文章
姚 鑫 · 六月 25, 2021 阅读大约需 10 分钟

第十八章 签署XML文档

第十八章 签署XML文档

本章介绍如何向XML文档添加数字签名。

关于数字签名文档

数字签名的XML文档包括一个或多个<Signature>元素，每个元素都是数字签名。
每个<Signature>元素对文档中的特定元素进行如下签名:

- 每个签名元素都有一个ID属性，该属性等于某个唯一值。例如：

```
<Person xmlns="http://mynamespace" Id="123456789">
```

- 一个<Signature>元素包含一个<Reference>元素，它指向该Id，如下所示:

```
<Reference URI="#123456789">
```

<Signature>元素是由私钥签名的。此元素包括由签名机构签署的X.509证书。如果已签名文档的接收方信任此签名机构，则接收方可以验证证书，并使用包含的公钥验证签名。

注意: IRIS还支持一种变体，其中有签名的元素有一个名为ID的属性，而不是ID。

下面是一个示例，为了便于阅读，添加了空格:

```
<?xml version="1.0" encoding="UTF-8"?>
<Person xmlns="http://mynamespace" Id="123456789">
  <Name>Persephone MacMillan</Name>
  <DOB>1976-02-20</DOB>
  <s01:Signature xmlns="http://www.w3.org/2000/09/xmldsig#"
                 xmlns:s01="http://mynamespace"
                 s02:Id="Id-BC0B1674-758D-40B9-84BF-F7BAA3AA19F4"
xmlns:s02="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">
    <SignedInfo>
      <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#">
      </CanonicalizationMethod>
      <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1">
      </SignatureMethod>
      <Reference URI="#123456789">
        <Transforms>
          <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature">
          </Transform>
```

```
          <Transform Algorithm="http://www.w3.org/TR/2001/REC-xml1317c14n-20010315">
          </Transform>
        </Transforms>
        <DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMeth
od>
        <DigestValue>FHwW2U58bztLI4cIE/mp+nsBNZg=</DigestValue>
      </Reference>
    </SignedInfo>
    <SignatureValue>MTha3zLoj8Tg content omitted</SignatureValue>
    <KeyInfo>
      <X509Data>
        <X509Certificate>MIICnDCCAYQCAWUwDQYJ content omitted</X509Certificate>
      </X509Data>
    </KeyInfo>
  </s01:Signature>
</Person>
```

要创建数字签名，可以使用类%XML.Security.Signature。
这是一个支持xml的类，它的投影是适当名称空间中的有效<Signature>元素。

创建数字签名XML文档

要创建数字签名的XML文档，请使用%XML.Writer为一个或多个适当定义的启用了XML的对象生成输出。

在为对象生成输出之前，必须创建所需的签名并将其写入对象，以便可以将信息写入目标。

签名的前提条件

在签署文档之前，必须至少创建一个IRIS凭据集。InterSystems
IRIS凭据集是存储在系统管理器数据库中的以下信息集的别名：

- 包含公钥的证书。证书应由文档接收者信任的签名机构签名。
- 关联的私钥， IRIS在需要时使用，但从不发送。签名需要私钥。
- (可选)私钥的密码， IRIS在需要时使用私钥，但从不发送。可以加载私钥，也可以在运行时提供私钥。

启用XML的类的要求

启用XML的类必须包括以下内容：
- 投影为ID属性的特性。
- 至少一个类型为%XML.Security的属性。投影为<Signature>元素的签名。(一个XML文档可以包含多个<Signature>元素。)

考虑以下类:

```
Class XMLEncryption.Simple Extends (%RegisteredObject, %XML.Adaptor)
{

Parameter NAMESPACE = "http://mynamespace";

Parameter XMLNAME = "Person";

Property Name As %String;
```

```
Property DOB As %String;

Property PersonId As %String(XMLNAME = "Id", XMLPROJECTION = "ATTRIBUTE");

Property MySig As %XML.Security.Signature(XMLNAME = "Signature");

}
```

生成和添加签名

要生成和添加数字签名，请执行以下步骤:

1. 可以选择包含%soap.inc包含文件，该文件定义可能需要使用的宏。
2. 创建%SYS.X509Credentials的实例在访问相应InterSystems IRIS凭据集。为此，调用%SYS.X509Credentials的GetByAlias()类方法。

```
classmethod GetByAlias(alias As %String, pwd As %String) as %SYS.X509Credentials
```

- alias 别名是证书的别名。
- pwd 是私钥密码。仅当关联的私钥已加密并且在加载私钥文件时未加载密码时，才需要私钥密码。

若要运行此方法，必须以该凭据集的OwnerList中包含的用户身份登录，否则OwnerList必须为空。

3. 在使用给定凭据集创建 %XML.Security.Signature的实例。为此，请调用该类的Createx509（）类方法：

```
classmethod CreateX509(credentials As %SYS.X509Credentials, signatureOption As %Integ
er, referenceOption As %Integer) as %XML.Security.Signature
```

- credentials 凭据是刚刚创建%SYS.X509Credentials的实例。
- signatureOption是$$$SOAPWSIncludeNone（还有其他选项，但它们不适用于此方案）
- referenceOption 指定对符号元素的引用的性质。

这里使用的宏在%soap.inc中定义包括文件。

4. 获取ID属性的值，对于此签名将点的ID。此详细信息取决于启用XML对象的定义。
5. 创建%XML.Security.Reference的实例，指向该ID。为此，请调用该类的Create（）类方法：

```
ClassMethod Create(id As %String, algorithm As %String, prefixList As %String)
```

- id是该参考应该指向的ID。
- algorithm 算法应该是以下之一：
  - $$$SOAPWSEnvelopedSignature","$$$SOAPWSexcc14n — 使用此版本获取独占规范化。
  - $$$SOAPWSEnvelopedSignature — 这相当于前面的选项。
  - $$$SOAPWSEnvelopedSignature","$$$SOAPWSexcc14n — 使用此版本进行包容性规范化。

6. 对于签名对象，调用AddReference()方法将此引用添加到签名：

```
Method AddReference(reference As %XML.Security.Reference)
```

7. 更新启用XML的类的相应属性以包含签名。

```
set object.MySig=signature
```

8. 创建%XML.Document的实例，该实例包含序列化为XML的启用了XML的对象。

这是必要的，因为签名必须包括有关签名文档的信息。

注意：本文档不包含空格。

9. 调用签名对象的SignDocument()方法：

```
Method SignDocument(document As %XML.Document) As %Status
```

此方法的参数是刚刚创建的中%XML.Document的实例。SignDocument()方法使用该实例中的信息更新签名对象。

10. 使用%XML.Writer中为对象生成输出。

注意：生成的输出必须包含与签名中使用的文档相同的空格(或不包含空格)。签名包含文档的摘要，如果将编写器中的缩进属性设置为1，则摘要将与文档不匹配。

例如：

放入到对应的实体类中，有一些属性需要替换

```
Method WriteSigned(filename As %String = "")
{
#Include %soap
    //??????
    set cred=##class(%SYS.X509Credentials).GetByAlias("servercred")
    set parts=$$$SOAPWSIncludeNone
    set ref=$$$KeyInfoX509Certificate

    set signature=##class(%XML.Security.Signature).CreateX509(cred,parts,ref,.status)
    if $$$ISERR(status) {do $system.OBJ.DisplayError(status) quit}

    // ???????????ID???
    set refid=$this.PersonId ; ???????????

    // ???????????ID???
    set algorithm=$$$SOAPWSEnvelopedSignature_","_$$$SOAPWSc14n
    set reference=##class(%XML.Security.Reference).Create(refid,algorithm)
    do signature.AddReference(reference)

    //??MySig?????$this?????????????????
    set $this.MySig=signature ; ???????????
```

```
    //??$this????????%XML.Document??????????????XML???
    set document=..GetXMLDoc($this)

    //??????XML???????????????????
    set status=signature.SignDocument(document)
    if $$$ISERR(status) {do $system.OBJ.DisplayError(status) quit}

    // ???????
    set writer=##class(%XML.Writer).%New()
    if (filename'="") {
        set status=writer.OutputToFile(filename)
        if $$$ISERR(status) {do $system.OBJ.DisplayError(status) quit}
    }
    do writer.RootObject($this)
}
```

前面的实例方法使用以下泛型类方法，该方法可以与任何启用了XML的对象一起使用：

```
ClassMethod GetXMLDoc2(object) As %XML.Document
{
    //??1-?????XML???
    set writer=##class(%XML.Writer).%New()
    set stream=##class(%GlobalCharacterStream).%New()
    set status=writer.OutputToStream(stream)
    if $$$ISERR(status) {do $System.Status.DisplayError(status) quit $$$NULLOREF}
    set status=writer.RootObject(object)
    if $$$ISERR(status) {do $System.Status.DisplayError(status) quit $$$NULLOREF}

    //??2-?????%XML.Document
    set status=##class(%XML.Document).GetDocumentFromStream(stream,.document)
    if $$$ISERR(status) {do $System.Status.DisplayError(status) quit $$$NULLOREF}
    quit document
}
```

变体：引用中带有URI=""的数字签名

作为一种变体，签名的<Reference>元素可以具有URI=""，这是对包含签名的XML文档根节点的引用。
要通过以下方式创建数字签名:

1. 可以选择包含%soap.inc包含文件，该文件定义可能需要使用的宏。
2. 创建%SYS.X509Credentials的实例在访问相应InterSystems IRIS凭据集。为此，请调用%SYS.X509Credentials的GetByAlias()类方法，如前面的步骤所述。
3. 创建使用给定凭据集的%XML.Security.Signature的实例。为此，请调用该类的CreateX509()类方法，如前面的步骤所述。
4. 按如下方式创建%XML.Security.X509Data的实例：

```
set valuetype=$$$KeyInfoX509SubjectName_","_$$$KeyInfoX509Certificate
set x509data=##class(%XML.Security.X509Data).Create(valuetype,cred)
```

其中，cred是%SYS的实例。
x509credentials在之前创建的新窗口中打开。
这些步骤创建了一个<X509Data>元素，其中包含一个<X509SubjectName>元素和一个<X509Certificate>元素。

5. 将<X509Data>元素添加到签名的<KeyInfo>元素中，方法如下:

```
do signature.KeyInfo.KeyInfoClauseList.Insert(x509data)
```

其中签名是%XML.Security的实例。
x509data是%XML.Security.X509Data的实例。

6. 创建%XML.Security的实例。
   参考如下:

```
set algorithm=$$$SOAPWSEnvelopedSignature
set reference=##class(%XML.Security.Reference).Create("",algorithm)
```

7. 在步骤6(调用AddReference())中继续上述步骤。

验证数字签名

对于收到的任何数字签名文档，都可以验证签名。不需要具有与文档内容匹配的启用XML的类。

验证签名的前提条件

若要验证数字签名，必须首先为签名者向InterSystems IRIS提供受信任的证书。如果InterSystems IRIS可以验证签名者的证书链(从签名者自己的证书到来自InterSystems IRIS信任的证书颁发机构(CA)的自签名证书)，包括中间证书(如果有)，则InterSystems IRIS可以验证签名。

验证签名

要验证数字签名的XML文档中的签名，请执行以下操作：

1. 创建%XML.Reader的实例并使用它打开文档。
2. 获取阅读器的Document属性。这是 %XML.Document的一个实例。包含作为DOM的XML文档的文档
3. 使用阅读器的correlation()方法将<Signature>元素或元素与类%XML.Security.Signature关联起来。
   例如:

```
 do reader.Correlate("Signature","%XML.Security.Signature")
```

4. 遍历文档以读取<Signature>元素或多个元素。为此，可以使用阅读器的Next()方法，该方法通过引用返回一个导入的对象(如果有的话)。
   例如:

```
if 'reader.Next(.isig,.status) {
    write !,"Unable to import signature",!
    do $system.OBJ.DisplayError(status)
    quit
    }
```

导入的对象是%XML.Security.Signature的实例

5. 调用导入签名的ValidateDocument()方法。该方法的参数必须是%XML的实例。先前检索到的文档。

```
 set status=isig.ValidateDocument(document)
```

例如:

```
ClassMethod ValidateDoc(filename As %String)
{
    set reader=##class(%XML.Reader).%New()
    set status=reader.OpenFile(filename)
    if $$$ISERR(status) {do $System.Status.DisplayError(status) quit }

    set document=reader.Document
    //?? <Signature> ??
    //????????
    do reader.Correlate("Signature","%XML.Security.Signature")
    if 'reader.Next(.isig,.status) {
        write !,"??????",!
        do $system.OBJ.DisplayError(status)
        quit
    }
    set status=isig.ValidateDocument(document)
    if $$$ISERR(status) {do $System.Status.DisplayError(status) quit }
}
```

变体:引用ID的数字签名

在典型的情况下，<Signature>元素包含一个<Reference>元素，该元素指向文档中其他地方的唯一Id。
InterSystems IRIS还支持一种变体，其中<Reference>元素指向名为ID(而不是ID)的属性。
在这种变体中，需要额外的工作来签署文档和验证文档。

要对文档进行数字签名，请遵循 “ 创建数字签名XML文档 ” 中的步骤，并进行以下更改:

- 对于支持xml的类，包含一个作为ID属性而不是ID属性投影的属性。
- 在生成和添加签名时，调用%XML的AddIDs()方法。文档实例。
  在获得序列化的XML文档之后，在调用签名对象的SignDocument()方法之前，执行此操作。
  例如:

```
//??MySig????$this???????????????
set $this.MySig=signature    ; ??????????

//??$this????????%XML??? ??????XML??????
set document=..GetXMLDoc($this)

//***** ?????ID??????? *****
do document.AddIDs()

//??????XML?????????????????
set status=signature.SignDocument(document)
if $$$ISERR(status) {do $system.OBJ.DisplayError(status) quit}
```

- 当验证文档时，在调用Correlate()之前包含以下步骤:

1. 调用 %XML.Document的AddIDs()方法。
2. 调用XML阅读器的Rewind()方法。

例如:

```
    set document=reader.Document

    //??????ID??????
    do document.AddIDs()
    do reader.Rewind()

    //?? <Signature> ??
    do reader.Correlate("Signature","%XML.Security.Signature")
```

#Caché

源URL: https://cn.community.intersystems.com/post/%E7%AC%AC%E5%8D%81%E5%85%AB%E7%AB%A0-%E7%AD%BE%E7%BD%B2xml%E6%96%87%E6%A1%A3