.Net Framework 下数据加解密方法的实现

吴永丰,钟 诚
(广西大学 计算机与电子信息学院,广西 南宁 530004)

摘 要:分析了.Net Framework环境提供的安全机制。基于C#.Net编程语言,通过建立DESCryptoServiceProvider对象、生成EncryptString()和DecryptString()函数的方法,给出实现对称密码系统DES数据加解密的程序示例;此外,通过建立RSACryptoServiceProvider对象、生成EncryptData()和DecryptData()函数的方法,给出实现非对称密码系统RSA数据加解密的程序示例。所给出的方法对于开发应用密码系统具有参考价值。

关键词:.Net Framework;密码算法;C#程序设计
中图分类号:TP309.7 **文献标识码**:A **文章编号**:1673-629X(2006)10-0137-02

Implementing Data Encryption and Decryption Methods for .Net Framework

WU Yong-feng, ZHONG Cheng
(School of Computer and Electronics and Information, Guangxi University, Nanning 530004, China)

Abstract: The security mechanism for .Net framework is analysed. Based on C#.Net programming language, it presents an example for implementing DES encryption and decryption by creating DESCryptoServiceProvider object, EncryptString() and DecryptString() functions; and proposes another example for implementing RSA encryption and decryption by creating RSACryptoServiceProvider object and EncryptData and DecryptDatafunctions. The proposed method is helpful to develop the applied cryptography system.

Key words:.Net framework; cryptography algorithms; C# programming

1 .Net Framework的安全机制

数据加解密是信息安全领域一项十分重要的工作[1]。应用Win32 API进行安全代码编程的最大问题是它难以理解和使用。Microsoft的.Net Framework安全机制提供了丰富的类库,以帮助软件开发人员构建安全的应用程序[2~4]。

.Net Framework类库提供了System.Security.Cryptography命名空间,此命名空间支持最重要的对称、非对称密码和安全散列以及随机数生成器算法(包括DES,Triple DES,Rijndael,RC2,RSA,DSA,MD5,SHA和伪随机数生成器PRNG等),其中RC2算法是RSA数据安全公司持有专利的采用可变长度密钥的对称密码算法,DSA为数字签名算法,Rijndael算法是高级加密标准(AES,Advanced Encryption Standard)算法,它使用128,192或256比特的密钥(分别称为AES-128,AES-192和AES-256)。DES,Triple DES,Rijndael和RC2算法由.Net Framework的SymmetricAlgorithm类封装,RSA和DSA算法由.Net Framework的AsymmetricAlgorithm类封装。System.Securyty.Cryptography.XML命名空间实现用于对XML进行数字签名的W3标准。而System.Securyty.Cryptography.X509Certificates类库实现X.509数字证书标准。

2 .Net Framework对称密码系统数据加解密方法的实现

2.1 .Net Framework对称密码算法类的层次结构

.Net Framework提供的各种对称密码算法类由抽象基类System.Security.Cryptography.SymmetricAlgorithm派生,其层次结构如图1所示。

SymmetricAlgorithm派生的DES,TripleDES,Rijndael和RC2对称密码算法类分别由.Net Framework提供的4个加密服务提供程序CSP类DES Crypto Service Provider,Triple DES Crypto Service Provider,RC2 Crypto Service Provider和Rijndael Managed来实现,其中以Crypto Service Provider为后缀的类是使用内在的Win32 Crypto API实现的包装类,而以Managed为后缀的类则是在托管代码中实现的。

收稿日期:2006-02-08
基金项目:广西科学基金(桂科自0339008);广西大学博士科研基金(B0309031)
作者简介:吴永丰(1975-),男,广西防城港人,硕士研究生,工程师,研究方向为网络信息安全;钟 诚,博士,教授,研究方向为网络信息安全、并行分布计算等。

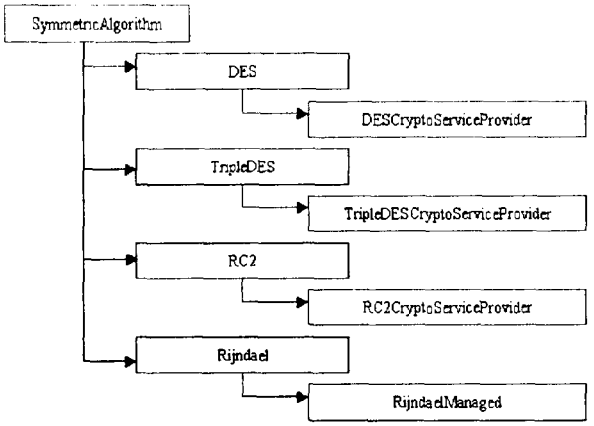

图 1 .Net Framework 的对称密码算法类层次结构

2.2 .Net Framework 对称密码系统数据加解密编程

下面给出的是以对称密码系统 DES 为例，在. Net Framework 下应用 C# 程序语言[5]编程实现数据加解密的方法：

(1)生成密钥及初始化向量 IV。

```
private const string sSecretKey = "Password";//使用口令初始化密钥
DESCryptoServiceProvider des = new DESCryptoService Provider();
//如果未提供密钥,服务提供程序就会随机生成一个密钥,该密钥可对文件进行编码;与密钥相似,如果未提供初始化向量 IV 的值,服务提供程序就会随机生成一个初始化向量。必须以字节数组的形式给加密服务提供程序提供密钥
des. Key = Encoding. Unicode. GetBytes ( sSecretKey);
des. IV = Encoding. Unicode. GetBytes ( sSecretKey);
```

(2)加密数据。

```
private string EncryptString(string text)
{
ICryptoTransform ct;//有了此接口才能在服务提供程序上调用 CreateEncryptor 方法,服务提供程序将返回定义该接口的实际 encryptor 对象
MemoryStream ms;
CryptoStream cs;
byte[ ]byt;
ct = des. CreateEncryptor(des. Key,des. IV);
byt = Encoding. Unicode. GetBytes(text);
//将正文字符串转换成字节数组(. NET 加密算法处理的是字节数组而非字符串)
ms = new MemoryStream();//创建 MemoryStream 对象 ms
cs = new CryptoStream(ms,ct,CryptoStreamMode. Write);
//创建 CryptoStream 对象 cs 并使用 Write 方法将加密数据块写入 MemoryStream 对象
cs. Write(byt,0,byt. Length);
cs. FlushFinalBlock();
cs. Close();
return Convert. ToBase64String(ms. ToArray());
}
```

(3)解密数据。

```
private string DecryptString(string text)
{
ICryptoTransform ct;
MemoryStream ms;
CryptoStream cs;
byte[ ]byt;
ct = des. CreateDecryptor(des. Key,des. IV);
//使用 CreateDecryptor 方法创建 ICtryptoTransform 对象 ct
byt = Convert. FromBase64String(text);//将字符串转换成字节数组
ms = new MemoryStream();
cs = new CryptoStream(ms, ct, CryptoStreamMode. Write);
cs. Write(byt, 0, byt. Length);
cs. FlushFinalBlock();
cs. Close();
return Encoding. Unicode. GetString(ms. ToArray());
// 将内存数据流的字节数组转换成可显示的普通字符串
}
```

3 .Net Framework 公钥密码系统数据加解密方法的实现

3.1 .Net Framework 公钥密码算法类的层次结构

. Net Framework 提供的公钥密码算法类由抽象基类 System. Security. Cryptography. Asymmetric Algorithm 派生，其层次结构如图 2 所示。

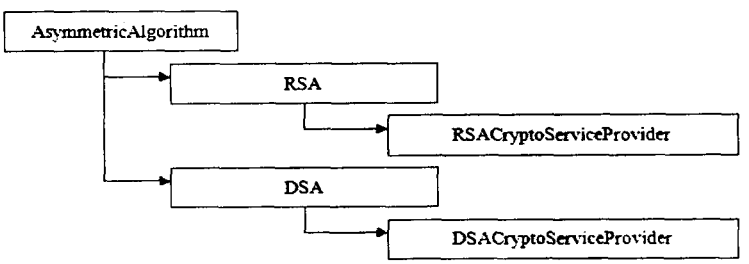

图 2 .Net Framework 的公钥密码算法类层次结构

3.2 .Net Framework 公钥密码系统数据加解密编程

下面是以公钥密码系统 RSA 为例，给出的在. Net Framework 下应用 C# 编程实现数据加解密的方法：

(1) 初始化 RSA 参数。

```
RSACryptoServiceProvider rsa = newRSACrypto ServiceProvider();//建立 RSA 加密服务对象
RSAParameters rsaParamsExcludePrivate = rsa. ExportParameters(false);//只保存 RSA 公钥
RSAParameters rsaParamsIncludePrivate = rsa. ExportParameters(true);//保存 RSA 公钥和私钥,若 ExportParameters()方法传递 true,则输出 RSA 公钥和私钥,用于解密;若传递 false,则只输出 RSA 公钥,用于加密
```

(2)加密数据。

```
private byte[ ]EncryptData(string text)
```

(下转第 142 页)

(3) 在申请资源的过程中,使用代理证书完成大部分操作。如果代理证书及关联的私钥被截获,攻击者便可假冒用户的身份进行破坏行为。目前,大多数服务器尚不为颁发的代理证书提供证书撤销的功能,而是将代理证书的有效期设置得很短,一般是几个小时,这样,即使证书被窃取,也很快就会过期,攻击者很难在剩余的时间里进行破坏,减少了损失。

此外,尚有以下问题和安全隐患未能得到很好解决。

a. 如上所述,GSI 中未提供代理证书的撤销功能,而只是将代理证书的有效期设置得非常短,一旦代理证书泄漏,只能通过重新签发证书来继续正常操作。

b. CAS 中,如果团体颁发给用户的证书被攻击,攻击者可以假冒用户身份退出团体,这样,用户将不能再从此团体中得到任何授权证书。

c. 引入 OCR 后,用户只需采用在线方式,通过 Web 浏览,输入用户名和口令即可得到代理证书,将原先采用公钥设施数字证书进行认证的安全体系又转变成简单的使用用户名口令这种原始的认证方式,降低了系统的安全性。

由上面的分析可以看出,虽然 GSI 是目前最为完善的网格安全体系,在授权与认证方面仍存在着一些问题,有待进行更深入的研究和改进。

5 结束语

安全问题一直以来都是网络通信中的重要问题,对于网格环境也不例外。授权与认证是网格安全中的核心技术,而网格环境的特殊性又为其提出了新的挑战,尤其是网格技术尚未成熟,仍在不断的探索之中。文中主要讨论分析了网格安全设施(GSI)中安全委托、单点登录、在线证书仓库和团体授权服务等技术,并通过图示说明认证授权在网格中的具体流程,讨论了其中仍存在的问题,为以后的研究提供可以参考的内容。

参考文献:

[1] Foster I, Kesselman C, Tuecke S. The Anatomy of the Grid - - Enabling Scalable Virtual Organizations[J]. International J. Supercomputer Applications, 2001, 15(3):1 - 2.

[2] Globus. Overview of the Grid Security Infrastructure[DB/OL]. http://www.globus.org/security/overview.html, 2002.

[3] CERN, CNRS, INFN, NIKHEF, PPARC. DataGrid Security Requirement and Testbed Security Implementation[EB/OL]. WP07: Network Service, 2002, DataGrid - 07 - D7.5 - 0111 - 4 - 0, PUBLIC, http://grid - auth.infn.it/docs/WP7 - security - requirements.pdf.2002.

[4] Novotny J, Tuecke S, Welch V. Online Credential Repository for the Grid: MyProxy[C/OL]. Proceedings of the Tenth International Symposium on High Performance Distributed Computing (HPDC - 10), IEEE Press, 2001, http://citeseer.ist.psu.edu/novotny01online.html.2001.

[5] Pearlman L, Kesselman C, Welch V, et al. The Community Authorization Service: Status and Future[A]. CHEP03[C]. La Jolla, California:[s.n.],2003.24 - 28.

(上接第 138 页)

```
}
RSACryptoServiceProvider rsa = new RSACryptoService Provider();
rsa.ImportParameters(rsaParamsExcludePrivate);//输入 RSA 公钥
byte[]bytes = Encoding.Unicode.GetBytes(text);//将正文字符串转换成字节数组
byte[]encryptedData = rsa.Encrypt(bytes,false);//调用 RSA 的 Encrypt()方法加密数据
return encryptedData;
}
```

(3)解密数据。

```
private byte[]DecryptData(byte[]data)//data 是密文
{
RSACryptoServiceProvider rsa = new RSACryptoService Provider();
rsa.ImportParameters(rsaParamsIncludePrivate);//输入包含有私钥的 RSA 参数
byte[] decryptedData = rsa.Decrypt(data,false);//调用 RSA 的 Decrypt()方法解密数据
return decryptedData;
}
```

4 结 语

如果使用非托管的 Win32API 开发数据加解密应用,那么即使是比较简单的操作也需要编写和执行大量的程序代码,不但处理过程复杂而且效率低下。.Net Framework 封装了用以实现数据加解密、数字签名等安全操作的一组类,大大简化编码工作,降低程序设计复杂度,进而提高信息安全软件系统的生产率。已经应用上述介绍的 .Net Framework 下数据加解密方法,开发了一个基于 Web 的网络通信系统的安全子系统。

参考文献:

[1] 钟 诚,赵跃华.信息安全概论[M].武汉:武汉理工大学出版社,2003.

[2] Thorsteinson P, Gnana Arun Ganesh G..NET 安全性与密码术[M].梁志敏,蔡 建译.北京:清华大学出版社,2004.

[3] O'Neill M. Web 服务安全技术与原理[M].冉 晓,郭文伟译.北京:清华大学出版社,2003.

[4] Dournaee B. XML 安全基础[M].周永彬,贺也平,刘 娟译.北京:清华大学出版社,2003.

[5] 吕文达.精通 C# 程序设计[M].北京:清华大学出版社,2004.