Archived:AES buffer encryption of files using Symbian C++

Archived: This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Description

According to the Advanced Encryption Standard (AES), a block of data can be encrypted and decrypted using the Archived:Symbian Cryptography APIs classes `CAESEncryptor` and `CAESDecryptor`, respectively. However, these classes can convert a data block of 16 bytes only and its not possible to encrypt/decrypt larger data blocks. For this purpose, the classes `CBufferedEncryptor` and `CBufferedDecryptor` can be used. These classes are intialised with an encryptor/decrytor class object, depending on the mode of operation, and a subclass of `CPadding`.

Solution

This example code encrypts a plain text file using AES encryption.

```
_LIT( KPlainInputFileName, "C:\\Data\\PlainInput.txt" );
_LIT( KEncryptedOutputFileName, "C:\\Data\\EnOutput.txt" );
_LIT8(KAESKey,"1234567890abcdef");   <br>
//Reading the plain text from the file
RFs fsSession;
User::LeaveIfError(fsSession.Connect());
RFile file;
User::LeaveIfError(file.Open(fsSession,KPlainInputFileName,EFileShareExclusive|EFileRead));
TInt plainTextsize;
file.Size(plainTextsize);
HBufC8* plainText = HBufC8::NewLC(plainTextsize);
TPtr8 plainTextPtr = plainText->Des();
file.Read(plainTextPtr);
file.Close(); <br>
//Initializing the encrytor and padding objects
CPaddingSSLv3* encryptPadding = CPaddingSSLv3::NewL(16);
CAESEncryptor* aesEncryptor = CAESEncryptor::NewL(KAESKey);
CBufferedEncryptor* bufEncryptor  = CBufferedEncryptor::NewL(aesEncryptor, encryptPadding);<br>
HBufC8* encryptData = HBufC8::NewLC(bufEncryptor->MaxFinalOutputLength(plainTextPtr.Size()));
TPtr8 encryptDataPtr = encryptData->Des();<br>
//Encrypting the text and writing into an output file
bufEncryptor->ProcessFinalL(plainTextPtr,encryptDataPtr);<br>
TInt err=file.Open(fsSession,KEncryptedOutputFileName,EFileShareExclusive|EFileWrite);
if (err==KErrNotFound) // file does not exist - create it
    err=file.Create(fsSession,KEncryptedOutputFileName,EFileShareExclusive|EFileWrite);
file.Write(encryptDataPtr);
file.Close();
CleanupStack::PopAndDestroy2);//encryptData,plainText
fsSession.Close();
```

This example code decrypts the file encrypted using the above code.

```
_LIT( KEncryptedOutputFileName, "C:\\Data\\EnOutput.txt" );
_LIT( KPlainOutputFileName, "C:\\Data\\PlainOutput.txt" );
_LIT8(KAESKey,"1234567890abcdef");  <br>
//Reading the encrypted text from the file
RFs fsSession;
User::LeaveIfError(fsSession.Connect());
RFile file;
User::LeaveIfError(file.Open(fsSession,KEncryptedOutputFileName,EFileShareExclusive|EFileRead));
TInt fileSize;
file.Size(fileSize);<br>
HBufC8* encryptData = HBufC8::NewLC(fileSize);
TPtr8 encryptDataPtr = encryptData->Des();
file.Read(encryptDataPtr);
file.Close();<br>
//Initializing the encrytor and padding objects
CPaddingSSLv3* decryptPadding = CPaddingSSLv3::NewL(16);
CAESDecryptor* aesDecryptor = CAESDecryptor::NewL(KAESKey);
CBufferedDecryptor* bufDecryptor = CBufferedDecryptor::NewL(aesDecryptor, decryptPadding);<br>
HBufC8* decryptedText = HBufC8::NewLC(bufDecryptor->MaxFinalOutputLength(encryptDataPtr.Size()));
TPtr8 decryptedTextPtr = decryptedText->Des();<br>
//Decrypting the text and writing the output into a file.
bufDecryptor->ProcessFinalL(encryptDataPtr,decryptedTextPtr); <br>
TInt err=file.Open(fsSession,KPlainOutputFileName,EFileShareExclusive|EFileWrite);
if (err==KErrNotFound) // file does not exist - create it
err=file.Create(fsSession,KPlainOutputFileName,EFileShareExclusive|EFileWrite);
file.Write(decryptedTextPtr);
file.Close();<br>
CleanupStack::PopAndDestroy(2); // encryptData,decryptedText
fsSession.Close();
```

Example application

File:BufEnEx.zip

(C) Copyright Nokia 2013. All rights reserved.

 (C) Copyright Nokia 2013. All rights reserved.