FÁBIO ANTÔNIO OLVEIRA NOVAIS

SYSTEM.SECURITY E SYSTEM.SECURITY.CRYPTOGRAPH NAMESPACES

Salvador
2009

FÁBIO ANTÔNIO OLVEIRA NOVAIS

SYSTEM.SECURITY E SYSTEM.SECURITY.CRYPTOGRAPH NAMESPACES

Atividade apresentada como requisito para avaliação semestral da disciplina linguagens para aplicação comercial no curso de Ciência da Computação da Universidade Federal da Bahia orientada pelo professor Adonai Medrado.

Salvador
2009

SUMÁRIO

1. INTRODUÇÃO

1.1.CONCEITOS DE SEGURANÇA E CRIPTOGRAFIA

Com o avanço da internet, o fluxo de informações entre pessoas e entidades tornou-se ainda mais intenso e a possibilidade de interceptação destas transforma-se, muitas vezes, em negócio lucrativo. Nesse cenário, onde nem sempre se reina a ética, a transmissão de dados deve ser feita de forma segura. Surge nesse contexto a criptografia, que é a técnica de codificar e transmitir dados de maneira que apenas pessoas devidamente aptas ou desejadas sejam capaz de decifrá-las. Com isso, informações podem ser trocadas com maior confiança, mas ainda sim, existem modos de "quebrar" essa codificação e recuperar os dados originais.

2. .NET FRAMEWORK

O .Net Framework fornece alguns conjuntos de classes, interfaces e enumeradores que ajudam a manter a integridade e segurança dos dados em aplicações desenvolvidas nas suas linguagens, geralmente C# ou VB.NET. Esses conjuntos estão disponíveis nos seguintes namespaces:

- System.Security
- System.Web.Security,
- System.Security.Cryptograph
- System.Security.Principal
- System.Security.Policy
- System.Security.Permissions

Esse documento abordará características de apenas dois namespaces: o System.Security e o System.Security.Cryptograph.

2.1.SYSTEM.SECURITY NAMESPACE

Esse namespace fornece dois tipos de maneiras de implementar segurança nas aplicações: a segurança baseada na função e a segurança de acesso ao código. A primeira diz respeito à autorização de usuários ou grupos de usuários para acessar certas partes do código de uma aplicação bem como suas permissões para realizar determinadas tarefas. Já a segunda se refere ao gerenciamento das permissões de determinados códigos a acessarem recursos protegidos e realizar operações privilegiadas dentro do sistema.

No site da MSDN, no seguinte link http://msdn.microsoft.com/en-us/library/system.security.aspx, encontra-se uma descrição completa das definições de classe para esse namespace, inclusive com exemplos de aplicação.

2.1.1. SecureString

Um objeto do tipo String nada mais é do que uma coleção de objetos Char agrupados. Por ser imutável, não permite mudança de valor após ser instanciado. Por esse motivo, quando um novo valor é atribuído ao objeto, uma nova instância da referida classe é criada e o valor antigo do objeto é teoricamente descartado pela CLR. Porém, isso nem sempre acontece, e um valor importante, como o de uma senha ou um número de cartão de crédito pode ficar armazenado na memória e eventualmente recuperado.

A classe SecureString cria uma espécie de string segura, mutável, que é encriptada automaticamente no momento de sua criação.

A seguir um exemplo de implementação dessa classe.

```
protected void SString()
{
      //Cria instância da classe
      SecureString senha = new SecureString();

      //Verifica se o objeto é read-only, caso seja, descarta
      //a instência atual e cria uma nova
      if (senha.IsReadOnly())
      {
            senha.Dispose();
            senha = new SecureString();
      }
      //adiciona caracter por caracter o valor da
      //textbox ao objeto SecureString
      if (txtSenha.Text != String.Empty)
      {
            foreach (char ch in txtSenha.Text)
            {
                senha.AppendChar(ch);
            }
      }
```

```
        //O objeto bstr terá uma cópia do objeto
        //SecureString decriptada
        IntPtr bstr = Marshal.SecureStringToBSTR(senha);

        txtStringSegura.Text = bstr.ToString();



        //Recuperando a senha

        try
        {
              txtStringRecuperada.Text = Marshal.PtrToStringBSTR(bstr);
        }
        finally
        {
              Marshal.ZeroFreeBSTR(bstr);
        }

}
```

Exemplo de aplicação da classe SecureString na manipulação de strings

Baseado no artigo Manipulado String com segurança de Bruno Silveira Cruz disponível em http://www.devmedia.com.br/articles/viewcomp.asp?comp=3649&hl=*Seguran%E7a*

2.2.SYSTEM.SECURITY.CRYPTOGRAPH NAMESPACE

Esse namespace fornece serviços de encriptação e desencriptação de dados bem como serviços de geração de números aleatórios e hashing.

É útil quando se quer mascarar algum tipo de informação colocando-o numa forma ilegível ao ser humano, sendo necessário um algoritmo para interpretação e, em alguns casos, uma chave de decodificação.

Exemplos de aplicação são: codificação de senhas em banco de dados, encriptação de query strings para recuperação de informações de forma segura, codificação de informações de configuração de sistemas que são transmitidas na forma de “texto-puro” e contém informações importantes referentes ao banco de dados de uma empresa por exemplo. Abaixo, um exemplo de uso desse namespace.

```
public class Encryption64
{
      private static byte[] key = { };
      private static byte[] IV = { 0x12, 0x34, 0x56, 0x78, 0x90,
      0xAB, 0xCD, 0xEF };
      public static string Encrypt(string stringToEncrypt, string
      sEncryptionKey)
      {
            byte[] inputByteArray;
            try
            {
                  key =
                  Encoding.UTF8.GetBytes(sEncryptionKey.Substring(0,
                  8));
                  DESCryptoServiceProvider des = new
                  DESCryptoServiceProvider();
                  inputByteArray =
                  Encoding.UTF8.GetBytes(stringToEncrypt);
                  MemoryStream ms = new MemoryStream();
                  CryptoStream cs = new CryptoStream(ms,
                  des.CreateEncryptor(key, IV),
                  CryptoStreamMode.Write);
                  cs.Write(inputByteArray, 0,
                  inputByteArray.Length);
                  cs.FlushFinalBlock();

                  return Convert.ToBase64String(ms.ToArray());
            }
            catch
            {
                throw new InvalidDataException("Esse valor de chave
não corresponde a nenhum registro válido.");
            }
      }
```

Métodos para encriptação de strings

```
      public static string Decrypt(string stringToDecrypt, string
sEncryptionKey)
        {
            byte[] inputByteArray = new
            byte[stringToDecrypt.Length];


            try
            {
                  key =
                  Encoding.UTF8.GetBytes(sEncryptionKey.Substring(0,
                  8));
                  DESCryptoServiceProvider des = new
                  DESCryptoServiceProvider();
                  inputByteArray =
                  Convert.FromBase64String(stringToDecrypt.Replace("
                  ","+"));
                  MemoryStream ms = new MemoryStream();
                  CryptoStream cs = new
                  CryptoStream(ms,des.CreateDecryptor(key,IV),
                  CryptoStreamMode.Write);
                  cs.Write(inputByteArray,0,inputByteArray.Length);
                  cs.FlushFinalBlock();



                  Encoding encoding = Encoding.UTF8;
                  return encoding.GetString(ms.ToArray());
            }
            catch
            {
                  throw new InvalidDataException("Esse valor de
                  chave não corresponde a nenhum registro válido.");
            }
      }
}
```

Métodos para desencriptação de strings

Código adaptado da revista .Net Magazine. Editora DevMedia, Ano 06, 60ª edição, p. 13. Adaptado do artigo do Tiberius OsBurn disponível em: http://www.devcity.net/articles/47/1/encrypt_querystring.aspx.

3. CONCLUSÃO

Do estudo desses dois conjuntos de rotinas pode-se concluir que, segurança deixou de ser um item opcional, tornando-se uma necessidade dentro de qualquer sistema, especialmente os sistemas web. O aprofundamento desse estudo é muito importante e essencial para qualquer associação ou pessoa que preze pela segurança de seus dados e informações. Em qualquer framework ou linguagem, que almeja uma maior confiabilidade, esse item deve ser incluído entre os requisitos essenciais do mesmo.

REFERÊNCIAS

Class SecureString. Disponível em < http://msdn.microsoft.com/pt-br/library/system.security.securestring.aspx> . Acessado em 11/05/09

Conceitos básicos de criptografia e o .Net Framework**. Revista Fórum Access**, nº 62, Jul/Ago. Disponível em < Conceitos criptografia e .Net Framework 05/05/09 20:09 http://velloso.com/materias/Crypto/cryptointro.htm>. Acessado em 05/05/09.

CRUZ, B.S. **Manipulando String com Segurança**. Disponível em < http://www.devmedia.com.br/articles/viewcomp.asp?comp=3649&hl=*Seguran%E7a* >. Acessado em 05/05/09

Secure String Members. Disponível em < http://msdn.microsoft.com/en-us/library/system.security.securestring_members.aspx>. Acessado em 11/05/09.

SENDIN, R.Criptografando a QueryString. **.NET Magazine**, Grajaú, RJ, ano 6, n. 60. p. 12-13, 2009.

System.Security Namespace. Disponível em < http://msdn.microsoft.com/en-us/library/system.security.aspx>. Acessado em 05/05/09.

System.Security.Cryptograph Namespace. Disponível em < http://msdn.microsoft.com/en-us/library/system.security.aspx>. Acessado em 05/05/09.

VOLODARSKY, M. **Projeto e Distribuição Seguras de Aplicações Web com ASP.NET 2.0 e IIS 6.0**. Disponível em < http://www.devmedia.com.br/articles/viewcomp.asp?comp=845&hl=*Seguran%E7a>. Acessado em 05/05/2009.