原文地址：https://www.cnblogs.com/Laggage/p/11028614.html

RSA在非对称加密中较为流行，RSA比对称加密慢很多很多，一般使用公钥加密，私钥解密

public class RSAHelper

{

    /// <summary>

    /// 使用公钥加密

    /// </summary>

    /// <param name="publicKey">公钥</param>

    /// <param name="content">待加密文本</param>

    /// <returns></returns>

    public static string RSAEncrypt (string publicKey, string content)

    {

        string encryptedContent = string.Empty;

        using (RSACryptoServiceProvider rsa = new RSACryptoServiceProvider ()) {

            rsa.ImportCspBlob (Convert.FromBase64String (publicKey));

            byte[] encryptedData = rsa.Encrypt (Encoding.Default.GetBytes (content), false);

            encryptedContent = Convert.ToBase64String (encryptedData);

        }

        return encryptedContent;

    }

    /// <summary>

    /// 使用私钥加密

    /// </summary>

    /// <param name="privateKey">私钥</param>

    /// <param name="content">待解密文本</param>

    /// <returns></returns>

    public static string RSADecrypt (string privateKey, string content)

    {

        string decryptedContent = string.Empty;

        using (RSACryptoServiceProvider rsa = new RSACryptoServiceProvider ()) {

            rsa.ImportCspBlob (Convert.FromBase64String (privateKey));

            byte[] decryptedData = rsa.Decrypt (Convert.FromBase64String (content), false);

            decryptedContent = Encoding.Default.GetString (decryptedData);

        }

        return decryptedContent;

}

    /// <summary>

    /// 生成密钥

    /// </summary>

    /// <param name="keySize">384位到16384位，以8为倍数</param>

    /// <returns>私钥Key，公钥Value</returns>

    public static KeyValuePair<string, string> GenerateRSASecretKey (int keySize)

    {

        KeyValuePair<string, string> rsaKey;

        using (RSACryptoServiceProvider rsa = new RSACryptoServiceProvider (keySize)) {

            rsaKey = new KeyValuePair<string, string> (

                // 导出Blob，此结构包含了密钥信息

                Convert.ToBase64String (rsa.ExportCspBlob (true)),

                // 导出Blob，此结构包含了公钥信息

                Convert.ToBase64String (rsa.ExportCspBlob (false)));

        }

        return rsaKey;

    }

}

// 示例

static void Main(string[] args)

{

    var key = RSAHelper.GenerateRSASecretKey(384);

    var encryptText = RSAHelper.RSAEncrypt(key.Value, "hello world");

var decryptText = RSAHelper.RSADecrypt(key.Key, encryptText);

    Console.WriteLine("Hello World!");

}