<http://www.cnblogs.com/JamesLi2015/archive/2013/05/03/3055998.html>

许可机制是ERP框架中必不可少的一部分，可以有效的保护框架资源在授权范围内应用，增加企业投资的回报。在研究了几种类型的许可机制(序列号注册码，Web服务联机验证，授权License文件)后，最后选定以Signed Xml配合RSA算法，作为许可机制的主要技术实现。

主要达到的目的如下

1  可以实现版本控制。企业版可使用所有的功能，专业版只可用部分功能，个人版免费使用，但功能集更少。

public enum Version {  Enterprise, Professional, Personal }

2  功能点的控制上，同时在线用户数量控制，帐套数量控制，硬件验证控制，试用过期控制，虚拟机控制。

* 在线用户数量 可以控制同时在线的用户数量，超过许可数量，则无法登陆
* 帐套数量控制 比如，只可以建立10套帐，超过此限制则无法登陆
* 硬件验证控制 生成许可文件时，会绑定硬件信息(硬盘，CPU，内存，主板），以此硬件信息生成的许可文件，不可以在别的电脑上运行，以控制用户数量。
* 试用过期控制 超过期限则停止进入系统，有效阻止未授权用户的继续使用，收回投资
* 虚拟机控制  因为虚拟机中安装与还原操作系统非常容易方便，我们常以此来试用软件，当软件试用到期后还想继续用，则只需要的还原一下虚拟机中的系统，则可以继续体验。以此选项，控制软件不可以运行于虚拟机中。常见的虚拟机即VMware Workstation和Virtual PC。

以此理论，设计如下格式的License.lic文件，以作为要颁发的许可证文件。

[](https://images0.cnblogs.com/blog/17042/201305/03091715-f7db8e59b9314050874b47b2bcc24691.png)

在程序编写过程中，参考了CodeProject网站中的文章

Using XML Digital Signatures for Application Licensing - CodeProject   
<http://www.codeproject.com/Articles/4940/Using-XML-Digital-Signatures-for-Application-Licen>

几乎就是对这篇文章的定制，就可以完成以上所需要达到的目的。以下分享几个遇到的实际问题，供您参考。

1 Xml序列时，元素的顺序。通过Google得知，请仔细阅读以下的几段话。

XmlSerializer takes all fields in the order that they are declared.   
the order problem on the Compact Framework.Unfortunately, this is by Design.

The order of elements serialized by the NETCF xml serializer is not   
guaranteed to match that of the desktop. There is nothing in the generated   
schema class included in the attached project that specifies the order of   
the elements.

In order to accomplish this you should add the /order option to xsd.exe and   
then regenerate the schema class   
(xsd.exe /order /c foo.xsd)

By doing this all the particle members will have their explicit order   
identifiers and then the serializer will honor the order of the schema. The   
new schema generated by with the /order switch will have the orders   
property specified on the XmlElementAttribute   
e.g. [System.Xml.Serialization.XmlElementAttribute(Order = 2)]

Xml序列化以元素声明的顺序，但是Compack Framework不一样，需要手动指定它们的顺序。

2 硬件信息的集成绑定

这应该是一个名值对，比如

CPU: Intel Pentium T440

Hard Driver: WD Elements

所以，需要设计一个List<HardwareInfo>,或是继承于CollectionBase。

public struct HardwareInfo

{

public string HardwareId { get;set;}

public string Description { get;set;}

}

这个List<T>要可以序列化，它要绑定到License.lic文件中。

3 类型中，有些对象不需要序列化的，要加上标签以阻止序列化。

[NonSerialized]

private string \_hashValue;

4  生成公钥和私匙配对，然后放到代码中去。生成的内容如下

public static string PrivateKey

{

get

{

return "<BitStrength>1024</BitStrength><RSAKeyValue><Modulus>uCMDxXTd0bNbiAFrOYjbiGyQpqfZY2Znn70hoQZsprNoXV8tSZ6mM8VswoTNh6S+0qfYntzxpQq29mqv+8mUIuGN/30YpUq9tZFR1bIHEJnPqSRHcQa0ezimTilBN7EN7J6wnQBQqFyt3ZRnLYUsRta1Vjdn4eEc50Q4EfEOlO8=</Modulus><Exponent>AQAB</Exponent><P>99QWQo0ulkBCDyHwL3amXKahDSmcGa3bJHz23M++65jtxYp0LViGH+ngr5FYSxp7oAj37dKTiw4h6NO/+J6amw==</P><Q>vjVO29oMfKynSHZgRIeRhcInt6ReHm19of8YIsvBVYgasg9qi0lONFUvmW51fPrXdTPWz4fHmlnv3leWN7AaPQ==</Q><DP>tiyKHGvJthsQNC1/cHRogCzgsFtI6zt4no7ZrKFtt6PYDODk27x6A5WZW5Wc8MBL5e0RyxmC6bH+zTZypGB6Rw==</DP><DQ>Rr/bYkl75Y/u9TQa4MKwbVlnnpZD7/t4BJ63IpI5ipACpgK39bFBppOdDewZRXCkXdL3buApbY9QepqHpJUbXQ==</DQ><InverseQ>zZu/5jmI8PSbo1e6nXfaAtzZQiSUO0q3D1Dm30w51lukRw8OlkmrMOszLF7LontM/4kLhmri2BU5yeTChppXLQ==</InverseQ><D>q5JsrCqlmQRfEA4KY9Siga5u5epWA2liupOW5xw+VuGqJ/5MC2HZCTo2idUGURJvf4dHr1a9jgO60UY9bgW4kWOkLdZ3xzn0wqYyt/VCdvQE1qH/YnVEeLUZqjrbH14Zw8isR2Yxf33QCFfvHWTqIvwtm0ZdniH+cEIRgEwsPNk=</D></RSAKeyValue>";

}

}

public static string PublicKey

{

get

{

return "<BitStrength>1024</BitStrength><RSAKeyValue><Modulus>uCMDxXTd0bNbiAFrOYjbiGyQpqfZY2Znn70hoQZsprNoXV8tSZ6mM8VswoTNh6S+0qfYntzxpQq29mqv+8mUIuGN/30YpUq9tZFR1bIHEJnPqSRHcQa0ezimTilBN7EN7J6wnQBQqFyt3ZRnLYUsRta1Vjdn4eEc50Q4EfEOlO8=</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>";

}

}

公钥只可以拿来验证许可文件，私匙可以验证，但主要的是用来生成Xml文件签名。

5 请注意License文件的最后一个节，它放的是绑定的硬件信息，这一节信息是加密的。为了可以解密，这里用RSA算法。这个过程如下，用户试用软件，申请许可文件，给用户一个EXE文件程序，用于生成hardware.id文件，然后用户将此文件发给软件公司，软件公司依据此文件，向用户发放试用许可文件。

string hardware = RSACryptionHelper.EncryptString(inputString, publicKey);

当用户将绑定有硬件信息的许可拿到别的电脑上运行时，会报异常

if ((configuration.Items.Count == 0) && (this.LicenseType == LicenseType.Enterprise))

{

throw new LicenseException("Hardware fingerprint is missing in license file");

}

6  许可类型

public enum LicenseType

{

Internal,

Enterprise,

Professional,

Personal

}

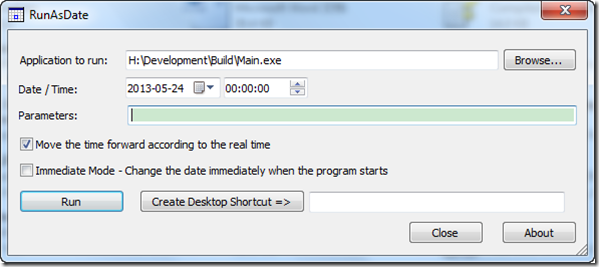
所有类型的license都会过期，在license中指定的ExpiredDate之后，都将无法运行。所以没有加Trial类型的许可。

我手头有个数据库工具软件，还有800多天的试用期，可还可使用2年多一点。IT这个行业既很传统，10年的技术，C#.NET WinForms技术，现在还在普及使用中，又很超前，大量的新技术，新知识注入到这个行业中。

2年的时间内可以做很多事，看很多书，走很多路，且行且看。

7  Signed Xml技术的要点主要是明文查看，但可以防止篡改。眼睁睁的看着2013-5-30号就过期了，你就是没有办法把它改成2100-5-30号。这一下子可以100年后，你改了，这企业再也不能从你这里收取费用，可怎么养活程序员呢。

但是，有两个软件可以做到一个，就是篡改系统的当前时间，然后注入到你的进程中。推荐软件RunAsDate。这软件的功能强大，可以设置当前时间，然后启动软件。如果你试用一个很好的软件，又不想付费，也找不到Cracker或是KEY的话，可以试下这个办法。

[](https://images0.cnblogs.com/blog/17042/201305/03091732-57f7975de4e04f26a32718ac864e3cc4.png)

8   ExpiryDate是2013-5-30时，但是到了这一天，用户把时间又改回到2013-4-30号，又可以继续试用一个月。对于这个问题，也要处理。对于MIS/ERP类型的软件，也可以不用处理。对数据的主要要求之一是准确，现在是5-2号了，你把时间改回到4-2号，继续使用软件，系统的日记帐时间也是4-2号。以后查帐的时候，这是很严重的问题。5-2号的进仓单，实际对应的是系统中的4-2号的单据，很不方便与追踪问题。

如果要控制用户改时间，则需要记住用户第一次使用系统的时间，发现用户系统时间，直接报错，异常退出。

<IssuedDate>2013-04-30T10:58:52.5456701+08:00</IssuedDate>

<ExpiryDate>2013-05-30T23:59:59.997</ExpiryDate>

请到如下的网页中下载源代码，供您参考。

Using XML Digital Signatures for Application Licensing - CodeProject   
http://www.codeproject.com/Articles/4940/Using-XML-Digital-Signatures-for-Application-Licen