1

保证WebService的安全

验证/认证用户
- 使用用户名令牌**<UsernameToken>**
- 设置Web Server 应用使用password

加密消息
- 消息头
- 消息体

使用数字签名
- 确认消息是否被改动
- 确认消息中的某一部分真的由认证的实体所产生

使用HTTPS

2

SOAP调试

3

开启SOAP log

用于SOAP debug，尤其是开发使用WSS时

在SOAP服务器命名空间或SOAP客户端命名空间：
- 设置log文件路径：Set ^ISCSOAP("LogFile") = "c:\temp\my_soap.log"
- 设置log内容： Set ^ISCSOAP("Log") = "ios"
  - I：input；O： output； S：Security

4

IRIS对WS-Security的支持

5

IRIS的WS-Security支持

支持

- WS-Security头
- X.509 令牌
- XML加密
- XML签名
- 用户名令牌
- WS-Security SAML令牌

支持明细

- https://docs.intersystems.com/latest/csp/docbook/DocBook.UI.Page.cls?KEY=GSOAPSEC_intro#GSOAPSEC_standards

6

SOAP用户认证

7

SOAP用户认证

SOAP服务器端开启用户认证

- 1. 服务器端SOAP 类，增加Parameter SECURITYIN = "REQUIRE";
- 2. 修改服务器端SOAP应用的认证模式为password
- 3. 为SOAP客户端访问，创建用户账户

SOAP客户端

- 1. 向SOAP请求消息头增加用户名令牌
  - Set utoken=##class(%SOAP.Security.UsernameToken).Create("_SYSTEM","SYS")
  - Do client.SecurityOut.AddSecurityElement(utoken)

8

演示

9

X.509证书和准备工作

10

IRIS使用X.509证书

通常X.509证书用于：

- 加密SOAP消息
- 数字签名

如何使用X.509证书

- 证书通常在SOAP服务器端和客户端共享
- 私钥保存在SOAP服务器端用于加密

11

准备工作 - 准备X.509证书

从CA机构获得X.509证书

使用OpenSSL生成私有证书

- 第一步，生成私钥：*openssl genrsa -des3 -out rootCA.key 4096*
- 第二步，生成X509证书: *openssl req -new -x509 -key rootCA.key -days 3650 -out IRIS.cer*

12

准备工作 - IRIS注册X.509证书

X.509证书应该是PEM编码格式

证书应以IRIS.cer文件保存在mgr子目录下

- IRIS.cer文件里可以保存过个证书

通过System > Security Management > X.509 Credentials注册证书

13

IRIS如何获取证书

获取保存并注册的证书

- set credset=##class(%SYS.X509Credentials).GetByAlias(alias,password)
- alias是IRIS保存的X.509证书的别名
- Password是私钥的密码，可选填

获取输入SOAP消息安全头里的证书

- set credset=..SecurityIn.Signature.X509Credentials
- if $CLASSNAME(credset)'="%SYS.X509Credentials" {set credset=""}

14

SOAP消息加密

15

加密SOAP消息

加密消息头

- 通常是SOAP客户端，加密消息头的元素，例如用户名令牌

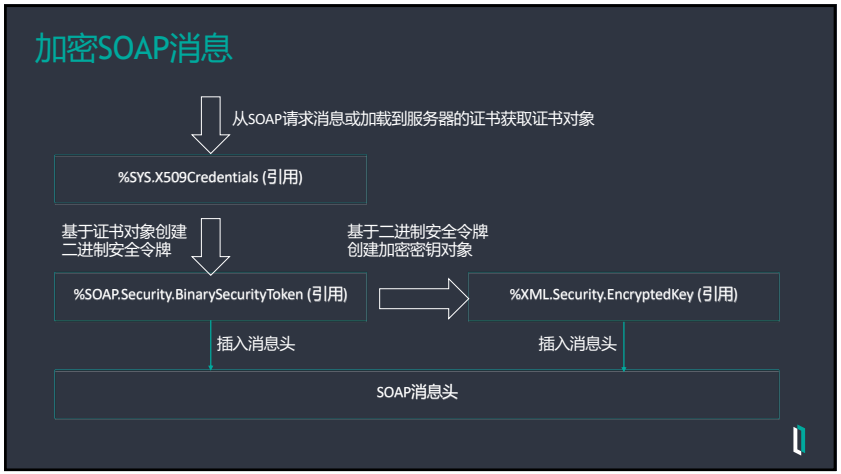
加密消息体

- 通常是SOAP服务器端和客户端，加密消息体数据

可以同时加密消息头和消息体

16

加密SOAP消息



17

加密SOAP消息 - SOAP服务器端示例代码

```
Method EncryptBody()
{
 //从注册的证书产生证书对象
 set clientcred = ##class(%SYS.X509Credentials).GetByAlias("Server","demo")
 // 创建X.509二进制安全令牌
 set bst=##class(%SOAP.Security.BinarySecurityToken).CreateX509Token(clientcred)
 do ..SecurityOut.AddSecurityElement(bst)
 // 产生对称密钥，用证书包含的公钥加密，并产生对此令牌的直接引用<EncryptedKey>
 set encropt=$$$SOAPWSEncryptDefault
 Set enc=##class(%XML.Security.EncryptedKey).CreateX509(bst,encropt)
 //将 <EncryptedKey> 加入SOAP安全头
 Do ..SecurityOut.AddSecurityElement(enc)
}
```

18

加密SOAP消息 - SOAP客户端示例代码

```
ClassMethod Test()
{ Set client=##class(HISSOAPService.HISSOAPServiceSoap).%New()
 // 创建WS-Security用户名令牌
 set user="SOAPUser",pwd="demo"
 set userToken=##class(%SOAP.Security.UsernameToken).Create(user,pwd)
 // 获取保存的X.509证书
 set cred = ##class(%SYS.X509Credentials).GetByAlias("Server")
 // 创建加密密钥
set enckey=##class(%XML.Security.EncryptedKey).CreateX509(cred, $$$SOAPWSEncryptDefault)
 // 创建EncryptedData 并使用密钥对其加密
 set encdata=##class(%XML.Security.EncryptedData).Create(,userToken)
 set dataref=##class(%XML.Security.DataReference).Create(encdata)
 do enckey.AddReference(dataref)
 //向SOAP安全头增加密钥
 do client.SecurityOut.AddSecurityElement(enckey)
 // 向SOAP安全头增加用户名令牌, 并加密
 do client.SecurityOut.AddSecurityElement(userToken,enckey)
 Quit client.Test() }
```

19

演示

20

其它设置

设置块加密算法：

- **$$$SOAPWSaes128cbc** (默认)，**$$$SOAPWSaes192cbc** and **$$$SOAPWSaes256cbc**
- set enckey.Algorithm=$$$SOAPWSaes256cbc

设置密钥传输机制：

- **$$$SOAPWSrsaoaep**，**$$$SOAPWSrsa15**
- do enckey.SetEncryptionMethod($$$SOAPWSrsa15)

设置加密范围：

- $$$SOAPWSEncryptNone（不加密），$$$SOAPWSEncryptSoapBody（消息体+消息头，默认值），$$$SOAPWSEncryptNoBody (仅消息头）
- set encropt=$$$SOAPWSEncryptNone
- set enckey=##class(%XML.Security.EncryptedKey).CreateX509(cred,encropt)

21

SOAP签名

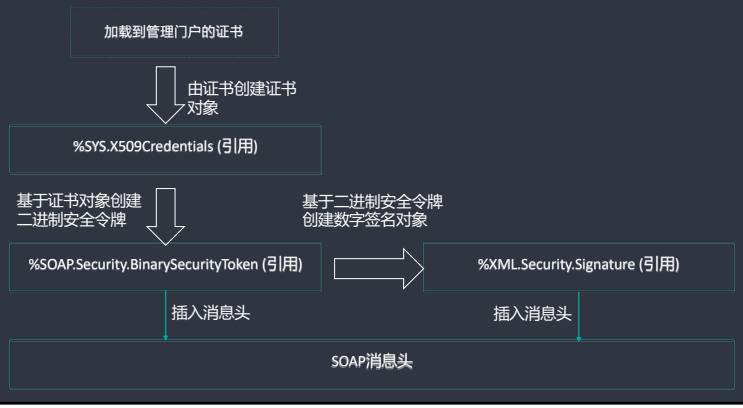
22

SOAP签名

通常使用数字签名来发现消息是否被改动，或简单验证是否消息中的某一部分真的由认证的实体所产生

23

加入数字签名



24

数字签名 - SOAP服务器端示例代码

```
Method EncryptBodyWithSignature()
{
        //从请求消息头的签名中获取X.509 证书
        Set clientsig = ..SecurityIn.Signature
        Set clientcred = clientsig.X509Credentials
        // 创建X.509二进制按全令牌
        set bst=##class(%SOAP.Security.BinarySecurityToken).CreateX509Token(clientcred)
        do ..SecurityOut.AddSecurityElement(bst)
        // 产生对称密钥，用证书包含的公钥加密，并产生对此令牌的直接引用<EncryptedKey>
        Set enc=##class(%XML.Security.EncryptedKey).CreateX509(bst)
        //将 <EncryptedKey> 加入SOAP头
        Do ..SecurityOut.AddSecurityElement(enc)
}
```

25

数字签名 - SOAP客户端示例代码

```
ClassMethod TestSignature()
{       Set client=##class(HISSOAPService.HISSOAPServiceSoap).%New()
        // 创建WS-Security用户名令牌
        // 获取保存的X.509证书
        // 创建加密密钥
        // 创建EncryptedData 并使用密钥对其加密
        // 向SOAP消息头增加密钥
        ...
        //创建WS-Security签名对象，并向SOAP消息头增加签名
        Set signature=##class(%XML.Security.Signature).CreateX509(cred)
        Do client.SecurityOut.AddSecurityElement(signature)
        // 向SOAP消息头增加用户名令牌, 并加密
        do client.SecurityOut.AddSecurityElement(userToken,enckey)
        Quit client.TestSignature()
}
```

26

其它设置 - 确定签名存放位置

- **$$$SOAPWSIncludeNone**
- **$$$SOAPWSIncludeDefault** (which equals **$$$SOAPWSIncludeSoapBody + $$$SOAPWSIncludeTimestamp + $$$SOAPWSIncludeAddressing**)
- **$$$SOAPWSIncludeSoapBody**
- **$$$SOAPWSIncludeTimestamp**
- **$$$SOAPWSIncludeAddressing**
- **$$$SOAPWSIncludeAction**
- **$$$SOAPWSIncludeFaultTo**
- **$$$SOAPWSIncludeFrom**
- **$$$SOAPWSIncludeMessageId**
- **$$$SOAPWSIncludeRelatesTo**
- **$$$SOAPWSIncludeReplyTo**
- **$$$SOAPWSIncludeTo**
- **$$$SOAPWSIncludeRMHeaders**

- set parts=$$$SOAPWSIncludeSoapBody + $$$SOAPWSIncludeTimestamp set signature=##class(%XML.Security.Signature).CreateX509(cred,parts)

27

其它设置 - 确定签名算法

- **$$$SOAPWSsha1** (默认值)
- **$$$SOAPWSsha256**
- **$$$SOAPWSsha384**
- **$$$SOAPWSsha512**

- do sig.SetDigestMethod($$$SOAPWSsha256)

28

其它SOAP 消息安全头元素

29

其它SOAP 消息安全头元素

<Timestamp>

- **用于避免重放攻击(replay** attacks)，或自定义log

30

SOAP 消息安全头的元素顺序

31

SOAP 消息安全头的元素顺序

使用不对称密钥

签名后加密	加密后签名
1. 其它头元素 2. <EncryptedKey> 3. <Signature>	*1.其它头元素* 2. <EncryptedKey> 3. <Signature> 4. <ReferenceList>

使用对称密钥

签名后加密	加密后签名
1. 其它头元素 2. <EncryptedKey> 3. <DerivedKeyToken> 4. <DerivedKeyToken> 5. <ReferenceList> 6. <Signature>	*1. 其它头元素* 2. <EncryptedKey> 3. <DerivedKeyToken> 4. <DerivedKeyToken> 5. <Signature> 6. <ReferenceList>

32


The power behind what matters.
InterSystems
Health | Business | Government
Thank you.

33