Hi community,

Some years ago i put in place a route that published a http endpoint as https with basic authentification in Talend ESB. My idea was to do a https proxy for a http (soap) service with as simple route.

The route exposes a webservice in https with basic authentification.

After some help from this great Talend community, I was able to make the route work correctly.

Here the link with the details:

https://community.talend.com/t5/Design-and-Development/ESB-routes-https-proxy-ccxf/td-p/71460

I used Talend Open Studio for ESB 5.5.0 on java 7

Now I tried to migrated the route in Talend Open Studio for ESB 6.5.1. The route works fine when I run it in Talend Studio. The problem occurs when I build the kar file and try to deploy it in the Runtime.

here the screenshot of the route and what i put in the spring:

route_tos.jpg

here my spring configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Used to inject external resources, beans or define more CamelContext and RouteBuilder here--><beans xmlns="http://www.springframework.org/sche
<import resource="classpath*:META-INF/cxf/cxf.xml"/>
<import resource="classpath*:META-INF/cxf/cxf-extension-*.xml" />
<import resource="classpath*:META-INF/cxf/cxf-servlet.xml"/>

<bean id="jmxEventNotifier" class="org.apache.camel.management.JmxNotificationEventNotifier">
<property name="source" value="BERDOZ_QUEUE"/>
<property name="ignoreCamelContextEvents" value="true"/>
<property name="ignoreRouteEvents" value="true"/>
```

```
<property name="ignoreServiceEvents" value="true"/>
<property name="ignoreExchangeEvents" value="true"/></bean>

<http:destination name="https://localhost:8041/services/MyService">
</http:destination>

<httpj:engine-factory bus="cxf">
<!-- you just need to specify the TLS Server configuration for the certain port -->
<httpj:engine port="8041">
<httpj:tlsServerParameters>
<sec:keyManagers keyPassword="password">
<sec:keyStore type="JKS" password="password" file="C:/Talend/TOS_ESB-20180116_1512-V6.5.1/Runtime_ESBSE/container/etc/keystores/keystore.jks"/>
</sec:keyManagers>
<sec:trustManagers>
<sec:keyStore type="JKS" password="password" file="C:/Talend/TOS_ESB-20180116_1512-V6.5.1/Runtime_ESBSE/container/etc/keystores/keystore.jks"/>
</sec:trustManagers>
<sec:cipherSuitesFilter>
<!-- these filters ensure that a ciphersuite with
export-suitable or null encryption is used,
but exclude anonymous Diffie-Hellman key change as
this is vulnerable to man-in-the-middle attacks -->
<sec:include>.*_EXPORT_.*</sec:include>
<sec:include>.*_EXPORT1024_.*</sec:include>
<sec:include>.*_WITH_DES_.*</sec:include>
<sec:include>.*_WITH_AES_.*</sec:include>
<sec:include>.*_WITH_NULL_.*</sec:include>
<sec:exclude>.*_DH_anon_.*</sec:exclude>
</sec:cipherSuitesFilter>
<!--sec:clientAuthentication want="true" required="true"/-->
</httpj:tlsServerParameters>
</httpj:engine>
</httpj:engine-factory>

<bean id="authenticationInterceptor" class="org.apache.cxf.interceptor.security.JAASLoginInterceptor">
<property name="contextName" value="karaf"/>
<property name="roleClassifier" value="admin"/>
</bean>



</beans>
```

At the beginning I had some missing class errors,

So what I did is to add manually the jar dependencies in a cConfig element as proposed in this forum entry:

https://community.talend.com/t5/Design-and-Development/deployed-route-misses-dependencies/td-p/90269

I took the jar I found in the {path to workspace}/.java/lib/ and added them as external source in the cConfig.

Now I am stuck with this error:

```
2018-02-21T11:46:28,852 | ERROR | SpringOsgiExtenderThread-18 | BundleApplicationContextListener 50 | 194 - org.springframework.osgi.extender - 1.
org.springframework.beans.factory.BeanDefinitionStoreException: Unexpected exception parsing XML document from URL [bundleentry://698.fwk86633084
at org.springframework.beans.factory.xml.XmlBeanDefinitionReader.doLoadBeanDefinitions(XmlBeanDefinitionReader.java:413) [138:org.apache.servicemi
at org.springframework.beans.factory.xml.XmlBeanDefinitionReader.loadBeanDefinitions(XmlBeanDefinitionReader.java:335) [138:org.apache.servicemix.
at org.springframework.beans.factory.xml.XmlBeanDefinitionReader.loadBeanDefinitions(XmlBeanDefinitionReader.java:303) [138:org.apache.servicemix.
at org.springframework.beans.factory.support.AbstractBeanDefinitionReader.loadBeanDefinitions(AbstractBeanDefinitionReader.java:174) [138:org.apac
at org.springframework.beans.factory.support.AbstractBeanDefinitionReader.loadBeanDefinitions(AbstractBeanDefinitionReader.java:209) [138:org.apac
at org.springframework.beans.factory.support.AbstractBeanDefinitionReader.loadBeanDefinitions(AbstractBeanDefinitionReader.java:180) [138:org.apac
at org.springframework.osgi.context.support.OsgiBundleXmlApplicationContext.loadBeanDefinitions(OsgiBundleXmlApplicationContext.java:164) [193:org
at org.springframework.osgi.context.support.OsgiBundleXmlApplicationContext.loadBeanDefinitions(OsgiBundleXmlApplicationContext.java:136) [193:org
at org.springframework.context.support.AbstractRefreshableApplicationContext.refreshBeanFactory(AbstractRefreshableApplicationContext.java:130) [1
at org.springframework.context.support.AbstractApplicationContext.obtainFreshBeanFactory(AbstractApplicationContext.java:545) [139:org.apache.serv
at org.springframework.osgi.context.support.AbstractDelegatedExecutionApplicationContext.access$800(AbstractDelegatedExecutionApplicationContext.j
at org.springframework.osgi.context.support.AbstractDelegatedExecutionApplicationContext$3.run(AbstractDelegatedExecutionApplicationContext.java:2
at org.springframework.osgi.util.internal.PrivilegedUtils.executeWithCustomTCCL(PrivilegedUtils.java:85) [193:org.springframework.osgi.core:1.2.1]
at org.springframework.osgi.context.support.AbstractDelegatedExecutionApplicationContext.startRefresh(AbstractDelegatedExecutionApplicationContext
at org.springframework.osgi.extender.internal.dependencies.startup.DependencyWaiterApplicationContextExecutor.stageOne(DependencyWaiterApplicatio
at org.springframework.osgi.extender.internal.dependencies.startup.DependencyWaiterApplicationContextExecutor.refresh(DependencyWaiterApplication
at org.springframework.osgi.context.support.AbstractDelegatedExecutionApplicationContext.refresh(AbstractDelegatedExecutionApplicationContext.java
at org.springframework.osgi.extender.internal.activator.ContextLoaderListener$2.run(ContextLoaderListener.java:716) [194:org.springframework.osgi.
at java.lang.Thread.run(Thread.java:748) [?:?]
Caused by: org.springframework.beans.FatalBeanException: Class [org.apache.cxf.transport.http_jetty.spring.NamespaceHandler] for namespace [http:/
at org.springframework.beans.factory.xml.DefaultNamespaceHandlerResolver.resolve(DefaultNamespaceHandlerResolver.java:126) ~[?:?]
at org.springframework.osgi.context.support.DelegatedNamespaceHandlerResolver.resolve(DelegatedNamespaceHandlerResolver.java:56) ~[?:?]
at org.springframework.beans.factory.xml.BeanDefinitionParserDelegate.parseCustomElement(BeanDefinitionParserDelegate.java:1427) ~[?:?]
at org.springframework.beans.factory.xml.BeanDefinitionParserDelegate.parseCustomElement(BeanDefinitionParserDelegate.java:1422) ~[?:?]
at org.springframework.beans.factory.xml.DefaultBeanDefinitionDocumentReader.parseBeanDefinitions(DefaultBeanDefinitionDocumentReader.java:187) ~[
at org.springframework.beans.factory.xml.DefaultBeanDefinitionDocumentReader.doRegisterBeanDefinitions(DefaultBeanDefinitionDocumentReader.java:14
at org.springframework.beans.factory.xml.DefaultBeanDefinitionDocumentReader.registerBeanDefinitions(DefaultBeanDefinitionDocumentReader.java:101)
at org.springframework.beans.factory.xml.XmlBeanDefinitionReader.registerBeanDefinitions(XmlBeanDefinitionReader.java:495) ~[?:?]
at org.springframework.beans.factory.xml.XmlBeanDefinitionReader.doLoadBeanDefinitions(XmlBeanDefinitionReader.java:391) ~[?:?]
... 18 more
```

the problem seems to be related with this jar:

cxf-rt-transports-http-3.1.14.jar

I found a post that explains the error here:

https://stackoverflow.com/questions/28719036/spring-namespacehandler-http-cxf-apache-org-transports-http-configuration

there seems to be a file inside the jar that maps the namespaces:

META-INF/spring.handlers

->
http\://cxf.apache.org/transports/http/configuration=org.apache.cxf.transport.http.spring.NamespaceHandler

I dont know how to make this part of the deployed kar file. (the jar is inside but it is not used on runtime)

I would be grateful with any idea / hint you could give me. As I am stuck with this since some time.