

View Point



Custom Encryption in Siebel & Siebel Web Service Security Test Guide 1.0

Muralidhar Reddy

Introduction

Siebel (7.5 onwards and upto 8.1) natively supports 2 Types of Inbound web Service Security –

- 1. WS Security UserName Token Profile Support – This requires the UserName and Password to be passed in clear text format in the SOAP Header Message.*
- 2. HTTPS Security – This requires a combination of Server and Client Certificates to perform the authentication.*

The first option is used for basic Security Authentication whereas the second one is used for fool proof Security Implementation where additional licenses are required for the Server and Client Security Certificates.

This Guide provides detail technical information on implementation of Custom Encryption of the data for any XML Tags received from External System using the external Encryption algorithm AES-128. This Encryption mode can be used as an additional level of security for Web Services along with the UserName Token Profile Support WS Security.

Executive Overview

The ubiquity of webservices in a corporate infrastructure has become an order of the day. While this has given great flexibility in managing integration of multiple applications, this also exposes an enterprise to increased data security threats. Hence it is imperative to address the threats arising to sensitive data like accounts, pricing etc. There are multiple ways of enforcing the security on the data exchanged across different systems. The most prominent solution is to implement the WS-Security.

WS-Security specification is a standard that builds on W3C's generic XML encryption and signature standards for the purpose of securing SOAP messages. The purpose of WS-Security is to enforce confidentiality, integrity or include authentication information in SOAP messages.

Secure Socket Layer (SSL) is a commonly used protocol for providing confidentiality, integrity and authentication for messages transmitted over the network.

Siebel endorses the WS-Security standards and provides its own mechanism for implementing the WS-Security using the above mentioned SSL Approach and the UserName Token mechanism. Both the approaches have certain Pros and Cons.

- **Option 1:** Implementation of SSL (using HTTPS) protocol involves importing of installation of Server and Client certificates in Siebel and external System. This mechanism ensures foolproof security by encrypting the complete XML payload along with Server Certificate details passed which can be unlocked only by relevant Client Certificate. This approach requires procurement of additional licenses for the Security certificates and will have impact on the performance
- **Option 2:** UserName Token Profile Support. This requires the UserName and Password to be passed only in clear text format in the SOAP Header Message. This has its own limitation as currently Siebel does not support a way to read the data in the SOAP Header, if passed in some encrypted manner. The advantage of using this option is it does not have any overhead unlike the Option 1, but has its own limitations.

The paper discusses another creative approach, beyond the two standard solutions mentioned above. It provides an alternate approach in which an additional level of Security built above the Siebel provided UserName Token Profile Support approach i.e., by passing certain security tag in an encrypted format from an External System to Siebel and then decrypting the tag in Siebel using any standard Encryption algorithm. (like AES 128)

The paper elucidates the above approach using Siebel Integrating with Java for providing the Custom Encryption using External Encryption algorithm AES-128.

Please note that the configuration steps detailed in this document are illustrative to present the approach.

Highlights

The document lists steps for Custom Encryption using AES-128 algorithm. The same procedure can be used with other Encryption algorithm as well. This will involve change in the underlying Java File. Since the custom Encryption is done by Integrating Siebel Product with External Java file, hence it can be used with Siebel version higher than 7.5.

Getting Started

Following are the configuration steps for implementing Custom Encryption:

1. Java File Create/Setup:

Step1. Create a Java file <StringEncryption.java> with following code.

```
import com.siebel.data.SiebelPropertySet;
import com.siebel.eai.SiebelBusinessServiceException;

import org.apache.commons.codec.binary.Base64;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
```

```

public class StringEncryption extends com.siebel.eai.SiebelBusinessService
{
public void doInvokeMethod(String methodName, SiebelPropertySet input,SiebelPropertySet output)
throws SiebelBusinessServiceException
{
if (!methodName.equals ("Decrypt"))
throw new SiebelBusinessServiceException("NO_SUCH_METHOD", "No such method");

else {
try
{
String encryptedstr = input.getProperty("Password");
String key = input.getProperty("Key");
SecretKeySpec skeySpec = new SecretKeySpec(Base64.decodeBase64(key), "AES");
byte[] encrypted = Base64.decodeBase64(encryptedstr);
Cipher cipher = Cipher.getInstance("AES");

cipher.init(Cipher.DECRYPT_MODE, skeySpec);
byte[] original = cipher.doFinal(encrypted);
String originalString = new String(original);
output.setProperty("PasswordText", originalString);
}
catch (Exception e) { }
}
} //end of void
} //end of class

```

Step2. Compile this Java code into a class file (StringEncryption.class) using javac command:

Command: javac StringEncryption.java

Step3. Jar the .class file created above.

Command: jar -cvf StringEncryption.class

2. Siebel Configuration:

Step1. Create reference LOV as shown below.

Type	Display Value	LIC
SECURITY_TOKEN	Yes	Security Req
SECURITY_TOKEN	Invalid Security Code	SECURITY ERROR CODE
SECURITY_TOKEN	Authentication Failed	SECURITY ERROR MSG

Step 2. Copy vanilla Business Service <EAI Java Business Service> and name it as EAI2Java BS. Add User Property to this BS with following details:

Name: @class

Value: StringEncryption

Note: The BS User Property value is the name of Java class. This should be given as pacakagename.Class. The Java Class created (StringEncryption.java) in step 1 under

Java File Create/Setup is not inside any package, hence the Value is mentioned without java package prefix.

Step 3. Create another Business Service: Test EAI Java BS. This Business Service invokes the java class and receives the Security Code Value in cleartext. Then it builds the Response as SUCCESS/FAILURE. Below is the code for this Business Service's PreInvoke Method:

```
function Service_PreInvokeMethod (MethodName, Inputs, Outputs)
{
    if(MethodName == "ValidateUser")
    {

        var pwd = Inputs.GetProperty("Password");
        var key;
        var result;
        var result1;

        //Start of Code to receive Decrypted value for Password

        var oBS = TheApplication().GetService("EAI2Java BS");
        var inPS = TheApplication().NewPropertySet();
        var outPS = TheApplication().NewPropertySet();

        key=TheApplication().InvokeMethod("LookupValue", "SECURITY_TOKEN", "Token");
        inPS.SetProperty("Password", pwd);
        inPS.SetProperty("Key", key);
        oBS.InvokeMethod("Decrypt", inPS, outPS);

        result = outPS.GetProperty("PasswordText");

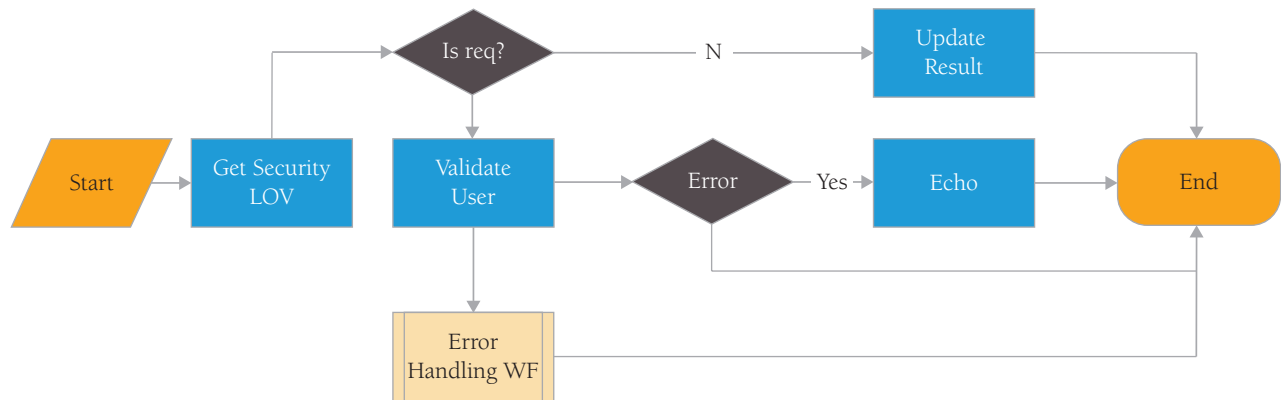
        var pwd=TheApplication().InvokeMethod("LookupValue", "SECURITY_TOKEN", "Password");

        if(result != null && result == pwd)
            result1 = "SUCCESS";
        else
            result1 = "FAILURE";

        Outputs.SetProperty("Result", result1);

        return(CancelOperation);
    }
    return (ContinueOperation);
}
```

Step 4. Create a workflow: Security Check WF. This WF is used to invoke the above Business Service.



Step 1. Get Security LOV

Name	Type	Business Service Name	Business Service Method	Sub Process Name	Business Components	Operation
Get Security LOV	Business Service	Workflow Utilities	Echo			
Output Arguments						
Property Name	Type	Value		Output Arguments		
IsSecurityRequired	Expression			Lookup Value ("SECURITY_TOKEN", Security Req")		

Step 2. Decision Step (Is required)

Name	Type	Business Service Name	Business Service Method	Sub Process Name	Business Component	Operation
Is Required						

If IsSecurityRequired = 'N', Go to Step 3 else goto Step 4

Step 3. Update Result

Name	Type	Business Service Name	Business Service Method	Sub Process Name	Business Component	Operation
Update Result	Business Service	Workflow Utilities	Echo			

Output Arguments			
Property Name	Type	Value	Output Arguments
Result	Literal		SUCCESS

Step 4. Validate User

Name	Type	Business Service Name	Business Service Method	Sub Process Name	Business Component	Operation
Validate User	Business Service	Test EAI Java BS	Validate User			

Input Argument	Type	Value	Property Name
Password	Process Property		Password

Output Arguments			
Property Name	Type	Value	Output Arguments
Result	Output Argument		Result

Note: The Output Process Property from Step 4 ie., Result will have value as “SUCCESS” or “FAILURE”.

Step 5. Decision Step

Name	Type	Business Service Name	Business Service Method	Sub Process Name	Business Component	Operation
Error?						

If Result ="SUCCESS", go to Step 6, else go to END Step

Step 6. Echo

Name	Type	Business Service Name	Business Service Method	Sub Process Name	Business Component	Operation
Echo	Business Service	Workflow Utilities	Echo			

Output Arguments			
Property Name	Type	Value	Output Arguments
Error Code	Expression		LookupValue("SECURITY_TOKEN ", "SECURITY ERROR CODE")
Error Code	Expression		LookupValue("SECURITY_TOKEN ", "SECURITY ERROR MSG")

3. Siebel Server Setup:

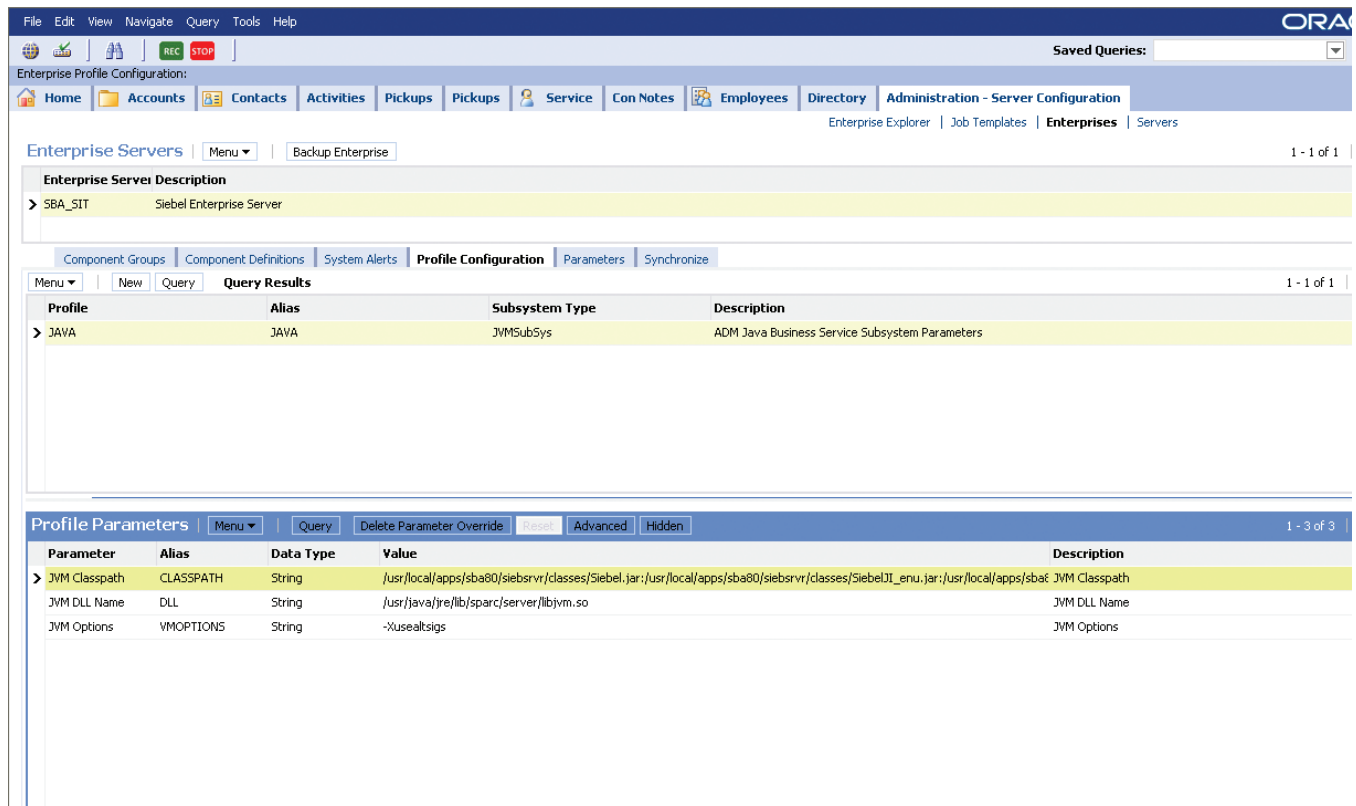
Step1. Navigate to SiteMap - Server Configuration> Enterprise Servers> Profile Configuration.

- Query for the JAVA Subsystem and update the following parameter

Parameter: JVM Classpath

Value:

`/usr/local/apps/sba80/siebsrvr/classes/Siebel.jar:/usr/local/apps/sba80/siebsrvr/classes/SiebelJI_enu.jar:/usr/local/apps/sba80/siebsrvr/classes/StringEncryption.jar:/usr/local/apps/sba80/siebsrvr/classes/commons-codec-1.4.jar`



The screenshot shows the Oracle Enterprise Server Administration interface. The 'Profile Configuration' tab is selected, displaying a table of Profile Parameters for the JAVA subsystem. The table has columns for Parameter, Alias, Data Type, Value, and Description.

Parameter	Alias	Data Type	Value	Description
JVM Classpath	CLASSPATH	String	<code>/usr/local/apps/sba80/siebsrvr/classes/Siebel.jar:/usr/local/apps/sba80/siebsrvr/classes/SiebelJI_enu.jar:/usr/local/apps/sba80/siebsrvr/classes/StringEncryption.jar:/usr/local/apps/sba80/siebsrvr/classes/commons-codec-1.4.jar</code>	JVM Classpath
JVM DLL Name	DLL	String	<code>/usr/java/jre/lib/sparc/server/libjvm.so</code>	JVM DLL Name
JVM Options	VMOPTIONS	String	<code>-Xusealtsigs</code>	JVM Options

4. Process Flow

1. Trigger the Security Check WF with Encrypted String as an Input Parameter
2. The step "Validate User" of the Workflow will invoke the Test EAI Java BS with the Encrypted String as an input parameter
3. The BS Test EAI Java BS will invoke make a call to the Business Service EAI2Java
4. The EAI2Java BS will invoke the method "Decrypt" of the Java class using its User Property @class.
5. The method "Decrypt" in the Java class will return the decrypted value of the Encrypted String to the BS Test EAI Java BS
6. The BS Test EAI Java BS will then compare the returned decrypted string with a value stored in Siebel and will return SUCCESS if the value is matching else FAILURE.
7. The Security Check WF will retrieve the value SUCCESS/FAILURE and if it is FAILURE then this writes the Error Message indicating "Invalid Authentication Code".

5. Unit Testing

- Unit Testing of the Custom Encryption using the Java File

1. Add the Main Method in the Java file as shown below.

```
public static void main(String[] args) {  
    StringEncryption st = new StringEncryption();  
    SiebelPropertySet inpSet = new SiebelPropertySet();  
    SiebelPropertySet outSet = new SiebelPropertySet();  
    inpSet.setProperty("Password","MoM1rQ2FgwAZ3O9qC/qc9Q==");  
    try  
    {  
        st.doInvokeMethod("Decrypt", inpSet, outSet);  
        String result = outSet.getProperty("PasswordText");  
        System.out.println("result in response: " + result);  
    }  
    catch (Exception e){}  
}
```

2. Set the classpath in you local machine as follows

```
set classpath=%classpath%;D:\Siebel.jar;D:\commons-codec-1.4.jar;.
```

3. Compile the Java file using the command `javac StringEncryption.java`.

4. Run the Java file using the command `java StringEncryption`

- Unit Testing of the Custom Encryption using the BS Simulator in Thin Client

1. Navigate to Business Service Simulator and provide the data as shown below

The screenshot displays the Oracle Business Service Simulator interface. The main window shows a test case configuration for 'Workflow Process Manager' with the method 'RunProcess' and 1 iteration. The 'Input Arguments' section is expanded, showing a table with the following data:

Test Case #	Type	Value	Child Type	Child Value	Property Name	Property Value
>					SecurityCode	MoM1rQ2FgwAZ3O9qC/qc9Q==

A 'Property Set Properties' dialog box is open, showing a table with the following data:

Property Name	Value
> SecurityCode	MoM1rQ2FgwAZ3O9qC/qc9Q==
ProcessName	Security Check WF

The 'Output Arguments' section at the bottom shows 'No Records'.

- Unit Testing of the Custom Encryption using the Workflow Simulator using Thick Client

Before testing using the Thick Client, update the relevant cfg file with the below section

[JAVA]

```
DLL = C:\Progra~1\Java\jdk1.5.0_17\re\bin\server\jvm.dll
```

```
CLASSPATH = D:\Siebel\Tools\CLASSES\Siebel.jar;D:\Siebel\Tools\CLASSES\SiebelJL_enu.jar;D:\StringEncryption.jar;  
D:\commons-codec-1.4.jar;
```

(Note : Change the above location based on the Installation path and the jar file location)

Other Features

- The Workflow provided in this Document can be used as a Child Process in the Parent Workflow, which is exposed as a Siebel Inbound WebService.
- The First Step in the Parent Workflow will be to invoke this Workflow and retrieve the Encrypted String and pass on this value to the Child Process (Security WF).
- Based on the response received from the Child Workflow (SUCCESS/FAILURE) the Parent Workflow can continue with Next Operation or terminated (or throw an Error) in case of Invalid Security Code received from External System.

Conclusion

The document offers a cost effective security implementation mechanism for Siebel interfaces. The decision to utilize the custom security implementation provided in this paper should be based on specific budgetary constraints and the criticality of secure interfaces. While the document illustrates the security implementation solution using AES-128 algorithm, the idea can easily be extended to other encryption routines.

References

Reference 1:

Title: Using AES with Java Technology

URL: http://java.sun.com/developer/technicalArticles/Security/AES/AES_v1.html

Reference 2:

Siebel Bookshelf: Transports and Interfaces: Siebel Enterprise Application Integration

Refer Chapter 5: Java Business Service

About the Author

[Muralidhar Reddy Kohir](#), Technology Architect- Siebel, Oracle Practice, Infosys. Muralidhar has over 10 years IT experience, primarily in the Siebel CRM space. He has worked and successfully delivered multiple end-to-end Siebel implementations for Logistics and eEnergy clients. His core skill-set includes Siebel EAI. He is a B.Tech. in Electronics and Communication from JNTU Hyderabad.



For more information, contact askus@infosys.com

About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit www.infosys.com.