Webservice Ws Security

Example webservice-ws-security can be browsed at https://github.com/apache/tomee/tree/master/examples/webservice-ws-security

Help us document this example! Click the blue pencil icon in the upper right to edit this page.

CalculatorImpl

```
package org.superbiz.calculator;

import javax.annotation.security.DeclareRoles;
import javax.annotation.security.RolesAllowed;
import javax.ejb.Stateless;
import javax.jws.WebService;

/**
 * This is an EJB 3 style pojo stateless session bean
 * Every stateless session bean implementation must be annotated
 * using the annotation @Stateless
 * This EJB has a single interface: CalculatorWs a webservice interface.
 */
//START SNIPPET: code
@DeclareRoles(value = {"Administrator"})
@Stateless
@WebService(
        portName = "CalculatorPort",
        serviceName = "CalculatorWsService",
        targetNamespace = "http://superbiz.org/wsdl",
        endpointInterface = "org.superbiz.calculator.CalculatorWs")
public class CalculatorImpl implements CalculatorWs, CalculatorRemote {

    @RolesAllowed(value = {"Administrator"})
    public int sum(int add1, int add2) {
        return add1 + add2;
    }

    public int multiply(int mul1, int mul2) {
        return mul1 * mul2;
    }
}
```

CalculatorRemote

```
package org.superbiz.calculator;

import javax.ejb.Remote;

@Remote
public interface CalculatorRemote {

    public int sum(int add1, int add2);

    public int multiply(int mul1, int mul2);
}
```

CalculatorWs

```
package org.superbiz.calculator;

import javax.jws.WebService;

//END SNIPPET: code

/**
 * This is an EJB 3 webservice interface
 * A webservice interface must be annotated with the @Local
 * annotation.
 */
//START SNIPPET: code
@WebService(targetNamespace = "http://superbiz.org/wsdl")
public interface CalculatorWs {

    public int sum(int add1, int add2);

    public int multiply(int mul1, int mul2);
}
```

ejb-jar.xml

```
<ejb-jar xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/ejb-jar_3_0.xsd"
         version="3.0" id="simple" metadata-complete="false">

  <enterprise-beans>

    <session>
```

```
    <ejb-name>CalculatorImplTimestamp1way</ejb-name>
    <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
    <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Container</transaction-type>
  </session>

  <session>
    <ejb-name>CalculatorImplTimestamp2ways</ejb-name>
    <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
    <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Container</transaction-type>
  </session>

  <session>
    <ejb-name>CalculatorImplUsernameTokenPlainPassword</ejb-name>
    <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
    <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Container</transaction-type>
  </session>

  <session>
    <ejb-name>CalculatorImplUsernameTokenHashedPassword</ejb-name>
    <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
    <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Container</transaction-type>
  </session>

  <session>
    <ejb-name>CalculatorImplUsernameTokenPlainPasswordEncrypt</ejb-name>
    <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
    <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Container</transaction-type>
  </session>

  <session>
    <ejb-name>CalculatorImplSign</ejb-name>
    <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
    <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
    <session-type>Stateless</session-type>
    <transaction-type>Container</transaction-type>
  </session>

  <session>
    <ejb-name>CalculatorImplEncrypt2ways</ejb-name>
    <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
    <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
```

```
      <session-type>Stateless</session-type>
      <transaction-type>Container</transaction-type>
    </session>

    <session>
      <ejb-name>CalculatorImplSign2ways</ejb-name>
      <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
      <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
      <session-type>Stateless</session-type>
      <transaction-type>Container</transaction-type>
    </session>

    <session>
      <ejb-name>CalculatorImplEncryptAndSign2ways</ejb-name>
      <service-endpoint>org.superbiz.calculator.CalculatorWs</service-endpoint>
      <ejb-class>org.superbiz.calculator.CalculatorImpl</ejb-class>
      <session-type>Stateless</session-type>
      <transaction-type>Container</transaction-type>
    </session>

  </enterprise-beans>

</ejb-jar>
```

openejb-jar.xml

```
<openejb-jar xmlns="http://www.openejb.org/openejb-jar/1.1">

  <ejb-deployment ejb-name="CalculatorImpl">
    <properties>
      # webservice.security.realm
      # webservice.security.securityRealm
      # webservice.security.transportGarantee = NONE
      webservice.security.authMethod = WS-SECURITY
      wss4j.in.action = UsernameToken
      wss4j.in.passwordType = PasswordText
      wss4j.in.passwordCallbackClass = org.superbiz.calculator.CustomPasswordHandler

      # automatically added
      wss4j.in.validator.{http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-secext-1.0.xsd}UsernameToken =
org.apache.openejb.server.cxf.OpenEJBLoginValidator
    </properties>
  </ejb-deployment>
  <ejb-deployment ejb-name="CalculatorImplTimestamp1way">
    <properties>
      webservice.security.authMethod = WS-SECURITY
      wss4j.in.action = Timestamp
    </properties>
```

```
  </ejb-deployment>
  <ejb-deployment ejb-name="CalculatorImplTimestamp2ways">
    <properties>
      webservice.security.authMethod = WS-SECURITY
      wss4j.in.action = Timestamp
      wss4j.out.action = Timestamp
    </properties>
  </ejb-deployment>
  <ejb-deployment ejb-name="CalculatorImplUsernameTokenPlainPassword">
    <properties>
      webservice.security.authMethod = WS-SECURITY
      wss4j.in.action = UsernameToken
      wss4j.in.passwordType = PasswordText
      wss4j.in.passwordCallbackClass=org.superbiz.calculator.CustomPasswordHandler
    </properties>
  </ejb-deployment>
  <ejb-deployment ejb-name="CalculatorImplUsernameTokenHashedPassword">
    <properties>
      webservice.security.authMethod = WS-SECURITY
      wss4j.in.action = UsernameToken
      wss4j.in.passwordType = PasswordDigest
      wss4j.in.passwordCallbackClass=org.superbiz.calculator.CustomPasswordHandler
    </properties>
  </ejb-deployment>
  <ejb-deployment ejb-name="CalculatorImplUsernameTokenPlainPasswordEncrypt">
    <properties>
      webservice.security.authMethod = WS-SECURITY
      wss4j.in.action = UsernameToken Encrypt
      wss4j.in.passwordType = PasswordText
      wss4j.in.passwordCallbackClass=org.superbiz.calculator.CustomPasswordHandler
      wss4j.in.decryptionPropFile = META-
INF/CalculatorImplUsernameTokenPlainPasswordEncrypt-server.properties
    </properties>
  </ejb-deployment>
  <ejb-deployment ejb-name="CalculatorImplSign">
    <properties>
      webservice.security.authMethod = WS-SECURITY
      wss4j.in.action = Signature
      wss4j.in.passwordCallbackClass=org.superbiz.calculator.CustomPasswordHandler
      wss4j.in.signaturePropFile = META-INF/CalculatorImplSign-server.properties
    </properties>
  </ejb-deployment>

</openejb-jar>
```

webservices.xml

```
<webservices xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"
```

```
            xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://www.ibm.com/webservices/xsd/j2ee_web_services_1_1.xsd"
            xmlns:ger="http://ciaows.org/wsdl" version="1.1">

  <webservice-description>
    <webservice-description-name>CalculatorWsService</webservice-description-name>
    <port-component>
      <port-component-name>CalculatorImplTimestamp1way</port-component-name>
      <wsdl-port>CalculatorImplTimestamp1way</wsdl-port>
      <service-endpoint-interface>org.superbiz.calculator.CalculatorWs</service-
endpoint-interface>
      <service-impl-bean>
        <ejb-link>CalculatorImplTimestamp1way</ejb-link>
      </service-impl-bean>
    </port-component>
    <port-component>
      <port-component-name>CalculatorImplTimestamp2ways</port-component-name>
      <wsdl-port>CalculatorImplTimestamp2ways</wsdl-port>
      <service-endpoint-interface>org.superbiz.calculator.CalculatorWs</service-
endpoint-interface>
      <service-impl-bean>
        <ejb-link>CalculatorImplTimestamp2ways</ejb-link>
      </service-impl-bean>
    </port-component>
    <port-component>
      <port-component-name>CalculatorImplUsernameTokenPlainPassword</port-component-
name>
      <wsdl-port>CalculatorImplUsernameTokenPlainPassword</wsdl-port>
      <service-endpoint-interface>org.superbiz.calculator.CalculatorWs</service-
endpoint-interface>
      <service-impl-bean>
        <ejb-link>CalculatorImplUsernameTokenPlainPassword</ejb-link>
      </service-impl-bean>
    </port-component>
    <port-component>
      <port-component-name>CalculatorImplUsernameTokenHashedPassword</port-component-
name>
      <wsdl-port>CalculatorImplUsernameTokenHashedPassword</wsdl-port>
      <service-endpoint-interface>org.superbiz.calculator.CalculatorWs</service-
endpoint-interface>
      <service-impl-bean>
        <ejb-link>CalculatorImplUsernameTokenHashedPassword</ejb-link>
      </service-impl-bean>
    </port-component>
    <port-component>
      <port-component-name>CalculatorImplUsernameTokenPlainPasswordEncrypt</port-
component-name>
      <wsdl-port>CalculatorImplUsernameTokenPlainPasswordEncrypt</wsdl-port>
      <service-endpoint-interface>org.superbiz.calculator.CalculatorWs</service-
endpoint-interface>
      <service-impl-bean>
```

```
        <ejb-link>CalculatorImplUsernameTokenPlainPasswordEncrypt</ejb-link>
      </service-impl-bean>
    </port-component>
  </webservice-description>

</webservices>
```

CalculatorTest

```
package org.superbiz.calculator;

import junit.framework.TestCase;
import org.apache.cxf.binding.soap.saaj.SAAJInInterceptor;
import org.apache.cxf.binding.soap.saaj.SAAJOutInterceptor;
import org.apache.cxf.endpoint.Client;
import org.apache.cxf.endpoint.Endpoint;
import org.apache.cxf.frontend.ClientProxy;
import org.apache.cxf.ws.security.wss4j.WSS4JInInterceptor;
import org.apache.cxf.ws.security.wss4j.WSS4JOutInterceptor;
import org.apache.ws.security.WSConstants;
import org.apache.ws.security.WSPasswordCallback;
import org.apache.ws.security.handler.WSHandlerConstants;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.security.auth.callback.Callback;
import javax.security.auth.callback.CallbackHandler;
import javax.security.auth.callback.UnsupportedCallbackException;
import javax.xml.namespace.QName;
import javax.xml.ws.Service;
import javax.xml.ws.soap.SOAPBinding;
import java.io.IOException;
import java.net.URL;
import java.util.HashMap;
import java.util.Map;
import java.util.Properties;

public class CalculatorTest extends TestCase {

    //START SNIPPET: setup
    protected void setUp() throws Exception {
        Properties properties = new Properties();
        properties.setProperty(Context.INITIAL_CONTEXT_FACTORY,
"org.apache.openejb.core.LocalInitialContextFactory");
        properties.setProperty("openejb.embedded.remotable", "true");

        new InitialContext(properties);
    }
    //END SNIPPET: setup
```

```
    //START SNIPPET: webservice
    public void testCalculatorViaWsInterface() throws Exception {
        Service calcService = Service.create(new URL(
"http://127.0.0.1:4204/CalculatorImpl?wsdl"),
                new QName("http://superbiz.org/wsdl", "CalculatorWsService"));
        assertNotNull(calcService);

        CalculatorWs calc = calcService.getPort(CalculatorWs.class);

        Client client = ClientProxy.getClient(calc);
        Endpoint endpoint = client.getEndpoint();
        endpoint.getOutInterceptors().add(new SAAJOutInterceptor());

        Map<String, Object> outProps = new HashMap<String, Object>();
        outProps.put(WSHandlerConstants.ACTION, WSHandlerConstants.USERNAME_TOKEN);
        outProps.put(WSHandlerConstants.USER, "jane");
        outProps.put(WSHandlerConstants.PASSWORD_TYPE, WSConstants.PW_TEXT);
        outProps.put(WSHandlerConstants.PW_CALLBACK_REF, new CallbackHandler() {

            public void handle(Callback[] callbacks) throws IOException,
UnsupportedCallbackException {
                WSPasswordCallback pc = (WSPasswordCallback) callbacks[0];
                pc.setPassword("waterfall");
            }
        });

        WSS4JOutInterceptor wssOut = new WSS4JOutInterceptor(outProps);
        endpoint.getOutInterceptors().add(wssOut);

        assertEquals(10, calc.sum(4, 6));
    }

    public void testCalculatorViaWsInterfaceWithTimestamp1way() throws Exception {
        Service calcService = Service.create(new URL(
"http://127.0.0.1:4204/CalculatorImplTimestamp1way?wsdl"),
                new QName("http://superbiz.org/wsdl", "CalculatorWsService"));
        assertNotNull(calcService);

        // for debugging (ie. TCPMon)
        calcService.addPort(new QName("http://superbiz.org/wsdl",
                "CalculatorWsService2"), SOAPBinding.SOAP12HTTP_BINDING,
                "http://127.0.0.1:8204/CalculatorImplTimestamp1way");

//        CalculatorWs calc = calcService.getPort(
//          new QName("http://superbiz.org/wsdl", "CalculatorWsService2"),
//      CalculatorWs.class);

        CalculatorWs calc = calcService.getPort(CalculatorWs.class);

        Client client = ClientProxy.getClient(calc);
```

```
        Endpoint endpoint = client.getEndpoint();
        endpoint.getOutInterceptors().add(new SAAJOutInterceptor());

        Map<String, Object> outProps = new HashMap<String, Object>();
        outProps.put(WSHandlerConstants.ACTION, WSHandlerConstants.TIMESTAMP);
        WSS4JOutInterceptor wssOut = new WSS4JOutInterceptor(outProps);
        endpoint.getOutInterceptors().add(wssOut);

        assertEquals(12, calc.multiply(3, 4));
    }

    public void testCalculatorViaWsInterfaceWithTimestamp2ways() throws Exception {
        Service calcService = Service.create(new URL(
"http://127.0.0.1:4204/CalculatorImplTimestamp2ways?wsdl"),
                new QName("http://superbiz.org/wsdl", "CalculatorWsService"));
        assertNotNull(calcService);

        // for debugging (ie. TCPMon)
        calcService.addPort(new QName("http://superbiz.org/wsdl",
                "CalculatorWsService2"), SOAPBinding.SOAP12HTTP_BINDING,
                "http://127.0.0.1:8204/CalculatorImplTimestamp2ways");

//        CalculatorWs calc = calcService.getPort(
//          new QName("http://superbiz.org/wsdl", "CalculatorWsService2"),
//      CalculatorWs.class);

        CalculatorWs calc = calcService.getPort(CalculatorWs.class);

        Client client = ClientProxy.getClient(calc);
        Endpoint endpoint = client.getEndpoint();
        endpoint.getOutInterceptors().add(new SAAJOutInterceptor());
        endpoint.getInInterceptors().add(new SAAJInInterceptor());

        Map<String, Object> outProps = new HashMap<String, Object>();
        outProps.put(WSHandlerConstants.ACTION, WSHandlerConstants.TIMESTAMP);
        WSS4JOutInterceptor wssOut = new WSS4JOutInterceptor(outProps);
        endpoint.getOutInterceptors().add(wssOut);

        Map<String, Object> inProps = new HashMap<String, Object>();
        inProps.put(WSHandlerConstants.ACTION, WSHandlerConstants.TIMESTAMP);
        WSS4JInInterceptor wssIn = new WSS4JInInterceptor(inProps);
        endpoint.getInInterceptors().add(wssIn);

        assertEquals(12, calc.multiply(3, 4));
    }

    public void testCalculatorViaWsInterfaceWithUsernameTokenPlainPassword() throws
Exception {
        Service calcService = Service.create(new URL(
"http://127.0.0.1:4204/CalculatorImplUsernameTokenPlainPassword?wsdl"),
                new QName("http://superbiz.org/wsdl", "CalculatorWsService"));
```

```
        assertNotNull(calcService);

        // for debugging (ie. TCPMon)
        calcService.addPort(new QName("http://superbiz.org/wsdl",
                "CalculatorWsService2"), SOAPBinding.SOAP12HTTP_BINDING,
                "http://127.0.0.1:8204/CalculatorImplUsernameTokenPlainPassword");

//        CalculatorWs calc = calcService.getPort(
//          new QName("http://superbiz.org/wsdl", "CalculatorWsService2"),
//          CalculatorWs.class);

        CalculatorWs calc = calcService.getPort(CalculatorWs.class);

        Client client = ClientProxy.getClient(calc);
        Endpoint endpoint = client.getEndpoint();
        endpoint.getOutInterceptors().add(new SAAJOutInterceptor());

        Map<String, Object> outProps = new HashMap<String, Object>();
        outProps.put(WSHandlerConstants.ACTION, WSHandlerConstants.USERNAME_TOKEN);
        outProps.put(WSHandlerConstants.USER, "jane");
        outProps.put(WSHandlerConstants.PASSWORD_TYPE, WSConstants.PW_TEXT);
        outProps.put(WSHandlerConstants.PW_CALLBACK_REF, new CallbackHandler() {

            @Override
            public void handle(Callback[] callbacks) throws IOException,
UnsupportedCallbackException {
                WSPasswordCallback pc = (WSPasswordCallback) callbacks[0];
                pc.setPassword("waterfall");
            }
        });

        WSS4JOutInterceptor wssOut = new WSS4JOutInterceptor(outProps);
        endpoint.getOutInterceptors().add(wssOut);

        assertEquals(10, calc.sum(4, 6));
    }

    public void testCalculatorViaWsInterfaceWithUsernameTokenHashedPassword() throws
Exception {
        Service calcService = Service.create(new URL(
"http://127.0.0.1:4204/CalculatorImplUsernameTokenHashedPassword?wsdl"),
                new QName("http://superbiz.org/wsdl", "CalculatorWsService"));
        assertNotNull(calcService);

        // for debugging (ie. TCPMon)
        calcService.addPort(new QName("http://superbiz.org/wsdl",
                "CalculatorWsService2"), SOAPBinding.SOAP12HTTP_BINDING,
                "http://127.0.0.1:8204/CalculatorImplUsernameTokenHashedPassword");

//        CalculatorWs calc = calcService.getPort(
//          new QName("http://superbiz.org/wsdl", "CalculatorWsService2"),
```

```
//          CalculatorWs.class);

        CalculatorWs calc = calcService.getPort(CalculatorWs.class);

        Client client = ClientProxy.getClient(calc);
        Endpoint endpoint = client.getEndpoint();
        endpoint.getOutInterceptors().add(new SAAJOutInterceptor());

        Map<String, Object> outProps = new HashMap<String, Object>();
        outProps.put(WSHandlerConstants.ACTION, WSHandlerConstants.USERNAME_TOKEN);
        outProps.put(WSHandlerConstants.USER, "jane");
        outProps.put(WSHandlerConstants.PASSWORD_TYPE, WSConstants.PW_DIGEST);
        outProps.put(WSHandlerConstants.PW_CALLBACK_REF, new CallbackHandler() {

            @Override
            public void handle(Callback[] callbacks) throws IOException,
UnsupportedCallbackException {
                WSPasswordCallback pc = (WSPasswordCallback) callbacks[0];
                pc.setPassword("waterfall");
            }
        });

        WSS4JOutInterceptor wssOut = new WSS4JOutInterceptor(outProps);
        endpoint.getOutInterceptors().add(wssOut);

        assertEquals(10, calc.sum(4, 6));
    }

    public void testCalculatorViaWsInterfaceWithUsernameTokenPlainPasswordEncrypt()
throws Exception {
        Service calcService = Service.create(new URL(
"http://127.0.0.1:4204/CalculatorImplUsernameTokenPlainPasswordEncrypt?wsdl"),
                new QName("http://superbiz.org/wsdl", "CalculatorWsService"));
        assertNotNull(calcService);

        // for debugging (ie. TCPMon)
        calcService.addPort(new QName("http://superbiz.org/wsdl",
                "CalculatorWsService2"), SOAPBinding.SOAP12HTTP_BINDING,
                "
http://127.0.0.1:8204/CalculatorImplUsernameTokenPlainPasswordEncrypt");

//        CalculatorWs calc = calcService.getPort(
//          new QName("http://superbiz.org/wsdl", "CalculatorWsService2"),
//          CalculatorWs.class);

        CalculatorWs calc = calcService.getPort(CalculatorWs.class);

        Client client = ClientProxy.getClient(calc);
        Endpoint endpoint = client.getEndpoint();
        endpoint.getOutInterceptors().add(new SAAJOutInterceptor());
```

```
        Map<String, Object> outProps = new HashMap<String, Object>();
        outProps.put(WSHandlerConstants.ACTION, WSHandlerConstants.USERNAME_TOKEN
                + " " + WSHandlerConstants.ENCRYPT);
        outProps.put(WSHandlerConstants.USER, "jane");
        outProps.put(WSHandlerConstants.PASSWORD_TYPE, WSConstants.PW_TEXT);
        outProps.put(WSHandlerConstants.PW_CALLBACK_REF, new CallbackHandler() {

            @Override
            public void handle(Callback[] callbacks) throws IOException,
UnsupportedCallbackException {
                WSPasswordCallback pc = (WSPasswordCallback) callbacks[0];
                pc.setPassword("waterfall");
            }
        });
        outProps.put(WSHandlerConstants.ENC_PROP_FILE, "META-
INF/CalculatorImplUsernameTokenPlainPasswordEncrypt-client.properties");
        outProps.put(WSHandlerConstants.ENCRYPTION_USER, "serveralias");

        WSS4JOutInterceptor wssOut = new WSS4JOutInterceptor(outProps);
        endpoint.getOutInterceptors().add(wssOut);

        assertEquals(10, calc.sum(4, 6));
    }

    public void testCalculatorViaWsInterfaceWithSign() throws Exception {
        Service calcService = Service.create(new URL(
"http://127.0.0.1:4204/CalculatorImplSign?wsdl"),
                new QName("http://superbiz.org/wsdl", "CalculatorWsService"));
        assertNotNull(calcService);

        // for debugging (ie. TCPMon)
        calcService.addPort(new QName("http://superbiz.org/wsdl",
                "CalculatorWsService2"), SOAPBinding.SOAP12HTTP_BINDING,
                "http://127.0.0.1:8204/CalculatorImplSign");

//      CalculatorWs calc = calcService.getPort(
//  new QName("http://superbiz.org/wsdl", "CalculatorWsService2"),
//  CalculatorWs.class);

        CalculatorWs calc = calcService.getPort(CalculatorWs.class);

        Client client = ClientProxy.getClient(calc);
        Endpoint endpoint = client.getEndpoint();
        endpoint.getOutInterceptors().add(new SAAJOutInterceptor());

        Map<String, Object> outProps = new HashMap<String, Object>();
        outProps.put(WSHandlerConstants.ACTION, WSHandlerConstants.SIGNATURE);
        outProps.put(WSHandlerConstants.USER, "clientalias");
        outProps.put(WSHandlerConstants.PW_CALLBACK_REF, new CallbackHandler() {

            @Override
```

```
            public void handle(Callback[] callbacks) throws IOException,
UnsupportedCallbackException {
                WSPasswordCallback pc = (WSPasswordCallback) callbacks[0];
                pc.setPassword("clientPassword");
            }
        });
        outProps.put(WSHandlerConstants.SIG_PROP_FILE, "META-INF/CalculatorImplSign-
client.properties");
        outProps.put(WSHandlerConstants.SIG_KEY_ID, "IssuerSerial");

        WSS4JOutInterceptor wssOut = new WSS4JOutInterceptor(outProps);
        endpoint.getOutInterceptors().add(wssOut);

        assertEquals(24, calc.multiply(4, 6));
    }
    //END SNIPPET: webservice
}
```

CustomPasswordHandler

```
package org.superbiz.calculator;

import org.apache.ws.security.WSPasswordCallback;

import javax.security.auth.callback.Callback;
import javax.security.auth.callback.CallbackHandler;
import javax.security.auth.callback.UnsupportedCallbackException;
import java.io.IOException;

public class CustomPasswordHandler implements CallbackHandler {
    @Override
    public void handle(Callback[] callbacks) throws IOException,
UnsupportedCallbackException {
        WSPasswordCallback pc = (WSPasswordCallback) callbacks[0];

        if (pc.getUsage() == WSPasswordCallback.USERNAME_TOKEN) {
            // TODO get the password from the users.properties if possible
            pc.setPassword("waterfall");
        } else if (pc.getUsage() == WSPasswordCallback.DECRYPT) {
            pc.setPassword("serverPassword");
        } else if (pc.getUsage() == WSPasswordCallback.SIGNATURE) {
            pc.setPassword("serverPassword");
        }
    }
}
```

Running

```
generate keys:
```

```
do.sun.jdk:
     [echo] *** Running on a Sun JDK ***
     [echo] generate server keys
     [java] Certificate stored in file </Users/dblevins/examples/webservice-ws-
security/target/classes/META-INF/serverKey.rsa>
     [echo] generate client keys
     [java] Certificate stored in file </Users/dblevins/examples/webservice-ws-
security/target/test-classes/META-INF/clientKey.rsa>
     [echo] import client/server public keys in client/server keystores
     [java] Certificate was added to keystore
     [java] Certificate was added to keystore
```

```
do.ibm.jdk:
```

```
run:
     [echo] Running JDK specific keystore creation target
```

```
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
Running org.superbiz.calculator.CalculatorTest
Apache OpenEJB 4.0.0-beta-1    build: 20111002-04:06
http://tomee.apache.org/
INFO - openejb.home = /Users/dblevins/examples/webservice-ws-security
INFO - openejb.base = /Users/dblevins/examples/webservice-ws-security
INFO - Configuring Service(id=Default Security Service, type=SecurityService,
provider-id=Default Security Service)
INFO - Configuring Service(id=Default Transaction Manager, type=TransactionManager,
provider-id=Default Transaction Manager)
INFO - Found EjbModule in classpath: /Users/dblevins/examples/webservice-ws-
security/target/classes
INFO - Beginning load: /Users/dblevins/examples/webservice-ws-security/target/classes
INFO - Configuring enterprise application: /Users/dblevins/examples/webservice-ws-
security/classpath.ear
INFO - Configuring Service(id=Default Stateless Container, type=Container, provider-
id=Default Stateless Container)
INFO - Auto-creating a container for bean CalculatorImplTimestamp1way:
Container(type=STATELESS, id=Default Stateless Container)
INFO - Enterprise application "/Users/dblevins/examples/webservice-ws-
security/classpath.ear" loaded.
```

```
INFO - Assembling app: /Users/dblevins/examples/webservice-ws-security/classpath.ear
INFO - Jndi(name=CalculatorImplTimestamp1wayRemote) --> Ejb(deployment-
id=CalculatorImplTimestamp1way)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplTimestamp1way!org.superbiz.calcula
tor.CalculatorRemote) --> Ejb(deployment-id=CalculatorImplTimestamp1way)
INFO - Jndi(name=global/classpath.ear/simple/CalculatorImplTimestamp1way) -->
Ejb(deployment-id=CalculatorImplTimestamp1way)
INFO - Jndi(name=CalculatorImplTimestamp2waysRemote) --> Ejb(deployment-
id=CalculatorImplTimestamp2ways)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplTimestamp2ways!org.superbiz.calcul
ator.CalculatorRemote) --> Ejb(deployment-id=CalculatorImplTimestamp2ways)
INFO - Jndi(name=global/classpath.ear/simple/CalculatorImplTimestamp2ways) -->
Ejb(deployment-id=CalculatorImplTimestamp2ways)
INFO - Jndi(name=CalculatorImplUsernameTokenPlainPasswordRemote) --> Ejb(deployment-
id=CalculatorImplUsernameTokenPlainPassword)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplUsernameTokenPlainPassword!org.sup
erbiz.calculator.CalculatorRemote) --> Ejb(deployment-
id=CalculatorImplUsernameTokenPlainPassword)
INFO - Jndi(name=global/classpath.ear/simple/CalculatorImplUsernameTokenPlainPassword)
--> Ejb(deployment-id=CalculatorImplUsernameTokenPlainPassword)
INFO - Jndi(name=CalculatorImplUsernameTokenHashedPasswordRemote) --> Ejb(deployment-
id=CalculatorImplUsernameTokenHashedPassword)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplUsernameTokenHashedPassword!org.su
perbiz.calculator.CalculatorRemote) --> Ejb(deployment-
id=CalculatorImplUsernameTokenHashedPassword)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplUsernameTokenHashedPassword) -->
Ejb(deployment-id=CalculatorImplUsernameTokenHashedPassword)
INFO - Jndi(name=CalculatorImplUsernameTokenPlainPasswordEncryptRemote) -->
Ejb(deployment-id=CalculatorImplUsernameTokenPlainPasswordEncrypt)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplUsernameTokenPlainPasswordEncrypt!
org.superbiz.calculator.CalculatorRemote) --> Ejb(deployment-
id=CalculatorImplUsernameTokenPlainPasswordEncrypt)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplUsernameTokenPlainPasswordEncrypt)
--> Ejb(deployment-id=CalculatorImplUsernameTokenPlainPasswordEncrypt)
INFO - Jndi(name=CalculatorImplSignRemote) --> Ejb(deployment-id=CalculatorImplSign)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplSign!org.superbiz.calculator.Calcu
latorRemote) --> Ejb(deployment-id=CalculatorImplSign)
INFO - Jndi(name=global/classpath.ear/simple/CalculatorImplSign) --> Ejb(deployment-
id=CalculatorImplSign)
INFO - Jndi(name=CalculatorImplEncrypt2waysRemote) --> Ejb(deployment-
id=CalculatorImplEncrypt2ways)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplEncrypt2ways!org.superbiz.calculat
```

```
or.CalculatorRemote) --> Ejb(deployment-id=CalculatorImplEncrypt2ways)
INFO - Jndi(name=global/classpath.ear/simple/CalculatorImplEncrypt2ways) -->
Ejb(deployment-id=CalculatorImplEncrypt2ways)
INFO - Jndi(name=CalculatorImplSign2waysRemote) --> Ejb(deployment-
id=CalculatorImplSign2ways)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplSign2ways!org.superbiz.calculator.
CalculatorRemote) --> Ejb(deployment-id=CalculatorImplSign2ways)
INFO - Jndi(name=global/classpath.ear/simple/CalculatorImplSign2ways) -->
Ejb(deployment-id=CalculatorImplSign2ways)
INFO - Jndi(name=CalculatorImplEncryptAndSign2waysRemote) --> Ejb(deployment-
id=CalculatorImplEncryptAndSign2ways)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImplEncryptAndSign2ways!org.superbiz.c
alculator.CalculatorRemote) --> Ejb(deployment-id=CalculatorImplEncryptAndSign2ways)
INFO - Jndi(name=global/classpath.ear/simple/CalculatorImplEncryptAndSign2ways) -->
Ejb(deployment-id=CalculatorImplEncryptAndSign2ways)
INFO - Jndi(name=CalculatorImplRemote) --> Ejb(deployment-id=CalculatorImpl)
INFO -
Jndi(name=global/classpath.ear/simple/CalculatorImpl!org.superbiz.calculator.Calculato
rRemote) --> Ejb(deployment-id=CalculatorImpl)
INFO - Jndi(name=global/classpath.ear/simple/CalculatorImpl) --> Ejb(deployment-
id=CalculatorImpl)
INFO - Created Ejb(deployment-id=CalculatorImplUsernameTokenHashedPassword, ejb-
name=CalculatorImplUsernameTokenHashedPassword, container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImpl, ejb-name=CalculatorImpl,
container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImplSign, ejb-name=CalculatorImplSign,
container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImplEncryptAndSign2ways, ejb-
name=CalculatorImplEncryptAndSign2ways, container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImplTimestamp1way, ejb-
name=CalculatorImplTimestamp1way, container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImplSign2ways, ejb-
name=CalculatorImplSign2ways, container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImplEncrypt2ways, ejb-
name=CalculatorImplEncrypt2ways, container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImplUsernameTokenPlainPassword, ejb-
name=CalculatorImplUsernameTokenPlainPassword, container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImplTimestamp2ways, ejb-
name=CalculatorImplTimestamp2ways, container=Default Stateless Container)
INFO - Created Ejb(deployment-id=CalculatorImplUsernameTokenPlainPasswordEncrypt, ejb-
name=CalculatorImplUsernameTokenPlainPasswordEncrypt, container=Default Stateless
Container)
INFO - Started Ejb(deployment-id=CalculatorImplUsernameTokenHashedPassword, ejb-
name=CalculatorImplUsernameTokenHashedPassword, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImpl, ejb-name=CalculatorImpl,
container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImplSign, ejb-name=CalculatorImplSign,
container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImplEncryptAndSign2ways, ejb-
```

```
name=CalculatorImplEncryptAndSign2ways, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImplTimestamp1way, ejb-
name=CalculatorImplTimestamp1way, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImplSign2ways, ejb-
name=CalculatorImplSign2ways, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImplEncrypt2ways, ejb-
name=CalculatorImplEncrypt2ways, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImplUsernameTokenPlainPassword, ejb-
name=CalculatorImplUsernameTokenPlainPassword, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImplTimestamp2ways, ejb-
name=CalculatorImplTimestamp2ways, container=Default Stateless Container)
INFO - Started Ejb(deployment-id=CalculatorImplUsernameTokenPlainPasswordEncrypt, ejb-
name=CalculatorImplUsernameTokenPlainPasswordEncrypt, container=Default Stateless
Container)
INFO - Deployed Application(path=/Users/dblevins/examples/webservice-ws-
security/classpath.ear)
INFO - Initializing network services
INFO - Creating ServerService(id=httpejbd)
INFO - Creating ServerService(id=cxf)
INFO - Creating ServerService(id=admin)
INFO - Creating ServerService(id=ejbd)
INFO - Creating ServerService(id=ejbds)
INFO - Initializing network services
  ** Starting Services **
  NAME                 IP              PORT
  httpejbd             127.0.0.1       4204
  admin thread         127.0.0.1       4200
  ejbd                 127.0.0.1       4201
  ejbd                 127.0.0.1       4203
-------
Ready!
Tests run: 7, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 4.582 sec

Results :

Tests run: 7, Failures: 0, Errors: 0, Skipped: 0
```