Bitomakey のソースファイル

（C）宇山　靖政

2012.04.14

BmpKey.exe　について。

このソフトは、BmpEC.exe と BmpDC.exe で使用する暗号鍵を作成するソフトです。さらに、暗号化と復号化のテストができます。

暗号化したものは、16 進数の列として表示します。もちろん、ビットマップファイルも作成されますので、その画像を確認することもできます。

“何でもビトマ”のバージョンアップと考えて下さい。一番大きな違いは、暗号化鍵の長さです。

暗号に楽しさを加えようと、どんなファイルもビットマップに変換するソフト BmpEC.exe を作りました。

暗号化鍵は、128,192,256 ビットから選択できます。このソフトを利用して、メールデータを暗号化してから送信します。拡張子は、現在は変更していませんが、最後にこのソフトで暗号化されて送られてきたデータは、拡張子を bmp にすれば、画像として表示されます。ほかのメールソフトで受信すれば単に画像ファイルの拡張子が不適切なものが送られてきたということになります。受信後に、BmpDC.exe で復号化します。

ただし、送信のときに、拡張子を bmp にすることも現在検討中です。その場合はまた、ソースコードを公開いたします。

暗号化の過程では乱数を利用して、同一データも暗号化する時間が異なれば結果も異なりますので、暗号化されたものが同一化データか否かは判定が難しくなっています。

BmpKey.exe のソースコード

最初は、ヘッダーファイル

Bmpkey.h

```
////////////////////////////
// BmpKey.h : PROJECT_NAME アプリケーションのメインヘッダーファイルです。
//

#pragma once
```

```
#ifndef __AFXWIN_H__
	#error "PCH に対してこのファイルをインクルードする前に'stdafx.h' をインクルードしてください"
#endif

#include "resource.h"		// メインシンボル


// CBmpKeyApp:
// このクラスの実装については、BmpKey.cpp を参照してください。
//

class CBmpKeyApp : public CWinApp
{
public:
	CBmpKeyApp();

// オーバーライド
	public:
	virtual BOOL InitInstance();

// 実装

	DECLARE_MESSAGE_MAP()
};

extern CBmpKeyApp theApp;

// 暗号化;  復号化
void bmp1(char* St1, char* St2, char* St3);
void bmp2(char* keyfn, char* srcfn, char* dstfn);
```

Bmpkeydig.h

```
////////////////////////////////
// BmpKeyDlg.h : ヘッダーファイル
//

#pragma once


// CBmpKeyDlg ダイアログ
class CBmpKeyDlg : public CDialog
{
// コンストラクション
public:
	CBmpKeyDlg(CWnd* pParent = NULL);	// 標準コンストラクタ

// ダイアログデータ
```

```
        enum { IDD = IDD_BMPKEY_DIALOG };
        int madekey;
        CString keylength;
        int i_KeyLen;
        int i_Mode;
        int i_Disp;
        CString s_key;

        CString s_fname1;// key_file
        CString s_fname2;// key_file

        CString planedata_file;
        CString encryptdata_file;
        CString decryptdata_file;
        CString encryptkey_file;

        CEdit   datadisp;

    void yDisplay(const char *cp);

        protected:
        virtual void DoDataExchange(CDataExchange* pDX);   // DDX/DDV サポート


// 実装
protected:
        HICON m_hIcon;

        // 生成された、メッセージ割り当て関数
        virtual BOOL OnInitDialog();
        afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
        afx_msg void OnPaint();
        afx_msg HCURSOR OnQueryDragIcon();

        virtual void OnOK();
        afx_msg void MakeKey();
        afx_msg void TestKey();

        DECLARE_MESSAGE_MAP()
};
```

Resource.h

```
//////////////////////////////////
//{{NO_DEPENDENCIES}}
// Microsoft Visual C++ generated include file.
// Used by BmpKey.rc
//
#define IDM_ABOUTBOX                    0x0010
#define IDD_ABOUTBOX                    100
```

```
#define IDS_ABOUTBOX                    101
#define IDD_BMPKEY_DIALOG               102
#define IDR_MAINFRAME                   128
#define IDC_RADIO1                      1000
#define IDC_RADIO2                      1001
#define IDC_RADIO3                      1002
#define IDC_RADIO4                      1003
#define IDC_RADIO5                      1004
#define IDC_RADIO6                      1005
#define IDC_ENCRYPTKEY_FILE             1006
#define IDC_DECRYPTKEY_FILE             1007
#define ID_MAKEKEY                      1008
#define IDC_SECRETKEY                   1009
#define IDC_DISPLAY                     1010
#define IDC_RADIO7                      1011
#define ID_TESTKEY                      1016
#define IDC_PLANEDATA_FILE              1017
#define IDC_ENCRYPTDATA_FILE            1018
#define IDC_DECRYPTDATA_FILE            1019

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE        129
#define _APS_NEXT_COMMAND_VALUE         32771
#define _APS_NEXT_CONTROL_VALUE         1012
#define _APS_NEXT_SYMED_VALUE           101
#endif
#endif
```

Stdafx.h

```
////////////////////////////////
// stdafx.h : 標準のシステムインクルードファイルのインクルードファイル、または
// 参照回数が多く、かつあまり変更されない、プロジェクト専用のインクルードファイル
// を記述します。

#pragma once

#ifndef _SECURE_ATL
#define _SECURE_ATL 1
#endif

#ifndef VC_EXTRALEAN
#define VC_EXTRALEAN            // Windows ヘッダーから使用されていない部分を除外します。
#endif

// 下で指定された定義の前に対象プラットフォームを指定しなければならない場合、以下の定義を変更してください。
```

```
// 異なるプラットフォームに対応する値に関する最新情報については、MSDN を参照してください。
#ifndef WINVER                          //Windows XP 以降のバージョンに固有の機能の使用を許可します。
#define WINVER 0x0501           // これをWindows の他のバージョン向けに適切な値に変更してください。
#endif

#ifndef _WIN32_WINNT            // Windows XP 以降のバージョンに固有の機能の使用を許可します。
#define _WIN32_WINNT 0x0501     // これをWindows の他のバージョン向けに適切な値に変更してください。
#endif

#ifndef _WIN32_WINDOWS          // Windows 98 以降のバージョンに固有の機能の使用を許可します。
#define _WIN32_WINDOWS 0x0410 // これをWindows Me またはそれ以降のバージョン向けに適切な値に変更してください。
#endif

#ifndef _WIN32_IE                       // IE 6.0 以降のバージョンに固有の機能の使用を許可します。
#define _WIN32_IE 0x0600 // これをIE の他のバージョン向けに適切な値に変更してください。
#endif

#define _ATL_CSTRING_EXPLICIT_CONSTRUCTORS      // 一部のCString コンストラクタは明示的です。

// 一般的で無視しても安全なMFC の警告メッセージの一部の非表示を解除します。
#define _AFX_ALL_WARNINGS

#include <afxwin.h>         // MFC のコアおよび標準コンポーネント
#include <afxext.h>         // MFC の拡張部分


#include <afxdisp.h>        // MFC オートメーションクラス



#ifndef _AFX_NO_OLE_SUPPORT
#include <afxdtctl.h>           // MFC のInternet Explorer 4 コモンコントロールサポート
#endif
#ifndef _AFX_NO_AFXCMN_SUPPORT
#include <afxcmn.h>                     // MFC のWindows コモンコントロールサポート
#endif // _AFX_NO_AFXCMN_SUPPORT









#ifdef _UNICODE
#if defined _M_IX86
#pragma comment(linker,"/manifestdependency:¥"type='win32' name='Microsoft.Windows.Common-Controls'
version='6.0.0.0' processorArchitecture='x86' publicKeyToken='6595b64144ccf1df' language='*'¥"")
#elif defined _M_IA64
#pragma comment(linker,"/manifestdependency:¥"type='win32' name='Microsoft.Windows.Common-Controls'
```

```
version='6.0.0.0' processorArchitecture='ia64' publicKeyToken='6595b64144ccf1df' language='*'¥"")
#elif defined _M_X64
#pragma comment(linker,"/manifestdependency:¥"type='win32' name='Microsoft.Windows.Common-Controls'
version='6.0.0.0' processorArchitecture='amd64' publicKeyToken='6595b64144ccf1df' language='*'¥"")
#else
#pragma comment(linker,"/manifestdependency:¥"type='win32' name='Microsoft.Windows.Common-Controls'
version='6.0.0.0' processorArchitecture='*' publicKeyToken='6595b64144ccf1df' language='*'¥"")
#endif
#endif
```

次は、cppファイル

Bmpecdc.cpp

```
//////////////////////////
// Misty1.cpp : コンソールアプリケーション用のエントリポイントの定義
//

#include "stdafx.h"

#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#include <math.h>
#include <direct.h>


typedef unsigned char uchar;
typedef unsigned short ushort;
typedef unsigned long DWORD;

        FILE* keyfile;
        FILE* srcfile;
        FILE* dstfile;

void bmp1(char* keyfn, char* pt, char* ct)  // 暗号化
{
        int mode,klen; //,blen,rc;
        char c_mode[4], c_klen[8], c_key[64];

int nsize, fsize, sidesize;
char FName[256];
unsigned int j, k, l, jj;
int i, mn;
unsigned char tmpch[16];
unsigned char tmprbuf[16];
```

```
char key[64];//32];

unsigned char bmpHeader[54] = {
'B', 'M', /* [ 0] ファイルタイプ*/
54, 4, 0, 0, /* [ 2] ファイルサイズ 54+4*16*16=1078*/
0, 0, 0, 0, /* [ 6] 予約*/
54, 0, 0, 0, /* [10] ビットマップデータのシーク位置*/
40, 0, 0, 0, /* [14] ここから始まるヘッダの高さ*/
16, 0, 0, 0, /* [18] ビットマップの幅*/
16, 0, 0, 0, /* [22] ビットマップの高さ*/
0x01, 0, /* [26] プレーン数*/
32, 0, /* [28] 1ピクセルあたりのビット数（課題がバイト指定されていたのでbitに変更）*/
0, 0, 0, 0, /* [30] 圧縮タイプ*/
0, 1, 0, 0, /* [34] ビットマップデータの長さ16*16=256*/
0, 0, 0, 0, /* [38] 水平解像度(px/m) */
0, 0, 0, 0, /* [42] 垂直解像度(px/m) */
0, 0, 0, 0, /* [46] カラーインデックス数*/
0, 0, 0, 0, /* [50] 重要なカラーインデックス数*/
};

        if((keyfile = fopen(keyfn, "rt")) == NULL){
//              printf("Can not find key file for encryption. ¥n");
                return;
        }
        fgets( c_mode, 4, keyfile );
        fgets( c_klen, 8, keyfile );
        fgets( c_key, 64, keyfile );

        fclose(keyfile);

        strcpy(key, c_key);

        mode = atoi(c_mode);
        klen = atoi(c_klen);

        mn = 8;
        if(klen == 128){ mn = 16; }
        if(klen == 192){ mn = 24; }
        if(klen == 256){ mn = 32; }

        if((srcfile = fopen(pt, "rb")) == NULL){
//              printf("Can not open plane file. ¥n");
                return;
        }

        strcpy(FName, ".¥¥Encrypt¥¥");
        strcat(FName, ct);
        if((dstfile = fopen(FName, "wb")) == NULL){
            if(0 == _mkdir(".¥¥Encrypt")){
                        if((dstfile = fopen(FName, "wb")) == NULL){
                                return;
                        }
```

```
		}
		else{
			return;
		}
	}

	srand( (unsigned)time(NULL) );  //in constracter

	strcpy(FName, pt);// = srcfile.GetFileName();
	nsize = strlen(FName); //.GetLength();// length of file name.
// 平文
	fseek(srcfile, 0, SEEK_END);
	long filelen = ftell(srcfile);
	fseek(srcfile,0,0);
	fsize = filelen; //fsize = srcfile.GetLength();// as char 8 bit
	sidesize = 1 + (int)sqrt((double)(4+1+(fsize/2)+1+(nsize/2)));// as short int 16 bit

	// Write the header of bitmap file.
	*((DWORD*)(&(bmpHeader[2]))) = 54+4*sidesize*sidesize;
	*((DWORD*)(&(bmpHeader[18]))) = sidesize;
	*((DWORD*)(&(bmpHeader[22]))) = sidesize;
	*((DWORD*)(&(bmpHeader[34]))) = sidesize*sidesize;

	fwrite( bmpHeader,sizeof(char),54, dstfile); //dstfile.Write(bmpHeader, 54);

	j = rand(); // 16 bits
	k = nsize & 0x0000ffff;
	jj = (j<<16) + (j^k);
	*((unsigned int*)&tmpch) = jj;//32 bit
	fwrite( tmpch,sizeof(char),4, dstfile); //dstfile.Write(tmpch, 4);//32 bit
	j = rand(); // 16 bits
	k = nsize & 0xffff0000;
	jj = (j<<16) + (j^(k>>16));
	*((unsigned int*)&tmpch) = jj;//32 bit
	fwrite( tmpch,sizeof(char),4, dstfile); //dstfile.Write(tmpch, 4);//32 bit

	j = rand(); // 16 bits
	k = fsize & 0x0000ffff;
	jj = (j<<16) + (j^k);
	*((unsigned int*)&tmpch) = jj;//32 bit
	fwrite( tmpch,sizeof(char),4, dstfile); //dstfile.Write(tmpch, 4);//32 bit
	j = rand(); // 16 bits
	k = fsize & 0xffff0000;
	jj = (j<<16) + (j^(k>>16));
	*((unsigned int*)&tmpch) = jj;//32 bit
	fwrite( tmpch,sizeof(char),4, dstfile); //dstfile.Write(tmpch, 4);//32 bit

	for(i=0; i<nsize/2 ; i++){
		j = rand(); // 16 bits
		k = (unsigned char)FName[2*i];
		l = (unsigned char)FName[2*i+1];
		jj = (j<<16) + (j^((k<<8)+ l));
```

```
                *((unsigned int*)&tmpch) = jj;//32 bit
                fwrite( tmpch,sizeof(char),4, dstfile); //dstfile.Write(tmpch, 4);//32 bit
        }
        if(nsize%2 == 1){
                j = rand(); // 16 bits
                k = (unsigned char)FName[nsize-1];
                jj = (j<<16) + (j^((k<<8)+ 0));
                *((unsigned int*)&tmpch) = jj;//32 bit
                fwrite( tmpch,sizeof(char),4, dstfile); //dstfile.Write(tmpch, 4);//32 bit
        }
        if(nsize%2 == 0){
                j = rand(); // 16 bits
                k = 0;
                jj = (j<<16) + (j^((k<<8)+ 0));
                *((unsigned int*)&tmpch) = jj;//32 bit
                fwrite( tmpch,sizeof(char),4, dstfile); //dstfile.Write(tmpch, 4);//32 bit
        }

        for(i=4+1+nsize/2; i<sidesize*sidesize ; i++){
                j = rand(); // 16 bits
                if(1 == fread(tmprbuf, sizeof(char), 1, srcfile)){//1 == srcfile.Read(tmprbuf,1)){
                        k = (unsigned char)tmprbuf[0];
                        k ^= (unsigned char)key[(i-(4+1+nsize/2))%mn];
                }
                else{k = 0;}
                if(1 == fread(tmprbuf, sizeof(char), 1, srcfile)){//1 == srcfile.Read(tmprbuf,1)){
                        l = (unsigned char)tmprbuf[0];
                        l ^= (unsigned char)key[(i-(4+1+nsize/2))%mn];
                        jj = (j<<16) + (j^((k<<8)+ l));
                }
                else{jj = (j<<16) + (j^((k<<8)+ 0));}
                *((unsigned int*)&tmpch) = jj;//32 bit
                fwrite( tmpch,sizeof(char),4, dstfile); //dstfile.Write(tmpch, 4);//32 bit
        }

        fclose(srcfile);//.Close();
        fclose(dstfile);//.Close();

}




void bmp2(char* keyfn, char* srcfn, char* dstfn)  // 復号化
{
        int mode,klen; //,blen,rc;
        char c_mode[4], c_klen[8], c_key[64], folderfile[256];

int nsize, fsize, sidesize;
char FName[256];
unsigned int j, k, jj;
```

```
int i, mn;
unsigned char tmpch[16];
unsigned char tmprbuf[16];
char key[64];//32];
char* pfName;
int q;


unsigned char bmpHeader[54] = {
'B', 'M', /* [ 0] ファイルタイプ*/
54, 4, 0, 0, /* [ 2] ファイルサイズ 54+4*16*16=1078*/
0, 0, 0, 0, /* [ 6] 予約*/
54, 0, 0, 0, /* [10] ビットマップデータのシーク位置*/
40, 0, 0, 0, /* [14] ここから始まるヘッダの高さ*/
16, 0, 0, 0, /* [18] ビットマップの幅*/
16, 0, 0, 0, /* [22] ビットマップの高さ*/
0x01, 0, /* [26] プレーン数*/
32, 0, /* [28] 1ピクセルあたりのビット数（課題がバイト指定されていたのでbitに変更）*/
0, 0, 0, 0, /* [30] 圧縮タイプ*/
0, 1, 0, 0, /* [34] ビットマップデータの長さ16*16=256*/
0, 0, 0, 0, /* [38] 水平解像度(px/m) */
0, 0, 0, 0, /* [42] 垂直解像度(px/m) */
0, 0, 0, 0, /* [46] カラーインデックス数*/
0, 0, 0, 0, /* [50] 重要なカラーインデックス数*/
};


        if((keyfile = fopen(keyfn, "rt")) == NULL){
//              printf("Can not find key file for encryption. ¥n");
                return;
        }
        fgets( c_mode, 4, keyfile );
        fgets( c_klen, 8, keyfile );
        fgets( c_key, 64, keyfile );
        fclose(keyfile);

        strcpy(key, c_key);
        mode = atoi(c_mode);
        klen = atoi(c_klen);

        mn = 8;
        if(klen == 128){ mn = 16; }
        if(klen == 192){ mn = 24; }
        if(klen == 256){ mn = 32; }

        strcpy(folderfile, ".¥¥Encrypt¥¥");
        strcat(folderfile, srcfn);
        if( fopen_s(&srcfile, folderfile, "rb") != 0){
//              printf("Can not open plane file. ¥n");
                return;
        }
```

```
        fseek(srcfile, 0, SEEK_END);
        long filelen = ftell(srcfile);
        fseek(srcfile,0,0);

        fread(bmpHeader, sizeof(char), 54, srcfile);//srcfile.Read(bmpHeader, 54);
        fread(tmpch, sizeof(char), 4, srcfile);//srcfile.Read(tmpch, 4);//32 bit
        jj = *((unsigned int*)&tmpch);
        j = jj^(jj>>16);
        nsize = j & 0x0000ffff;
        fread(tmpch, sizeof(char), 4, srcfile);//srcfile.Read(tmpch, 4);//32 bit
        jj = *((unsigned int*)&tmpch);
        j = jj^(jj>>16);
        k = j & 0x0000ffff;
        nsize = nsize + (k<<16);

        fread(tmpch, sizeof(char), 4, srcfile);//srcfile.Read(tmpch, 4);//32 bit
        jj = *((unsigned int*)&tmpch);
        j = jj^(jj>>16);
        fsize = j & 0x0000ffff;
        fread(tmpch, sizeof(char), 4, srcfile);//srcfile.Read(tmpch, 4);//32 bit
        jj = *((unsigned int*)&tmpch);
        j = jj^(jj>>16);
        k = j & 0x0000ffff;
        fsize = fsize + (k<<16);

        pfName = new(char[nsize+8]);
        if(pfName == NULL){
//              printf("メモリー不足です。¥n");
                return;
        }

        for(i=0; i<nsize/2 ; i++){
                fread(tmpch, sizeof(char), 4, srcfile);//srcfile.Read(tmpch, 4);//32 bit
                jj = *((unsigned int*)&tmpch);
                j = jj^(jj>>16);
                k = j & 0x0000ffff;
                pfName[2*i] = (char)(k>>8);
                pfName[2*i+1] = (char)(k&0x000000ff);
        }
        for(i=nsize/2; i<(nsize+5)/2 ; i++){
                pfName[2*i] = NULL;
                pfName[2*i+1] = NULL;
        }
        if(nsize%2 == 0){
                fread(tmpch, sizeof(char), 4, srcfile);//srcfile.Read(tmpch, 4);//32 bit , pointer  を
進める
        }
        if(nsize%2 == 1){
                fread(tmpch, sizeof(char), 4, srcfile);//srcfile.Read(tmpch, 4);//32 bit
                jj = *((unsigned int*)&tmpch);
                j = jj^(jj>>16);
```

```
                k = j & 0x0000ffff;
                pfName[nsize-1] = (char)(k>>8);
        }

        strcpy(FName, ".¥¥Decrypt¥¥");
        strcat(FName, pfName);
        if((dstfile = fopen(FName, "wb")) == NULL){
            if(0 == _mkdir(".¥¥Decrypt")){
//                      printf("サブフォルダーDecrypt を作りました。¥n");
                        if((dstfile = fopen(FName, "wb")) == NULL){
                                return;
                        }
                }
                else{
                        return;
                }
        }

        for(i=0 ; i<fsize ; i+=2){
                q = fsize - i - 2;
                if(q >= 0){
                        if(4 == fread(tmpch, sizeof(char), 4, srcfile)){//.Read(tmpch, 4)){
                                jj = *((unsigned int*)&tmpch);//32 bit
                                j = jj^(jj>>16);
                                k = j & 0x0000ffff;
                                tmprbuf[0] = (unsigned char)(k>>8);
                                tmprbuf[0] ^= (unsigned char)key[(i/2)%mn];
                                tmprbuf[1] = (unsigned char)(k&0x000000ff);
                                tmprbuf[1] ^= (unsigned char)key[(i/2)%mn];
                                fwrite( tmprbuf,sizeof(char),2, dstfile);//dstfile.Write(tmprbuf, 2);
                        }
                }
                if(q < 0){ // q==-1
                        if(4 == fread(tmpch, sizeof(char), 4, srcfile)){//srcfile.Read(tmpch, 4)){
                                jj = *((unsigned int*)&tmpch);//32 bit
                                j = jj^(jj>>16);
                                k = j & 0x0000ffff;
                                tmprbuf[0] = (unsigned char)(k>>8);
                                tmprbuf[0] ^= (unsigned char)key[(i/2)%mn];
                                fwrite( tmprbuf,sizeof(char),1, dstfile);//dstfile.Write(tmprbuf, 1);
                        }
                }
        }

        fclose(srcfile);//.Close();
        fclose(dstfile);//.Close();

}
```

Bmpkey.cpp

```
////////////////////////////
// BmpKey.cpp : アプリケーションのクラス動作を定義します。
//

#include "stdafx.h"
#include "BmpKey.h"
#include "BmpKeyDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#endif


// CBmpKeyApp

BEGIN_MESSAGE_MAP(CBmpKeyApp, CWinApp)
	ON_COMMAND(ID_HELP, &CWinApp::OnHelp)
END_MESSAGE_MAP()


// CBmpKeyApp コンストラクション

CBmpKeyApp::CBmpKeyApp()
{
	// TODO: この位置に構築用コードを追加してください。
	// ここにInitInstance 中の重要な初期化処理をすべて記述してください。
}


// 唯一のCBmpKeyApp オブジェクトです。

CBmpKeyApp theApp;


// CBmpKeyApp 初期化

BOOL CBmpKeyApp::InitInstance()
{
	// アプリケーションマニフェストがvisual スタイルを有効にするために、
	// ComCtl32.dll Version 6 以降の使用を指定する場合は、
	// Windows XP にInitCommonControlsEx() が必要です。さもなければ、ウィンドウ作成はすべて失敗し
ます。
	INITCOMMONCONTROLSEX InitCtrls;
	InitCtrls.dwSize = sizeof(InitCtrls);
	// アプリケーションで使用するすべてのコモンコントロールクラスを含めるには、
	// これを設定します。
	InitCtrls.dwICC = ICC_WIN95_CLASSES;
	InitCommonControlsEx(&InitCtrls);
```

```
	CWinApp::InitInstance();

	AfxEnableControlContainer();

	// 標準初期化
	// これらの機能を使わずに最終的な実行可能ファイルの
	// サイズを縮小したい場合は、以下から不要な初期化
	// ルーチンを削除してください。
	// 設定が格納されているレジストリキーを変更します。
	// TODO: 会社名または組織名などの適切な文字列に
	// この文字列を変更してください。
	SetRegistryKey(_T("アプリケーションウィザードで生成されたローカルアプリケーション"));

	CBmpKeyDlg dlg;
	m_pMainWnd = &dlg;
	INT_PTR nResponse = dlg.DoModal();
	if (nResponse == IDOK)
	{
		// TODO: ダイアログが<OK> で消された時のコードを
		//  記述してください。
	}
	else if (nResponse == IDCANCEL)
	{
		// TODO: ダイアログが<キャンセル> で消された時のコードを
		//  記述してください。
	}

	// ダイアログは閉じられました。アプリケーションのメッセージポンプを開始しないで
	//  アプリケーションを終了するためにFALSE を返してください。
	return FALSE;
}
```

Bmpkeydlg.cpp

```
////////////////////////////
// BmpKeyDlg.cpp : 実装ファイル
//

#include "stdafx.h"
#include "BmpKey.h"
#include "BmpKeyDlg.h"

#include "fstream"
#include <iostream>

using namespace std;

FILE *stream0;//plane
FILE *stream1;//encrypt
```

```
FILE *stream2;//decrypt

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

unsigned char k[128];
unsigned char e[256];
//typedef unsigned long Word;


// 暗号文のHEX表示用
char toChar( int c )
{
        if( c >= 0 && c <= 9 )                  // 0～ならば
                return char( c + 0x30 ); // ASCIIに変換して返す
        else if( c >= 10 && c <= 15 )     // 10～ならば
                return char( c + 0x37 ); // A～FのASCIIを返す
        else
                return ' ';
}

// アプリケーションのバージョン情報に使われるCAboutDlg ダイアログ

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

// ダイアログデータ
        enum { IDD = IDD_ABOUTBOX };

        protected:
        virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV サポート

// 実装
protected:
        DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
}

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

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
END_MESSAGE_MAP()
```

```
// CBmpKeyDlg ダイアログ



CBmpKeyDlg::CBmpKeyDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CBmpKeyDlg::IDD, pParent)
{
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);

	i_KeyLen = 0;
	i_Mode = 0;
	i_Disp = 0;
	s_fname1 = "BmpkeyEC.bin";
	s_fname2 = "BmpkeyDC.bin";
	srand( (unsigned)time(NULL) );//乱数初期化
	madekey = 0;
	planedata_file = "plane.txt";
	encryptdata_file = "test.bmp";
	decryptdata_file = "平文名を使用";

}

void CBmpKeyDlg::OnOK()
{
	if(madekey==0){
		MessageBox("鍵を作成してください。");
		return;
	}

	CDialog::OnOK();

	ofstream ofs(s_fname1, ios::out);
	if(i_Mode == 0) ofs << 1 << endl;
	if(i_Mode == 1) ofs << 2 << endl;
	if(i_KeyLen == 0) ofs << 128 << endl;
	if(i_KeyLen == 1) ofs << 192 << endl;
	if(i_KeyLen == 2) ofs << 256 << endl;
	ofs << s_key << endl;
	ofs.close();

	ofstream ofs2(s_fname2, ios::out);
	if(i_Mode == 0) ofs2 << 1 << endl;
	if(i_Mode == 1) ofs2 << 2 << endl;
	if(i_KeyLen == 0) ofs2 << 128 << endl;
	if(i_KeyLen == 1) ofs2 << 192 << endl;
	if(i_KeyLen == 2) ofs2 << 256 << endl;
	ofs2 << s_key << endl;
	ofs2.close();
}
```

```
void CBmpKeyDlg::DoDataExchange(CDataExchange* pDX)
{
        CDialog::DoDataExchange(pDX);

        DDX_Radio(pDX, IDC_RADIO1, i_KeyLen);
        DDX_Radio(pDX, IDC_RADIO4, i_Mode);
        DDX_Radio(pDX, IDC_RADIO6, i_Disp);

        DDX_Text(pDX, IDC_SECRETKEY, s_key);
        DDX_Text(pDX, IDC_ENCRYPTKEY_FILE, s_fname1);
        DDX_Text(pDX, IDC_DECRYPTKEY_FILE, s_fname2);
        DDX_Text(pDX, IDC_PLANEDATA_FILE, planedata_file);
    DDV_MaxChars(pDX, planedata_file, 128);
    DDX_Text(pDX, IDC_ENCRYPTDATA_FILE, encryptdata_file);
    DDV_MaxChars(pDX, encryptdata_file, 128);
    DDX_Text(pDX, IDC_DECRYPTDATA_FILE, decryptdata_file);
    DDV_MaxChars(pDX, decryptdata_file, 128);
        DDX_Control(pDX, IDC_DISPLAY, datadisp);
        //}}AFX_DATA_MAP
}


BEGIN_MESSAGE_MAP(CBmpKeyDlg, CDialog)
        ON_WM_SYSCOMMAND()
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
        ON_COMMAND(ID_MAKEKEY, MakeKey)
        ON_COMMAND(ID_TESTKEY, TestKey)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()


// CBmpKeyDlg メッセージハンドラ

void CBmpKeyDlg::yDisplay(const char *cp)
{
  int nEditLength = datadisp.GetWindowTextLength();
  datadisp.SetSel(nEditLength, nEditLength); //テキストの終端にカーソルを移動する
  datadisp.ReplaceSel(cp); //新しいテキストを追加する
}



void CBmpKeyDlg::MakeKey()
{
        int i, n;
        char j , k;
        char a[64];
        char b[128];
        for(i=0; i<128; i++){ b[i] = 0;}
```

```
	if(IDC_RADIO1 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO3) ){
		n= 16;
	}
	if(IDC_RADIO2 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO3) ){
		n= 24;
	}
	if(IDC_RADIO3 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO3) ){
		n= 32;
	}

	for(i=0; i<n ; i++){
		*(a+i) = (char)rand();
	}
	a[n] = NULL;

	for(i=0; i<n; i++){
		j = *(a+i);
		k = (j>>4)&0x0f;
		if(k>=0 && k<=9) b[2*i] = k + 0x30;
		else if(k>=10 && k<=15) b[2*i] = k + 0x37;
		k = j & 0x0f;
		if(k>=0 && k<=9) b[2*i+1] = k + 0x30;
		else if(k>=10 && k<=15) b[2*i+1] = k + 0x37;
	}
	b[2*n] = NULL;

	s_key = b;

	CWnd * pwnd = GetDlgItem(IDC_SECRETKEY);
	pwnd->SetWindowText(b);

	madekey = 1;
}


void CBmpKeyDlg::TestKey()
{
	char lencrypt[256], ldecrypt[256];
	int  cnt, c, block, i, j;
	long filelen, mesLength; // 平文長（バイト）
	char* bufp;
	unsigned char* bufc;
	char* bufdc;
	int n, lmode;
	int show;
	CWnd* pwnd;
	CFileStatus fs;

	datadisp.SetLimitText(INT_MAX);

	if(IDC_RADIO1 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO3) )
	{
```

```
        n= 16;                  keylength = "128";
}
if(IDC_RADIO2 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO3) )
{
        n= 24;                  keylength = "192";
}
if(IDC_RADIO3 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO3) )
{
        n= 32;                  keylength = "256";
}

if(IDC_RADIO4 == GetCheckedRadioButton(IDC_RADIO4, IDC_RADIO5) )
{
        lmode = 0;
}
if(IDC_RADIO5 == GetCheckedRadioButton(IDC_RADIO4, IDC_RADIO5) )
{
        lmode = 1;
}

if(madekey == 0){               // 鍵の生成
        yDisplay("¥r¥n");
        yDisplay("鍵を生成中¥r¥n");
        yDisplay("鍵長    = "); yDisplay(keylength); yDisplay(" bit");
        yDisplay("¥r¥n");
        yDisplay("¥r¥n");
MakeKey();
}

ofstream ofs("tmp.key", ios::out);
if(lmode == 0) ofs << 0 << endl;
if(lmode == 1) ofs << 1 << endl;
if(n == 16) ofs << 128 << endl;
if(n == 24) ofs << 192 << endl;
if(n == 32) ofs << 256 << endl;
ofs << s_key << endl;
ofs.close();

char lplan_file[256];
char lencry_file[256];
char ldecry_file[256];

pwnd = GetDlgItem(IDC_PLANEDATA_FILE);
pwnd->GetWindowText(planedata_file);
pwnd = GetDlgItem(IDC_ENCRYPTDATA_FILE);
pwnd->GetWindowText(encryptdata_file);
pwnd = GetDlgItem(IDC_DECRYPTDATA_FILE);
pwnd->GetWindowText(decryptdata_file);

strcpy(lplan_file, planedata_file);
strcpy(lencry_file, encryptdata_file);
strcpy(ldecry_file, decryptdata_file);
```

```
	bmp1("tmp.key", lplan_file, lencry_file);
	bmp2("tmp.key", lencry_file, ldecry_file);

	if(IDC_RADIO6 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO7) ){
		show = 0;
	}
	if(IDC_RADIO7 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO7) ){
		show = 1;
	}
	if(show == 0){
		yDisplay("終了");
		return;
	}



/* 読み出すファイルを開く
 *（ファイルが存在しないときは、呼び出しが失敗）
 */
	pwnd = GetDlgItem(IDC_PLANEDATA_FILE);
	pwnd->GetWindowText(planedata_file);
	if( (stream0 = fopen( planedata_file, "rb" )) == NULL ){
		yDisplay("ファイル"); yDisplay(planedata_file); yDisplay(" は開けませんでした。¥r¥n");
		return;//(-1);
		}
else
		{yDisplay("ファイル"); yDisplay(planedata_file); yDisplay(" は開けました。¥r¥n");}

 // 平文
	CFile::GetStatus(planedata_file, fs);
	filelen = fs.m_size;
	mesLength = filelen + sizeof(long);
	block = mesLength/16 + ((mesLength%16)?1:0);
	bufp = (char*)new(char[block*16 + 1]);
	if(bufp == NULL){
		yDisplay("メモリ不足¥r¥n");
		return;//(-1);
		}
	for(j=0; j<(block*16 + 1); j++){
		bufp[j] = 0;
	}
	*(long*)(bufp) = filelen;
	fseek(stream0,0,0);
	i = sizeof(long);
	do{
		c = fgetc(stream0);
		bufp[i]=c;
		i=i+1;
	}while(c!=EOF);
	bufp[i-1]=NULL;

	mesLength = i-1; // 平文長（バイト）+ sizeof(long)
```

```
        char  *sd;
        CString  Ssd;

        sd = (char*)new(char[256]);
        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->GetWindowText(Ssd);
        strcpy(sd, Ssd);

        yDisplay("秘密鍵  = "); yDisplay(sd); yDisplay("¥r¥n");
        yDisplay("¥r¥n");

        char buff[16];
        itoa(mesLength-4, buff, 10);
    yDisplay("平文長  = "); yDisplay(buff); yDisplay(" byte¥r¥n");
        yDisplay("¥r¥n");
        yDisplay("¥r¥n");

    yDisplay("平文    = ¥r¥n");
        yDisplay(bufp+sizeof(long));
        yDisplay("¥r¥n");
        yDisplay("¥r¥n");



// 暗号文
        strcpy(lencrypt, ".¥¥Encrypt¥¥");
        strcat(lencrypt, encryptdata_file);

        pwnd = GetDlgItem(IDC_ENCRYPTDATA_FILE);
        pwnd->GetWindowText(encryptdata_file);
        if( (stream1  = fopen(lencrypt, "rb" )) == NULL ){
                yDisplay("ファイル"); yDisplay(lencrypt); yDisplay(" は開けませんでした。¥r¥n");
                return;//(-1);
                }
   else
                {yDisplay("ファイル"); yDisplay(lencrypt); yDisplay(" は開けました。¥r¥n");}

        CString ecStr;
        ecStr = lencrypt;

        CFile::GetStatus(ecStr, fs);
        filelen = fs.m_size;
        mesLength = filelen + sizeof(long);
        block = mesLength/16 + ((mesLength%16)?1:0);
        bufc = (unsigned char*)new(char[block*16 + 1]);
        if(bufc == NULL){
                yDisplay("メモリ不足¥r¥n");
                return;//(-1);
                }
        for(j=0; j<(block*16 + 1); j++){
                bufc[j] = 0;
```

```
        }
        *(long*)(bufc) = filelen;
        fseek(stream1,0,0);
        i = sizeof(long);
        do{
                c = fgetc(stream1);
                bufc[i]=c;
                i=i+1;
        }while(c!=EOF);
        bufc[i-1]=NULL;

        mesLength = filelen; // 平文長（バイト）+ sizeof(long)

        // 暗号文を進数で表示 0xa4  は  A4  と表示される
    yDisplay("暗号文(HEX)= ");
        for( cnt = 0; cnt<(block*16); cnt++ ){
                char c1, c2;
                if(cnt%30 ==0){yDisplay("¥r¥n");}

                c1 = toChar((int(bufc[cnt]) >> 4) & 0xf);
                c2 = toChar(int(bufc[cnt]) & 0xf);
                CString sc = (CString)c1;
                sc += c2;
                yDisplay(sc);
                fwrite( &(bufc[cnt]),sizeof(char),1, stream1);
        }
        yDisplay("¥r¥n");
        yDisplay("¥r¥n");


// 復号文
        char lpname[256];
        strcpy(lpname, planedata_file);
        strcpy(ldecrypt, ".¥¥Decrypt¥¥");
        strcat(ldecrypt, lpname);

        if( (stream2  = fopen(ldecrypt, "rb" )) == NULL ){
                yDisplay("ファイル"); yDisplay(decryptdata_file); yDisplay(" は開けませんでした。
¥r¥n");
                return;//(-1);
                }
   else
                {yDisplay("ファイル"); yDisplay(decryptdata_file); yDisplay(" は開けました。¥r¥n");}

        CFile::GetStatus(decryptdata_file, fs);
        filelen = fs.m_size;

        mesLength = filelen + sizeof(long);
        block = mesLength/16 + ((mesLength%16)?1:0);
        bufdc = (char*)new(char[block*16 + 1]);
        if(bufdc == NULL){
                yDisplay("メモリ不足¥r¥n");
```

```
                return;//(-1);
                }
        for(j=0; j<(block*16 + 1); j++){
                bufdc[j] = 0;
        }
        *(long*)(bufdc) = filelen;
        fseek(stream2,0,0);
        i = sizeof(long);
        do{
                c = fgetc(stream2);
                bufdc[i]=c;
                i=i+1;
        }while(c!=EOF);
        bufdc[i-1]=NULL;

        // 復号文出力
    yDisplay("復号文  = ¥r¥n");
        yDisplay(bufdc+sizeof(long));
        yDisplay("¥r¥n");


        if( fclose( stream0 ) )
      MessageBox( "ファイル'p-data' は閉じられませんでした。¥n" );
        if( fclose( stream1 ) )
      MessageBox( "ファイル'c-data' は閉じられませんでした。¥n" );
        if( fclose( stream2 ) )
      MessageBox( "復号化ファイルは閉じられませんでした。¥n" );

        delete[] bufp;
        delete[] bufdc;
        delete[] bufc;

        return;// 0;
}


BOOL CBmpKeyDlg::OnInitDialog()
{
        CDialog::OnInitDialog();

        // "バージョン情報..." メニューをシステムメニューに追加します。

        // IDM_ABOUTBOX は、システムコマンドの範囲内になければなりません。
        ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
        ASSERT(IDM_ABOUTBOX < 0xF000);

        CMenu* pSysMenu = GetSystemMenu(FALSE);
        if (pSysMenu != NULL)
        {
                CString strAboutMenu;
                strAboutMenu.LoadString(IDS_ABOUTBOX);
                if (!strAboutMenu.IsEmpty())
```

```
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
		}
	}

	// このダイアログのアイコンを設定します。アプリケーションのメインウィンドウがダイアログでない
場合、
	//  Framework は、この設定を自動的に行います。
	SetIcon(m_hIcon, TRUE);			// 大きいアイコンの設定
	SetIcon(m_hIcon, FALSE);		// 小さいアイコンの設定

	// TODO: 初期化をここに追加します。

	return TRUE;  // フォーカスをコントロールに設定した場合を除き、TRUE を返します。
}

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

// ダイアログに最小化ボタンを追加する場合、アイコンを描画するための
//  下のコードが必要です。ドキュメント/ビューモデルを使うMFC アプリケーションの場合、
//  これは、Framework によって自動的に設定されます。

void CBmpKeyDlg::OnPaint()
{
	if (IsIconic())
	{
		CPaintDC dc(this); // 描画のデバイスコンテキスト

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

		// クライアントの四角形領域内の中央
		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;

		// アイコンの描画
		dc.DrawIcon(x, y, m_hIcon);
```

```
	}
	else
	{
		CDialog::OnPaint();
	}
}

// ユーザーが最小化したウィンドウをドラッグしているときに表示するカーソルを取得するために、
//  システムがこの関数を呼び出します。
HCURSOR CBmpKeyDlg::OnQueryDragIcon()
{
	return static_cast<HCURSOR>(m_hIcon);
}
```

Stdafx.cpp

```
//////////////////////////////
// stdafx.cpp : 標準インクルードBmpKey.pch のみを
// 含むソースファイルは、プリコンパイル済みヘッダーになります。
// stdafx.obj にはプリコンパイルされた型情報が含まれます。

#include "stdafx.h"
```

おわり。