رمز نگاري در .NET

در اين مقاله قصد داريم بطور خلاصه نگاهي داشته باشيم بر رمزنگاري و كارهايي كه در .NET Framework براي حمايت از رمزنگاري انجام شده است . در اينجا كار را با معرفي رمزنگاري شروع خواهيم كرد و سپس به بررسي روش هاي مختلف اين كار مي پردازيم و در قسمت دوم به بررسي الگوريتم ها و شي هاي مختلفي كه در .NET براي اين كار تدارك ديده شده است مي پردازيم .

رمزنگاري

ما اغلب بدون آنكه بدانيم از رمزنگاري استفاده مي كنيم براي مثال ، زماني را به ياد آوريد كه سعي مي كنيد با استفاده از علائم پيامي را به دوستتان برسانيد ، در حاليكه معني علامت ها را فقط شما و دوستتان مي دانيد و يا نوشته اي كه به گونه اي نوشته شده است كه فقط خودتان و كساني كه شما را مي شناسند مي توانند بخوانند ؛ بنابراين همه ما در زندگي از رمزنگاري استفاده كرده ايم و اين كار جديدي نيست .

رمزنگاري دانش بهم ريختن كاراكترهاي بامعني براي ايجاد رشته اي بي معني براي كساني كه نبايد به متن و داده اصلي دسترسي داشته باشند مي باشد . دانش رمزنگاري سال هاست كه وجود دارد ، حتي قبل از بوجود آمدن كامپيوترها . رمزنگاري ، در طول سالیان دراز هنري بوده كه توسط افراد مختلف به كار برده مي شده است و تكنيك هاي مختلفي براي ايجاد امنيت اطلاعات از آنها باقي مانده است . در بيست سال گذشته اين هنر تبديل به دانش فراگيري شده است . با ورود كامپيوترها علم توانسته رمزهاي نسبتاً غير قابل شكستي ايجاد كند .

رمزنگاري به عنوان يكي از پيچيده ترين جنبه هاي برنامه نويسي توسط برنامه نويسان در نظر گرفته مي شود . استفاده از الگوريتم هاي رمزنگاري به مانند بازي كودكانه اي كه هر كسي بتواند انجام دهد نمي باشد ؛ بلكه به دانش سطح بالايي از رياضيات نياز دارد . خوشبختانه در MICROSOFT .NET كلاس هاي جديدي ايجاد شده كه اين الگوريتم هاي پيچيده و مشكل را به پروپرتي ها و متد هاي سهل الاستفاده اي براي كاربران تبديل كرده است . اين مقاله نگاه كلي بر روش هاي رمزنگاري تدارك ديده شده توسط .NET Framework خواهد داشت .

در ابتدا به معرفي اصطلاحات فني بكار برده شده در رمزنگاري مي پردازيم :

• PlainText يا ClearText : به داده هايي كه بدون وجود معيار خاصي بتوان آنها را خواند و فهميد PlainText مي گويند .

• Encryption (رمز بندي) : به روش تغيير شكل دادن PlainText به فرمي كه مفهوم آن مشخص نباشد رمزبندي اطلاعات مي گويند .

• Key (كليد) : كليد رشته اي از بيت ها مي باشد كه براي رمزنگاري و رمزگشايي اطلاعاتي كه بايد انتقال يابد استفاده مي شود . اين رشته به صورت مجموعه اي از عددها / كاراكترها مي باشد كه به صورت تصادفي توليد مي شود تا بوسيله آن بتوان اطلاعات را رمزنگاري / رمزگشايي نمود .

انواع رمزنگاري

پس از آشنايي با واژگان فني ، اجازه دهيد تا با انواع رمزنگاري آشنا شويم . دو روش براي رمزنگاري وجود دارد : رمزنگاري با استفاده از كليد خصوصي و رمزنگاري با استفاده از كليد عمومي .

رمزنگاري با استفاده از كليد خصوصي

رمزنگاري با استفاده از كليد خصوصي ، تا قبل از ظهور رمزنگاري با استفاده از كليد عمومي در سال ١٩٧٦ تنها روش مرسوم براي اينكار بوده است . امپراتوراني همچون ژوليس سزار از اين روش براي انتقال پيام هاي رمز استفاده مي كرده اند . در اين روش نياز است كه طرفين ارتباط از يك كليد مشترك استفاده كنند كه همان كليد خصوصي مي باشد . در روش كليد خصوصي، شما براي به رمز درآوردن پيامتان از يك كليد كه فقط شما از آن اطلاع داريد استفاده مي كنيد و براي رمزگشايي پيام نيز به همان كليد نياز مي باشد . استفاده از روش كليد خصوصي تنها هنگامي كه كليد رمز محرمانه باشد موثر مي باشد . با وجود اين ضعف باطني وليكن براي ايجاد و اجراي اين روش منابع زيادي مورد نياز نيست و كار نسبتاً ساده اي مي باشد . اجازه دهيد مثالي را بيان كنم – در نظر بگيريد كه ژوليس سزار مي خواهد با استفاده از كليد خصوصي پيام محرمانه اي را به فرماندهان ارتش خود برساند ؛ براي اينكه فرماندهان بتوانند پيام او را رمزگشايي كنند نياز است تا كليد خصوصي را بدانند ؛ بنابراين لازم است كه ژوليس كليد خصوصي را نيز برايشان بفرستد . اما اگر اين كليد و رمز آن به دست دشمنان سزار بيفتد ، ديگر پيام امنيت خود را از دست مي دهد . علاوه بر اين، اگر فرمانده كليد خصوصي را به افسر زيردست خود بگويد او نيز مي تواند پيام را رمزگشايي نمايد .

رمزنگاري با استفاده از كليد عمومي

رمزنگاري با كليد عمومي بر اين اساس بنا شده است كه فرستنده و گيرنده پيام هر كدام كليد خصوصي خواهند داشت كه فقط خودشان مي دانند و يك كليد عمومي وجود دارد كه هر كسي مي تواند آن را بداند . هر فرآيند رمزنگاري /رمزگشايي حداقل به يك كليد عمومي و يك كليد خصوصي نياز دارد ، كه هر كدام به هم با قوانين رياضي مرتبط هستند . در اين صورت پيامي را كه توسط كليد عمومي به رمز در آورده شده باشد فقط با استفاده از كليد خصوصي مرتبط به آن ، مي توان رمزگشايي كرد .

در اينجا به بررسي نمونه اي مي پردازيم – قبل از اينكه ژوليس پيامي براي فرمانده سپاهش بفرستد ، او نياز دارد كه جفت كليد خصوصي و عمومي را ايجاد كند . بنابراين فرمانده آزادانه مي تواند كليد عمومي را در بين افراد زيردستش پخش كند ، اما كليد خصوصي را نزد خود نگه مي دارد . هنگامي كه ژوليس مي خواهد پيامي براي فرمانده سپاه بفرستد ، از كليد عمومي براي رمزنگاري پيام استفاده مي كند و سپس اين پيام را مي فرستد . در اين مورد او تنها كسي است كه مي تواند پيام را رمزگشايي كند . از آنجايي كه جفت كليد به اين صورت عمل مي كنند كه تنها پيام هايي كه با كليد عمومي رمزنگاري شده باشند را مي توان با كليد خصوصي رمزگشايي كرد بنابراين ديگر نيازي به انتقال كليد هاي محرمانه نمي باشد . به اين ترتيب خطر لو رفتن كليد از بين مي رود .

عكس اين عمل هم به خوبي انجام مي شود . فرض كنيد كه فرمانده پيامي را كه توسط كليد خصوصي اش رمزنگاري شده است را براي ژوليس بفرستد .در اين صورت ژوليس با استفاده از كليد عمومي مي تواند پيام را رمزگشايي كند .اما مي دانيم كه كليد عمومي محرمانه نيست و هر كسي مي تواند آن را داشته باشد . به هر حال استفاده از اين متد دستكاري نشدن پيام را توسط ديگران تضمين مي كند چون در صورت هر گونه تغييري ديگر نمي توان پيام را با استفاده از كليد عمومي رمزگشايي كرد.

.NET و رمز نگاري

.NET مجموعه اي از اشيا را براي رمزنگاري تدارك ديده است كه الگوريتم هاي مشهور و پركاربردي همچون hashing (درهم سازي) ، encryption (رمزبندي) و توليد امضاي ديجيتال را حمايت مي كنند . اين اشيا طوري طراحي شده اند كه بتوان با تركيب اين توانايي هاي ابتدايي اعمال پيچيده تري مانند امضا و رمزنگاري مستندات را به راحتي انجام داد . اشيا رمزنگاري در .NET براي پشتيباني از سرويس هاي

داخلي استفاده مي شوند اما همچنين براي برنامه نويساني كه به حمايت هاي رمزنگاري نياز دارند نيز در دسترس مي باشند . در .Framework NET بسياري از اين الگوريتم ها و اشيا ي استاندارد رمزنگاري پياده سازي شده است و قابل استفاده مي باشند . مانند دسترسي آماده به قابليت هاي ساده تاييد اعتبار در .NET Framework ؛ اشياي اصلي رمزنگاري نيز همچنين به سادگي از طريق استفاده از كتابخانه كدهاي مديريت شده اي كه بر پايه جريان داده (Stream Based) مي باشد امكان پذير مي باشد . اين سرويس هاي رمزنگاري در فضاي نام System.Security.Cryptography در .NET Framework تدارك ديده شده اند . الگوريتم هاي پياده سازي شده در اين فضاي نام شامل موارد زير مي باشد :

• الگوريتم هاي رمزنگاري با استفاده از كليد عمومي (نامتقارن) RSA و DSA : الگوريتم هاي نا متقارن بر روي يك بافر ثابت عمليات انجام مي دهند . آنها از كليد عمومي براي رمزنگاري / رمزگشايي استفاده مي كنند . از الگوريتم RSA مي توان به عنوان يك الگوريتم نامتقارن نام برد كه RSA حروف ابتدايي نام سه مخترع آن يعني Rivest، Shamir، Adleman مي باشد . اين يك الگوريتم كليد عمومي مشهور مي باشد كه از نسخه استاندارد غير رسمي آن براي امضاي ديجيتال و همچنين رمزنگاري به خوبي مي توان استفاده كرد . DSA_CSP يك پياده سازي براي الگوريتم امضاي ديجيتال DSA (Digital Signature Algorithm) مي باشد ؛ اين يك الگوريتم كليد عمومي مي باشد كه از آن مي توان براي ايجاد و تاييد امضاي ديجيتال استفاده كرد .

• الگوريتم هاي رمزنگاري با استفاده از كليد خصوصي (متقارن) RSA، DES و TripleDES : از الگوريتم هاي متفارن براي تغيير دادن بافرهاي با طول متغير و انجام يك عمل بر روي داده هاي ورودي تكرار شونده استفاده مي شود . در اين روش ها از يك كليد محرمانه براي رمزنگاري و رمزگشايي داده استفاده مي كنند . استاندارد رمزنگاري داده Data Encryption Standard) DES)يك استاندارد گسترده جهاني براي رمزنگاري داده مي باشد كه در ابتداي دهه ۱۹۷۰ منتشر شد . اين مشهورترين الگوريتم رمزنگاري مي باشد كه با استفاده از كلاس DES_CSP پياده سازي شده است . اين كلاس جرياني را ارائه مي كند كه شما آن را در داده قرار مي دهيد و آن با استفاده از يك كليد واحد عمليات رمزنگاري / رمزگشايي را انجام مي دهد . الگوريتم TripleDES سه بار بر روي يك بلوك داده با استفاده از يك كليد عملياتي را انجام مي دهد . RC2 كه سر نام Rivest Cipher يا Rivest) Ron’s Code و Ron نام مخترعان اين الگوريتم مي باشد) مي باشد . RC2 يك الگوريتم متقارن مي باشد كه با كليد با سايز متفاوت كار مي كند .اين روش مانند

بسياري از روش هاي ديگر رمزنگاري در .NET. مي باشد كه ابتدا يك بلوك داده محرمانه ايجاد مي كند و سپس بر روي آن عملياتش را انجام مي دهد كه اين روش در مقابل الگوريتم هاي جريان داده محرمانه ساز قرار دارد .

• الگوريتم هاي درهم سازي MD5 و SHA1: يك الگوريتم درهم سازي يك طرفه مي باشد كه داده اي با طول متغير را از ورودي مي گيرد و يك مقدار Hash، 128 بيتي توليد مي كند. الگوريتم SHA1 نيز يك روش Hashing مي باشد كه يك مقدار ١٦٠ بيتي Hash ايجاد مي كند كه از مقداري كه MD5 توليد مي كند طولاني تر است .

(Cryptographic Service Provider (CSP **چيست ؟**

CSP يك موجوديت براي اجراي محاسبات رمزنگاري مي باشد . كلاس CSP از كلاس هاي متناظر پايه مشتق شده است و راه حل هايي را براي يك الگوريتم مشخص پياده سازي مي كند . براي مثال، كلاس DESCryptoServiceProvider از كلاس DES مشتق شده است و استاندارد رمزنگاري ديجيتال را پياده سازي كرده است . شما مي توانيد هم از كلاس هاي تدارك ديده شده و هم از راه حل هاي خودتان استفاده نماييد .

از كدام الگوريتم استفاده كنيم ؟

در اينجا به ارائه يك راهبرد عمومي براي كمك در تصميم گيري در مورد استفاده از روش هاي مختلف رمزنگاري مي پردازيم .
الگوريتم هاي متقارن يا كليد خصوصي الگوريتم هاي خيلي سريعي هستند كه براي رمزنگاري جريان هاي بزرگ داده بسيار مفيد مي باشند . از اين الگوريتم ها هم براي رمزنگاري و هم رمزگشايي استفاده مي شود . اين روش ها با آنكه بطور منصفانه اي امن مي باشند ولي در صورت وجود وقت كافي كاملاً قابل شكستن مي باشند . به عنوان مثال يك شخص مي تواند با انجام يك جستجو بر روي تركيب هاي مشخصي از مقادير به كلمه رمز دست يابد . از آنجايي كه هر كدام از اين الگوريتم ها از يك كليد با طول ثابت يا كاراكترهاي ASCII استفاده مي كنند بنابراين يك برنامه رمزياب مي تواند با امتحان كردن تركيب هاي متفاوت در نهايت به يك مقدار رمز دست يابد . يكي از استفاده هاي معمول از اين نوع الگوريتم ها براي ذخيره و بازيابي رشته هاي ارتباطي به پايگاه داده مي باشد .

الگوريتم هاي نامتقارن يا كليد عمومي به سرعت الگوريتم هاي متقارن عمل نمي كنند . اما شكستن آنها بسيار مشكل تر مي باشد . اين الگوريتم ها از دو كليد ، يكي خصوصي و ديگري عمومي استفاده مي كنند . از كليد عمومي براي به رمز در آوردن پيام و از كليد خصوصي فقط براي رمزگشايي پيام استفاده مي شود . كليدهاي خصوصي و عمومي با قوانين رياضي به هم پيوند خورده اند و بنابراين براي انجام موفقيت آميز عمليلت رمزنگاري به هر دوي اين كليد ها نياز است . الگوريتم هاي نامتقارن به خاطر كاهش كارآيي خيلي براي استفاده همراه مقادير زياد داده مناسب نمي باشند . يكي از استفاده هاي معمول الگوريتم هاي نامتقارن در به رمز درآوردن و انتقال يك كليد رمز متقارن به بخش ديگر و آغاز مسير انتقال اطلاعات مي باشد و سپس از الگوريتم هاي متقارن براي ارسال پيام ها در مسيري آزاد و امن استفاده مي شود .

مقادير درهم سازي شده هنگامي استفاده مي شود كه شما به هيچ وجه نمي خواهيد مقادير اصلي بازيابي شوند و يا كسي به راحتي بتواند آن را كشف كند. درهم سازي بر روي هز زشته اي با طول دلخواه انجام مي شود . كه حاصل آن مجموعه اي ثابت از بايت ها مي باشد . از اين روش مي توان بر روي مقادير كوچكي از داده مانند كلمات رمز استفاده كرد . اگر كاربر يك كلمه رمز را در يك صفحه ورودي امن وارد كند برنامه مي تواند اين مقدار را Hash كرده و در پايگاه داده نگهداري كند . حتي اگر اين پايگاه داده نيز هك شود ، از آنجايي كه اين كلمه رمز به صورت Hash شده ذخيره شده است بنابراين ديگر كسي نمي تواند به مقدار واقعي آن دسترسي پيدا كند . وقتيكه كاربر براي ورود به سيستم اقدام مي كند و كلمه رمز خود را وارد مي كند اين كلمه Hash مي شود و با كلمه رمز Hash شده اي كه در پايگاه داده نگهداري مي شود مقايسه مي شود و در صورتي كه برابر باشند تاييد و وارد سيستم مي گردد .

حال پس از اينكه با الگوريتم هاي مختلف رمزنگاري تاحدودي آشنا شديد اجازه دهيد تا به بررسي يك مثال از رمزنگاري و رمزگشايي فابل ها با استفاده از فضاي نام System.Security.Cryptography بپردازيم . در اين مثال از روش مديريت شده ريجنديل (Rijndeal) استفاده شده است . كلاس مديريت شده Rijndeal از نسخه مديريت شده الگوريتم ريجنديل استفاده مي كند . اين كلاس غير قابل ارث بري مي باشد . اين كلاس در فضاي نام زير قرار دارد .

System.Object

System.Security.Cryptography.SymmetricAlgorithm

System.Security.Cryptography.Rijndael

System.Security.Cryptography.RijndaelManaged

کد اين نمونه به صورت زير مي باشد .

```
// Encrypting and decrypting files using the Rijndael Managed
// encryption method.
using System;
using System.IO;
using System.Security.Cryptography;

class CryptoEx
{
   public static void Main(string[] args)
   {
      if (args.Length != 1)
      {
         Console.WriteLine("FileName Not Entered. Specify a
            filename to encrypt.");
         return;
      }

      string file = args[0];
      string tempfile = Path.GetTempFileName();
```

```
// Open the file to read
FileStream fsIn  = File.Open(file,FileMode.Open,FileAccess.Read);
FileStream fsOut =
    File.Open(tempfile, FileMode.Open,FileAccess.Write);



// Create an instance and calling the CreateEncryptor method which
// creates a symmetric encryptor object

SymmetricAlgorithm symm = new RijndaelManaged();
ICryptoTransform transform = symm.CreateEncryptor();
CryptoStream cstream =
    new CryptoStream(fsOut,transform,CryptoStreamMode.Write);


BinaryReader br = new BinaryReader(fsIn);


cstream.Write(br.ReadBytes((int)fsIn.Length), 0, (int)fsIn.Length);


cstream.FlushFinalBlock();
cstream.Close();
fsIn.Close();
fsOut.Close();


Console.WriteLine("Created Encrypted File {0}", tempfile);
```

```
        fsIn = File.Open(tempfile,FileMode.Open,FileAccess.Read);

        transform = symm.CreateDecryptor();

        cstream = new CryptoStream(fsIn,transform,CryptoStreamMode.Read);


        StreamReader sr = new StreamReader(cstream);

        Console.WriteLine("Decrypted the File: " + sr.ReadToEnd());

        fsIn.Close();
}
```