AESKey のソースファイル

(C) 宇山　靖政

2012.04.11

AEScrypt は以下のソースコードを、AESKey のソースコードに追加したものです。

```
void CAESKeyDlg::DCKRead()
{
	int i,k, Keylen2, Blocklen2;

	CString strNewFileName;
	CString sFName;
	CWnd* pwnd;


	strNewFileName = "*.*";
	CFileDialog fileDlg(TRUE, NULL,"*.bin", OFN_HIDEREADONLY, strNewFileName);
	if(fileDlg.DoModal() == IDOK){
		strNewFileName = fileDlg.GetPathName();

		// If file doesn't already exist, then create it.
		CFile file;
		CFileStatus status;
		if (!file.GetStatus(strNewFileName, status)){
			CString strMessage;
//			AfxFormatString1(strMessage, IDS_MAKENEWFILE,
//				strNewFileName);
			if(AfxMessageBox(strMessage, MB_YESNO) == IDNO)    {
				return;
			}
			if (!file.Open(strNewFileName, CFile::modeCreate)){
				CString strMessage;
//				AfxFormatString1(strMessage, IDS_NOADBOOKMAILBOXTEMPLATE,
//					strNewFileName);
				AfxMessageBox(strMessage);
				return;
			}
			file.Close();
		}
	}

	FILE *fkey = 0;
	char c_klen[8], c_blen[8], c_skey[128];

	if((fkey = fopen(strNewFileName, "rt")) == NULL){
		MessageBox( "Can not find key file for encryption. ¥n");
		return ;
	}

	fgets( c_klen, 8, fkey );
	fgets( c_blen, 8, fkey );
```

```
	fgets( c_skey, 128, fkey );

	if(fkey){ fclose(fkey);  }

	i = atoi(c_klen);
	k = i;

	if(i == 128 ){
		Keylen2 = 128; keylength = "128";
		CheckRadioButton(IDC_RADIO1, IDC_RADIO5, IDC_RADIO1);
	}else{
		if(i==160){
			Keylen2 = 160; keylength = "160";
			CheckRadioButton(IDC_RADIO1, IDC_RADIO5, IDC_RADIO2);
		}else{
			if(i==192){
				Keylen2 = 192; keylength = "192";
				CheckRadioButton(IDC_RADIO1, IDC_RADIO5, IDC_RADIO3);
			}else{
				if(i==224){
					Keylen2 = 224; keylength = "224";
					CheckRadioButton(IDC_RADIO1, IDC_RADIO5, IDC_RADIO4);
				}else{
					if(i==256){
						Keylen2 = 256; keylength = "256";
						CheckRadioButton(IDC_RADIO1, IDC_RADIO5,
IDC_RADIO5);
					}else{
						MessageBox( "鍵の長さが不正です。¥n");
						return ;
					}
				}
			}
		}
	}

	i = atoi(c_blen);
	if(i == 128 ){
		Blocklen2 = 128; s_blocklength = "128";
		CheckRadioButton(IDC_RADIO6, IDC_RADIO10, IDC_RADIO6);
	}else{
		if(i==160){
			Blocklen2 = 160;  s_blocklength = "160";
			CheckRadioButton(IDC_RADIO6, IDC_RADIO10, IDC_RADIO7);
		}else{
			if(i==192){
				Blocklen2 = 192; s_blocklength = "192";
				CheckRadioButton(IDC_RADIO6, IDC_RADIO10, IDC_RADIO8);
			}else{
				if(i==224){
					Blocklen2 = 224; s_blocklength = "224";
					CheckRadioButton(IDC_RADIO6, IDC_RADIO10, IDC_RADIO9);
```

```
                                }else{
                                        if(i==256){
                                                Blocklen2 = 256; s_blocklength = "256";
                                                CheckRadioButton(IDC_RADIO6, IDC_RADIO10,
IDC_RADIO10);
                                        }else{
                                                MessageBox( "ブロックの長さが不正です。¥n");
                                                return ;
                                        }
                                }
                        }
                }
        }

//      c_skey[k/4] = NULL;
        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->SetWindowText(c_skey);

        sFName = strNewFileName;
        int jj = sFName.ReverseFind( '¥¥' );
        sFName.Delete(0,jj+1);
        pwnd = GetDlgItem(IDC_DECRYPTKEY_FILE);
        pwnd->SetWindowText(sFName);
        pwnd = GetDlgItem(IDC_ENCRYPTKEY_FILE);
        pwnd->SetWindowText(sFName);

        readdckey = 1;
        madekey = 1;

        return ;

        // Open the file now that it has been created.
//      strcpy_s(tmppath, strNewFileName);
//      OpenDocumentFile(strNewFileName);


}



void CAESKeyDlg::Search2(){
        CString strNewFileName;

        strNewFileName.LoadString(IDS_BSEARCH);
        CFileDialog fileDlg(TRUE, NULL, NULL, OFN_HIDEREADONLY, strNewFileName);
        if(fileDlg.DoModal() == IDOK){
                strNewFileName = fileDlg.GetPathName();
            l_bin.AddString(strNewFileName);
        }
}

void CAESKeyDlg::AddList2(){
```

```
        CWnd* pwnd;
        pwnd = GetDlgItem(IDC_ADDBIN2);
        pwnd->GetWindowText(s_addbin);
    l_bin.AddString(s_addbin);
        s_addbin = "";
        pwnd->SetWindowText(s_addbin);
}

void CAESKeyDlg:: DelList2(){
        int index = l_bin.GetCurSel();
        l_bin.DeleteString((UINT)index);
}

void CAESKeyDlg::OnBnClickedButton5(){//暗号化（連続）
        int  cnt, c, block, i,Keylen2=128, Blocklen2=128;
        char Key[128+2];
        char inputfile[256];
        char outputfile[256];
        long filelen, mesLength; // 平文長（バイト）
        char* bufp;
        unsigned char* bufc;
        int numclosed;

        CWnd* pwnd;
        CFileStatus fs, fsec, fsdc;

        datadisp.SetLimitText(INT_MAX);

        if(IDC_RADIO1 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 128; keylength = "128";
        }
        if(IDC_RADIO2 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 160; keylength = "160";
        }
        if(IDC_RADIO3 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 192; keylength = "192";
        }
        if(IDC_RADIO4 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 224; keylength = "224";
        }
        if(IDC_RADIO5 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 256; keylength = "256";
        }


        if(IDC_RADIO6 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
```

```
		Blocklen2 = 128; s_blocklength = "128";
	}
	if(IDC_RADIO7 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 160; s_blocklength = "160";
	}
	if(IDC_RADIO8 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 192; s_blocklength = "192";
	}
	if(IDC_RADIO9 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 224; s_blocklength = "224";
	}
	if(IDC_RADIO10 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 256; s_blocklength = "256";
	}

////////////////////////////////////////////////////////////////
		yDisplay("¥r¥n");
		yDisplay("鍵長    = "); yDisplay(keylength); yDisplay(" bit");
		yDisplay("¥r¥n");
		yDisplay("ブロック長 = "); yDisplay(s_blocklength); yDisplay(" bit");
		yDisplay("¥r¥n");
		yDisplay("¥r¥n");

	SetCurrentDirectory(bufpath);

	char buf[256];
	int i2, n2;
	n2 = l_bin.GetCount();

	for(i2=0; i2<n2; i2++){
	l_bin.SetCurSel(i2);
		l_bin.GetText(i2,buf);
////////////////////////////////////////////////////////////
	planedata_file = buf;

 /* 復文を書き込むファイルの名前*/
	CString sFName = planedata_file;
	int j = sFName.ReverseFind( '¥¥' );
	sFName.Delete(0,j+1);
	encryptdata_file = ".¥¥Encrypted¥¥";
	encryptdata_file += sFName;

	sFName = encryptdata_file;
	j = sFName.ReverseFind( '¥¥' );
	sFName.Delete(0,j+1);
	decryptdata_file = ".¥¥Decrypted¥¥";
	decryptdata_file += sFName;
```

```
        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->GetWindowText(s_key);
        strcpy_s(Key, s_key);
        strcpy_s(inputfile, planedata_file);
        strcpy_s(outputfile, encryptdata_file);

        rc=AES_Encrypt(Keylen2, Blocklen2, Key, inputfile, outputfile);


// 読み出すファイルを開く
        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");}

   /* 暗号文を読み出すファイルを開く*/
        if( (stream1 = fopen( encryptdata_file, "rb" )) == NULL ){
                yDisplay("ファイル"); yDisplay(encryptdata_file); yDisplay(" は開けませんでした。
¥r¥n");
                return;//(-1);
                }
   else
                {yDisplay("ファイル"); yDisplay(encryptdata_file); yDisplay(" は開けました。¥r¥n");}

        // Key disp
        char sd[256];
        CString  Ssd;

        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->GetWindowText(Ssd);
        strcpy_s(sd, Ssd);

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

```
  // 平文
        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 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");


// 暗文
        CFile::GetStatus(encryptdata_file, fsec);
        filelen = fsec.m_size;

        bufc = (unsigned char*)new(unsigned char[filelen + 2]);
        if(bufc == 0){
                yDisplay("メモリ不足¥r¥n");
                return;//(-1);
                }
        fseek(stream1,0,0);
        int cc;
        i=0;
        do{
                cc = fgetc(stream1);
                bufc[i]=cc;
                i=i+1;
        }while(cc!=EOF);
        bufc[i-1]=NULL;

        mesLength = filelen;
        block = mesLength/16 + ((mesLength%16)?1:0);
// 暗文
        // 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);
        }
        yDisplay("¥r¥n");
        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" );


   /* 他のすべてのファイルを閉じる*/
        numclosed = _fcloseall( );

        delete[] bufp;
        delete[] bufc;
        ////////////////////////////////////////////////////////////////////
        }
//////////////////////////////////////////////////////////////////////////


        yDisplay("¥r¥n");
        yDisplay("暗号化終了。¥n");
        yDisplay("¥r¥n");

        return;


}


void CAESKeyDlg::OnBnClickedButton6(){//復号化（連続）
        int  cnt, c, block, i, Keylen2=128, Blocklen2=128;
        char Key[128+2];
        char inputfile[256];
        char outputfile[256];
        unsigned long filelen, mesLength; // 平文長（バイト）
        unsigned char* bufc;
        char* bufdc;
        int numclosed;

        CWnd* pwnd;
        CFileStatus fs, fsec, fsdc;
```

```
	datadisp.SetLimitText(INT_MAX);

	if(IDC_RADIO1 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
	{
		Keylen2 = 128; keylength = "128";
	}
	if(IDC_RADIO2 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
	{
		Keylen2 = 160; keylength = "160";
	}
	if(IDC_RADIO3 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
	{
		Keylen2 = 192; keylength = "192";
	}
	if(IDC_RADIO4 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
	{
		Keylen2 = 224; keylength = "224";
	}
	if(IDC_RADIO5 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
	{
		Keylen2 = 256; keylength = "256";
	}


	if(IDC_RADIO6 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 128; s_blocklength = "128";
	}
	if(IDC_RADIO7 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 160; s_blocklength = "160";
	}
	if(IDC_RADIO8 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 192; s_blocklength = "192";
	}
	if(IDC_RADIO9 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 224; s_blocklength = "224";
	}
	if(IDC_RADIO10 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
	{
		Blocklen2 = 256; s_blocklength = "256";
	}


////////////////////////////////////////////////////////////////

		yDisplay("¥r¥n");
		yDisplay("鍵長    = "); yDisplay(keylength); yDisplay(" bit");
		yDisplay("¥r¥n");
		yDisplay("ブロック長 = "); yDisplay(s_blocklength); yDisplay(" bit");
```

```
                yDisplay("¥r¥n");
                yDisplay("¥r¥n");


        SetCurrentDirectory(bufpath);

        char buf[256];
        int n2,m;
        n2 = l_bin.GetCount();

        ////////////////////////////////////////////////////////////////////
        for(m=0; m<n2; m++){
        l_bin.SetCurSel(m);
                l_bin.GetText(m,buf);
//////////////////////////////////////////////////
        encryptdata_file = buf;

 /* 復文を書き込むファイルの名前*/
        CString sFName = encryptdata_file;
        int j = sFName.ReverseFind( '¥¥' );
        sFName.Delete(0,j+1);
        decryptdata_file = ".¥¥Decrypted¥¥";
        decryptdata_file += sFName;

        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->GetWindowText(s_key);
        strcpy_s(Key, s_key);

        strcpy_s(inputfile, encryptdata_file);
        strcpy_s(outputfile, decryptdata_file);
    rc=AES_Decrypt(Keylen2, Blocklen2, Key, inputfile, outputfile);



   /* 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）
    */

   /* 暗号文を読み出すファイルを開く*/
        if( (stream1 = fopen( encryptdata_file, "rb" )) == NULL ){
                yDisplay("ファイル"); yDisplay(encryptdata_file); yDisplay(" は開けませんでした。
¥r¥n");
                return;//(-1);
                }
   else
                {yDisplay("ファイル"); yDisplay(encryptdata_file); yDisplay(" は開けました。¥r¥n");}

   /* 復号文を読み出すファイルを開く*/
        if( (stream2 = fopen( decryptdata_file, "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");}

        // Key disp
        char sd[256];
        CString  Ssd;

        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->GetWindowText(Ssd);
        strcpy_s(sd, Ssd);

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


// 暗文
        CFile::GetStatus(encryptdata_file, fsec);
        filelen = fsec.m_size;

        bufc = (unsigned char*)new(unsigned char[filelen + 2]);
        if(bufc == 0){
                yDisplay("メモリ不足¥r¥n");
                return;//(-1);
                }
        fseek(stream1,0,0);
        int cc;
        i=0;
        do{
                cc = fgetc(stream1);
                bufc[i]=cc;
                i=i+1;
        }while(cc!=EOF);
        bufc[i-1]=NULL;

        mesLength = filelen;
        block = mesLength/16 + ((mesLength%16)?1:0);
// 暗文
        // 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);
        }
        yDisplay("¥r¥n");
        yDisplay("¥r¥n");
```

```
        if( fclose( stream1 ) )
      MessageBox( "ファイル'c-data' は閉じられませんでした。¥n" );

  // 復文
        CFile::GetStatus(decryptdata_file, fsdc);
        filelen = fsdc.m_size;

        bufdc = (char*)new(char[filelen + 2]);
        if(bufdc == 0){
                yDisplay("メモリ不足¥r¥n");
                return;//(-1);
                }
        fseek(stream2,0,0);
        i = 0;
        do{
                c = fgetc(stream2);
                bufdc[i]=c;
                i=i+1;
        }while(c!=EOF);
        bufdc[i-1]=NULL;

    yDisplay("復文    = ¥r¥n");
        yDisplay(bufdc);
        yDisplay("¥r¥n");
        yDisplay("¥r¥n");

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


   /* 他のすべてのファイルを閉じる*/
        numclosed = _fcloseall( );

        delete[] bufdc;
        delete[] bufc;
////////////////////////////////////////////////////
        }
        ////////////////////////////////////////////

        yDisplay("¥r¥n");
        yDisplay("復号化終了。¥n");
        yDisplay("¥r¥n");

        return;// 0;

}




void CAESKeyDlg::OnBnClickedButton1(){//暗号化（連続　非表示）
```

```
int  Keylen2=128, Blocklen2=128;
char Key[128+2];
char inputfile[256];
char outputfile[256];
int numclosed;


CWnd* pwnd;
CFileStatus fs, fsec, fsdc;

datadisp.SetLimitText(INT_MAX);

if(IDC_RADIO1 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
{
        Keylen2 = 128; keylength = "128";
}
if(IDC_RADIO2 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
{
        Keylen2 = 160; keylength = "160";
}
if(IDC_RADIO3 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
{
        Keylen2 = 192; keylength = "192";
}
if(IDC_RADIO4 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
{
        Keylen2 = 224; keylength = "224";
}
if(IDC_RADIO5 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
{
        Keylen2 = 256; keylength = "256";
}


if(IDC_RADIO6 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
{
        Blocklen2 = 128; s_blocklength = "128";
}
if(IDC_RADIO7 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
{
        Blocklen2 = 160; s_blocklength = "160";
}
if(IDC_RADIO8 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
{
        Blocklen2 = 192; s_blocklength = "192";
}
if(IDC_RADIO9 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
{
        Blocklen2 = 224; s_blocklength = "224";
}
if(IDC_RADIO10 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
{
```

```
                Blocklen2 = 256; s_blocklength = "256";
        }

////////////////////////////////////////////////////////////////
                yDisplay("¥r¥n");
                yDisplay("鍵長     = "); yDisplay(keylength); yDisplay(" bit");
                yDisplay("¥r¥n");
                yDisplay("ブロック長 = "); yDisplay(s_blocklength); yDisplay(" bit");
                yDisplay("¥r¥n");
                yDisplay("¥r¥n");

        SetCurrentDirectory(bufpath);

        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->GetWindowText(s_key);
        strcpy_s(Key, s_key);

        char buf[256];
        int i2, n2;
        n2 = l_bin.GetCount();

        for(i2=0; i2<n2; i2++){
        l_bin.SetCurSel(i2);
                l_bin.GetText(i2,buf);
////////////////////////////////////////////////////////
        planedata_file = buf;

 /* 暗文を書き込むファイルの名前*/
        CString sFName = planedata_file;
        int j = sFName.ReverseFind( '¥¥' );
        sFName.Delete(0,j+1);
        encryptdata_file = ".¥¥Encrypted¥¥";
        encryptdata_file += sFName;

        strcpy_s(inputfile, planedata_file);
        strcpy_s(outputfile, encryptdata_file);
        rc=AES_Encrypt(Keylen2, Blocklen2, Key, inputfile, outputfile);

   /* 他のすべてのファイルを閉じる*/
        numclosed = _fcloseall( );
////////////////////////////////////////////////////////////////
        }
////////////////////////////////

        yDisplay("¥r¥n");
        yDisplay("暗号化終了。¥n");
        yDisplay("¥r¥n");

        return;


}
```

```
void CAESKeyDlg::OnBnClickedButton2(){//復号化（連続　非表示）
        int  Keylen2=128, Blocklen2=128;
        char Key[128+2];
        char inputfile[256];
        char outputfile[256];
        int numclosed;
        CWnd* pwnd;
        CFileStatus fs, fsec, fsdc;

        datadisp.SetLimitText(INT_MAX);

        if(IDC_RADIO1 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 128; keylength = "128";
        }
        if(IDC_RADIO2 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 160; keylength = "160";
        }
        if(IDC_RADIO3 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 192; keylength = "192";
        }
        if(IDC_RADIO4 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 224; keylength = "224";
        }
        if(IDC_RADIO5 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 256; keylength = "256";
        }


        if(IDC_RADIO6 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 128; s_blocklength = "128";
        }
        if(IDC_RADIO7 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 160; s_blocklength = "160";
        }
        if(IDC_RADIO8 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 192; s_blocklength = "192";
        }
        if(IDC_RADIO9 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 224; s_blocklength = "224";
        }
        if(IDC_RADIO10 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
```

```
        {
                Blocklen2 = 256; s_blocklength = "256";
        }
        yDisplay("¥r¥n");
        yDisplay("鍵長    = "); yDisplay(keylength); yDisplay(" bit");
        yDisplay("¥r¥n");
        yDisplay("ブロック長 = "); yDisplay(s_blocklength); yDisplay(" bit");
        yDisplay("¥r¥n");
        yDisplay("¥r¥n");

        SetCurrentDirectory(bufpath);

        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->GetWindowText(s_key);
        strcpy_s(Key, s_key);

        char buf[256];
        int n2,m;
        n2 = l_bin.GetCount();

        ////////////////////////////////////////////////////////////////////
        for(m=0; m<n2; m++){
        l_bin.SetCurSel(m);
                l_bin.GetText(m,buf);
////////////////////////////////////////////////////
        encryptdata_file = buf;

 /* 復文を書き込むファイルの名前*/
        CString sFName = encryptdata_file;
        int j = sFName.ReverseFind( '¥¥' );
        sFName.Delete(0,j+1);
        decryptdata_file = ".¥¥Decrypted¥¥";
        decryptdata_file += sFName;

        strcpy_s(inputfile, encryptdata_file);
        strcpy_s(outputfile, decryptdata_file);
    rc=AES_Decrypt(Keylen2, Blocklen2, Key, inputfile, outputfile);

   /* 他のすべてのファイルを閉じる*/
        numclosed = _fcloseall( );
////////////////////////////////////////////////////
        }
        ////////////////////////////////////////////

        yDisplay("¥r¥n");
        yDisplay("復号化終了。¥n");
        yDisplay("¥r¥n");

        return;// 0;

}
```

以上が、AESKey のソースコードに追加したコード。

AESKey.exe は、AES 暗号 （Rijndael 暗号） の暗号鍵作成とそのテストが可能です。

ソースファイルは、ヘッダーファイルと CPP のファイルを公開します。暗号機能について理解するには、これがあれば十分です。リソースファイルはご自由にお作りください。

このソフトウェアは、ベクターから無料でダウンロードできます。

最初は、ヘッダーファイルです。

AES_ECDC.h

```
////////////////////////////////////////////////////////////////////////
/*  aes_ecdc.h  */


/*  Includes:
        Standard include files
*/

#include <stdio.h>


/*  Function protoypes  */

int AES_Encrypt(int Keylen, int Blocklen, char* key, char* inputfile, char* outputfile);
int AES_Decrypt(int Keylen, int Blocklen, char* key, char* inputfile, char* outputfile);
```

AESKey.h

```
////////////////////////////////////////////////////////////////////////////
// AESKey.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


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

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

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

// 実装

	DECLARE_MESSAGE_MAP()
};

  extern CAESKeyApp theApp;
```

AESKeyDlg.h

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

#pragma once
#include "aes_ecdc.h"

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

// ダイアログデータ
	enum { IDD = IDD_AESKEY_DIALOG };
	int madekey;
	CString keylength;

	int i_KeyLen, keylen;
	int i_BlockLen, blocklen;
	CString s_key;
	CString s_fname1;
```

```
        CString s_fname2;

        CString planedata_file;
        CString encryptdata_file;
        CString decryptdata_file;
//      CString encryptkey_file;
//      CString decryptkey_file;
//      CString s_cipherinit;
        CEdit   datadisp;

    void yDisplay(const char *cp);
    int rc;
    unsigned long x[4];

        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
        //{{AFX_MSG(CRijnKeyDlg)
        virtual void OnOK();
        virtual BOOL OnInitDialog();
        afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
        afx_msg void OnPaint();
        afx_msg HCURSOR OnQueryDragIcon();
        afx_msg void OnDataChange();
        afx_msg void MakeKey();
        afx_msg void TestKey();
        //}}AFX_MSG
        DECLARE_MESSAGE_MAP()

};
```

Resource.h

```
//////////////////////////////////////////////////////////////////////////////
//{{NO_DEPENDENCIES}}
// Microsoft Visual C++ generated include file.
// Used by AESKey.rc
//
#define IDM_ABOUTBOX                    0x0010
#define IDD_ABOUTBOX                    100
#define IDS_ABOUTBOX                    101
#define IDD_AESKEY_DIALOG               102
```

```
#define IDR_MAINFRAME                   128
#define IDC_DISPLAY                     1000
#define IDC_DECRYPTDATA_FILE            1001
#define ID_MAKEKEY                      1003
#define IDC_OPENKEY                     1005
#define IDC_SECRETKEY                   1005
#define IDC_RADIO1                      1006
#define IDC_RADIO2                      1007
#define IDC_RADIO3                      1008
#define IDC_RADIO4                      1009
#define IDC_RADIO5                      1010
#define IDC_RADIO6                      1011
#define IDC_RADIO7                      1012
#define IDC_RADIO8                      1013
#define IDC_RADIO9                      1014
#define IDC_RADIO10                     1015
#define ID_TESTKEY                      1016
#define IDC_PLANEDATA_FILE              1017
#define IDC_ENCRYPTDATA_FILE            1018
#define IDC_ENCRYPTKEY_FILE             1025
#define IDC_DECRYPTKEY_FILE             1026

// 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         1000
#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
```

つぎは、CPP のファイルです。

AES_ECDC.cpp

```
////////////////////////////////////////////////////
// AESECDC.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 <stdio.h>
#include <time.h>
#include <string.h>
#include <limits.h>
#include <stdlib.h>
#include <malloc.h>

#define file_len(x) (unsigned long)x
#define MAXBC 8
#define MAXKC 8
#define MAXROUNDS 14

typedef unsigned char word8;
typedef unsigned int word32;

int BC, KC, ROUNDS;
int s;

word8 Logtable[256] = {
          0,   0,  25,   1,  50,   2,  26,198,  75,199,  27,104,  51,238,223,   3,
        100,   4,224,  14,  52,141,129,239,  76,113,   8,200,248,105,  28,193,
        125,194,  29,181,249,185,  39,106,  77,228,166,114,154,201,   9,120,
```

```
          101,  47, 138,   5,  33,  15, 225,  36,  18, 240, 130,  69,  53, 147, 218, 142,
          150, 143, 219, 189,  54, 208, 206, 148,  19,  92, 210, 241,  64,  70, 131,  56,
          102, 221, 253,  48, 191,   6, 139,  98, 179,  37, 226, 152,  34, 136, 145,  16,
          126, 110,  72, 195, 163, 182,  30,  66,  58, 107,  40,  84, 250, 133,  61, 186,
           43, 121,  10,  21, 155, 159,  94, 202,  78, 212, 172, 229, 243, 115, 167,  87,
          175,  88, 168,  80, 244, 234, 214, 116,  79, 174, 233, 213, 231, 230, 173, 232,
           44, 215, 117, 122, 235,  22,  11, 245,  89, 203,  95, 176, 156, 169,  81, 160,
          127,  12, 246, 111,  23, 196,  73, 236, 216,  67,  31,  45, 164, 118, 123, 183,
          204, 187,  62,  90, 251,  96, 177, 134,  59,  82, 161, 108, 170,  85,  41, 157,
          151, 178, 135, 144,  97, 190, 220, 252, 188, 149, 207, 205,  55,  63,  91, 209,
           83,  57, 132,  60,  65, 162, 109,  71,  20,  42, 158,  93,  86, 242, 211, 171,
           68,  17, 146, 217,  35,  32,  46, 137, 180, 124, 184,  38, 119, 153, 227, 165,
          103,  74, 237, 222, 197,  49, 254,  24,  13,  99, 140, 128, 192, 247, 112,   7};

word8 Alogtable[256] = {
            1,   3,   5,  15,  17,  51,  85, 255,  26,  46, 114, 150, 161, 248,  19,  53,
           95, 225,  56,  72, 216, 115, 149, 164, 247,   2,   6,  10,  30,  34, 102, 170,
          229,  52,  92, 228,  55,  89, 235,  38, 106, 190, 217, 112, 144, 171, 230,  49,
           83, 245,   4,  12,  20,  60,  68, 204,  79, 209, 104, 184, 211, 110, 178, 205,
           76, 212, 103, 169, 224,  59,  77, 215,  98, 166, 241,   8,  24,  40, 120, 136,
          131, 158, 185, 208, 107, 189, 220, 127, 129, 152, 179, 206,  73, 219, 118, 154,
          181, 196,  87, 249,  16,  48,  80, 240,  11,  29,  39, 105, 187, 214,  97, 163,
          254,  25,  43, 125, 135, 146, 173, 236,  47, 113, 147, 174, 233,  32,  96, 160,
          251,  22,  58,  78, 210, 109, 183, 194,  93, 231,  50,  86, 250,  21,  63,  65,
          195,  94, 226,  61,  71, 201,  64, 192,  91, 237,  44, 116, 156, 191, 218, 117,
          159, 186, 213, 100, 172, 239,  42, 126, 130, 157, 188, 223, 122, 142, 137, 128,
          155, 182, 193,  88, 232,  35, 101, 175, 234,  37, 111, 177, 200,  67, 197,  84,
          252,  31,  33,  99, 165, 244,   7,   9,  27,  45, 119, 153, 176, 203,  70, 202,
           69, 207,  74, 222, 121, 139, 134, 145, 168, 227,  62,  66, 198,  81, 243,  14,
           18,  54,  90, 238,  41, 123, 141, 140, 143, 138, 133, 148, 167, 242,  13,  23,
           57,  75, 221, 124, 132, 151, 162, 253,  28,  36, 108, 180, 199,  82, 246,   1};

word8 S[256] = {
           99, 124, 119, 123, 242, 107, 111, 197,  48,   1, 103,  43, 254, 215, 171, 118,
          202, 130, 201, 125, 250,  89,  71, 240, 173, 212, 162, 175, 156, 164, 114, 192,
          183, 253, 147,  38,  54,  63, 247, 204,  52, 165, 229, 241, 113, 216,  49,  21,
            4, 199,  35, 195,  24, 150,   5, 154,   7,  18, 128, 226, 235,  39, 178, 117,
            9, 131,  44,  26,  27, 110,  90, 160,  82,  59, 214, 179,  41, 227,  47, 132,
           83, 209,   0, 237,  32, 252, 177,  91, 106, 203, 190,  57,  74,  76,  88, 207,
          208, 239, 170, 251,  67,  77,  51, 133,  69, 249,   2, 127,  80,  60, 159, 168,
           81, 163,  64, 143, 146, 157,  56, 245, 188, 182, 218,  33,  16, 255, 243, 210,
          205,  12,  19, 236,  95, 151,  68,  23, 196, 167, 126,  61, 100,  93,  25, 115,
           96, 129,  79, 220,  34,  42, 144, 136,  70, 238, 184,  20, 222,  94,  11, 219,
          224,  50,  58,  10,  73,   6,  36,  92, 194, 211, 172,  98, 145, 149, 228, 121,
          231, 200,  55, 109, 141, 213,  78, 169, 108,  86, 244, 234, 101, 122, 174,   8,
          186, 120,  37,  46,  28, 166, 180, 198, 232, 221, 116,  31,  75, 189, 139, 138,
          112,  62, 181, 102,  72,   3, 246,  14,  97,  53,  87, 185, 134, 193,  29, 158,
          225, 248, 152,  17, 105, 217, 142, 148, 155,  30, 135, 233, 206,  85,  40, 223,
          140, 161, 137,  13, 191, 230,  66, 104,  65, 153,  45,  15, 176,  84, 187,  22};

word8 Si[256] = {
           82,   9, 106, 213,  48,  54, 165,  56, 191,  64, 163, 158, 129, 243, 215, 251,
```

```
        124,227, 57,130,155, 47,255,135, 52,142, 67, 68,196,222,233,203,
         84,123,148, 50,166,194, 35, 61,238, 76,149, 11, 66,250,195, 78,
          8, 46,161,102, 40,217, 36,178,118, 91,162, 73,109,139,209, 37,
        114,248,246,100,134,104,152, 22,212,164, 92,204, 93,101,182,146,
        108,112, 72, 80,253,237,185,218, 94, 21, 70, 87,167,141,157,132,
        144,216,171,  0,140,188,211, 10,247,228, 88,  5,184,179, 69,  6,
        208, 44, 30,143,202, 63, 15,  2,193,175,189,  3,  1, 19,138,107,
         58,145, 17, 65, 79,103,220,234,151,242,207,206,240,180,230,115,
        150,172,116, 34,231,173, 53,133,226,249, 55,232, 28,117,223,110,
         71,241, 26,113, 29, 41,197,137,111,183, 98, 14,170, 24,190, 27,
        252, 86, 62, 75,198,210,121, 32,154,219,192,254,120,205, 90,244,
         31,221,168, 51,136,  7,199, 49,177, 18, 16, 89, 39,128,236, 95,
         96, 81,127,169, 25,181, 74, 13, 45,229,122,159,147,201,156,239,
        160,224, 59, 77,174, 42,245,176,200,235,187, 60,131, 83,153, 97,
         23, 43,  4,126,186,119,214, 38,225,105, 20, 99, 85, 33, 12,125};


word32 RC[30] = {
        0x00,0x01,0x02,0x04,0x08,0x10,0x20,0x40,0x80,
        0x1b,0x36,0x6c,0xd8,0xab,0x4d,0x9a,0x2f,0x5e,
        0xbc,0x63,0xc6,0x97,0x35,0x6a,0xd4,0xb3,0x7d,
        0xfa,0xef,0xc5};



static word8 shifts[5][4] = {
        0,1,2,3,
        0,1,2,3,
        0,1,2,3,
        0,1,2,4,
        0,1,3,4};

static int numrounds[5][5] = {
        10,11,12,13,14,
        11,11,12,13,14,
        12,12,12,13,14,
        13,13,13,13,14,
        14,14,14,14,14};



word8 mul(word8 a, word8 b) {
        if(a && b) return Alogtable[(Logtable[a] + Logtable[b])%255];
        else return 0;
}

void AddRoundKey(word8 a[4][MAXBC], word8 rk[4][MAXBC]) {
        int i, j;

        for(i = 0; i<4; i++)
                for(j=0;j<BC;j++) a[i][j] ^= rk[i][j];
}
```

```
void SubBytes(word8 a[4][MAXBC], word8 box[256]){
        int i,j;

        for(i=0;i<4;i++)
                for(j=0;j<BC;j++) a[i][j] = box[a[i][j]] ;
}

void ShiftRows(word8 a[4][MAXBC], word8 d){
        word8 tmp[MAXBC];
        int i,j;

        if(d==0){
                for(i=1;i<4;i++){
                        for(j=0;j<BC;j++)
                                tmp[j] = a[i][(j+shifts[BC-4][i]) % BC];
                        for(j=0;j<BC;j++) a[i][j] = tmp[j];
                }
        }
        else{
                for(i=1;i<4;i++){
                        for(j=0;j<BC;j++)
                                tmp[j] = a[i][(BC+j-shifts[BC-4][i]) % BC];
                        for(j=0;j<BC;j++) a[i][j] = tmp[j];
                }
        }
}

void MixColumns(word8 a[4][MAXBC]){
        word8 b[4][MAXBC];
        int i,j;

        for(j=0;j<BC;j++)
                for(i=0;i<4;i++)
                        b[i][j] = mul(2,a[i][j])
                                ^ mul(3,a[(i+1) % 4][j])
                                ^ a[(i+2) % 4][j]
                                ^ a[(i+3) % 4][j];
        for(i=0;i<4;i++)
                for(j=0;j<BC;j++) a[i][j] = b[i][j];
}

void InvMixColumns(word8 a[4][MAXBC]){
        word8 b[4][MAXBC];
        int i,j;

        for(j=0;j<BC;j++)
        for(i=0;i<4;i++)
                b[i][j] = mul(0xe, a[i][j])
                        ^ mul(0xb, a[(i+1) % 4][j])
                        ^ mul(0xd, a[(i+2) % 4][j])
                        ^ mul(0x9, a[(i+3) % 4][j]);
```

```
        for(i=0;i<4;i++)
                for(j=0;j<BC;j++) a[i][j] = b[i][j];
}

int KeyExpansion(word8 k[4][MAXKC],
                                word8 W[MAXROUNDS+1][4][MAXBC]){
        int i,j,t,RCpointer = 1;
        word8 tk[4][MAXKC];

        for(j=0;j<KC;j++)
                for(i=0;i<4;i++)
                        tk[i][j] = k[i][j];
        t = 0;

        for(j=0;(j<KC) && (t<(ROUNDS+1)*BC); j++, t++)
                for(i=0;i<4;i++) W[t / BC][i][t % BC] = tk[i][j];

        while(t<(ROUNDS+1)*BC){
                for(i=0;i<4;i++)
                        tk[i][0] ^= S[tk[(i+1)%4][KC-1]];
                tk[0][0] ^= RC[RCpointer++];

                if(KC<=6)
                        for(j=1;j<KC;j++)
                                for(i=0;i<4;i++) tk[i][j] ^= tk[i][j-1];
                else{
                        for(j=1;j<4;j++)
                                for(i=0;i<4;i++) tk[i][j] ^= tk[i][j-1];
                        for(i=0;i<4;i++) tk[i][4] ^= S[tk[i][3]];
                        for(j=5;j<KC;j++)
                                for(i=0;i<4;i++) tk[i][j] ^= tk[i][j-1];
                }
                for(j=0; (j<KC) && (t<(ROUNDS+1)*BC); j++, t++)
                        for(i=0;i<4;i++) W[t/BC][i][t%BC] = tk[i][j];
        }
        return 0;
}

int Encrypt(word8 a[4][MAXBC], word8 rk[MAXROUNDS+1][4][MAXBC]){
        int r;

        AddRoundKey(a,rk[0]);

        for(r=1;r<ROUNDS;r++){
                SubBytes(a,S);
                ShiftRows(a,0);
                MixColumns(a);
                AddRoundKey(a,rk[r]);
        }

        SubBytes(a,S);
        ShiftRows(a,0);
```

```
        AddRoundKey(a,rk[ROUNDS]);

        return 0;
}

int Decrypt(word8 a[4][MAXBC], word8 rk[MAXROUNDS+1][4][MAXBC]){
        int r;

        AddRoundKey(a,rk[ROUNDS]);
        SubBytes(a,Si);
        ShiftRows(a,1);

        for(r=ROUNDS-1;r>0;r--){
                AddRoundKey(a,rk[r]);
                InvMixColumns(a);
                SubBytes(a,Si);
                ShiftRows(a,1);
        }
        AddRoundKey(a,rk[0]);

        return 0;
}

int AES_Encrypt(int Keylen, int Blocklen, char* Key, char* inputfile, char* outputfile)
{
        int i,j,klen,blen;
        word8 a[4][MAXBC], rk[MAXROUNDS+1][4][MAXBC], sk[4][MAXKC];
        char c_klen[8], c_blen[8], pass[128];
        FILE *fkey =0, *fin = 0, *fout = 0;
    fpos_t flen;
        char *dbuf;
    unsigned long len, rlen, blen4;


        klen = Keylen/32;
        blen = Blocklen/32;
        blen4 = (unsigned int)blen*4;

        KC = klen;
        if(KC<4 || 8<KC){
//              printf("Wrong key size. ¥n");
                return (-1);
        }

        BC = blen;
        if(BC<4 || 8<BC){
//              printf("Wrong block size. ¥n");
                return (-1);
        }

        dbuf = (char *)malloc(2*MAXBC*4);
        if(dbuf == NULL){
```

```
//              printf("No memory. ¥n");
                return (-1);
        }


        if((fin = fopen(inputfile, "rb")) == NULL){
//              printf("Can not open plane text file. ¥n");
                return (-1);
        }

    fseek(fin, 0, SEEK_END);
    fgetpos(fin, &flen);
    rlen = file_len(flen);
    // reset to start
    fseek(fin, 0, SEEK_SET);

        if((fout = fopen(outputfile, "wb")) == NULL){
//              printf("Can not open encrypted data file. ¥n");
                return (-1);
        }

        ROUNDS = numrounds[KC-4][BC-4];

        char cl,cr;
        int k = 0;
        for(j=0;j<KC;j++){
                for(i=0;i<4;i++){
                        if(pass[k]>=0x30 && pass[k]<=0x39){cl = pass[k]-0x30;}
                        else if(pass[k]>=0x41 && pass[k]<=0x46){cl = pass[k]-0x37;}
                        else if(pass[k]>=0x61 && pass[k]<=0x66){cl = pass[k]-0x57;}
                        else cl = 0;
                        if(pass[k+1]>=0x30 && pass[k+1]<=0x39){cr = pass[k+1]-0x30;}
                        else if(pass[k+1]>=0x41 && pass[k+1]<=0x46){cr = pass[k+1]-0x37;}
                        else if(pass[k+1]>=0x61 && pass[k+1]<=0x66){cr = pass[k+1]-0x57;}
                        else cr = 0;
                        sk[i][j] = ((cl<<4)|(cr));
                        k += 2;
                }
        }

        KeyExpansion(sk,rk);

////////////////////////////////////////////////////////////
        s = sizeof(unsigned int);
        // write the bytes of the file
        *((unsigned int*)dbuf) = rlen;
        len = (unsigned long) fread(dbuf+s, 1, blen4-s , fin);
    rlen -= len;
    // pad the file bytes with zeroes
    for(i = len+s; (unsigned int)i < blen4 ; ++i)
        dbuf[i] = 0;
    // encrypt the top 16 bytes of the buffer
```

```
    k=0;
    for(j=0;j<BC;j++){
            for(i=0;i<4;i++){
                    a[i][j] = dbuf[k];
                    k++;
            }
    }
    Encrypt(a,rk);
    k=0;
    for(j=0;j<BC;j++){
            for(i=0;i<4;i++){
                    dbuf[k] = a[i][j];
                    k++;
            }
    }
if(fwrite(dbuf, 1, blen4, fout) != (unsigned int)blen4)
    return -2;


if((rlen <= (unsigned int)blen4) && (rlen > 0))
{   // if the file length is less than or equal to 16 bytes
    // read the bytes of the file into the buffer and verify length
    len = (unsigned long) fread(dbuf, 1, blen4 , fin);
    rlen -= len;

    if(rlen > 0)
        return -1;

    // pad the file bytes with zeroes
    for(i = len; (unsigned int)i < blen4 ; ++i)
        dbuf[i] = 0;

    // encrypt the top 16 bytes of the buffer
            k=0;
            for(j=0;j<BC;j++){
                    for(i=0;i<4;i++){
                            a[i][j] = dbuf[k];
                            k++;
                    }
            }

            Encrypt(a,rk);

    // write the IV and the encrypted file bytes
            k=0;
            for(j=0;j<BC;j++){
                    for(i=0;i<4;i++){
                            dbuf[k] = a[i][j];
                            k++;
                    }
            }
    if(fwrite(dbuf, 1, blen4, fout) != (unsigned int)blen4)
```

```
        return -2;
}
else
{   // if the file length is more 16 bytes
    // read the file a block at a time
    while(rlen > 0 && !feof(fin))
    {
        // read a block and reduce the remaining byte count
        len = (unsigned long)fread(dbuf, 1, blen4 , fin);
        rlen -= len;

        // verify length of block
        if(len != (unsigned int)blen4 ){
        // pad the file bytes with zeroes
                        for(i = len; (unsigned int)i < blen4 ; ++i)
                                dbuf[i] = 0;
                }

        // encrypt the block
                k=0;
                for(j=0;j<BC;j++){
                        for(i=0;i<4;i++){
                                a[i][j] = dbuf[k];
                                k++;
                        }
                }
                Encrypt(a,rk);
        // write the encrypted block
                k=0;
                for(j=0;j<BC;j++){
                        for(i=0;i<4;i++){
                                dbuf[k] = a[i][j];
                                k++;
                        }
                }
        if(fwrite(dbuf, 1, blen4, fout) != blen4)
           return -2;

        // if there is only one more block do ciphertext stealing
        if(rlen > 0 && rlen < (unsigned int)blen4 )
        {
                len = (unsigned long)fread(dbuf, 1, blen4 , fin);

               // clear the remainder of the bottom half of buffer
        for(i = 0; (unsigned int)i < (unsigned int)blen4  - len; ++i)
            dbuf[len +i] = 0;

            // encrypt the final block
                k=0;
                for(j=0;j<BC;j++){
                        for(i=0;i<4;i++){
                                a[i][j] = dbuf[k];
```

```
                                        k++;
                                }
                        }
                        Encrypt(a,rk);

                        k=0;
                        for(j=0;j<BC;j++){
                                for(i=0;i<4;i++){
                                        dbuf[k] = a[i][j];
                                        k++;
                                }
                        }
                if(fwrite(dbuf, 1, blen4, fout) != blen4)
                   return -2;

                        break;
                    // set the length of the final write
                }

        }
    }


        if(fout)
        {
        fclose(fout);
        }

        if(fin)
        {
                fclose(fin);
        }

        free(dbuf);

        return 0;
}


int AES_Decrypt(int Keylen, int Blocklen, char* Key, char* inputfile, char* outputfile)
{
        int i,j,klen,blen;
        word8 a[4][MAXBC], rk[MAXROUNDS+1][4][MAXBC], sk[4][MAXKC];
        char c_klen[8], c_blen[8], pass[128];
        FILE *fkey =0, *fin = 0, *fout = 0;
    fpos_t flen;
        char *dbuf;
    unsigned long len, rlen, blen4;

////////////////////////////////////////////////////////////

        klen = Keylen/32;
```

```
        blen = Blocklen/32;
        blen4 = (unsigned int)blen*4;


        dbuf = (char *)malloc(2*blen4);


        KC = klen;
        if(KC<4 || 8<KC){
//              printf("Wrong key size. ¥n");
                return (-1);
        }

        BC = blen;
        if(BC<4 || 8<BC){
//              printf("Wrong block size. ¥n");
                return (-1);
        }


        if((fin = fopen(inputfile, "rb")) == NULL){
//              printf("Can not open encrypted file. ¥n");
                return (-1);
        }
    fseek(fin, 0, SEEK_END);
    fgetpos(fin, &flen);
    rlen = file_len(flen);
    // reset to start
    fseek(fin, 0, SEEK_SET);

        if((fout = fopen(outputfile, "wb")) == NULL){
//              printf("Can not open decrypted file. ¥n");
                return (-1);
        }

        ROUNDS = numrounds[KC-4][BC-4];

        char cl,cr;
        int k = 0;
        for(j=0;j<KC;j++){
                for(i=0;i<4;i++){
                        if(pass[k]>=0x30 && pass[k]<=0x39){cl = pass[k]-0x30;}
                        else if(pass[k]>=0x41 && pass[k]<=0x46){cl = pass[k]-0x37;}
                        else if(pass[k]>=0x61 && pass[k]<=0x66){cl = pass[k]-0x57;}
                        else cl = 0;
                        if(pass[k+1]>=0x30 && pass[k+1]<=0x39){cr = pass[k+1]-0x30;}
                        else if(pass[k+1]>=0x41 && pass[k+1]<=0x46){cr = pass[k+1]-0x37;}
                        else if(pass[k+1]>=0x61 && pass[k+1]<=0x66){cr = pass[k+1]-0x57;}
                        else cr = 0;
                        sk[i][j] = ((cl<<4)|(cr));
                        k += 2;
                }
```

```
        }

        KeyExpansion(sk,rk);
        s = sizeof(unsigned int);
////////////////////////////////

        len = (unsigned long) fread(dbuf, 1, blen4 , fin);
    rlen -= len;

    // decrypt the top 16 bytes of the buffer
        k=0;
        for(j=0;j<BC;j++){
                for(i=0;i<4;i++){
                        a[i][j] = dbuf[k];
                        k++;
                }
        }

        Decrypt(a,rk);
    len = blen4 ;

        // write the IV and the encrypted file bytes
        k=0;
        for(j=0;j<BC;j++){
                for(i=0;i<4;i++){
                        dbuf[k] = a[i][j];
                        k++;
                }
        }

        unsigned int wlen = *((unsigned int*)dbuf);

        if(wlen <= len-s){
        if(fwrite(dbuf+s, 1, wlen, fout) != wlen)
                return -2;
        }
        else{
                wlen -= len-s;
        if(fwrite(dbuf+s, 1, len-s, fout) != len-s)
                return -2;
        }

    if((rlen <= blen4) && (rlen>0) )
    {   // if the original file length is less than or equal to 16 bytes
        // read the bytes of the file and verify length
        len = (unsigned long)fread(dbuf, 1, blen4 , fin);
        rlen -= len;

                if(rlen > 0)
           return -1;

         // decrypt from position len to position len + BLOCK_LEN
```

```
                k=0;
                for(j=0;j<BC;j++){
                        for(i=0;i<4;i++){
                                a[i][j] = dbuf[k];
                                k++;
                        }
                }

                Decrypt(a,rk);

                k=0;
                for(j=0;j<BC;j++){
                        for(i=0;i<4;i++){
                                dbuf[k] = a[i][j];
                                k++;
                        }
                }

        // output decrypted bytes
        if(fwrite(dbuf, 1, wlen, fout) != (unsigned long)wlen)
            return -2;
    }
    else
    {
        // read the encrypted file a block at a time
        while(rlen > 0 && !feof(fin))
        {
            // input a block and reduce the remaining byte count
            len = (unsigned long)fread(dbuf, 1, blen4, fin);
            rlen -= len;

            // verify the length of the read operation
            if(len != blen4 )
                return -1;

            // decrypt input buffer
                        k=0;
                        for(j=0;j<BC;j++){
                                for(i=0;i<4;i++){
                                        a[i][j] = dbuf[k];
                                        k++;
                                }
                        }

                        Decrypt(a,rk);

                        k=0;
                        for(j=0;j<BC;j++){
                                for(i=0;i<4;i++){
                                        dbuf[k] = a[i][j];
                                        k++;
                                }
```

```
                        }

                        if(wlen < len){
                                if(fwrite(dbuf, 1, wlen, fout) != wlen){
                                    return -2;
                                }
                                break;
                        }
                        else{
                                if(fwrite(dbuf, 1, blen4, fout) != blen4)
                                        return -2;
                        }
                        if(wlen > len){wlen -= len;}
                }
    }

        if(fout)
        {
        fclose(fout);
        }

        if(fin)
        {
                fclose(fin);
        }

        free(dbuf);

        return 0;
}
```

AESKey.cpp

```
/////////////////////////////////////////////////////////////
// AESKey.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 "AESKey.h"
#include "AESKeyDlg.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#endif


// CAESKeyApp
```

```
BEGIN_MESSAGE_MAP(CAESKeyApp, CWinApp)
	ON_COMMAND(ID_HELP, &CWinApp::OnHelp)
END_MESSAGE_MAP()


// CAESKeyApp コンストラクション

CAESKeyApp::CAESKeyApp()
{
	// 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CAESKeyApp オブジェクトです。

CAESKeyApp theApp;


// CAESKeyApp 初期化

BOOL CAESKeyApp::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"));

	CAESKeyDlg 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;
}
```

AESKeyDlg.cpp

```
// AESKeyDlg.cpp : 実装ファイル
//

#include "stdafx.h"
#include "AESKey.h"
#include "AESKeyDlg.h"
#include "fstream"
#include <iostream>

using namespace std;


FILE *stream0;//plane
FILE *stream1;//encrypt
FILE *stream2;//decrypt
FILE *stream3;//eckey
FILE *stream4;//dckey

#ifdef _DEBUG
#define new DEBUG_NEW
#endif



unsigned char k[128];
unsigned char e[256];
/////////////////////////////////////////////////////////////////////////////
// アプリケーションのバージョン情報で使われているCAboutDlg ダイアログ
// 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()


// CAESKeyDlg ダイアログ


/////////////////////////////////////////////////////////////////////////////
// CAESKeyDlg ダイアログ

CAESKeyDlg::CAESKeyDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CAESKeyDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CAESKeyDlg)
	i_KeyLen = 0;
```

```
	i_BlockLen = 0;
	s_fname1 = "AesKeyEC1.bin";
	s_fname2 = "AesKeyDC1.bin";

	madekey = 0;
	planedata_file = "plane.txt";
	encryptdata_file = "encrypt.bin";
	decryptdata_file = "decrypt.txt";
	srand( (unsigned)time(NULL) );
		// メモ: この位置にClassWizard によってメンバの初期化が追加されます。
	//}}AFX_DATA_INIT
	// メモ: LoadIcon はWin32 のDestroyIcon のサブシーケンスを要求しません。
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}


void CAESKeyDlg::OnOK()
{
	CDialog::OnOK();

	ofstream ofs(s_fname1, ios::out);
	if(i_KeyLen == 0) ofs << 128 << endl;
	if(i_KeyLen == 1) ofs << 160 << endl;
	if(i_KeyLen == 2) ofs << 192 << endl;
	if(i_KeyLen == 3) ofs << 224 << endl;
	if(i_KeyLen == 4) ofs << 256 << endl;
	if(i_BlockLen == 0) ofs << 128 << endl;
	if(i_BlockLen == 1) ofs << 160 << endl;
	if(i_BlockLen == 2) ofs << 192 << endl;
	if(i_BlockLen == 3) ofs << 224 << endl;
	if(i_BlockLen == 4) ofs << 256 << endl;
	ofs << s_key << endl;
	ofs.close();

	ofstream ofs2(s_fname2, ios::out);
	if(i_KeyLen == 0) ofs2 << 128 << endl;
	if(i_KeyLen == 1) ofs2 << 160 << endl;
	if(i_KeyLen == 2) ofs2 << 192 << endl;
	if(i_KeyLen == 3) ofs2 << 224 << endl;
	if(i_KeyLen == 4) ofs2 << 256 << endl;
	if(i_BlockLen == 0) ofs2 << 128 << endl;
	if(i_BlockLen == 1) ofs2 << 160 << endl;
	if(i_BlockLen == 2) ofs2 << 192 << endl;
	if(i_BlockLen == 3) ofs2 << 224 << endl;
	if(i_BlockLen == 4) ofs2 << 256 << endl;
	ofs2 << s_key << endl;
	ofs2.close();
}

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

```
	//{{AFX_DATA_MAP(CAESKeyDlg)
	DDX_Text(pDX, IDC_ENCRYPTKEY_FILE, s_fname1);
	DDX_Text(pDX, IDC_DECRYPTKEY_FILE, s_fname2);
	DDX_Text(pDX, IDC_OPENKEY, s_key);
	DDX_Radio(pDX, IDC_RADIO1, i_KeyLen);
	DDX_Radio(pDX, IDC_RADIO6, i_BlockLen);
	DDX_Text(pDX, IDC_PLANEDATA_FILE, planedata_file);
	DDX_Text(pDX, IDC_ENCRYPTDATA_FILE, encryptdata_file);
	DDX_Text(pDX, IDC_DECRYPTDATA_FILE, decryptdata_file);
	DDX_Control(pDX, IDC_DISPLAY, datadisp);
		// メモ: この場所にはClassWizard によってDDX とDDV の呼び出しが追加されます。
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CAESKeyDlg, CDialog)
	//{{AFX_MSG_MAP(CAESKeyDlg)
	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()

void CAESKeyDlg::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 CAESKeyDlg::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_RADIO5) ){ n= 16;}
	if(IDC_RADIO2 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) ){ n= 20;}
	if(IDC_RADIO3 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) ){ n= 24;}
	if(IDC_RADIO4 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) ){ n= 28;}
	if(IDC_RADIO5 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) ){ n= 32;}

//	srand( (unsigned)time(NULL) );  //in constracter
	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_OPENKEY);
        pwnd->SetWindowText(b);
        madekey = 1;
}

void CAESKeyDlg::OnDataChange()
{
}


void CAESKeyDlg::TestKey()
{
//      unsigned char key2[128+2];
        char c_ci[65];
        int  cnt, c, block, i, j, Keylen2, Blocklen2;
        char Key[128+2];
        char inputfile[256];
        char outputfile[256];
        long filelen, mesLength; // 平文長（バイト）
        char* bufp;
        unsigned char* bufc;
        char* bufdc;
        int numclosed;
        int n;
        CWnd* pwnd;
        CFileStatus fs, fsec, fsdc;

        datadisp.SetLimitText(INT_MAX);

        if(IDC_RADIO1 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 128; keylength = "128";
        }
        if(IDC_RADIO2 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 160; keylength = "160";
        }
        if(IDC_RADIO3 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
```

```
                Keylen2 = 192; keylength = "192";
        }
        if(IDC_RADIO4 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 224; keylength = "224";
        }
        if(IDC_RADIO5 == GetCheckedRadioButton(IDC_RADIO1, IDC_RADIO5) )
        {
                Keylen2 = 256; keylength = "256";
        }


        if(IDC_RADIO6 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 128;
        }
        if(IDC_RADIO7 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 160;
        }
        if(IDC_RADIO8 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 192;
        }
        if(IDC_RADIO9 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 224;
        }
        if(IDC_RADIO10 == GetCheckedRadioButton(IDC_RADIO6, IDC_RADIO10) )
        {
                Blocklen2 = 256;
        }

////////////////////////////////////////////////////////////////
        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();
        }
        pwnd = GetDlgItem(IDC_SECRETKEY);
        pwnd->GetWindowText(s_key);

        //int AES_Encrypt(int Keylen, int Blocklen, char* Key, char* inputfile, char* outputfile);
        strcpy(Key, s_key);
        strcpy(inputfile, planedata_file);
        strcpy(outputfile, encryptdata_file);
        rc=AES_Encrypt(Keylen2, Blocklen2, Key, inputfile, outputfile);

        strcpy(inputfile, encryptdata_file);
```

```
        strcpy(outputfile, decryptdata_file);
    rc=AES_Decrypt(Keylen2, Blocklen2, Key, inputfile, outputfile);



    /* 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");}

    /* 暗号文を書き込むファイルを開く*/
        pwnd = GetDlgItem(IDC_ENCRYPTDATA_FILE);
        pwnd->GetWindowText(encryptdata_file);
        if( (stream1 = fopen( encryptdata_file, "rb" )) == NULL ){
                yDisplay("ファイル"); yDisplay(encryptdata_file); yDisplay(" は開けませんでした。
¥r¥n");
                return;//(-1);
                }
    else
                {yDisplay("ファイル"); yDisplay(encryptdata_file); yDisplay(" は開けました。¥r¥n");}

    /* 復号文を書き込むファイルを開く*/
        pwnd = GetDlgItem(IDC_DECRYPTDATA_FILE);
        pwnd->GetWindowText(decryptdata_file);
        if( (stream2 = fopen( decryptdata_file, "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");}

        // Key disp
        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");
```

```
        // 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;

  // 平文
        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 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");


// 暗文
        CFile::GetStatus(encryptdata_file, fsec);
        filelen = fsec.m_size;

        bufc = (unsigned char*)new(unsigned char[filelen + 2]);
        if(bufc == 0){
                yDisplay("メモリ不足¥r¥n");
                return;//(-1);
                }
        fseek(stream1,0,0);
        int cc;
```

```
        i=0;
        do{
                cc = fgetc(stream1);
                bufc[i]=cc;
                i=i+1;
        }while(cc!=EOF);
        bufc[i-1]=NULL;

        mesLength = filelen;
        block = mesLength/16 + ((mesLength%16)?1:0);
// 暗文
        // 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);
        }
        yDisplay("¥r¥n");
        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" );

//return;

        // 復文
        CFile::GetStatus(planedata_file, fsdc);
        filelen = fsdc.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++){
                bufp[j] = 0;
        }
        *(long*)(bufdc) = filelen;

  // 復文
        fseek(stream2,0,0);
        i = sizeof(long);
```

```
        do{
                c = fgetc(stream2);
                bufdc[i]=c;
                i=i+1;
        }while(c!=EOF);
        bufdc[i-1]=NULL;


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

    yDisplay("復文     = ¥r¥n");
        yDisplay(bufdc+sizeof(long));
        yDisplay("¥r¥n");
        yDisplay("¥r¥n");

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


   /* 他のすべてのファイルを閉じる*/
        numclosed = _fcloseall( );

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

        return;// 0;
}




// CAESKeyDlg メッセージハンドラ

BOOL CAESKeyDlg::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 CAESKeyDlg::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 CAESKeyDlg::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 CAESKeyDlg::OnQueryDragIcon()
{
        return static_cast<HCURSOR>(m_hIcon);
}
```

Stdafx.cpp

```
//////////////////////////////////////////////////////////
// stdafx.cpp : 標準インクルードAESKey.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"
```

以上です。