nwzCryptoLib 매뉴얼

작성자. 서버인프라개발팀 이기로
검수자. 서버인프라개발팀 이기로

Version	Date	Modifier	Details
1.0	2010/11/11	이기로	Create Document
1.01	2011/04/22	이기로	함수 리턴값 변경

1. 소개

- 개요

  nwzCryptoLib는 대칭키 방식의 암복호화 기능을 간단하게 사용할 수 있도록 만든 라이브러리이며,
  암호화 라이브러리로는 openssl을 사용하며 외부 라이브러리 사용을 최소화 하여 대부분의 빌드환경에서 사용이 가능하다.
  DB계정/패스워드 통합인증 서비스를 위해서 인증서버에서 암호화된 계정/패스워드를 가져오는 기능이 포함되어 있다.

- 라이브러리 구성

  1) Win32

구성 요소	설명	비고
nwzCrypto.h	인터페이스 header file	
nwzCryptoLib.lib	Win32용 lib 파일	
nwzCryptoLib.dll	Win32용 dll	다른 외부 dll은 필요없음

  2) Unix / Linux

구성 요소	설명	비고
nwzCrypto.h	인터페이스 header file	
nwzCryptoLib.so	Shared object library	curl 과 openssl 라이브러리 필요

- 주의사항

  1) 대칭키는 라이브러리 내부에 포함되어 있으며 변경할 수 없다.
  2) 4096 바이트 이하의 문자열만 암복호화가 가능하다.
  3)

2. 함수 reference

2.1. Win32

2.1.1 nwzEncryptUC() 함수

1) syntax

```
int nwzEncryptUC (
                    LPCWSTR lpPlainBuf,
                    DWORD dwPlainBufSize,
                    LPWSTR lpEncryptedBuf,
                    DWORD dwEncryptedBufSize,
                    int nAlg=0
                    );
```

2) parameters

파라미터	설명
[in] lpPlainBuf	평문 문자열 포인터
[in]dwPlainBufSize	평문 문자열의 길이
[out] lpEncryptedBuf	암호화된 결과값을 받을 buffer 포인터
[out] dwEncryptedBufSize	암호화된 결과값을 받을 buffer 의 최대 크기
[in] nAlg	대칭키 알고리즘 (현재는 BLOWFISH= 0 만 지원)

3) return값

성공시	암호화된 문자열의 길이
실패시	실패인 경우 0
lpEncryptedBuf = NULL , dwEncryptedBufSize = 0 을 셋팅시	암호화된 문자열에 필요한 buffer 크기

4) remark

입력받은 문자열을 암호화한다.

Unicode(utf16) 값을 암호화하여 결과값을 Unicode(uft16)로 돌려준다.

2.1.2 nwzDecryptUC() 함수

1) syntax

```
int nwzDecryptUC (
                    LPCWSTR lpEncryptedBuf,
                    DWORD dwEncryptedBufSize,
                    LPWSTR lpPlainBuf,
                    DWORD dwPlainBufSize,
                    int nAlg=0
                    );
```

2) parameters

파라미터	설명
[in] lpEncryptedBuf	암호화된 문자열 포인터
[in] dwEncryptedBufSize	암호화된 문자열의 길이
[out] lpPlainBuf	복호화된 결과값을 받을 buffer 포인터
[out] dwPlainBufSize	복호화된 결과값을 받을 buffer 의 최대 크기
[in] nAlg	대칭키 알고리즘 (현재는 BLOWFISH= 0 만 지원)

3) return값

성공시	복호화된 문자열의 길이
실패시	실패인 경우 0
lpPlainBuf = NULL , dwPlainBufSize = 0 을 셋팅시	복호화된 문자열에 필요한 buffer 크기

4) remark

입력받은 암호화된 문자열을 복호화한다.

Unicode(utf16) 값을 복호화하여 결과값을 Unicode(uft16)로 돌려준다.

암호화된 문자열이 아니더라도 전달된 문자열이 암호화 알고리즘의 cipher block 크기의 배수라면 잘못된 값으로 복호화하여 리턴할 수도 있다. (garbage 입력 -> garbage 출력)

2.1.3 QueryToAuthServer() 함수

1) syntax

```
int QueryToAuthServerUC(
                        LPCWSTR lpHostName,
                        LPCWSTR lpSendBuf,
                        DWORD dwSendBufSize,
                        LPWSTR lpRecvBuf,
                        DWORD lpRecvBufSize
                        );
```

2) parameters

파라미터	설명
[in] lpHostName	인증서버의 hostname
[in] lpSendBuf	인증키값의 buffer 포인터
[in] dwSendBufSize	인증키값의 buffer 의 크기
[out] lpRecvBuf	암호화된 문자열을 받을 buffer 포인터
[out] lpRecvBufSize	암호화된 문자열을 받을 buffer 포인터의 최대크기 (Max값은 1024)

3) return값

성공시	암호화된 문자열의 크기를 리턴
실패시	-101 : parameter에 NULL 값을 입력한 경우 -102 : Buffer size가 작은 경우 -103 : 인증서버 통신 실패 -104 : 인증키값에 해당하는 value가 없음

4) remark

전달된 hostname 으로 http://hostname/getPwd.nwz?ikey=인증키 를 호출하여 암호화된 문자열을 받아온다.
받아올 암호화된 문자열의 크기를 알 수 없는 경우는 MAX값인 1024를 설정한다.
운영환경의 통합인증 서버는 dbdove.pmang.com 과 dbswan.pmang.com 을 사용
개발환경의 통합인증 서버는 dbcpubdev.neowiz.com 을 사용
현재의 인증서버는 DB의 접속 계정과 패스워드를 관리하고 있으며 접속하고자 하는 DB의 계정에 대한 인증키값과 패스워드에 인증키값을 DB 기술팀에 요청하여 받아야 한다.

2.2. Unix / Linux

2.2.1 nwzEncrypt() 함수

1) syntax

```
int nwzEncrypt(
                const char * input,
                const int inputSize,
                char * outputBuf,
                int outputBufSize,
                int alg
              );
```

2) parameters

파라미터	설명
[in] input	평문 문자열 포인터
[in] inputSize	평문 문자열의 길이
[out] outputBuf	암호화된 결과값을 받을 buffer 포인터
[out] outputBufSize	암호화된 결과값을 받을 buffer 의 최대 크기
[in] alg	대칭키 알고리즘 (현재는 BLOWFISH= 0 만 지원)

3) return값

성공시	암호화된 문자열의 길이
실패시	실패인 경우 0
outputBuf = NULL , outputBufSize = 0 을 셋팅시	암호화된 문자열에 필요한 buffer 크기

4) remark

입력받은 문자열을 암호화한다.

Ansi code 값을 암호화하여 결과값을 Ansi code 값으로 돌려준다.

2.2.2 nwzDecrypt() 함수

1) syntax

```
int nwzDecrypt(
                const char * input,
                const int inputSize,
                char * outputBuf,
                int outputBufSize,
                int alg
              );
```

2) parameters

파라미터	설명
[in] input	암호화된 문자열 포인터
[in] inputSize	암호화된 문자열의 길이
[out] outputBuf	복호화된 결과값을 받을 buffer 포인터
[out] outputBufSize	복호화된 결과값을 받을 buffer 의 최대 크기
[in] alg	대칭키 알고리즘 (현재는 BLOWFISH= 0 만 지원)

3) return값

성공시	복호화된 문자열의 길이
실패시	실패인 경우 0
outputBuf = NULL , outputBufSize = 0 을 셋팅시	복호화된 문자열에 필요한 buffer 크기

4) remark

입력받은 암호화된 문자열을 복호화한다.

Ansi code 값을 복호화하여 결과값을 Ansi code 값으로 돌려준다.

2.2.3 nwzDecryptUTF16 () 함수

1) syntax

```
int nwzDecryptUTF16 (
                const char * input,
                const int inputSize,
                char * outputBuf,
                int outputBufSize,
                int alg
              );
```

2) parameters

파라미터	설명
[in] input	암호화된 문자열 포인터 또는 인증키값
[in] inputSize	암호화된 문자열의 길이
[out] outputBuf	복호화된 결과값을 받을 buffer 포인터
[out] outputBufSize	복호화된 결과값을 받을 buffer 의 최대 크기
[in] alg	대칭키 알고리즘 (현재는 BLOWFISH= 0 만 지원)

3) return값

성공시	복호화된 문자열의 길이
실패시	실패인 경우 0
outputBuf = NULL , outputBufSize = 0 을 셋팅시	복호화된 문자열에 필요한 buffer 크기

4) remark

window에서 환경에서 Unicode(utf16)으로 암호화된 문자열을 복호화하여 ansi code로 리턴한다. Win32 와는 다르게 QueryToAuthServerUC() 기능을 함수내부로 넣어 인증키값을 input으로 할 경우 AuthServer 로 암호화된 값을 가져와서 복호화 하여 준다.
AuthServer는 Default 로 dbdove.pmang.com 으로 되어 있으며 AuthServer를 변경하려면 **PWD_AUTH_URL** 이라는 환경변수에 "http://변경하는hostname/getPwd.nwz?ikey=" 을 셋팅해 주면 된다.

3. 사용 예제

3.1. 단순 암호화 / 복호화 (win32)

```
// basic encrypt test
LPCWSTR inputText = L"1234";
LPWSTR outputText = NULL;
int ret = 0;
int sizeOfCh = ::wcslen(inputText);

int bufferSize = nwzEncryptUC( inputText, sizeOfCh, NULL, 0);
if ( bufferSize > 0 )
{
	outputText = new wchar_t[bufferSize+1];
	ZeroMemory(outputText,(bufferSize+1)*sizeof(wchar_t));
	ret = nwzEncryptUC(inputText,sizeOfCh,outputText,bufferSize);
}

if( ret > 0 )
	_tprintf(_T("basic encrypt : %s₩n"),outputText);

// basic decrypt test
LPCWSTR inputText2 = outputText;
outputText = NULL;
ret = 0;
sizeOfCh = ::wcslen (inputText2);

bufferSize = nwzDecryptUC(inputText2,sizeOfCh,NULL,0);
if ( bufferSize > 0 )
{
	outputText = new wchar_t[bufferSize+1];
	ZeroMemory(outputText,(bufferSize+1)*sizeof(wchar_t));
	ret = decryptPwdUC(inputText2,sizeOfCh,outputText,bufferSize);
}

if( ret > 0 )
	_tprintf(_T("basic decrypt : %s₩n"),outputText);

if( NULL != outputText ) delete [] outputText;
```

3.2. DB 통합 인증 시스템 연동 (win32)

QueryToAuthServerUC() 함수를 호출하여 인증서버에서 암호화 된 계정값과 패스워드 값을 받아온후에 nwzDecryptUC() 함수를 호출하여 복호화된 계정과 패스워드로 DB에 접속한다.

```
wchar_t *ikey = L"CODE1";   //test 인증키로 복호화하면 1234
wchar_t remotepwd[1024] = {0,};

int errcode = QueryToAuthServerUC(L"dbdove.pmang.com",   ikey, wcslen(ikey), remotepwd, 1024);
if( errcode < 0 )
{
   _tprintf(_T("fail to QueryToAuthServerUC()   errcode: %d₩n"), errcode );
   return ;
}
else
{
   _tprintf(_T("remote pwd:%s₩n"), remotepwd );
}

wchar_t * outputText = NULL;
int bufferSize = nwzDecryptUC( remotepwd, wcslen (remotepwd), NULL, 0 );
if ( bufferSize > 0 )
{
   outputText = new wchar_t[bufferSize+1];
   ZeroMemory(outputText, ( bufferSize+1 ) * sizeof ( wchar_t ) );
   ret = nwzDecryptUC(remotepwd, wcslen (remotepwd), outputText, bufferSize);
}

if( ret > 0 )
 _tprintf(_T("remote decrypt : %s₩n"),outputText);
if( NULL != outputText ) delete [] outputText;
```

3.3. MBCS 환경에서 복호화 (win32)

```
// for uft16 -> ansi decrypt
	LPCSTR inputText = "aFTexeG6jC7lNFgl1s9l+Q==";
	char outputText[1024] = {0,};
	int ret = 0;
	wchar_t ucInputText[1024]={0,};

	MultiByteToWideChar(CP_ACP,0,inputText,-1,ucInputText,1024);
	DWORD sizeOfCh = ::wcslen(ucInputText);
	int bufferSize = nwzDecryptUC(ucInputText,sizeOfCh,NULL,0);

	LPWSTR ucOutputText = NULL;
	if ( bufferSize > 0 )
	{
		ucOutputText = new wchar_t[bufferSize+1];
		ZeroMemory(ucOutputText,(bufferSize+1)*sizeof(wchar_t));
		ret = nwzDecryptUC(ucInputText,sizeOfCh,ucOutputText,bufferSize);
	}
	if( ret > 0 )
	{
		WideCharToMultiByte(CP_ACP,0,ucOutputText,-1,outputText,1024,NULL,NULL);
		_tprintf(_T("basic decrypt : %s₩n"),outputText);
	}

	if( NULL != ucOutputText ) delete [] ucOutputText;
	return 0;
```

4. 참고사항

1) win32 의 경우 이전 함수와 호환성을 위해서 다음 함수들은 남겨두었음

```
// 유니코드를사용하는경우
// encryptedBuffer = NULL , encryptedBufferSize = 0 을입력하면필요한encryptedBuffer 크기를리턴
int encryptPwdUC(const TCHAR * plain, const int plainSize, TCHAR * encryptedBuffer,int encryptedBufferSize,
int alg=0);
// plainBuffer = NULL , plainBufferSize = 0 을입력하면필요한plainBuffer 크기를리턴
int decryptPwdUC(const TCHAR * encrypted, const int encryptedSize, TCHAR * plainBuffer, int plainBufferSize,
int alg=0);

// ANSI char를사용하는경우
// encrypted = NULL , encryptedSize = 0 을입력하면필요한encrypted 크기를리턴
int encryptPwd(const char * plain, const int plainSize, char * encrypted, int encryptedSize, int alg=0);
// plain = NULL , plainSize = 0 을입력하면필요한plain 크기를리턴
int decryptPwd(const char * encrypted, const int encryptedSize, char * plain, int plainSize, int alg=0);
```