المقدمة

يعتبر الأمان شيء رئيسيُ للعديد مِنْ التطبيقاتِ. وعملية التحقُّق والتفويض للمستخدمين في التطبيقات جزء من الأمن العامِّ.ولكن ماذا عن البيانات المستخدمة والتي يتم إرسالها من والى التطبيق ؟ فهي في هذه الحالة تكون معرضة للتجسس والسرقة.
وهنا تبرز أهمية تشفير البيانات، علم التشفير موضوع ضخم . وفى هذا الدرس سنقوم فقط بشرح كيفية التشفير في NET Framework .

لماذا نستخدم التشفير ؟

في اغلب التطبيقات يتوفر فيها الأمان عن طريق شاشات الدخول وإعطاء الصلاحيات لكل مستخدم . على أية حال، ماذا لو أنّ شخص ما أعترض البياناتَ المرسلة عبر الشبكة ؟ (بواسطة أي أداة لمراقبة الشبكة أو برامج sniffer) ماذا لو أنّ شخص ما تلاعب بالبياناتِ التي تُرسَل على الشبكة ؟ ماذا لو أنّ شخص ما فَتحَ قاعدةَ بيانات SQL server التي تَخْزنُ كلماتَ السر؟

الحل هو تشفير البيانات :-

NET. Framework يعطيك فئات عن طريقها تُشفّرَ encrypt البياناتَ التي ترسل في نظامِكِ أوِ شبكتِكِ وبعد ذلك تفك التشفير decrypt فقط للمستخدم المخول بالتعديل أو القراءة. باختصار التشفير يزودنا بالميزّاتَ تاليةَ:

١- حماية البياناتَ المرسلة مِنْ القراءة من طرف ليس مخول
١- حماية البياناتَ المرسلة مِنْ أيّ تعديل
١- التأكّدْ بأنّ البياناتِ تَصِلُ مِنْ المكان الصحيح

أنواع فئات التشفير:-
وهى تصنف إلى أربع أنواع :-

١- الفئات التي تَتعاملُ مع المفتاح السري (يسمى التشفير المتناظر Symmetric Cryptography)
٢- الفئات التي تَتعاملُ مع المفتاح العامِّ (يسمى التشفير الغير متناظرCryptography Asymmetric)
٣- الفئات التي تَتعاملُ مع التواقيع الرقمية (signatures cryptographic)
٤- الفئات التي تَتعاملُ مع الأرقام المختلطة المشفّرة hashes cryptographic

كل هذه الفئات تجدها في فضاء الأسماء التالي:

System.Security.Cryptography

أولاً التشفير المتناظر :-

في التشفير المتناظر(المفتاح السري)، البيانات المحمية تكون مشفرة باستخدام مفتاح سري وحيد. هذا المفتاح معروف فقط للمرسل والمستلم. يشفّر المرسل البيانات باستعمال المفتاح السري. المستلم يفك تشفير البيانات باستخدام نفس المفتاح السري. ومن الهم جدا إخفاء المفتاح السري لان أي شخص يحصل عليه يصبح قادر على فك التشفير

إطار العمل يوفر الفئات التالية للتعامل مع التشفير المتناظر:-

* DESCryptoServiceProvider
* RC2CryptoServiceProvider
* RijndaelManaged
* TripleDESCryptoServiceProvider

وفى هذا المثال سنشرح كيفية التشفير المتناظر باستخدام algorithm Triple-DES

System.Security.Cryptography namespace يحتوي على الفئة TripleDESCryptoServiceProvider الذي يوفر إمكانية تشفير البيانات بخوارزميةTriple-DES.

يحتاج التشفير المتناظر لتشفير البيانات:-
١- مفتاح سري A secret key

٢-موجه تهيئةAn initialization vector

تستخدم خوارزميات التشفير سلسلة من التقنيات لتشفير البيانات. في هذه التقنية، كامل البيانات الّتي ستشفّر تقسم إلى كتل صغيرة. الكتلة المشفّرة سابقا من البيانات تستعمل لتشفير الكتلة الحالية وتتكرر العملية.

The Initialization Vector(IV) يستعمل لتشفير وفك تشفير الكتلة الأولى للبايتات. بهذا يضمن عدم تكرار الكتل المشفرة

لاستعمال TripleDESCryptoServiceProvider، مفتاح التشفير يجب أن يكون من حجم٢٤ بايت وموجه التهيئة يجب أن يكون من ٨ بايت.
مثال لاستخدام الفئةTripleDESCryptoServiceProvider

```
Imports System.Security.Cryptography
Imports System.IO
Imports System.Text

Public Class SecurityHelper
  Public Key() As Byte
  Public IV() As Byte

  Public Function Encrypt(ByVal strData As String) As Byte()
     Dim data() As Byte = ASCIIEncoding.ASCII.GetBytes(strData)
     Dim tdes As TripleDESCryptoServiceProvider = New  TripleDESCryptoServiceProvider
     If Key Is Nothing Then
        tdes.GenerateKey()
        tdes.GenerateIV()
        Key = tdes.Key
        IV = tdes.IV
     Else
        tdes.Key = Key
        tdes.IV = IV
     End If
     Dim encryptor As ICryptoTransform = tdes.CreateEncryptor()
     Dim ms As New MemoryStream
     Dim cs As CryptoStream = New CryptoStream(ms, encryptor, CryptoStreamMode.Write)
     cs.Write(data, 0, data.Length)
     cs.FlushFinalBlock()
     ms.Position = 0
     Dim result(ms.Length - 1) As Byte
     ms.Read(result, 0, ms.Length)
     cs.Close()
     Return result
  End Function

  Public Function Decrypt(ByVal data() As Byte) As String
     Dim tdes As TripleDESCryptoServiceProvider = New TripleDESCryptoServiceProvider
     tdes.Key = Key
     tdes.IV = IV
     Dim decryptor As ICryptoTransform = tdes.CreateDecryptor()
     Dim ms As New MemoryStream
     Dim cs As CryptoStream = New CryptoStream(ms, decryptor, CryptoStreamMode.Write)
     cs.Write(data, 0, data.Length)
     cs.FlushFinalBlock()
     ms.Position = 0
     Dim result(ms.Length - 1) As Byte
     ms.Read(result, 0, ms.Length)
     cs.Close()
     Return ASCIIEncoding.ASCII.GetString(result)
  End Function

End class
```

شرح الكود :-

قمنا بإنشاء فئة وأسميناها SecurityHelper وهى تحتوى على متغيرين عامين من النوع byte array والذين يمثلان المفتاح السري والموجه وكذلك يوجد به دالتين الأولى لتشفير البيانات والثانية لفك التشفير .

داله التشفير تستقبل نص لتشفيره وتعود بقيمة من النوع byte array تمثل النص المشفر
وفى هذه الدالة قمنا أولا بتعريف متغير من نوع byte array ووضعنا به النص المراد تشفيره بعد تحويله إلى bytes باستخدام الطريقة GetBytes()
ثم أنشئنا كائن من الفئة TripleDESCryptoServiceProvider
ثم نختبر إذا كان المستخدم أعطى قيمة للمفتاح والموجه فإذا لم يفعل نقوم بتوليدها عن طريق الفئة TripleDESCryptoServiceProvider
عرفنا متغير من الواجهة ICryptoTransform وهى تعرف العمليات الأساسية للتشفير واسندنا لها القيمة الناتجة من الطريقة CreateEncryptor()

عرفنا كائن من النوع MemoryStream حيث سنقوم بكتابة البيانات المشفرة إليه ويمكنك استخدام FileStream لتخزين البيانات على الهارد دسك

عرفنا كائن من النوع CryptoStream ومررنا له MemoryStreamو ICryptoTransformالسابق إنشائه ونمطه وهو هنا write
ثم قمنا بكتابة البيانات لتشفيرها بالكائن CryptoStream الذي يحتفظ بهذه البيانات في الذاكرة بواسطة الكائن MemoryStream

أخيرا قمنا بقراءة البيانات المشفرة من الـ memorystream ووضعها في متغير من byte array وهى القيمة التي تعود بها الدالة ، والآن أصبح بإمكانك تطوير هذه الفئة واستخدامها في تطبيقاتك لتشفير كلمات السر والبيانات المهمة .

ملاحظة :-
في الكود السابق داله التشفير تعود بمصفوفة من () byte وهى تمثل النص بعد تشفيره حيث قمنا بقراءتها من الـ MemoryStream وهو الذي وضعنا فيه النص المشفر ووضعناه في المتغير result الذي تعود به الدالة وبعد ذلك تستطيع إرسال النص مشفرا إلى قاعدة بيانات مثلا أو أي تطبيق أخر .

نتابع الحديث هنا عن التشفير الغير متناظر(المفتاح العام) :-

على خلاف التشفير المتناظر (بالمفتاح السري)، يستخدم في التشفير الغير متناظر مفتاحين . واحد يسمى المفتاح العام(public key) والأخر يسمى المفتاح الخاص (private key)

المفتاح العام(Public Key)

هو الرقم الذي يتم تداوله و نشره بين بقية المستخدمين لتشفير أي معلومات أو رسالة الكترونية مخصصة لك و يعتبر رقمك العام أساس عملية التشفير و لا يستطيع أحد فك رموز تلك المعلومة غيرك أنت لأنها تحتاج إلى الرقم السري و ليكن هو المفتاح الخاص بك لإكمال العملية الحسابية والوصول إلى الرقم الأساس وبالتالي فتح الملفات مرة أخرى

المفتاح الخاص (private key)

هو النصف الأخر المكمل للمفتاح العام للوصول إلى الرقم الأساس وإعادة المعلومات المشفرة إلى وضعها الطبيعي قبل التشفير ، و هذا المفتاح هو الذي يميز كل شخص عن غيره من المستخدمين ويكون بمثابة هوية الكترونية تمكن صاحبها من فك أي معلومة مشفرة مرسله إليه على أساس رقمه العام ولذلك يجب عليك الاحتفاظ بالمفتاح الخاص سرا .

يزودنا NET Framework . الفئات التالية للتعامل مع التشفير الغير متناظر:-

١ - Dsacryptoserviceprovider
٢ - Rsacryptoserviceprovider

تعتبر خوارزميةRSA إحدى اشهر الخوارزميات والتقنيات المستخدمة في التشفير المتناظر. الاختصار RSA يشير إلى , Rivest Shamir, and Adelman ،وهى أسماء مخترعي هذه التقنية. الدوت نت يزودنا بالفئة Rsacryptoserviceprovider الذي يغلّف هذه الخوارزمية. وسنحاول في هذا الدرس شرح هذه الفئة في تشفير البيانات.
حيث نحن كمبرمجين ومطورين لا تهمنا التركيبة الداخلية للخوارزم المستخدم ولكن ما يهمنا هو حماية بياناتنا من السرقة

سننشى فئةClass وسنسميها PublicKeySecurityHelper تحتوى على دالتين Method واحدة لتشفير البيانات والأخرى

لفك التشفير كما سننشى فئة أخرى وسنسميها MyRSAInfo وسنستخدمها في تخزين المفتاح العام والمفتاح الخاص

هذا هو الكود للفئتين

```
Imports System.Security.Cryptography
Imports System.IO
Imports System.Text

Public Class PublicKeySecurityHelper
  Public Function Encrypt(ByVal strData As String) As MyRSAInfo
     Dim myrsa As New MyRSAInfo
     Dim p As CspParameters = New CspParameters
     p.Flags = CspProviderFlags.UseMachineKeyStore
     myrsa.Parameters = p
     Dim rsa As RSACryptoServiceProvider = New RSACryptoServiceProvider(p)
     Dim data() As Byte = rsa.Encrypt(Encoding.Unicode.GetBytes(strData), False)
     myrsa.PublicKey = rsa.ToXmlString(False)
     myrsa.PrivateKey = rsa.ToXmlString(True)
     myrsa.Data = data
     Return myrsa
  End Function

  Public Function Decrypt(ByVal myrsa As MyRSAInfo) As Byte()
     Dim rsa As RSACryptoServiceProvider = New RSACryptoServiceProvider(myrsa.Parameters)
     rsa.FromXmlString(myrsa.PublicKey)
     Dim data() As Byte = rsa.Decrypt(myrsa.Data, False)
     Return data
  End Function

End Class

Public Class MyRSAInfo
  Public PublicKey As String
  Public PrivateKey As String
  Public Parameters As CspParameters
  Public Data() As Byte
End Class
```

شرح الكود :-

أولا تشفير البيانات Encrypting data :-

- في البداية قمنا باستيراد فضاءات الأسماء التي سنتعامل معها وأهمها هنا System.Security.Cryptography لأنه هو الذي يحتوى على الفئة RSACryptoServiceProvider

- قمنا بإنشاء الدالة Encrypt () وهى المستخدمة في التشفير وهى تستقبل نص (المراد تشفيره) وتعود بنسخة من الفئة MyRSAInfo وهى تحتوى على النص المشفر

- الفئة MyRSAInfo تحتوى على أربعة أعضاء أو متغيرات هي: PublicKey ويستخدم لتخزين المفتاح العام ، PrivateKey لتخزين المفتاح الخاص الذين يتم توليدهم ، Parametersمتغير من نوع CspParameters الذي يستخدم لتوليد المفاتيح العامة والخاصة ، Data()مصفوفة من البايتات وهى تمثل النص المشفر

- نعود لدالة التشفير حيث أنشئنا فيه كائن من الفئة CspParameters واسندنا للخاصية Flag القيمة CspProviderFlags.UseMachineKeyStore وهى إحدى قيم الـ enumeration التي تأخذها وهذه القيمة تحدد من أي مكان سيأخذ معلومات المفتاح

- ثم أنشئنا كائن من الفئة RSACryptoServiceProvider ومررنا لها الكائنCspParameters السابق إنشاءه والمسمى p

- قمنا باستدعاء الطريقة Encrypt() method للفئة RSACryptoServiceProvider ومررنا لها البيانات المراد تشفيرها وذلك طبعا بعد تحويلها إلى مصفوفة من البايتات باستخدام الطريقة GetBytes () التي تعود للفئة Encoding الموجودة في فضاء الأسماء Text والباروميتر الثاني للطريقة Encrypt () هو من النوع Boolean فإذا كان True يتم استخدام OAEP

padding في التشفير وللعلم هذا الشكل لا يعمل إلا على Windows XP ولذلك قمنا بتمرير القيمة False والتي تشير إلى استخدام PKCS#1 v1.5 padding ، وطبعا الدالة أو الطريقة() Encrypt تعود بمصفوفة من البايتات byte array تمثل البيانات بعد تشفيرها وقمنا بإسنادها للمتغير من نفس النوع أسميناه data

- أخيرا قمنا بتعبئة أعضاء الفئة MyRSAInfo التي أنشئنا نسخة منها وأسميناها myrsa

واستخدمنا الدالة ToXmlString والتي تعود بنص على هيئه xml يمثل كائن RSA وتأخذ بارومیتر من النوع Boolean فإذا كان true فستعود بنص يشمل حتى المفتاح الخاص

ثانيا فك التشفير Decrypting data :-

- قمنا بإنشاء الدالة () Decrypt التي تستقبل كائن من الفئة MyRSAInfo وهذا الكائن يجب أن يكون إحدى النسخ التي تعود بها الطريقة () Encrypt أي أنه يجب أن يكون الكائن الممرر للدالة () Decrypt هو نفس الكائن التي تعود به الدالة التي استخدمت لتشفير البيانات المراد فك تشفيرها وطبعا هذه الدالة تعود بمصفوفة من البايت تمثل البيانات الطبيعية

- داخل الدالة () Decrypt قمنا بإنشاء كائن من الفئة RSACryptoServiceProvider ومررنا له نفس البارومیتر

- قمنا باستدعاء الطريقة() FromXmlString للفئة RSACryptoServiceProvider ومررنا لها المفتاح العام الذي أنشئ سابقا

- أخيرا قمنا باستدعاء الطريقة () Decrypt أيضا الخاصة بالفئةRSACryptoServiceProvider ومررنا لها البارومیتر الخاص بها وهى تأخذ نفس البارومیتر الخاص بالطريقة () Encrypt ولكن تقوم بعكس الوظيفة .

تقنية الأرقام المختلطةHashes :-

خوارزميات الأرقام المختلطة تنشئ مخرجات ثابتة الطول لمعطيات متغيرة من البيانات. فإذا قام أي شخص بتغيير البيانات الأصلية فستكون الأرقام المولدة مختلفة عن الأرقام المولدة الأصلية وبهذه الطريقة تستطيع التأكد من صحة البيانات إذا قام احدهم بالتلاعب فيها. وهى غالبا تستخدم في التوقيع الرقمي .

وهذه هي الفئات التي تتعامل مع الأرقام المختلطة (hashes) :

١ - SHA1Managed
٢ - MD5CryptoServiceProvider
٣ - MACTripleDES

ملاحظة: خوارزمية SHA1 تم كسرها من قبل القراصنة فهي لم تعد مستخدمه عمليا .

والآن سنقوم بعمل مثال لاستخدام هذه الطريقة باستخدام خوارزمية MD5
وهذا هو الكود

```
Imports System
Imports System.Security.Cryptography
Imports System.Text
Public Class MD5HashHelper
  Public Function GetHash(ByVal message As String) As Byte()
     Dim data As Byte()
     data = UTF8Encoding.ASCII.GetBytes(message)
     Dim md5 As MD5CryptoServiceProvider = New MD5CryptoServiceProvider
     Return md5.ComputeHash(data, 0, data.Length)

  End Function

  Public Function VerifyHash(ByVal message As String, ByVal hash As Byte()) As Boolean
     Dim data As Byte()
     data = UTF8Encoding.ASCII.GetBytes(message)
     Dim md5 As MD5CryptoServiceProvider = New MD5CryptoServiceProvider
     Dim hashTemp As Byte() = md5.ComputeHash(data, 0, data.Length)
     For counter As Int32 = 0 To hash.Length - 1 Step 1
        If hash(counter) <> hashTemp(counter) Then
           Return False
        End If
     Next
     Return True

  End Function
End Class
```

شرح الكود :-

- في البداية قمنا باستيراد فضاءات الأسماء اللازمة
- ثم قمنا بإنشاء الفئة MD5HashHelper وهى تحتوى على الطريقتين ،GetHash VerifyHash
- الدالة ()GetHash تستقبل نص وتعود الدالة بمصفوفة من البايت تمثل قيمه الهاش المحسوب
- في داخل الدالة ()GetHash استخدمنا الفئة UTF8Encoding للحصول على byte array تمثل النص المرسل
- ثم أنشئنا كائن من الفئة MD5CryptoServiceProvider واستدعينا الطريقة ComputeHash ومررنا لها النص المحول إلى byte array
- الدالة ComputeHash تستقبل byte array وتقوم بحساب قيمة الهاش له وتعود أيضا ب byte array ويمثل قيمة الهاش
- الدالة VerifyHash تستقبل نص وهو النص الذي نريد التحقق من صحته وكذلك تستقبل byte array تمثل قيمة الهاش المولدة مسبقا لهذا النص وهى تعود بقيمة منطقية true إذا لم يتم العبث بالبيانات والعكس
- في هذه الدالة قمنا بمثل ما عملناه في الدالة السابقة حيث قمنا بتوليد قيمة هاش للنص وثم عملنا حلقة دوران وقمنا فيها بمقارنه كل بايت لكل من الهاش المولد وقيمة الهاش المولدة مسبقا فإذا كانت متطابقة عدنا بtrue وإذا اختلف عدنا بـ false وهكذا بهذه الطريقة تستطيع التأكد من صحة البيانات المرسلة إذا كانت سليمة أو تم العبث بها .

التواقيع الرقمية Digital Signatures :-

التواقيع الرقمية تستخدم للتحقق من هوية المرسل والتأكد من سلامة البيانات . وهو يستخدم غالبا مع التشفير الغير متناظر(المفتاح العام)

آلية عمل التواقيع الرقمية :-

يطبّق المرسل خوارزمية hash إلى البيانات المرسلة وينشئ رسالة ملخص . هذه الرسالة هي عبارة عن تمثيل وتوضيح للبيانات المرسلة. ثم يقوم المرسل بتشفير الرسالة مع المفتاح الخاص للحصول على التوقيع الرقمي
وبعد ذلك يقوم بإرسال البيانات ضمن قناة آمنه
يستلم المستلم البيانات ويفك تشفير التوقيع الرقمي باستخدام المفتاح العام ويسترجع الرسالة الملخصة
يطبّق المستلم نفس خوارزمية الـ hash التي استخدمها المرسل وينشئ رسالة ملخص جديدة للبيانات
إذا تطابق ملخص المستلم مع ملخص المرسل فان هذا يعنى أن الرسالة قادمة من المكان الصحيح

إطار العمل دوت نت يوفر لنا الفئات التالية للعمل مع التواقيع الرقمية :-

RSACryptoServiceProvider للتشفير اللا متناظر

RSAPKCS1SignatureFormatter للتواقيع الرقمية

في هذا المثال سننشئ فئة ونسميها DigitalSignatureHelper وهى التي سنستخدمها في إنشاء التواقيع الرقمية والتأكد منها وكذلك سنستخدم الفئة MD5HashHelper لتوليد قيم الهاش

وهذا هو الكود

```
Imports System
Imports System.Security.Cryptography

Public Class DigitalSignatureHelper
  Private m_private As RSAParameters
  Private m_public As RSAParameters
  Public Function CreateSignature(ByVal hash As Byte()) As Byte()
     Dim RSA As RSACryptoServiceProvider = New RSACryptoServiceProvider
     Dim RSAFormatter As RSAPKCS1SignatureFormatter = New RSAPKCS1SignatureFormatter(RSA)
     RSAFormatter.SetHashAlgorithm("MD5")
     m_public = RSA.ExportParameters(False)
     m_private = RSA.ExportParameters(True)
     Return RSAFormatter.CreateSignature(hash)

  End Function
  Public Function VerifySignature(ByVal hash As Byte(), ByVal signedhash As Byte()) As Boolean
     Dim RSA As RSACryptoServiceProvider = New RSACryptoServiceProvider
     Dim RSAKeyInfo As RSAParameters = New RSAParameters
     RSAKeyInfo.Modulus = m_public.Modulus
     RSAKeyInfo.Exponent = m_public.Exponent
     RSA.ImportParameters(RSAKeyInfo)
     Dim RSADeformatter As RSAPKCS1SignatureDeformatter = New _
RSAPKCS1SignatureDeformatter(RSA)
     RSADeformatter.SetHashAlgorithm("MD5")
     Return RSADeformatter.VerifySignature(hash, signedhash)
  End Function
End Class
```

شرح الكود

• أنشئنا الفئة وهى تحتوى على متغيرين من النوع RSAParameters يتم فيها تخزين المعلومات عن المفتاح الخاص والعام

• أنشئنا داله CreateSignature() وهى تستقبل مصفوفة بايت تمثل قيمة الهاش وتعود بمصفوفة بايت تمثل التوقيع الرقمي للهاش

• في داخل الدالة قمنا بإنشاء كائن من الفئة RSACryptoServiceProvide وأنشئنا كائن آخر من الفئة RSAPKCS1SignatureFormatter ومررنا له الكائن المنشئ من الفئة الأولى

• الفئة RSAPKCS1SignatureFormatter مسئولة عن إنشاء ما يسمى PKCS #1 أو(Public Key Cryptographic signature Signature version 1.5)

• قمنا باستدعاء الطريقة SetHashAlgorithm() التي تأخذ اسم للخوارزم المستخدم في توليد الهاش كباراميتر ونحن هنا نستخدم خوارزم MD5

• قمنا باستدعاء الطريقة ExportParameters () للحصول على المفتاح العام والخاص وتخزينهما في المتغيرات

• وأخر شيء قمنا باستدعاء الطريقة CreateSignature وهى التي تقوم بإنشاء التوقيع الرقمي وتعود به على هيئة مصفوفة بايت وهى القيمة التي تعود بها دالتنا

وبهذا أكون قد تطرقت إلى جميع طرق التشفير والتأكد من البيانات أرجو من الله العلى التوفيق

وبما أننا نحتاج لإنشاء مفاتيح للتشفير فنحن بحاجة إلى شيء لتوليد أرقام عشوائية تودي لنا هذا الغرض الدوت نت يقدم لنا هذه الفئة التي تقوم بهذا الغرض :-

RNGCryptoServiceProvider