성 및 마운트 언마운트 구현 소스)

```
// testdlgDlg.cpp : 구현 파일
//

#include "stdafx.h"
#include "testdlg.h"
#include "testdlgDlg.h"
#include "afxdialogex.h"

#include <InitGuid.h>
#include <virtdisk.h>
#pragma comment(lib, "VirtDisk.lib")

#ifdef _DEBUG
#define new DEBUG_NEW
#endif


// 응용 프로그램 정보에 사용되는 CAboutDlg 대화 상자입니다.
```

```
class CAboutDlg : public CDialogEx
{
public:
	CAboutDlg();

// 대화 상자 데이터입니다.
	enum { IDD = IDD_ABOUTBOX };

	protected:
	virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV 지원입니다.

// 구현입니다.
protected:
	DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)
{
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialogEx)
END_MESSAGE_MAP()


// CtestdlgDlg 대화 상자




CtestdlgDlg::CtestdlgDlg(CWnd* pParent /*=NULL*/)
	: CDialogEx(CtestdlgDlg::IDD, pParent)
	, m_szFilename(_T(""))
	//, m_sizeInMB(_T(""))

	, m_szVhdname(_T(""))
{
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
	m_sizeInMB = 0;
}

void CtestdlgDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
	//  DDX_Control(pDX, IDC_VHD_NAME, m_szFilename);
	//  DDX_Control(pDX, IDC_VHD_MBSIZE, m_sizeInMB);
	//  DDX_Text(pDX, IDC_VHD_MBSIZE, m_sizeInMB);
	//  DDX_Text(pDX, IDC_VHD_NAME, m_szFilename);
```

```
	DDX_Text(pDX, IDC_EDIT_VHD_NAME01, m_szFilename);
	//  DDX_Text(pDX, IDC_VHD_MBSIZE, m_sizeInMB);
	DDX_Text(pDX, IDC_EDIT_VHD_MBSIZE, m_sizeInMB);
	DDX_Text(pDX, IDC_EDIT_VHD_NAME02, m_szVhdname);
}

BEGIN_MESSAGE_MAP(CtestdlgDlg, CDialogEx)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_CREATE_VHD, &CtestdlgDlg::OnBnClickedCreateVhd)
	ON_BN_CLICKED(IDC_BUTTON_VHD_ATTACH, &CtestdlgDlg::OnBnClickedButtonVhdAttach)
	ON_BN_CLICKED(IDC_BUTTON_VHD_DETACH, &CtestdlgDlg::OnBnClickedButtonVhdDetach)
	ON_BN_CLICKED(IDC_BUTTON_CREATE_VHD_FILEPATH,
&CtestdlgDlg::OnBnClickedButtonCreateVhdFilepath)
	ON_BN_CLICKED(IDC_BUTTON_MOUNT_VHD_FILEPATH,
&CtestdlgDlg::OnBnClickedButtonMountVhdFilepath)
END_MESSAGE_MAP()


// CtestdlgDlg 메시지 처리기

BOOL CtestdlgDlg::OnInitDialog()
{
	CDialogEx::OnInitDialog();

	// 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.

	// IDM_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		BOOL bNameValid;
		CString strAboutMenu;
		bNameValid = strAboutMenu.LoadString(IDS_ABOUTBOX);
		ASSERT(bNameValid);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
		}
	}

	// 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는
	//  프레임워크가 이 작업을 자동으로 수행합니다.
	SetIcon(m_hIcon, TRUE);			// 큰 아이콘을 설정합니다.
	SetIcon(m_hIcon, FALSE);		// 작은 아이콘을 설정합니다.

	// TODO: 여기에 추가 초기화 작업을 추가합니다.
```

```
	return TRUE;  // 포커스를 컨트롤에 설정하지 않으면 TRUE를 반환합니다.
}

void CtestdlgDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
	if ((nID & 0xFFF0) == IDM_ABOUTBOX)
	{
		CAboutDlg dlgAbout;
		dlgAbout.DoModal();
	}
	else
	{
		CDialogEx::OnSysCommand(nID, lParam);
	}
}

// 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면
//  아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는
//  프레임워크에서 이 작업을 자동으로 수행합니다.

void CtestdlgDlg::OnPaint()
{
	if (IsIconic())
	{
		CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.

		SendMessage(WM_ICONERASEBKGND, reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);

		// 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.
		int cxIcon = GetSystemMetrics(SM_CXICON);
		int cyIcon = GetSystemMetrics(SM_CYICON);
		CRect rect;
		GetClientRect(&rect);
		int x = (rect.Width() - cxIcon + 1) / 2;
		int y = (rect.Height() - cyIcon + 1) / 2;

		// 아이콘을 그립니다.
		dc.DrawIcon(x, y, m_hIcon);
	}
	else
	{
		CDialogEx::OnPaint();
	}
}

// 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서
//  이 함수를 호출합니다.
HCURSOR CtestdlgDlg::OnQueryDragIcon()
{
	return static_cast<HCURSOR>(m_hIcon);
}
```

```
void CtestdlgDlg::OnBnClickedCreateVhd()
{
	// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
	 BOOL bRet = FALSE;
    HANDLE hvhd;
    CREATE_VIRTUAL_DISK_PARAMETERS  params;
    VIRTUAL_DISK_ACCESS_MASK          mask;
    VIRTUAL_STORAGE_TYPE              vst =
    {
        VIRTUAL_STORAGE_TYPE_DEVICE_VHD,
        VIRTUAL_STORAGE_TYPE_VENDOR_MICROSOFT
    };

        UpdateData(TRUE);
        PCWSTR szFilename = (PCWSTR)m_szFilename;
        //ULONG sizeInMB = (ULONG)atol((char*)(LPCTSTR)m_sizeInMB);
        ULONG sizeInMB = (ULONG)m_sizeInMB;

        CString msg1;

    //wprintf(L"CreateVHD_Fixed %s, size (MB) %d\n", szFilename, sizeInMB);
        msg1.Format(_T("CreateVHD_Fixed %s, size (MB) %d\n"), szFilename, sizeInMB);
        AfxMessageBox(msg1, MB_OK);

    params.Version1.UniqueId              = GUID_NULL;
    params.Version1.BlockSizeInBytes      = 0;
    params.Version1.MaximumSize                   = sizeInMB * 1024 * 1024;
    params.Version1.ParentPath                    = NULL;
    params.Version1.SourcePath             = NULL;
    params.Version1.SectorSizeInBytes     = 512;
    params.Version                                          = CREATE_VIRTUAL_DISK_VERSION_1;
    mask                                  = VIRTUAL_DISK_ACCESS_CREATE;

    DWORD ret = CreateVirtualDisk(&vst,
        szFilename,
        mask,
        NULL,
        // To create a dynamic disk, use CREATE_VIRTUAL_DISK_FLAG_NONE instead.
        CREATE_VIRTUAL_DISK_FLAG_FULL_PHYSICAL_ALLOCATION,
        0,
        &params,
        NULL,
        &hvhd);

    if (ret == ERROR_SUCCESS)
    {
        bRet = TRUE;
                    msg1.Format(_T("%s를 생성하였습니다."), szFilename);
                    AfxMessageBox(msg1, MB_OK);
```

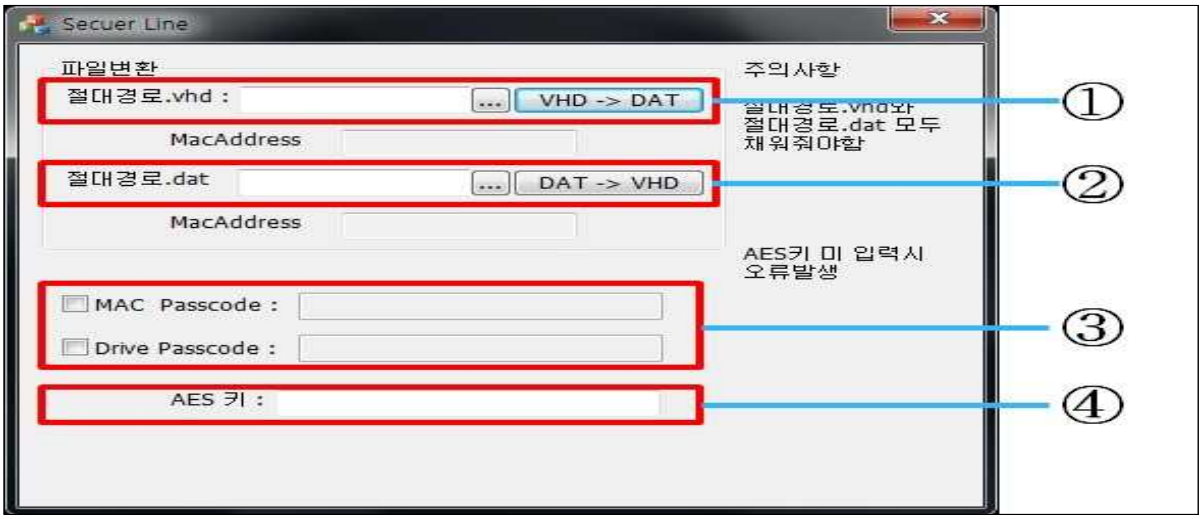
```
    }
    else
    {
        bRet = FALSE;
        //printf("failed to create vdisk...err 0x%x\n", ret);
       // PrintErrorMessage(GetLastError());
                    msg1.Format(_T("failed to create vdisk...err 0x%x\n"), ret);
                    AfxMessageBox(msg1, MB_OK);
    }

    if (INVALID_HANDLE_VALUE != hvhd)
    {
        CloseHandle(hvhd);
    }

    //return bRet;
}

void CtestdlgDlg::OnBnClickedButtonCreateVhdFilepath()
{
          // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

          CString FilePath, FolderPath, FileName;

          CFileDialog CreateVHD(false, NULL, _T("*.vhd"), OFN_HIDEREADONLY | OFN_OVERWRITEPROMPT,
NULL, NULL);

          if(CreateVHD.DoModal() == IDOK)
          {
                    FolderPath = CreateVHD.GetFolderPath();
                    FileName = CreateVHD.GetFileName();

                    FilePath.Format(_T("%s\\%s"), FolderPath, FileName);

                    m_szFilename = FilePath;
                    UpdateData(FALSE);
          }
}

void CtestdlgDlg::OnBnClickedButtonVhdAttach()
{
          // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
          BOOL bRet = FALSE;
    HANDLE hVhd = INVALID_HANDLE_VALUE;
    DWORD ret;
    OPEN_VIRTUAL_DISK_PARAMETERS oparams;
    ATTACH_VIRTUAL_DISK_PARAMETERS iparams;
    VIRTUAL_STORAGE_TYPE              vst =
    {
        VIRTUAL_STORAGE_TYPE_DEVICE_VHD,
        VIRTUAL_STORAGE_TYPE_VENDOR_MICROSOFT
    };
```

```
        UpdateData(TRUE);
        PCWSTR pszVhdPath = (PCWSTR)m_szVhdname;

        CString msg1;

    //wprintf(L"OpenAndAttachVHD %s\n", pszVhdPath);
        msg1.Format(_T("OpenAndAttachVHD %s\n"), pszVhdPath);
        AfxMessageBox(msg1, MB_OK);

    oparams.Version = OPEN_VIRTUAL_DISK_VERSION_1;
    oparams.Version1.RWDepth = OPEN_VIRTUAL_DISK_RW_DEPTH_DEFAULT;

    iparams.Version = ATTACH_VIRTUAL_DISK_VERSION_1;

    ret = OpenVirtualDisk(&vst, pszVhdPath,
        VIRTUAL_DISK_ACCESS_ATTACH_RW | VIRTUAL_DISK_ACCESS_GET_INFO |
VIRTUAL_DISK_ACCESS_DETACH,
        OPEN_VIRTUAL_DISK_FLAG_NONE, &oparams, &hVhd);

    if (ERROR_SUCCESS == ret)
    {
        //printf("success opening vdisk...\n");
                msg1.Format(_T("success opening vdisk...\n"));
                AfxMessageBox(msg1, MB_OK);
        ret = AttachVirtualDisk(hVhd, NULL,
            ATTACH_VIRTUAL_DISK_FLAG_PERMANENT_LIFETIME, 0, &iparams, NULL);

        if (ERROR_SUCCESS == ret)
        {
            //printf("success attaching vdisk...\n");
                        msg1.Format(_T("success attaching vdisk...\n"));
                        AfxMessageBox(msg1, MB_OK);
        }
        else
        {
            //printf("failed to attach vdisk...err 0x%x\n", ret);
                        msg1.Format(_T("failed to attach vdisk...err 0x%x\n"), ret);
                        AfxMessageBox(msg1, MB_OK);
            //PrintErrorMessage(GetLastError());
            bRet = FALSE;
        }
    }
    else
    {
        //printf("failed to open vdisk...err 0x%x\n", ret);
                msg1.Format(_T("failed to open vdisk...err 0x%x\n"), ret);
                AfxMessageBox(msg1, MB_OK);
        //PrintErrorMessage(GetLastError());
        bRet = FALSE;
    }
```

```
    if (INVALID_HANDLE_VALUE != hVhd)
    {
        CloseHandle(hVhd);
    }

    //return bRet;
}

void CtestdlgDlg::OnBnClickedButtonMountVhdFilepath()
{
        // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
        CString FilePath, FolderPath, FileName;

        TCHAR Filter[] = _T("VHD File(*.vhd) |*.vhd| 모든파일(*.*) |*.*|");

        CFileDialog MountVHD(true, NULL, _T("*.vhd"), OFN_HIDEREADONLY | OFN_FILEMUSTEXIST,
Filter, NULL);

        if(MountVHD.DoModal() == IDOK)
        {
                FolderPath = MountVHD.GetFolderPath();
                FileName = MountVHD.GetFileName();

                FilePath.Format(_T("%s\\%s"), FolderPath, FileName);

                m_szVhdname = FilePath;
                UpdateData(FALSE);
        }
}

void CtestdlgDlg::OnBnClickedButtonVhdDetach()
{
        // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
        BOOL bRet = FALSE;
    DWORD ret;
    DETACH_VIRTUAL_DISK_FLAG Flags;
    HANDLE hVhd = INVALID_HANDLE_VALUE;
    OPEN_VIRTUAL_DISK_PARAMETERS oparams;
    VIRTUAL_STORAGE_TYPE            vst =
    {
        VIRTUAL_STORAGE_TYPE_DEVICE_VHD,
        VIRTUAL_STORAGE_TYPE_VENDOR_MICROSOFT
    };

        UpdateData(TRUE);
        PCWSTR pszVhdPath = (PCWSTR)m_szVhdname;

        CString msg1;

    //wprintf(L"OpenAndDetachVHD %s\n", pszVhdPath);
        msg1.Format(_T("OpenAndDetachVHD %s\n"), pszVhdPath);
        AfxMessageBox(msg1, MB_OK);
```

```
    oparams.Version = OPEN_VIRTUAL_DISK_VERSION_1;
    oparams.Version1.RWDepth = OPEN_VIRTUAL_DISK_RW_DEPTH_DEFAULT;

    ret = OpenVirtualDisk(&vst, pszVhdPath,
        VIRTUAL_DISK_ACCESS_DETACH,
        OPEN_VIRTUAL_DISK_FLAG_NONE, NULL /*&oparams*/, &hVhd);

    if (ERROR_SUCCESS == ret)
    {
        //printf("success opening vdisk...\n");
                    msg1.Format(_T("success opening vdisk...\n"));
                    AfxMessageBox(msg1, MB_OK);
        Flags = DETACH_VIRTUAL_DISK_FLAG_NONE;
        ret = DetachVirtualDisk(hVhd, Flags, 0);
        if (ERROR_SUCCESS == ret)
        {
            //printf("success detaching vdisk...\n");
                            msg1.Format(_T("success detaching vdisk...\n"));
                            AfxMessageBox(msg1, MB_OK);
        }
        else
        {
            //printf("failed to detach vdisk... %d\n", ret);
                            msg1.Format(_T("failed to detach vdisk... %d\n"), ret);
                            AfxMessageBox(msg1, MB_OK);
            //PrintErrorMessage(GetLastError());
            bRet = FALSE;
        }
    }
    else
    {
        //printf("failed to open vdisk...err %d\n", ret);
                    msg1.Format(_T("failed to open vdisk...err %d\n"), ret);
                    AfxMessageBox(msg1, MB_OK);
        //PrintErrorMessage(GetLastError());
        bRet = FALSE;
    }

    if (INVALID_HANDLE_VALUE != hVhd)
    {
        CloseHandle(hVhd);
    }

    //return bRet;
}
```

3.2 암·복호화 및 Secure Line 인증방식



① VHD 파일이 생성된 경로를 지정 한 후 VHD -> DAT 버튼을 클릭한다. (암호화한 PC의 MAC주소를 첨부하여 VHD 파일을 암호화한 DAT파일을 생성)

② VHD 파일이 생성된 경로를 지정 한 후 DAT -> VHD 버튼을 클릭한다. (암호화된 DAT파일을 복호화 하고 암호화한 PC의 MAC주소를 확인한다.)

③ 비인증 PC에서 사용이 필요한 경우 체크를 하고 Passcode를 설정한다.

※ MAC Passcode : MAC주소 인증 필요 없이 비인증PC에서 사용이 필요한 경우 설정한 Passcode를 입력하면, 인증과정을 거치지 않고 사용이 가능하다.

※ Drive Passcode : 하드웨어 ID 인증 필요 없이 비인증PC 사용이 필요한 경우 설정한 Passcode를 입력하면, 인증과정을 거치지 않고 사용이 가능하다.

④ AES 키 : 사용자 본인임을 증명하기 위한 수단으로, 최초로 설정할 때 필히 지정해야한다. 지정하지 않을 경우 진행이 되지 않는다.

3.2.2 test-rwdlgDlg.cpp (암·복호화 및 Secure Line 인증방식 구현 소스)

```
// test-rwdlgDlg.cpp : 구현 파일
//

#include "stdafx.h"
#include "test-rwdlg.h"
#include "test-rwdlgDlg.h"
#include "afxdialogex.h"

#include "Rijndael.h"
#include "KISA_SHA256.hpp"

#include <IPHlpApi.h>
#pragma comment(lib, "IPHlpApi.lib")



#ifdef _DEBUG
#define new DEBUG_NEW
#endif


// 응용 프로그램 정보에 사용되는 CAboutDlg 대화 상자입니다.

class CAboutDlg : public CDialogEx
{
public:
	CAboutDlg();

// 대화 상자 데이터입니다.
	enum { IDD = IDD_ABOUTBOX };

	protected:
	virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV 지원입니다.

// 구현입니다.
protected:
	DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)
{
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
}
```

```
BEGIN_MESSAGE_MAP(CAboutDlg, CDialogEx)
END_MESSAGE_MAP()


// CtestrwdlgDlg 대화 상자




CtestrwdlgDlg::CtestrwdlgDlg(CWnd* pParent /*=NULL*/)
	: CDialogEx(CtestrwdlgDlg::IDD, pParent)
	, m_strVhdName(_T(""))
	, m_strDatName(_T(""))
	, m_strAMAC(_T(""))
	, m_strBMAC(_T(""))
	, m_strKey(_T(""))
{
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
	m_bChecked[0]=m_bChecked[1]=FALSE;
	m_TypeCheck=0;
	//  m_strMacPasscode = _T("");
	m_strDrivePasscode = _T("");
	m_strMacPasscode = _T("");
}

void CtestrwdlgDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialogEx::DoDataExchange(pDX);
	DDX_Text(pDX, IDC_EDIT_VHDNAME, m_strVhdName);
	DDX_Text(pDX, IDC_EDIT_DATNAME, m_strDatName);
	DDX_Text(pDX, IDC_EDIT_AMAC, m_strAMAC);
	DDX_Text(pDX, IDC_EDIT_BMAC, m_strBMAC);
	DDX_Text(pDX, IDC_EDIT_AESKEY, m_strKey);
	//  DDX_Text(pDX, IDC_EDIT_DRIVEPASSCODE, m_strMacPasscode);
	DDX_Text(pDX, IDC_EDIT_DRIVEPASSCODE, m_strDrivePasscode);
	DDX_Text(pDX, IDC_EDIT_MACPASSCODE, m_strMacPasscode);
}

BEGIN_MESSAGE_MAP(CtestrwdlgDlg, CDialogEx)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_BUTTON_VTOD, &CtestrwdlgDlg::OnBnClickedButtonVtod)
	ON_BN_CLICKED(IDC_BUTTON_DTOV, &CtestrwdlgDlg::OnBnClickedButtonDtov)
	ON_BN_CLICKED(IDC_BUTTON_VHD_FILEPATH, &CtestrwdlgDlg::OnBnClickedButtonVhdFilepath)
	ON_BN_CLICKED(IDC_BUTTON_DAT_FILEPATH, &CtestrwdlgDlg::OnBnClickedButtonDatFilepath)
	ON_BN_CLICKED(IDC_CHECK1, &CtestrwdlgDlg::OnBnClickedCheck1)
	ON_BN_CLICKED(IDC_CHECK2, &CtestrwdlgDlg::OnBnClickedCheck2)
	ON_EN_CHANGE(IDC_EDIT_MACPASSCODE, &CtestrwdlgDlg::OnEnChangeEditMacpasscode)
	ON_EN_CHANGE(IDC_EDIT_DRIVEPASSCODE, &CtestrwdlgDlg::OnEnChangeEditDrivepasscode)
	ON_EN_CHANGE(IDC_EDIT_AESKEY, &CtestrwdlgDlg::OnEnChangeEditAeskey)
```

```
END_MESSAGE_MAP()


// CtestrwdlgDlg 메시지 처리기

BOOL CtestrwdlgDlg::OnInitDialog()
{
	CDialogEx::OnInitDialog();

	// 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.

	// IDM_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		BOOL bNameValid;
		CString strAboutMenu;
		bNameValid = strAboutMenu.LoadString(IDS_ABOUTBOX);
		ASSERT(bNameValid);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
		}
	}

	// 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는
	//  프레임워크가 이 작업을 자동으로 수행합니다.
	SetIcon(m_hIcon, TRUE);			// 큰 아이콘을 설정합니다.
	SetIcon(m_hIcon, FALSE);		// 작은 아이콘을 설정합니다.

	// TODO: 여기에 추가 초기화 작업을 추가합니다.

	return TRUE;  // 포커스를 컨트롤에 설정하지 않으면 TRUE를 반환합니다.
}

void CtestrwdlgDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
	if ((nID & 0xFFF0) == IDM_ABOUTBOX)
	{
		CAboutDlg dlgAbout;
		dlgAbout.DoModal();
	}
	else
	{
		CDialogEx::OnSysCommand(nID, lParam);
	}
}
```

```
// 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면
//  아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는
//  프레임워크에서 이 작업을 자동으로 수행합니다.

void CtestrwdlgDlg::OnPaint()
{
	if (IsIconic())
	{
		CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.

		SendMessage(WM_ICONERASEBKGND, reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);

		// 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.
		int cxIcon = GetSystemMetrics(SM_CXICON);
		int cyIcon = GetSystemMetrics(SM_CYICON);
		CRect rect;
		GetClientRect(&rect);
		int x = (rect.Width() - cxIcon + 1) / 2;
		int y = (rect.Height() - cyIcon + 1) / 2;

		// 아이콘을 그립니다.
		dc.DrawIcon(x, y, m_hIcon);
	}
	else
	{
		CDialogEx::OnPaint();
	}
}

// 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서
//  이 함수를 호출합니다.
HCURSOR CtestrwdlgDlg::OnQueryDragIcon()
{
	return static_cast<HCURSOR>(m_hIcon);
}

/*************************************************************************************************
*****************************/
/*
/*		MAC주소 탐색 함수
/*                                                              컴퓨터의 MAC Address를 CString로 출력
/*************************************************************************************************
*****************************/
CString CtestrwdlgDlg::GetMacAddress(void)
{
	CString strMacAddress = _T("000000000000");
	IP_ADAPTER_INFO AdapterInfo[5];
	DWORD dwBufLen = sizeof(AdapterInfo);

	DWORD dwStatus = GetAdaptersInfo( AdapterInfo, &dwBufLen);

	if(dwStatus != ERROR_SUCCESS)
```

```
	{
		strMacAddress.Format(_T("Error : %d"), dwStatus);
		AfxMessageBox(strMacAddress, MB_OK);
	}
	PIP_ADAPTER_INFO pAdapterInfo = AdapterInfo;
	strMacAddress.Format(_T("%02X%02X%02X%02X%02X%02X"), pAdapterInfo->Address[0],
pAdapterInfo->Address[1], pAdapterInfo->Address[2], pAdapterInfo->Address[3], pAdapterInfo->Address[4],
pAdapterInfo->Address[5]);
	//AfxMessageBox(strMacAddress, MB_OK);

	//int n; CString str;
	//n=strMacAddress.GetLength();
	//str.Format(_T("strMacAddress의 길이 = %d"), n);
	//AfxMessageBox(str, MB_OK);

	return strMacAddress;
}

/*************************************************************************************************
*****************************/
/*
/*		TextHash
/*		CString 입력값 입력시 CString형식의 해시값 출력
/*************************************************************************************************
*****************************/
CString CtestrwdlgDlg::GetTextHash(CString IN_msg)
{
	int sz_HashBefore = IN_msg.GetLength();
	CString OUT_msg;  OUT_msg.Empty();

	char *input, output[32];
//	input = (char*)calloc(sz_HashBefore, sizeof(char));
//	input = (LPSTR)IN_msg.GetBuffer();
	input = (LPSTR)IN_msg.GetBuffer();

	/**/ //메시지박스 입력값과 그 길이
/*	CString tmp00;	tmp00.Format(_T("%d : %s"), sz_HashBefore, input);
	AfxMessageBox(tmp00, MB_OK);
	/**/


	SHA2_256 test;
	test.SHA256_Encrpyt(/*(const BYTE*)(LPCSTR)m_strHashBefore.GetBuffer()*/(const BYTE*)input,
/*sizeof(m_strHashBefore)*/sz_HashBefore, /*(BYTE*)(LPSTR)(LPCTSTR)m_strHashAfter*/(BYTE*)output);

	OUT_msg.Format(_T("%s"),output);


/****/	/********해시값 HEX로 재정렬********/
/*
	CString HashToTCHAR = m_strHashAfter.Left(16);
```

```
        CString returnvalue;            returnvalue.Empty();
    for (int x = 0; x < HashToTCHAR.GetLength(); x++)
     {
        CString temporary;
        int value = (int)(HashToTCHAR[x]);
        temporary.Format(_T("%04X"), value);
        //returnvalue += temporary;
                returnvalue.Format(_T("%s%s"), returnvalue, temporary);
     }

        m_strHashAfter = returnvalue;
/****/
        //m_strHashAfter = HashToTCHAR;

        /**/ //메시지박스 output값과 m_strHashAfter값과 값들의 길이
/*      int size = m_strHashAfter.GetLength();//m_strHashAfter.GetLength();
        CString tmp;        tmp.Format(_T("%d : %s\n%d : %s"), size, output, size, m_strHashAfter);
        AfxMessageBox(tmp, MB_OK);
        /**/
/**********/

/***/    /********해시값 길이와 내용 msgBox로 보는 코드********/
/*
        int s1, s2;                 s1=m_strHashAfter.GetLength();          s2=sizeof(output);
        CString msg;        msg.Format(_T("%d : %s\n%d : %s"),s1, m_strHashAfter, s2, output);
        AfxMessageBox(msg, MB_OK);

/***/

/*****/  /********해시값 txt파일로 출력실험하는 코드********/
/*
        CFile CFtest;

        CFtest.Open(_T("C:\\Users\\Administrator\\Desktop\\CFtest.txt"),        CFile::modeCreate        |
CFile::modeWrite);

        CFtest.Write(m_strHashAfter,16);
        CFtest.Write(m_strHashAfter,32);
        CFtest.Write(m_strHashAfter,64);
        CFtest.Write(m_strHashAfter,128);
        //CFtest.Write(m_strHashAfter,256);


        CFtest.Close();
/*****/
/**********/

        return OUT_msg;
}

/*************************************************************************************************
****************************/
```

```
/*
/*                  Get VolumSerial
/*                                                                  CString FilePath 입력시 해당파일이 있는
Volum의 SerialNumber CString로 출력
/*****************************************************************************************************
*****************************/
CString CtestrwdlgDlg::GetVolumSerial(CString IN_FilePath)
{
          CString DriveName, FileSystem;
          CString DriveVolum;                                       DriveVolum=IN_FilePath.Left(3);
          DWORD Serial=0;

          GetVolumeInformation(DriveVolum, (LPTSTR)(LPCTSTR)DriveName, 1024, (LPDWORD)Serial, NULL,
NULL, (LPTSTR)(LPCTSTR)FileSystem, 1024);

          return (CString)(LPCTSTR)Serial;
}

/*****************************************************************************************************
*****************************/
/*
/*                  VHD파일 경로
/*
/*****************************************************************************************************
*****************************/
void CtestrwdlgDlg::OnBnClickedButtonVhdFilepath()
{
          // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
          CString VhdFilePath, DatFilePath, FolderPath, FileName;

          CFileDialog GetVhdName(true, NULL, _T("*.vhd"), OFN_HIDEREADONLY | OFN_FILEMUSTEXIST,
NULL, NULL);

          if(GetVhdName.DoModal() == IDOK)
          {
                    FolderPath = GetVhdName.GetFolderPath();
                    FileName = GetVhdName.GetFileTitle();

                    VhdFilePath.Format(_T("%s\\%s.vhd"), FolderPath, FileName);
                    DatFilePath.Format(_T("%s\\%s.dat"), FolderPath, FileName);

                    m_strVhdName = VhdFilePath;
                    m_strDatName = DatFilePath;

                    UpdateData(FALSE);
          }
}

/*****************************************************************************************************
*****************************/
/*
/*                  VHD파일 -> DAT파일
```

```
/*
/*****************************************************************************************************
*****************************/
void CtestrwdlgDlg::OnBnClickedButtonVtod()
{
	// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
	char buffer_in[1024], buffer_out[1024];
	int ReadByte;
	CString strMacAddress, strVolumSerial, AESKey;
	CString Mac_Hash, VS_Hash, SK_Hash, MacPass_Hash, VSPass_Hash;

	CString TypeCheck;
	TypeCheck.Format(_T("%d"),m_TypeCheck);


	UpdateData(TRUE);



	//Read : opfile, Write : clfile 파일
	CFile opfile, clfile;

	opfile.Open(m_strVhdName, CFile::modeRead);
	clfile.Open(m_strDatName, CFile::modeCreate | CFile::modeWrite);

	//AES 객체 & 키
	CRijndael oRijndael;
	oRijndael.MakeKey(LPCSTR(LPCTSTR(m_strKey.GetBuffer())),     "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0",
32, 32);


	/******/ //헤더정보 작성
                                                                                    /**/
/**/	//HW인증정보(MAC Address, VolumSerial)읽기
                                                        /**/
	strMacAddress	= CtestrwdlgDlg::GetMacAddress();
//MacAddress 탐지		/**/
	strVolumSerial	= CtestrwdlgDlg::GetVolumSerial(m_strVhdName);  //VolumSerial 읽기
	/**/	//heap corruption detected에러발생
	AESKey			= m_strKey;
                                //암복호화 키
/**/

		/**/
/***/	//HW인증정보 Hash
                                                                                    /**/
	Mac_Hash= CtestrwdlgDlg::GetTextHash(strMacAddress);                //MACAddress 해시
	/**/
	VS_Hash		= CtestrwdlgDlg::GetTextHash(strVolumSerial);                //VolumSerial 해시
		/**/
	SK_Hash		= CtestrwdlgDlg::GetTextHash(AESKey);                                    //AES
Key 해시		/**/
```

```
/***/
        /******/


/******/ //MacAddress 헤더 입력
        /**/
        //clfile.Write(strMacAddress, 24);
        /**/
        clfile.Write(TypeCheck, 2);      //AfxMessageBox(TypeCheck, MB_OK);
        clfile.Write(Mac_Hash, 32);
                /**/
        if((int)m_TypeCheck==2 || (int)m_TypeCheck==3)   //MacPasscode 사용할 경우
        {
                MacPass_Hash = CtestrwdlgDlg::GetTextHash(m_strMacPasscode);
                clfile.Write(MacPass_Hash, 32);
        }
        clfile.Write(VS_Hash, 32);
                /**/
        if((int)m_TypeCheck==1 || (int)m_TypeCheck==3)   //DrivePasscode 사용할 경우
        {
                VSPass_Hash = CtestrwdlgDlg::GetTextHash(m_strDrivePasscode);
                clfile.Write(VSPass_Hash, 32);
        }
        clfile.Write(SK_Hash, 32);
                /**/
/******/
                                        /**/
        m_strAMAC = strMacAddress;
        UpdateData(FALSE);

        //파일 읽고 쓰기
        while(1)
        {
                if((ReadByte = opfile.Read(buffer_in, sizeof(buffer_in))) == 0) break;

                oRijndael.Encrypt(buffer_in, buffer_out, sizeof(buffer_in), 1);

                clfile.Write(buffer_out, ReadByte);
                //clfile.Write(buffer_in, ReadByte);
        }

        AfxMessageBox(_T("Success"), MB_OK);

        opfile.Close();
        clfile.Close();
}

/*************************************************************************************************
*****************************/
/*
/*              DAT파일 경로
/*
```

```
/*****************************************************************************************************
*****************************/
void CtestrwdlgDlg::OnBnClickedButtonDatFilepath()
{
	// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
	CString VhdFilePath, DatFilePath, FolderPath, FileName;

	CFileDialog GetDatName(true, NULL, _T("*.dat"), OFN_HIDEREADONLY | OFN_FILEMUSTEXIST,
NULL, NULL);

	if(GetDatName.DoModal() == IDOK)
	{
		FolderPath = GetDatName.GetFolderPath();
		FileName = GetDatName.GetFileTitle();

		VhdFilePath.Format(_T("%s\\%s.vhd"), FolderPath, FileName);
		DatFilePath.Format(_T("%s\\%s.dat"), FolderPath, FileName);

		m_strVhdName = VhdFilePath;
		m_strDatName = DatFilePath;

		UpdateData(FALSE);
	}
}

/*****************************************************************************************************
*****************************/
/*
/*		DAT파일 -> VHD파일
/*
/*****************************************************************************************************
*****************************/
void CtestrwdlgDlg::OnBnClickedButtonDtov()
{
	// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
	char buffer_in[1024], buffer_out[1024];
	TCHAR ReadBuffer[32];
	int ReadByte;
	CString strMacAddress, strVolumSerial, AESKey;		//인증을 위한 시스템값
	CString Mac_Hash, VS_Hash, SK_Hash;					//시스템값을 해시한
값
	CString MacPass_Hash, VSPass_Hash;					//Passcode 해시값
	CString	HeaderGetMac, HeaderGetVS, HeaderGetSK;		//헤더에서 읽은 해시값
	CString	HeaderGetMacPasscode, HeaderGetVSPasscode;	//헤더에서 읽은 Passcode
	CString TypeCheck;
	int		nTypeCheck;

	CString msg;

	UpdateData(TRUE);
```

```
        //입출력 파일
        CFile opfile, clfile;

        opfile.Open(m_strDatName, CFile::modeRead);
        clfile.Open(m_strVhdName, CFile::modeCreate | CFile::modeWrite);

        //AES 객체 & 키 생성
        CRijndael oRijndael;
        oRijndael.MakeKey(LPCSTR(LPCTSTR(m_strKey.GetBuffer())),    "\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0",
32, 32);


        /******/ //헤더 비교정보 작성
/**/    //HW인증정보(MAC Address, VolumSerial)읽기
        strMacAddress       = CtestrwdlgDlg::GetMacAddress();
//MacAddress 탐지
        strVolumSerial      = CtestrwdlgDlg::GetVolumSerial(m_strVhdName);  //VolumSerial 읽기
        AESKey                      = m_strKey;
/**/
/***/   //HW인증 비교정보 Hash
        Mac_Hash= CtestrwdlgDlg::GetTextHash(strMacAddress);                //MACAddress 해시
        Mac_Hash= Mac_Hash.Left(16);
        VS_Hash             = CtestrwdlgDlg::GetTextHash(strVolumSerial);               //VolumSerial 해시
        VS_Hash             = VS_Hash.Left(16);
        SK_Hash             = CtestrwdlgDlg::GetTextHash(AESKey);                               //AES
Key 해시
        SK_Hash             = SK_Hash.Left(16);
                                                                            /**/

/***/
        /******/


//TypeCheck                 (2 || 3) : MacPasscode 사용 타입
//                                  (1 || 3) : DrivePasscode(or VSPasscode)사용타입
        opfile.Read(ReadBuffer, 2);
        TypeCheck = (LPCTSTR)ReadBuffer;
        TypeCheck = TypeCheck.Left(1);

//      msg.Format(_T("%d"),TypeCheck);
//      AfxMessageBox(msg,MB_OK);
//      msg.Format(_T("%s"),TypeCheck);
//      AfxMessageBox(msg,MB_OK);

        nTypeCheck = _ttoi(TypeCheck);
//      msg.Format(_T("%d"),nTypeCheck);
//      AfxMessageBox(msg,MB_OK);

//Mac주소 비교

        opfile.Read(ReadBuffer, 32);
        HeaderGetMac = (LPCTSTR)ReadBuffer;                     //헤더에서 MacAddress 읽음
```

```
		HeaderGetMac = HeaderGetMac.Left(16);

		if(nTypeCheck==2 || nTypeCheck==3)	//MacPasscode 사용할 경우
		{
			opfile.Read(ReadBuffer, 32);
			HeaderGetMacPasscode = (LPCTSTR)ReadBuffer;
			HeaderGetMacPasscode = HeaderGetMacPasscode.Left(16);

			MacPass_Hash = CtestrwdlgDlg::GetTextHash(m_strMacPasscode);
		}

		m_strBMAC = HeaderGetMac.Left(16);
		UpdateData(FALSE);

		if(strcmp((LPCSTR)(LPCTSTR)HeaderGetMac.GetBuffer(), (LPCSTR)(LPCTSTR)Mac_Hash.GetBuffer()) !=
0)			//추출한 Mac와 헤더에서 읽은 Mac을 검증
		{
			msg.Format(_T("MacAddress Mismatch\nMac1 = %s\nMac2 = %s"), HeaderGetMac,
Mac_Hash);
			AfxMessageBox(msg, MB_OK);

			if(nTypeCheck==2 || nTypeCheck==3)	//MacPasscode 사용할 경우
			{
				if(strcmp((LPCSTR)(LPCTSTR)HeaderGetMacPasscode.GetBuffer(),
(LPCSTR)(LPCTSTR)MacPass_Hash.GetBuffer()) != 0)
				{
					msg.Format(_T("MacAddress Passcode Mismatch\nMac1 = %s\nMac2
= %s"), HeaderGetMacPasscode, MacPass_Hash);
					AfxMessageBox(msg, MB_OK);
					goto Fail; //패스코드를 사용하지만 불일치
				}
			}
			else
			{
				goto Fail; //패스코드를 사용하지 않음
			}
		}


//VS비교
		opfile.Read(ReadBuffer, 32);
		HeaderGetVS = (LPCTSTR)ReadBuffer;				//헤더에서 VolumSerial 읽음
		HeaderGetVS = HeaderGetVS.Left(16);

		if(nTypeCheck==1 || nTypeCheck==3)	//DrivePasscode 사용할 경우
		{
			opfile.Read(ReadBuffer, 32);
			HeaderGetVSPasscode = (LPCTSTR)ReadBuffer;
			HeaderGetVSPasscode = HeaderGetVSPasscode.Left(16);

			VSPass_Hash = CtestrwdlgDlg::GetTextHash(m_strDrivePasscode);
		}
```

```
	if(strcmp((LPCSTR)(LPCTSTR)HeaderGetVS.GetBuffer(), (LPCSTR)(LPCTSTR)VS_Hash.GetBuffer()) != 0)
				//VolumSerial 비교
	{
		msg.Format(_T("VolumSerial Mismatch\nMac1 = %s\nMac2 = %s"), HeaderGetVS,
VS_Hash);
		AfxMessageBox(msg, MB_OK);

		if(nTypeCheck==1 || nTypeCheck==3)	//DrivePasscode 사용할 경우
		{
			if(strcmp((LPCSTR)(LPCTSTR)HeaderGetVSPasscode.GetBuffer(),
(LPCSTR)(LPCTSTR)VSPass_Hash.GetBuffer()) != 0)
			{
				msg.Format(_T("VolumSerial Passcode Mismatch\nMac1 = %s\nMac2
= %s"), HeaderGetVSPasscode, VSPass_Hash);
				AfxMessageBox(msg, MB_OK);
				goto Fail; //패스코드를 사용하지만 불일치
			}
		}
		else
		{
			goto Fail; //패스코드를 사용안함
		}
	}


//PW 비교
	opfile.Read(ReadBuffer, 32);
	HeaderGetSK = (LPCTSTR)ReadBuffer;				//헤더에서 PassWord 읽음
	HeaderGetSK = HeaderGetSK.Left(16);

	if(strcmp((LPCSTR)(LPCTSTR)HeaderGetSK.GetBuffer(), (LPCSTR)(LPCTSTR)SK_Hash.GetBuffer()) != 0)
				//
	{
		msg.Format(_T("Password Mismatch\nMac1 = %s\nMac2 = %s"), HeaderGetSK, SK_Hash);
		AfxMessageBox(msg, MB_OK);
	}
	else
	{
		//파일 읽고 쓰기
		while(1)
		{
			if((ReadByte = opfile.Read(buffer_in, sizeof(buffer_in))) == 0) break;

			oRijndael.Decrypt(buffer_in, buffer_out, sizeof(buffer_in), 1);

			clfile.Write(buffer_out, ReadByte);
			//clfile.Write(buffer_in, ReadByte);
		}
		AfxMessageBox(_T("Success"), MB_OK);
	}
```

```
		goto Jmp00;

Fail:			//인증 실패시 할 행동
		//CFile.Remove( ... );
		//or
		//DeleteFile(strPath);
		clfile.Remove(m_strVhdName);

Jmp00:

		opfile.Close();
		clfile.Close();


}


void CtestrwdlgDlg::OnBnClickedCheck1()		//MAC Passcode	//default m_bChecked[0]=FALSE
{
		// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

		if(m_bChecked[0])
		{
			m_bChecked[0]=FALSE;
			((CEdit*) GetDlgItem(IDC_EDIT_MACPASSCODE)->EnableWindow(FALSE));

			if(!m_bChecked[1])  //m_bChecked[0]=FALSE//m_bChecked[1]=FASLE  00
			{
				m_TypeCheck=0;
			}
			else					//m_bChecked[0]=FALSE//m_bChecked[1]=TRUE
		01
			{
				m_TypeCheck=1;
			}
		}
		else
		{
			m_bChecked[0]=TRUE;
			((CEdit*) GetDlgItem(IDC_EDIT_MACPASSCODE)->EnableWindow(TRUE));


			if(!m_bChecked[1])  //m_bChecked[0]=TRUE//m_bChecked[1]=FASLE   10
			{
				m_TypeCheck=2;
			}
			else					//m_bChecked[0]=TRUE//m_bChecked[1]=TRUE
		11
			{
				m_TypeCheck=3;
			}
		}
```

```
	//CString msg;
	//msg.Format(_T("%d"),m_TypeCheck);
	//AfxMessageBox(msg,MB_OK);
}

void CtestrwdlgDlg::OnBnClickedCheck2()		//Drive Passcode	//default m_bChecked[1]=FALSE
{
	// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
	if(m_bChecked[1])
	{
		m_bChecked[1]=FALSE;
		((CEdit*) GetDlgItem(IDC_EDIT_DRIVEPASSCODE)->EnableWindow(FALSE));

		if(!m_bChecked[0])  //m_bChecked[0]=FALSE//m_bChecked[1]=FASLE  00
		{
			m_TypeCheck=0;
		}
		else					//m_bChecked[0]=TRUE//m_bChecked[1]=FALSE
	10
		{
			m_TypeCheck=2;
		}
	}
	else
	{
		m_bChecked[1]=TRUE;
		((CEdit*) GetDlgItem(IDC_EDIT_DRIVEPASSCODE)->EnableWindow(TRUE));

		if(!m_bChecked[0])  //m_bChecked[0]=FALSE//m_bChecked[1]=TRUE  01
		{
			m_TypeCheck=1;
		}
		else					//m_bChecked[0]=TRUE//m_bChecked[1]=TRUE
	11
		{
			m_TypeCheck=3;
		}
	}

	//CString msg;
	//msg.Format(_T("%d"),m_TypeCheck);
	//AfxMessageBox(msg,MB_OK);
}


void CtestrwdlgDlg::OnEnChangeEditMacpasscode()
{
	// TODO:  RICHEDIT 컨트롤인 경우, 이 컨트롤은
	// CDialogEx::OnInitDialog() 함수를 재지정
	//하고 마스크에 OR 연산하여 설정된 ENM_CHANGE 플래그를 지정하여 CRichEditCtrl().SetEventMask()를
호출하지 않으면
```

```
	// 이 알림 메시지를 보내지 않습니다.

	// TODO:  여기에 컨트롤 알림 처리기 코드를 추가합니다.
	UpdateData(TRUE);
}


void CtestrwdlgDlg::OnEnChangeEditDrivepasscode()
{
	// TODO:  RICHEDIT 컨트롤인 경우, 이 컨트롤은
	// CDialogEx::OnInitDialog() 함수를 재지정
	//하고 마스크에 OR 연산하여 설정된 ENM_CHANGE 플래그를 지정하여 CRichEditCtrl().SetEventMask()를
호출하지 않으면
	// 이 알림 메시지를 보내지 않습니다.

	// TODO:  여기에 컨트롤 알림 처리기 코드를 추가합니다.
	UpdateData(TRUE);
}


void CtestrwdlgDlg::OnEnChangeEditAeskey()
{
	// TODO:  RICHEDIT 컨트롤인 경우, 이 컨트롤은
	// CDialogEx::OnInitDialog() 함수를 재지정
	//하고 마스크에 OR 연산하여 설정된 ENM_CHANGE 플래그를 지정하여 CRichEditCtrl().SetEventMask()를
호출하지 않으면
	// 이 알림 메시지를 보내지 않습니다.

	// TODO:  여기에 컨트롤 알림 처리기 코드를 추가합니다.
	UpdateData(TRUE);
}
```

4. 결　　론

정보가 방대해지고, 각 개인의 정보, 기업의 정보 등 여러 가지 정보가 증대됨에 따라 그 정보를 보관, 관리, 처리가 점점 더 전문화 되고 보안이 더욱 중요해지는 추세이다. 여러 매체에서 조사한 결과 USB메모리에 회사의 주요 정보나 고객 정보저장하고 있다는 조사 결과도 있으며, 이는 USB 메모리 분실 시 회사의 중요정보 및 고객의 개인정보가 유출될 수 있는 중대한 문제이다.

Secure Line은 USB를 일반영역과 보안영역으로 나누어서, 분실 시 타인이 습득하더라도 제한적 접근이 가능하도록 의견을 모아 개발에 착수하게 되었다. 개발결과 일반영역은 일반 USB처럼 별다른 인증 없이 접근이 가능하고, 보안영역은 사용자가 지정한 인증방식을 통해 접근이 가능하도록 구현을 하였고, 사용자가 지정된 PC이외의 다른 PC에서 보안영역의 접근을 원할 경우, 별도의 인증을 통해서 보안영역에 접근이 가능하도록 구현하여 사용자의 부주의나 그 외의 요인으로 타인이 습득한 USB메모리에 접근하더라도 중요 정보가 저장되어 있는 보안영역으로의 접근이 제한되기 때문에 피해예방을 할 수 있도록 구현하였다. 각종 자료 수집을 통해 NTFS에 적용이 가능한 시스템을 개발하였지만, 시스템에 적용한 VHD API가 FAT32를 지원하지 않아 FAT32에는 적용할 수 없었고, 교수님의 조언을 통해 FAT32를 NTFS로 변환하는 방법으로 해결하였다.

이번 연구를 통하여 암호화 프로그래밍과 하드웨어 인증 구현을 위해 관련자료 수집과 개인연구 및 의논을 통해 개인적인 프로그래밍 실력, 암호화, 인증방식에 대해 좀 더 깊숙이 공부할 수 있는 기회가 되었다.

지금 개발한 보안영역 탑재 USB는 여러 하드웨어 인증 방안의 한 가지 방법을 제안하였다. 이번 연구에서 부족한 점을 보완하고 개선 한다면 향후에 나올 저장매체는 좀 더 편리하고 안전한 매체가 되는데 기여할 수 있길 기대해 본다.

5. 참고 문헌

[1] www.kisa.or.kr : 해쉬 함수의 역사 및 개요

[2] www.nexpert.net : [JPEG] 05_HASH_&_HMAC

[3] 네이버 백과사전

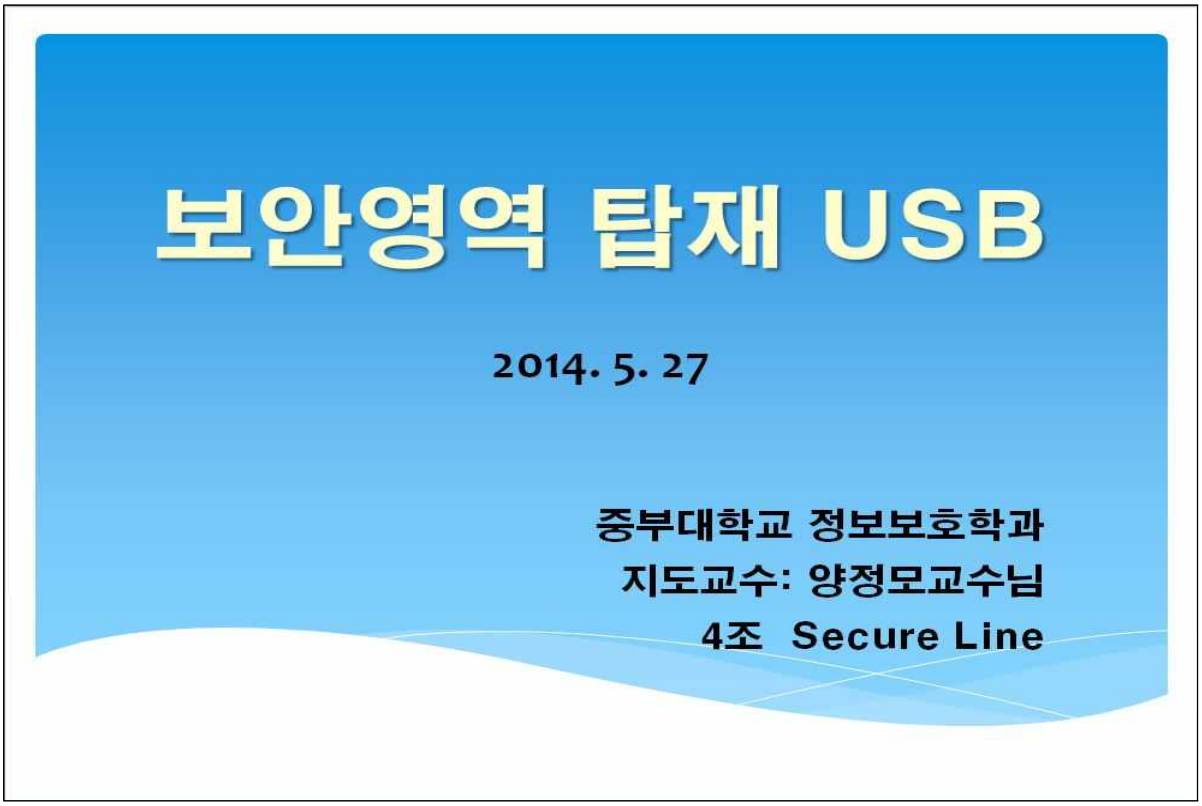
[4] http://blog.naver.com/jsjhahi : MFC란? / MFC의 구조도

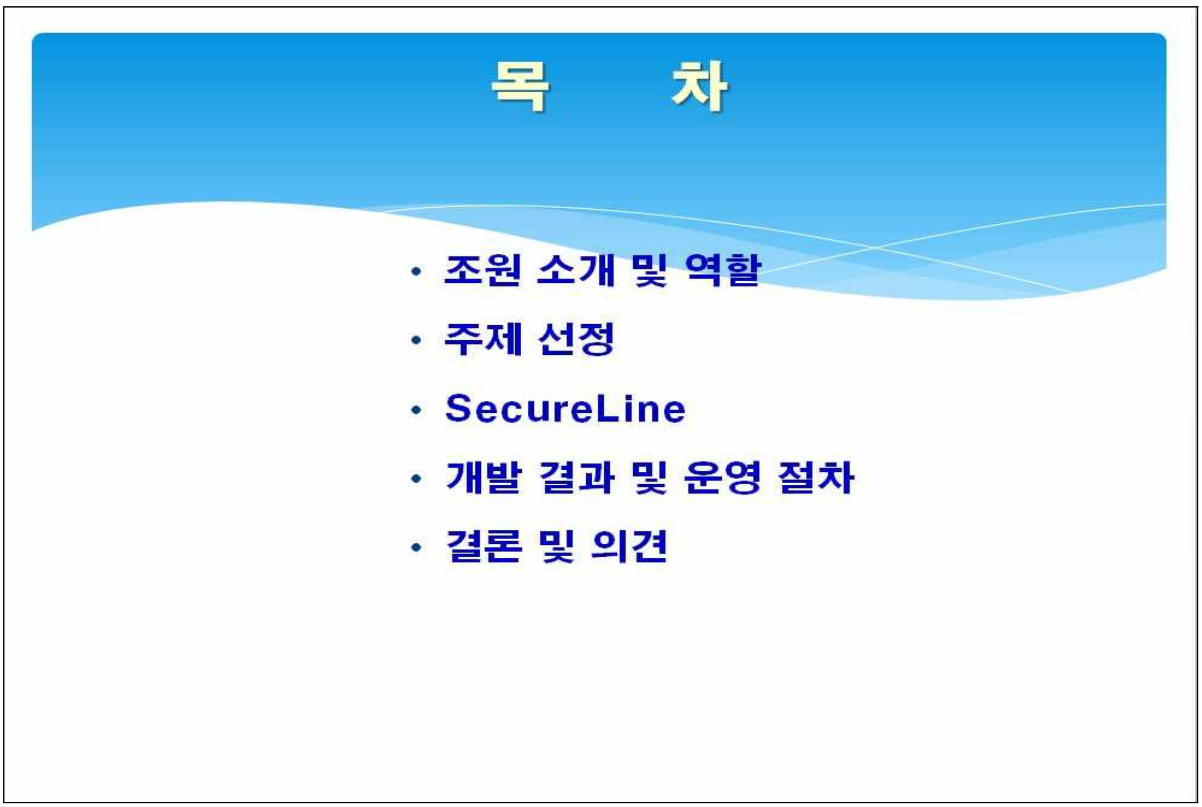
[5] http://leejiwon09.inlive.co.kr : AES의 기본구조

[6] http://cappleblog.co.kr : VHD의 개요 및 설명

[7] http://msdn.microsoft.com : Virtual Hard Disk

5. 발표 PPT 자료


보안영역 탑재 USB
2014. 5. 27
중부대학교 정보보호학과
지도교수: 양정모교수님
4조 Secure Line


목 차
• 조원 소개 및 역할
• 주제 선정
• SecureLine
• 개발 결과 및 운영 절차
• 결론 및 의견

조원 소개 및 역할

- **손민규 : 자료수집 및 보고서 종합**
- **송무건 : 기술 연구 및 프로그래밍**
- **송민섭 : 기술 연구 및 프로그래밍**

주제 선정

보안 2011.08.10

USB 메모리 분실 피해액 350만 달러 육박

John P. Mello Jr | PC Advisor

해커들에 의한 데이터 유출이 헤드라인을 장식하고 있는 요즘, 또 다른 형태의 데이터 유출도 기업에 큰 타격을 줄 수 있다는 조사 결과가 발표됐다. 바로, USB 메모리 스틱의 분실이다.

포니몬 인스티튜트(Ponemon Institute)가 기업 400곳 이상을 조사해 발표한 보고서에 따르면, 메모리 스틱 분실로 인한 회사의 피해액이 250만 달러에 이르는 것으로 나타났다.

평균적으로 분실된 USB 스틱에는 1만 2,000명의 고객, 소비자, 직원 기록이 포함되어 있었고, 기록당 가치는 214달러로 전부 합치면 250만 달러 정도가 된다.

보고서는 "USB 스틱은 매우 작지만, 여기에 포함된 데이터의 유출은 매우 큰 결과로 이어질 수 있다. 조사 응답자의 70% 이상이 도난 당하거나 분실된 USB 드라이브에 저장된 주요 정보가 데이터 유출의 큰 원인이라고 답했다"라고 설명한다.

게다가 조사에 참여한 743명의 IT 및 IT 보안 실무자들 중 거의 절반이 "지난 2년 동안 USB 메모리 도난 또는 분실로 피해를 입은 경험이 있다"고 답했다.

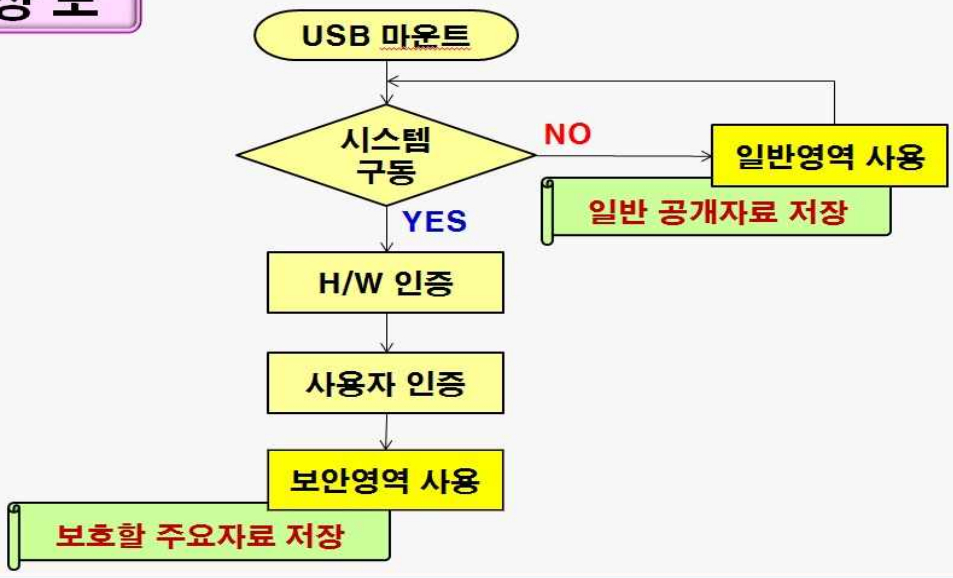
주제 선정

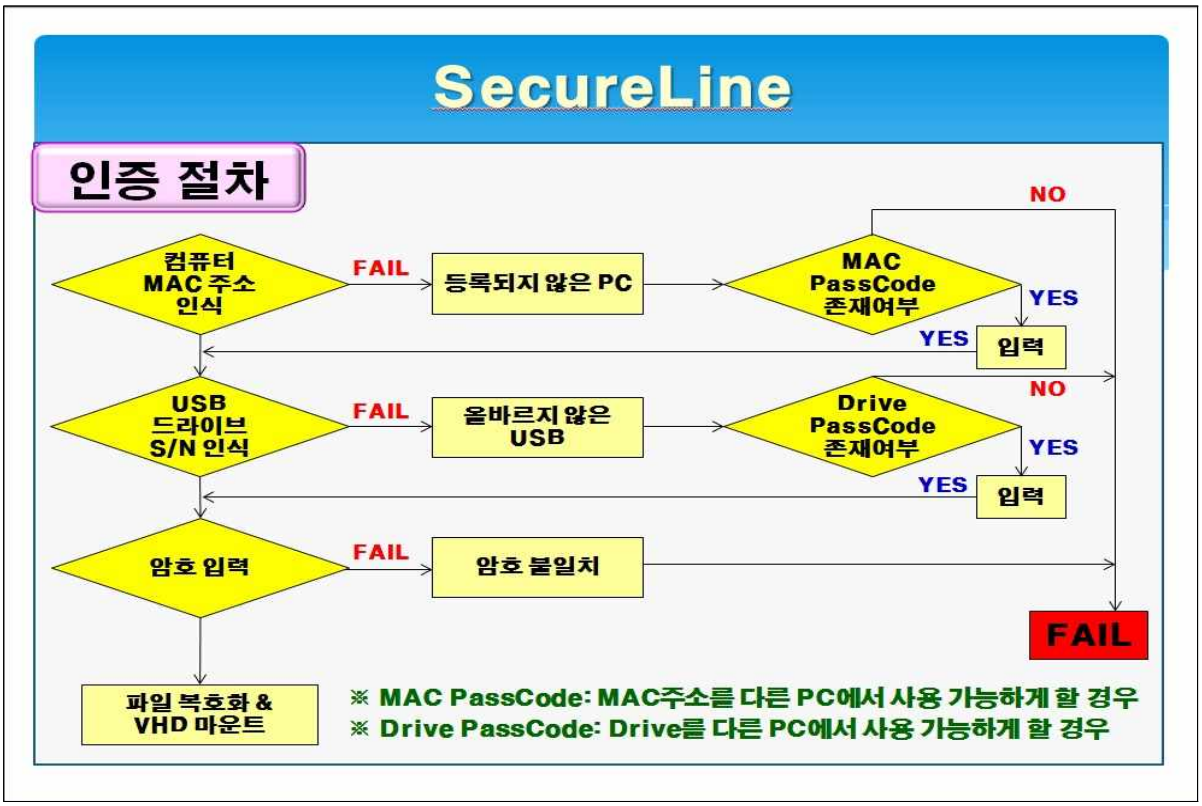
- USB은 사용의 편리성 및 휴대의 간편성이 보장되나 분실 요인이 많음
- 특히 분실시 수록 자료에 대한 보호가 불가능
- 이에 대한 대책으로
  - USB를 일반영역과 보안영역으로 구분 할당
  - 공개 자료는 일반영역, 보호할 자료는 보안영역에 수록, USB 분실시에도 주요자료를 보호

⇨ 개발 시스템은 SecureLine으로 명명

SecureLine

구 상 도




SecureLine
인증 절차
컴퓨터 MAC 주소 인식
FAIL
등록되지 않은 PC
MAC PassCode 존재여부
NO
YES
입력
YES
USB 드라이브 S/N 인식
FAIL
올바르지 않은 USB
Drive PassCode 존재여부
NO
YES
입력
YES
암호 입력
FAIL
암호 불일치
FAIL
파일 복호화 & VHD 마운트
※ MAC PassCode: MAC주소를 다른 PC에서 사용 가능하게 할 경우
※ Drive PassCode: Drive를 다른 PC에서 사용 가능하게 할 경우


SecureLine
개발 및 운영 환경
❍개발체제 ⇨ Visual Studio 2008, 2010
- MFC
- API
- C++
❍운영환경 ⇨ Windows 7
❍암호체계 ⇨ AES, SHA-256

SecureLine

AES 알고리즘?

- 고급 암호화 표준(Advanced Encryption Standard)은 대칭키 암호 알고리즘
- 미국 표준 기술 연구소에서 2001.11 암호화 기술표준으로 선정
- 다양한 컴퓨터 환경에서도 우수한 성능을 보여주고 메모리를 적게 차지해 메모리 용량이 적은 장치 (USB, 스마트 카드 등)에서도 손쉽게 쓸 수 있음
- 2014.05 현재 알려진 보안공격 미확인

SecureLine

SHA-256 (해쉬)?

- 해쉬 알고리즘 개념
  - ▶ 가변사이즈의 데이터를 작은 일정길이의 데이터 사이즈로 변환시켜 주는 알고리즘
- 해쉬 알고리즘 특징
  - ▶ 다양한 가변 길이의 입력에 적용될 수 있어야 함
  - ▶ 고정된 길이의 출력을 만듦
  - ▶ 해쉬 결과값으로 입력값 을 계산하는 것은 불가능
  - ▶ 동일한 해쉬값 을 가지는 서로 다른 메시지 쌍이 없음

SecureLine

VHD(Virtual Hard Disk)?

❖ VHD의 개념

▶ 가상 하드디스크의 약자로 파일 자체가 가상 하드디스크 역할을 담당

❖ VHD의 특징

▶ 실제 존재하지 않는 디스크를 디스크 이미지 파일을 통해 논리적으로 마치 실제 물리 디스크를 사용하는 것과 같은 효과

▶ 단순 자료저장 뿐만 아니라 운영체제 구동까지 가능

개발 결과 및 운영 절차

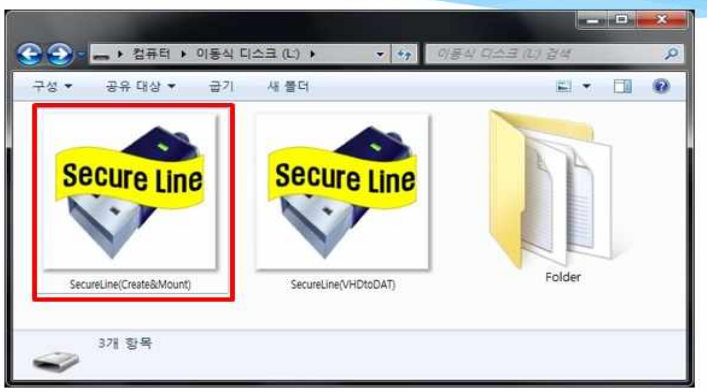
USB 드라이브 탐색

❍ SecureLine 실행 준비화면

개발 결과 및 운영 절차

VHD 파일 생성

❍ SecureLine(Creat&Mount) 실행



개발 결과 및 운영 절차

❍ VHD 생성 및 Attach 작업 화면

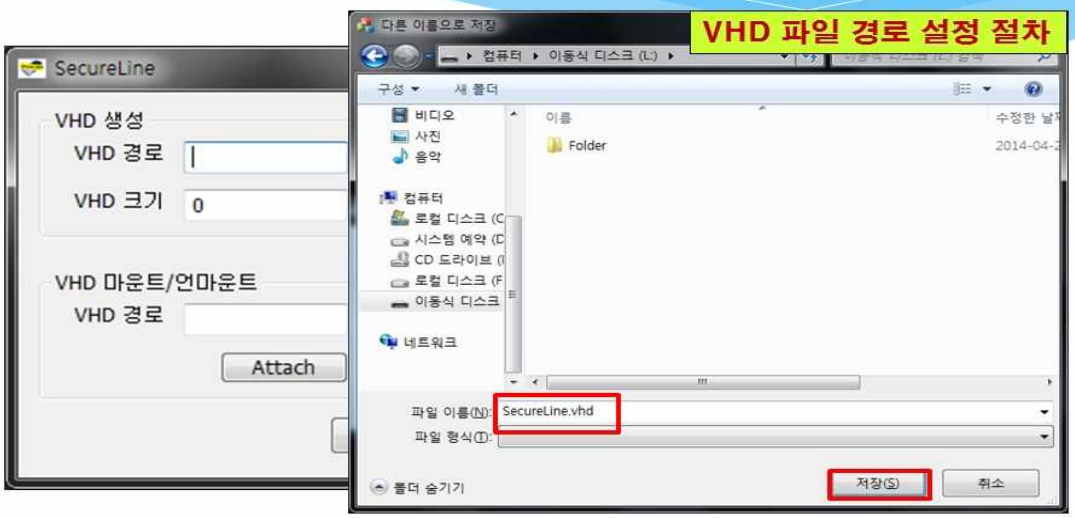
VHD 경로와 크기를 지정 하여 VHD 파일을 생성

VHD 파일을 마운트하면 가상드라이브가 생성됨

FAT32 USB 이용시 NTFS로 변환 하여 사용

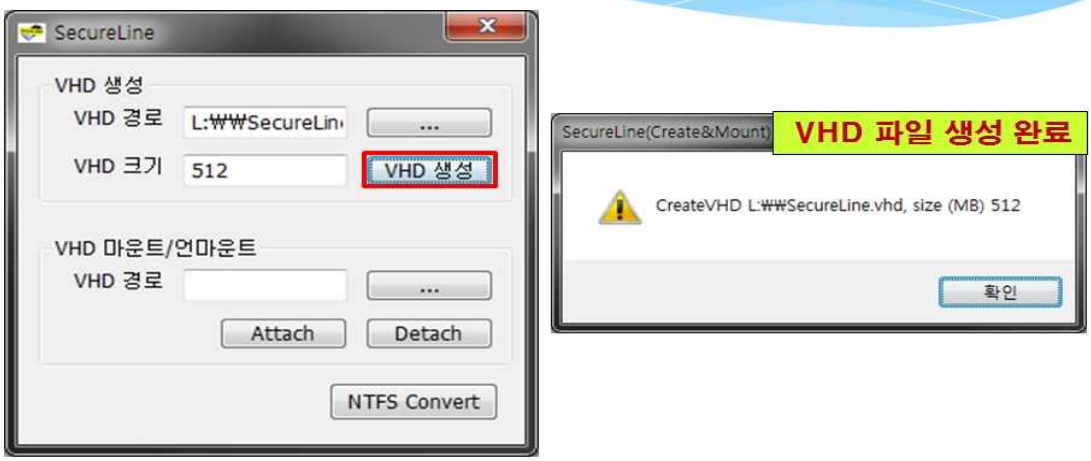
개발 결과 및 운영 절차

❍ VHD 파일 경로 설정



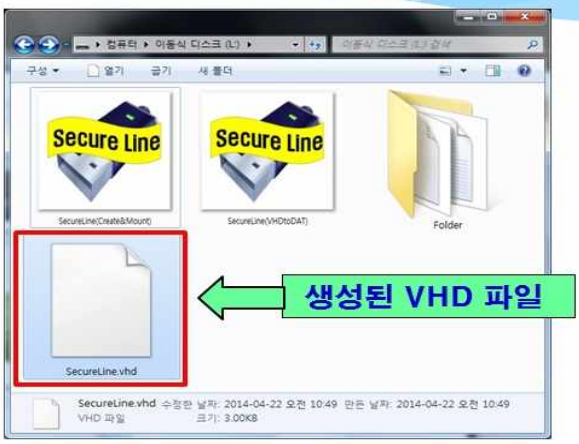
개발 결과 및 운영 절차

❍ VHD 파일 크기 설정 및 VHD 파일 생성



개발 결과 및 운영 절차

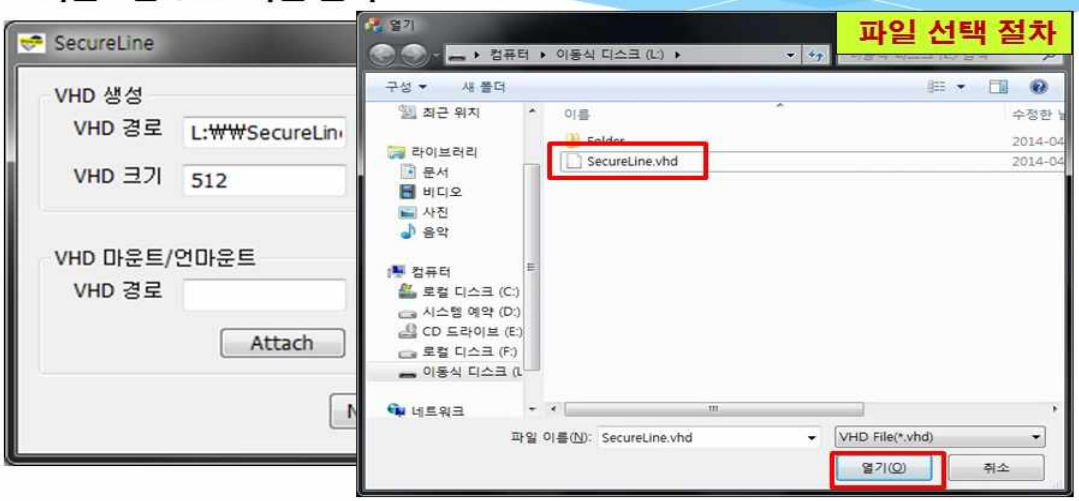
❍ USB 드라이브에 VHD 파일이 생성된 결과



개발 결과 및 운영 절차

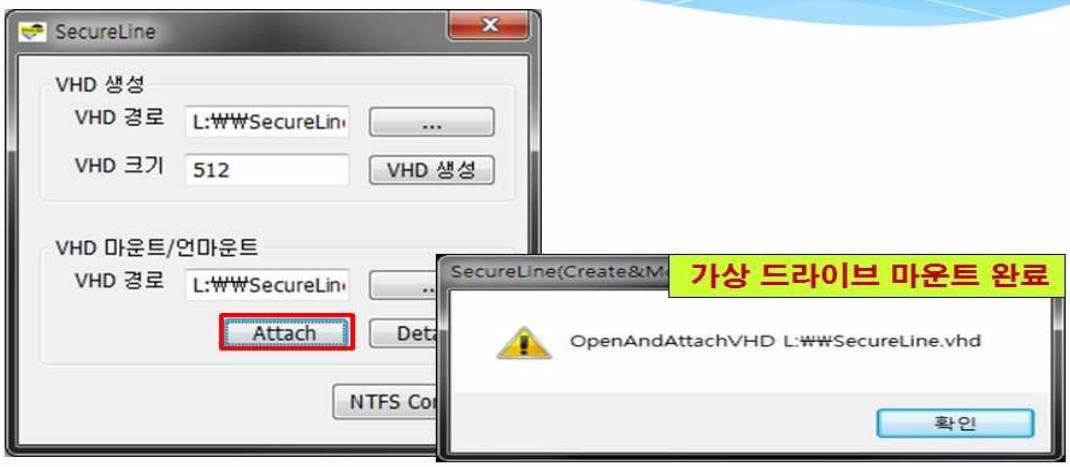
VHD 파일 마운트

❍ 마운트할 VHD 파일 선택



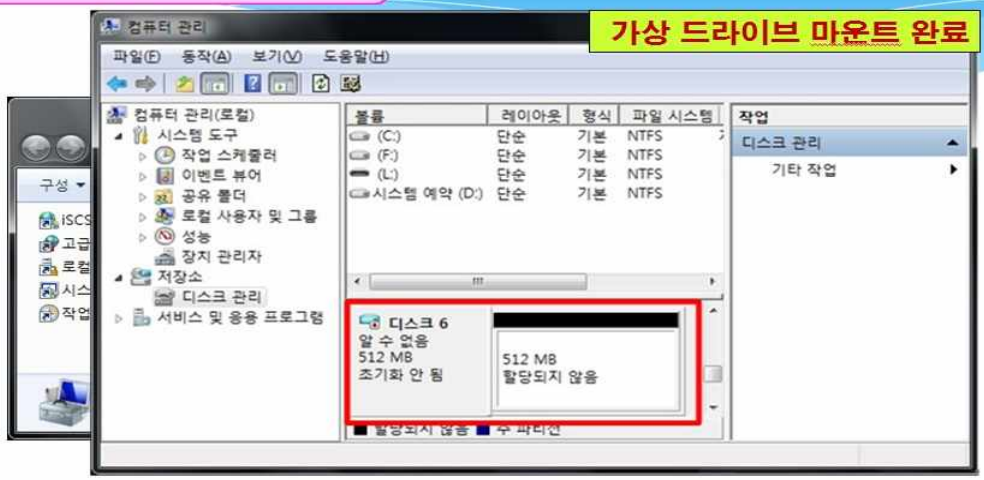
개발 결과 및 운영 절차

❍ VHD 파일을 가상 드라이브로 마운트



개발 결과 및 운영 절차

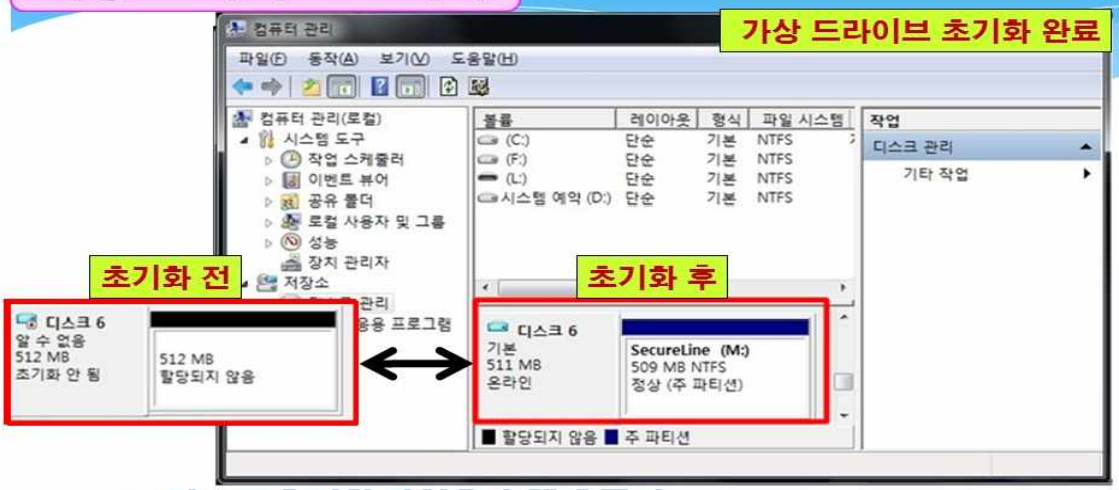
가상 드라이브 초기화



- 제어판 – 관리 도구 – 컴퓨터 관리 실행

개발 결과 및 운영 절차

가상 드라이브 초기화

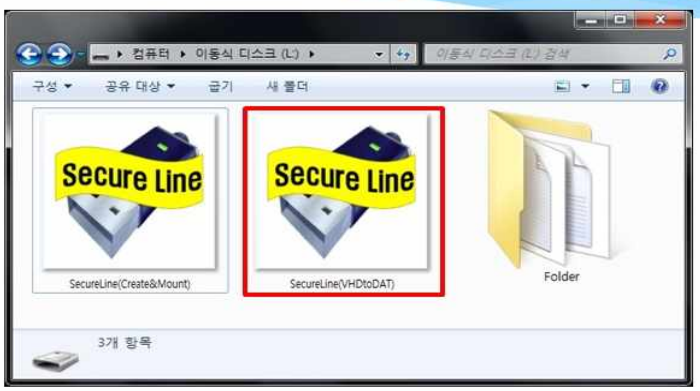


- 디스크 초기화 작업을 수행해 준다

개발 결과 및 운영 절차

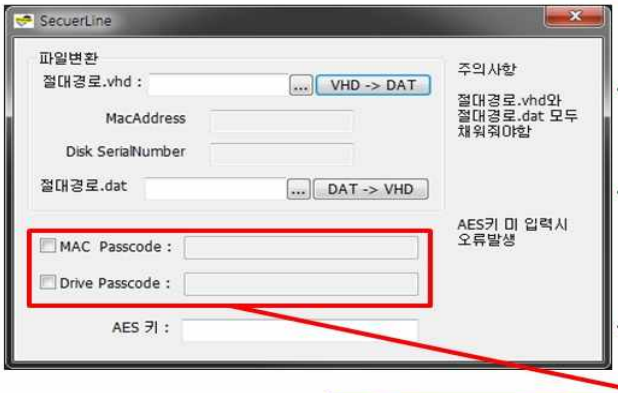
파일 암/복호화

❍ SecureLine(VHDtoDAT) 실행



개발 결과 및 운영 절차

❍ VHD 파일 암/복호화 작업화면



VHD 파일을 AES로 암호화
암호화한 PC의 MAC주소,
드라이브 시리얼을 저장

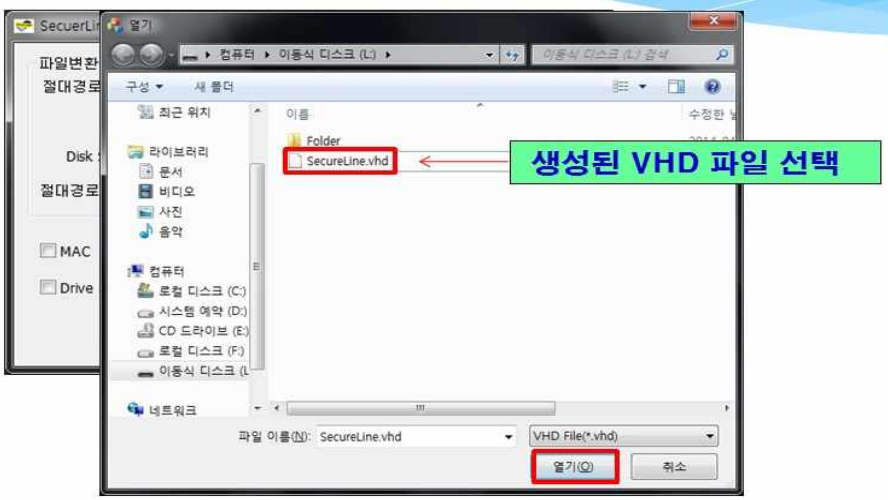
암호화된 DAT파일을 복호화

암/복호화에 사용할 키를 입력

비인증 PC사용이 필요할 경우 Passcode 설정
(미설정시 인증된PC에서만 사용가능)

개발 결과 및 운영 절차

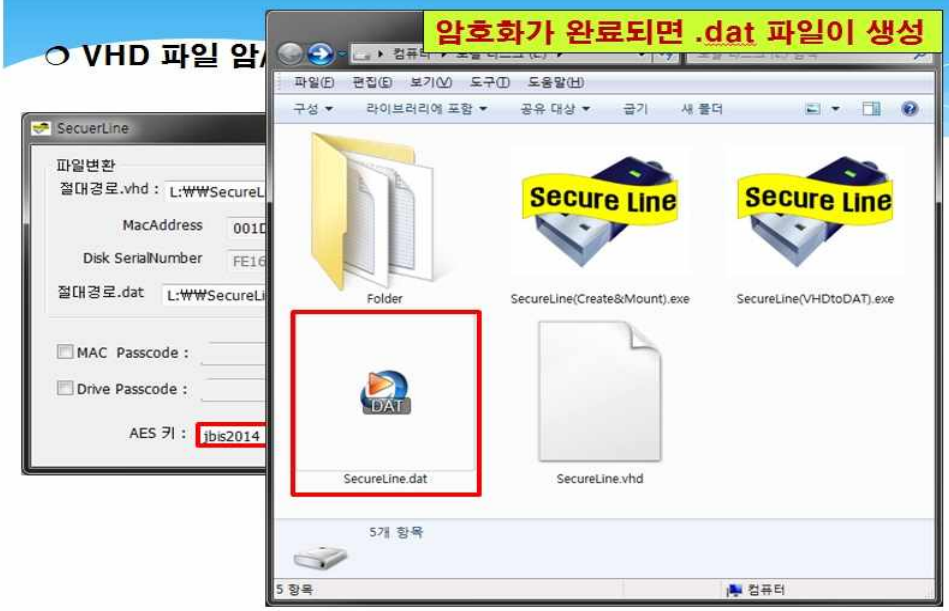
❍ VHD 파일 암/복호화 경로설정 (1/3)



생성된 VHD 파일 선택

개발 결과 및 운영 절차

❍ VHD 파일 암/



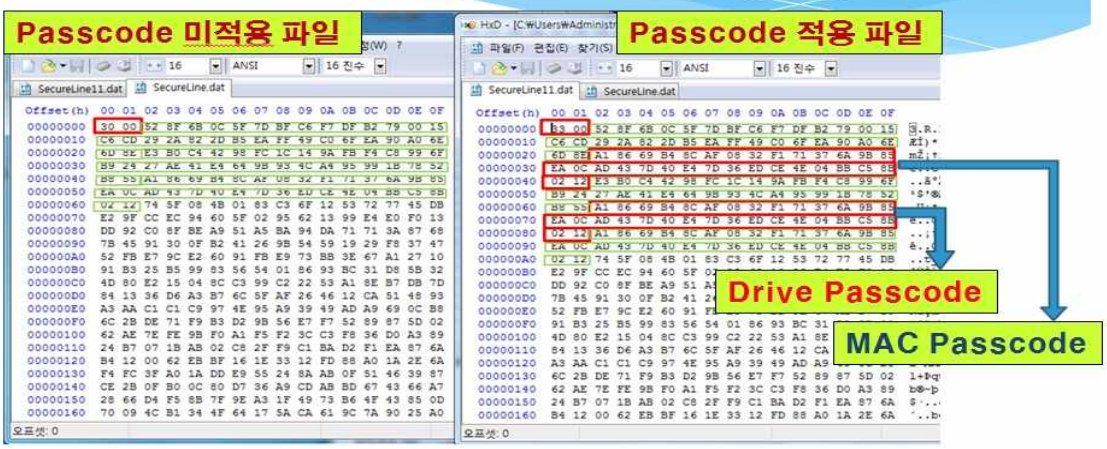
개발 결과 및 운영 절차

❍ VHD 파일 암/복호화 경로설정 (3/3)

DAT -> VHD의 경로설정 방법도 위의 과정과 동일

개발 결과 및 운영 절차

❍ Passcode 적용 / 미적용 파일 헥스 코드 비교화면



결론 및 의견

❍ SecureLine 개발 성과

- 암호화 시 첨부한 MAC 주소 등을 사전 확인함으로써 인증되지 않은 시스템에서는 파일 접근을 1차적으로 거부

- USB에 보안영역을 설정, 수록된 자료 열람을 거부함으로써 USB 분실시에도 주요자료를 안전하게 보호

❍ Secure Line 개발과정에서 파일 암·복호화, 하드웨어 인증 등을 직접 프로그램으로 구현, 기술개발 역량을 배양


Q & A
감사합니다