Virus Scan with SAP Process Integration Using Custom EJB Adapter Module

Applies to:

SAP Process Integration 7.0 and Above Versions. Custom Adapter Module, Virus Scan Adapter Module,

Virus Scan in SAP PI. For more information, visit the SOA Management homepage.

Summary

In many situation of file transfer we usually poll the files from different FTP locations and these FTP locations we maintain in the PI File Adapter. In some cases we need to check the file for any Viruses, if we are sending the file to a Business Critical target system. In SAP we have two standard scan features for upload and download program in ABAP and JAVA. But to scan the file at PI level before transferring the file content to Integration Server we can achieve this at PI Adapter level.

This Article will provide the detail description to use Virus Scan feature using SAP Process Integration and the SAP Certified Antivirus Products through Custom EJB module.

Author: Sugata Bagchi Majumder

Company: IBM Global Business Services, India.

Created on: 12 October 2010

Author Bio

Sugata B Majumder is a SAP Technical Consultant in IBM Global Business Services. He is working with IBM since 2007 and presently involves with multiple SAP implementation projects. His technical expertise includes SAP Process Integration, eSOA and JAVA/J2EE. He has experience of working in diverse SAP Landscapes and with different integrations, such as SRM, CRM an MDM along with various JMS, Third party systems and B2B Integrations.

Table of Contents

Introduction

In SAP PI for all of us a very common scenario is File to ABAP Proxy Scenario. This is one of the Basic scenario through which we can exchange message between two systems. We use generally a file adapter to pick up the files from the source location (usually a FTP location). In one of my project I faced a requirement to scan the files for any Virus Infection before sending it to PI Integration Engine. This module will provide this functionality if added to the specific adapter.

Assumption

This document is for an audience who has been working with SAP Netweaver Developer Studio (NWDS), know how to use the NWDS and perform the generic activities. The readers are expected to understand what an EJB (Enterprise JAVA Beans) is, Basic PI Adapter concepts and Experience in Developing Applications. This document will not explain all the associated concepts like EJB, VISUAL Admin activities, Deployment of Module etc from scratch or in detail.

Prerequisite

NWDS 7.0 has to be installed in the system.

Overview

We have a File to File Scenario as a Business Case. We need to perform the Virus Scan after picking up the file from Source FTP location (or NFS).Here we are using File Adapter for exchanging the messages.

Restriction

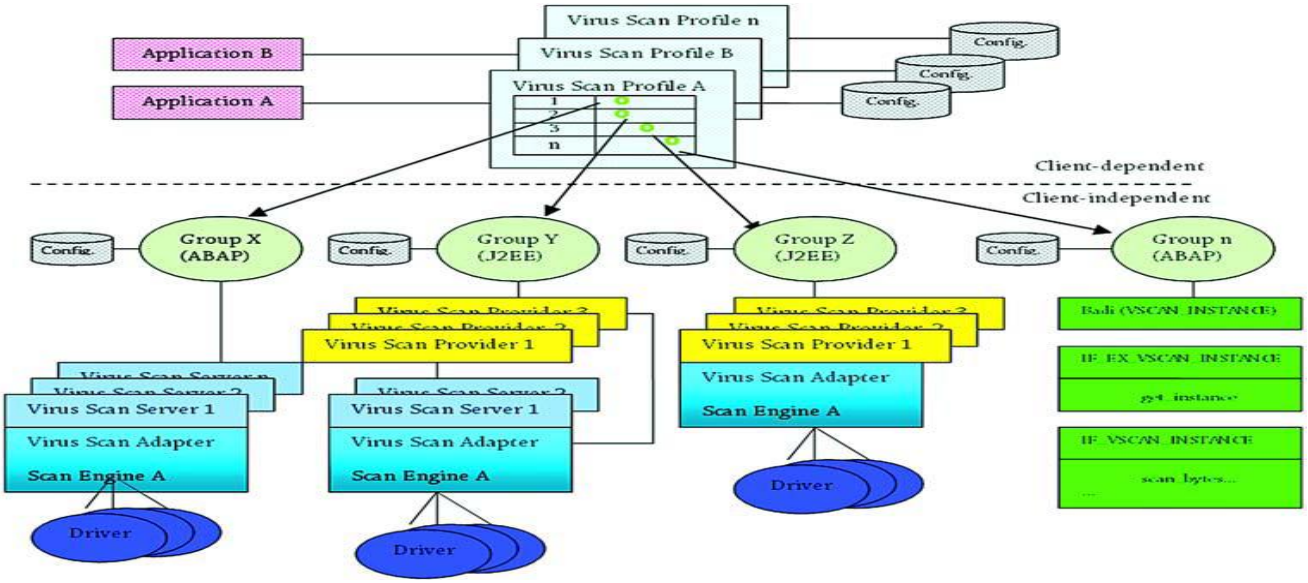
This module is prepared to work with File and JMS Adapters for single file transfer in one transaction. To use the Virus scan functionality in case Mail Adapter (Mail with Attachment) the EJB module needs to be changed a little bit. In case of Mail Adapter this EJB Module will be enhanced with the code to check the content of the attachment(s) and that is why code for attachment handling is also required.

For attachment the below code fragment can be added –

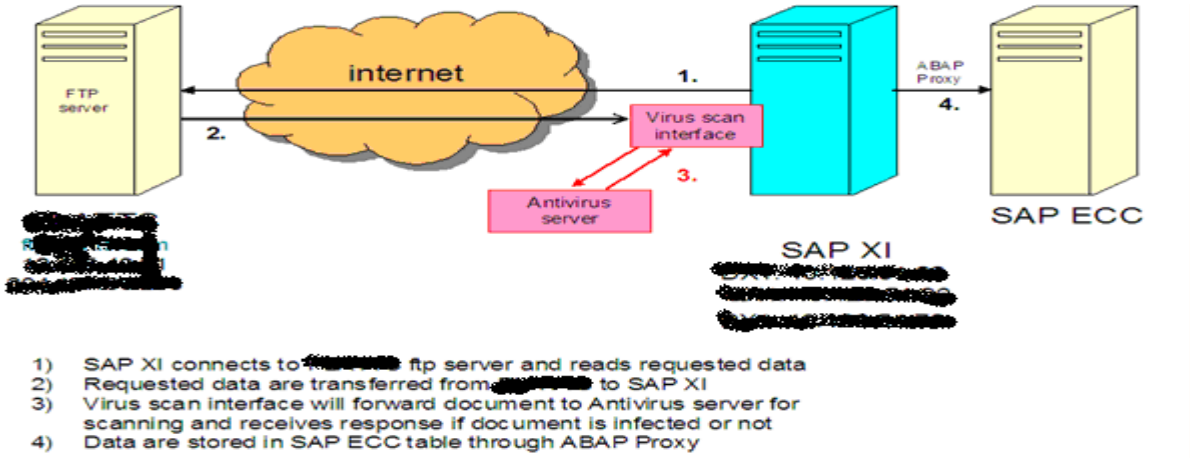
```
String attach = "MailAttachment-1";//+Integer.toString(i); if Handling single
attachment else //count the attachment and use loop to iterate
//int numOfAttachemtns= msg.countAttachments();
//int xmlflag=0;
//for(int i=1;i<=numOfAttachemtns;i++)
Payload pay= msg.getAttachment(attach);
byte [] paybyte= pay.getContent();// pass this byte Array to each scanByte() method
```

Architecture and Design of Landscape

Below is a representation of the Antivirus Product Specific Design-



The landscape for the business case scenario is mentioned below-



Step by Step Solution

AV Product Installation

This is the First step for using the Virus Scan facility. There are several Anti Virus Product in the Market any one of them can be used. These AV Products are SAP ICC (certified). E.g. - AVIRA, McAfee etc.

In my project we have used McAfee. The AV Product needs to be installed first in the same Application Server or some other windows server which can be accessed through RFC defined in the J2EE engine of SAP PI. Based on the choice one can use either Virus scan Adapter or Server. We have used Virus Scan Server, hence some configuration are needed.

Most of these are done by SAP BASIS team as it requires J2EE Visual Admin access. As a prerequisite the AV Product should be installed in some Windows / Application server.

Service Identification

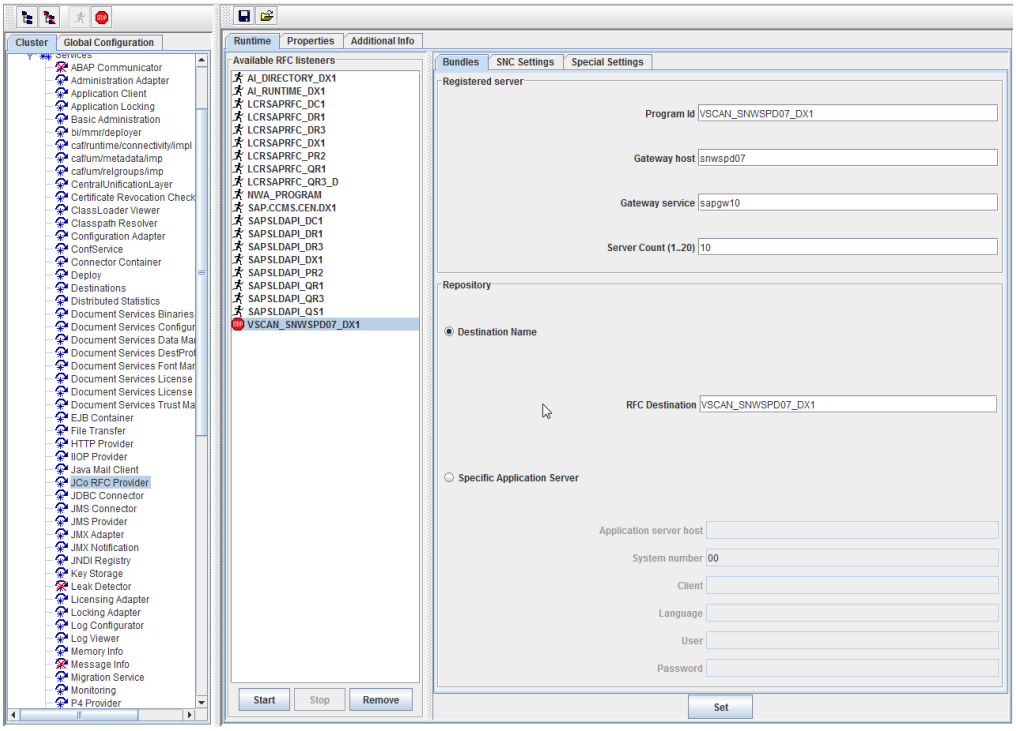
The AS Java service “Virus Scan Provider” (tc/sec/vsi/service) provides the interface “Virus Scan Interface”. The Java interface “Virus Scan Interface” (tc/sec/vsi/interface) provides the scan API for applications within the AS Java. The virus scanning is performed using an instance object previously obtained from the virus scan provider service. These services can be found in the J2EE Visual Admin as the JNDI path.

Visual Admin Set up

Please follow the screenshots below to setup the Virus Scan Provider –

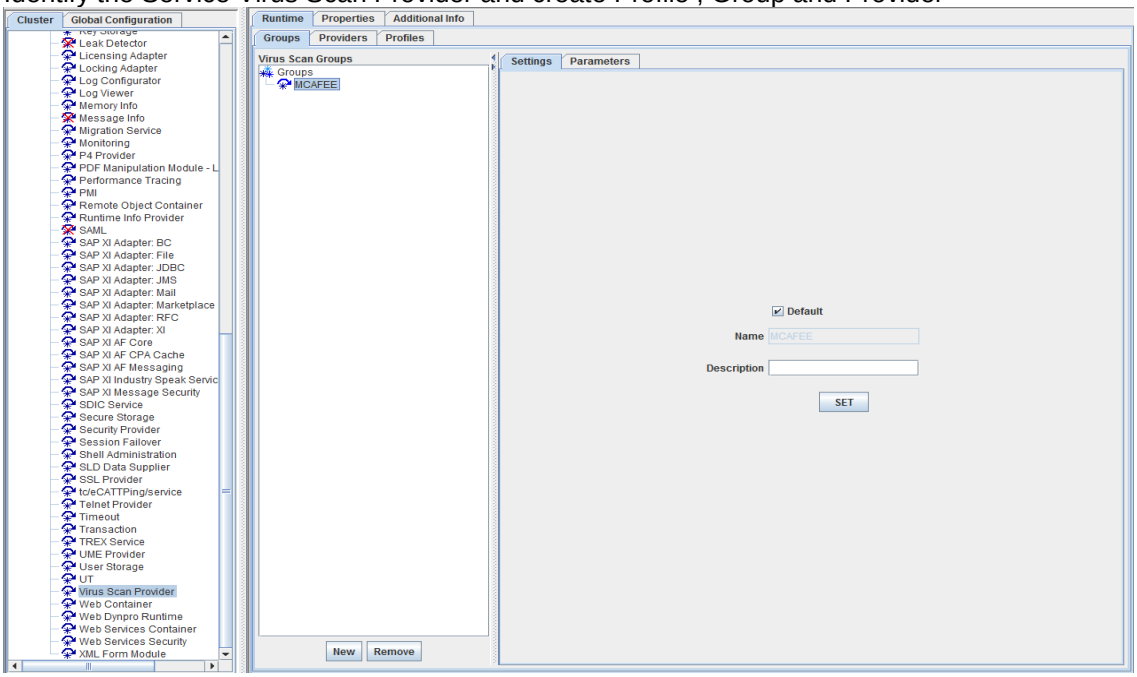
Step 1:

Create JCo-RFC Provider. Give the required details like – Program ID, RFC Destination, gateway Host and gateway Service. RFC Destination should start with VSCAN_.



Step 2:

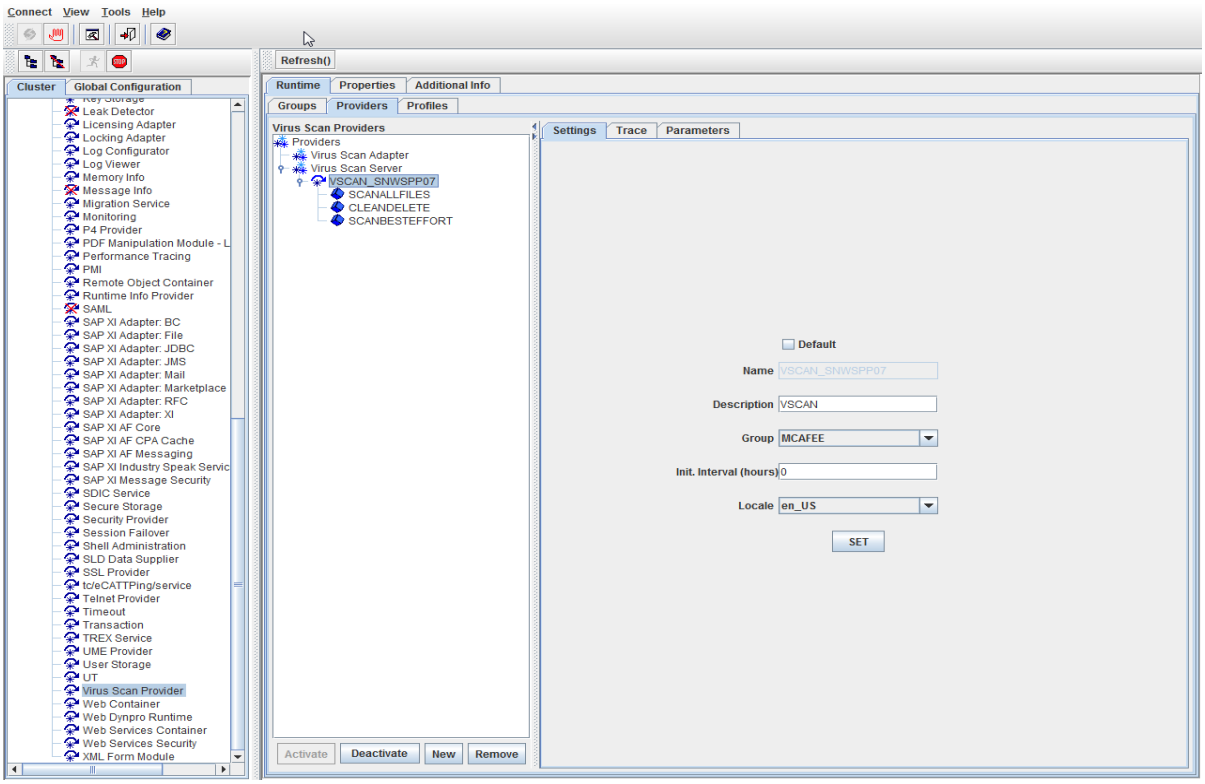
Identify the Service Virus Scan Provider and create Profile , Group and Provider-



Here a Group name is given as MCAFEE.

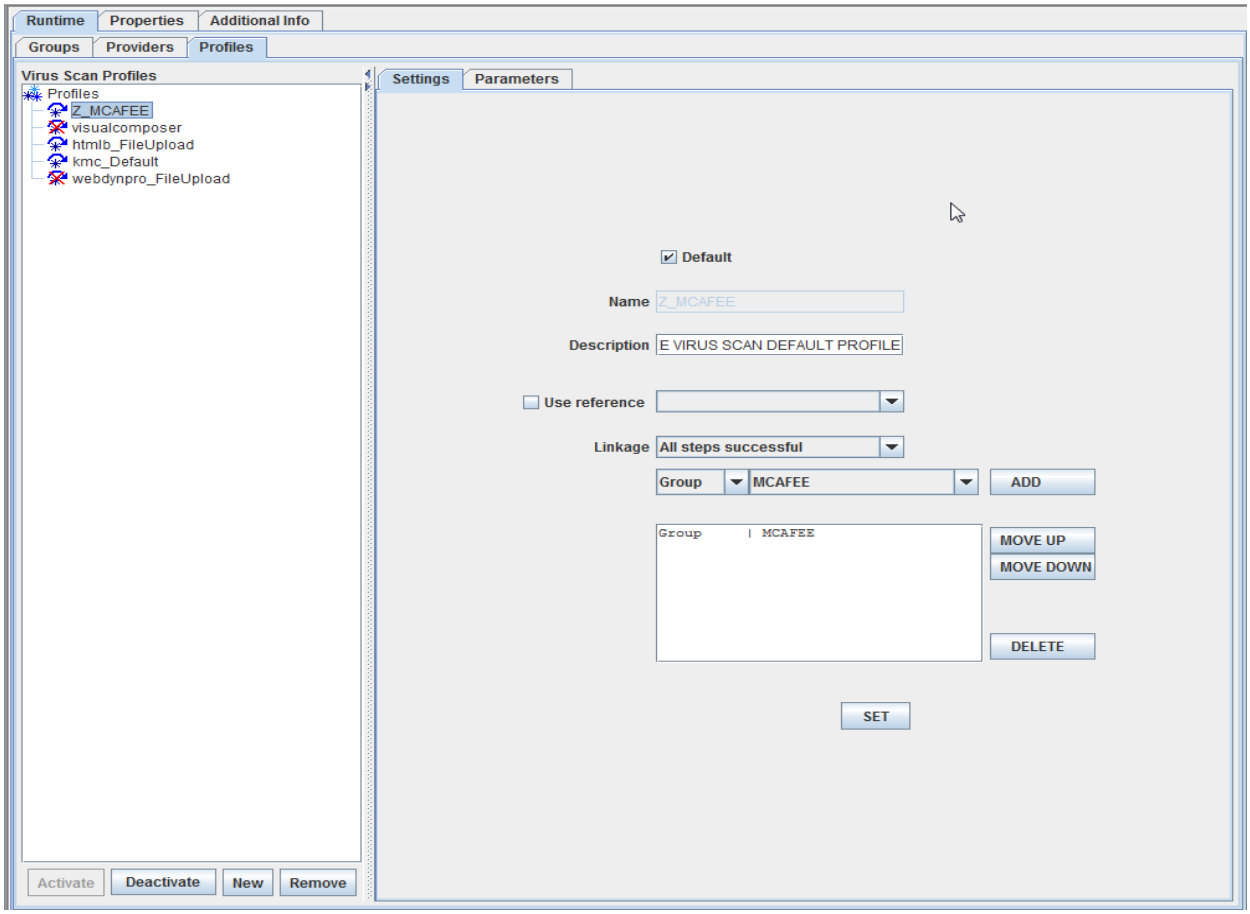
Step 3:

Identify the AV Provider . The AV Provider can be found in the JCo-RFC provider and that was created earlier in the first step.



Step 4:

Create one Profile which will be referred in the adapter module code-



Custom EJB Code for Adapter Module

To create the adapter module a EJB project should be created in NWDS, This module is written to work with SAP PI 7.0, hence required J2EE Library Changes and API Package Imports are needed to change if it is being used with higher version of PI. As per the requirement this module will read byte arrays from the incoming file and scan the same for viruses. If any viruses found it will throw adapter level exception and polling of file will be stopped. The infected file name with full source path will also be shown in the adapter level. I have used - NWDS Version: 7.0.22 Build id: 201008181154.

While writing the module one must refer the below JARS in the NWDS project build path along with the others-

```
tc_sec_vsi_service.jar;
tc_sec_vsi_interface.jar;
tc_sec_ssf.jar;
ejb20.jar;
exception. jar
```

*XIScan.java

```
/**
 * Created on 09/11/2010
 * Author Sugata B Majumder
 * NWDS Version: 7.0.22 Build id: 201008181154
 */
// Interface Object - MM20-009, FMS Fuel Prices
// Technical Object Owner - Sugata B Majumder
package com.usermodule;
import java.rmi.RemoteException;
import java.util.Hashtable;
import javax.ejb.EJBException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;
//import javax.ejb.TimedObject;
//import javax.ejb.Timer;
import com.sap.aii.af.ra.cci.XIDeliveryException;
//import com.sap.aii.af.mp.module.*;
import com.sap.aii.af.ra.ms.api.*;
//import com.sap.aii.af.service.auditlog.*;
//import com.sap.aii.af.mp.module.Module;
import com.sap.aii.af.mp.module.ModuleContext;
import com.sap.aii.af.mp.module.ModuleData;
import com.sap.aii.af.mp.module.ModuleException;
import com.sap.aii.af.ra.ms.api.Message;
//import com.sap.aii.af.ra.ms.api.MessageDirection;
import com.sap.aii.af.service.auditlog.Audit;
import com.sap.aii.af.service.auditlog.AuditDirection;
import com.sap.aii.af.service.auditlog.AuditLogStatus;
import com.sap.aii.af.service.auditlog.AuditMessageKey;
import javax.naming.*;
import com.sap.aii.af.ra.ms.api.TextPayload;
import com.sap.security.core.server.vsi.api.*;
import com.sap.security.core.server.vsi.api.exception.*;
import com.sap.security.core.server.vsi.api.exception.VirusInfectionException;
//import java.lang.Throwable;
```

```
 */
public class XIScan implements SessionBean {
    /**
    *
    */
    private static final long serialVersionUID = 3L;

    /* (non-Javadoc)
     * @see javax.ejb.SessionBean#ejbActivate()
     */
    public void ejbActivate() throws EJBException, RemoteException {
        // TODO Auto-generated method stub

    }

    /* (non-Javadoc)
     * @see javax.ejb.SessionBean#ejbPassivate()
     */
    public void ejbPassivate() throws EJBException, RemoteException {
        // TODO Auto-generated method stub

    }

    /* (non-Javadoc)
     * @see javax.ejb.SessionBean#ejbRemove()
     */
    public void ejbRemove() throws EJBException, RemoteException {
        // TODO Auto-generated method stub

    }
```

```
/* (non-Javadoc)
 * @see javax.ejb.TimedObject#ejbTimeout(javax.ejb.Timer)
 */
//public void ejbTimeout(Timer arg0) {
// TODO Auto-generated method stub

//}

public void ejbCreate() throws javax.ejb.CreateException {

}

public ModuleData process(
    ModuleContext moduleContext,
    ModuleData inputModuleData)
    throws ModuleException {

    try {
        Hashtable mp =
            (Hashtable) inputModuleData.getSupplementalData(
                "module.parameters");
        String fileName = (String) mp.get("FileName");
        Message msg = (Message) inputModuleData.getPrincipalData();
        AuditMessageKey amk =
            new AuditMessageKey(msg.getMessageId(), AuditDirection.INBOUND);
        Audit.addAuditLogEntry(
            amk,
            AuditLogStatus.SUCCESS,
            "XIScan: Module called");
        Payload payload = msg.getDocument();
        byte[] scancontent = payload.getContent();
        //TextPayload txtPayload=msg.getDocument();
        //String txt_str=txtPayload.getText();
        //Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, "Payload Conversion to String: Done");
```

```
        Context ctx = new InitialContext();
        VSIService vsiService =
            (VSIService) ctx.lookup(VSIService.JNDI_NAME);
        if (vsiService != null) {
            Instance vsiInstance = vsiService.getInstance("Z_MCAFEE");
            Instance vsiInstance1 = vsiService.getInstance("Z_MCAFEE");
            if (vsiInstance != null || vsiInstance1 != null) {
                Audit.addAuditLogEntry(
                    amk,
                    AuditLogStatus.SUCCESS,
                    "VSI Instances are Created: Done");
                try {
                    Audit.addAuditLogEntry(
                        amk,
                        AuditLogStatus.SUCCESS,
                        "XISBM Scan Started");
                    boolean b1 = vsiInstance.scanBytes(scancontent);
                    if (b1 == true) {
                        Audit.addAuditLogEntry(
                            amk,
                            AuditLogStatus.SUCCESS,
                            "No VIRUS Found");
                    }
                } catch (VirusInfectionException vie) { //Infection[] virusInfections = vie.getInfections();
                    //if (virusInfections.length!=0)
                    { //System.exit(3);
                        Audit.addAuditLogEntry(
                            amk,
                            AuditLogStatus.ERROR,
                            "VIRUS Found" + vie.getLastErrorRC());
                        XIDeliveryException adpterexp =
                            new XIDeliveryException(
                                "Virus found: Exiting Module, Please DELETE the FILE Named:"
                                    + fileName);
                        throw adpterexp;
```

```
                        }
                    } finally {
                        vsiService.releaseInstance(vsiInstance);
                        vsiService.releaseInstance(vsiInstance1);
                    }
                }
            }
            Message msg1 = (Message) inputModuleData.getPrincipalData();
            inputModuleData.setPrincipalData(msg1);
            return inputModuleData;
        } catch (Exception e) {
            ModuleException me = new ModuleException(e);
            throw me;
        } finally {
            System.gc();
        }
    }
}
```

Please find the entire code given below-

```
 /**
 *
 */
package com.usermodule;
import java.rmi.RemoteException;
import java.util.Hashtable;
import javax.ejb.EJBException;
import javax.ejb.SessionBean;
import javax.ejb.SessionContext;
//import javax.ejb.TimedObject;
//import javax.ejb.Timer;
import com.sap.aii.af.ra.cci.XIDeliveryException;
//import com.sap.aii.af.mp.module.*;
import com.sap.aii.af.ra.ms.api.*;
//import com.sap.aii.af.service.auditlog.*;
//import com.sap.aii.af.mp.module.Module;
import com.sap.aii.af.mp.module.ModuleContext;
import com.sap.aii.af.mp.module.ModuleData;
import com.sap.aii.af.mp.module.ModuleException;
import com.sap.aii.af.ra.ms.api.Message;
//import com.sap.aii.af.ra.ms.api.MessageDirection;
import com.sap.aii.af.service.auditlog.Audit;
import com.sap.aii.af.service.auditlog.AuditDirection;
import com.sap.aii.af.service.auditlog.AuditLogStatus;
import com.sap.aii.af.service.auditlog.AuditMessageKey;
import javax.naming.*;
import com.sap.aii.af.ra.ms.api.TextPayload;
import com.sap.security.core.server.vsi.api.*;
import com.sap.security.core.server.vsi.api.exception.*;
import com.sap.security.core.server.vsi.api.exception.VirusInfectionException;
//import java.lang.Throwable;


/**
```

```
 * @author Sugata B Majumder //NWDS CE 7.1 SP05
 *
 */
public class XIScan implements SessionBean{
        /**
    *
    */
   private static final long serialVersionUID = 3L;

   /* (non-Javadoc)
    * @see javax.ejb.SessionBean#ejbActivate()
    */
   public void ejbActivate() throws EJBException, RemoteException {
        // TODO Auto-generated method stub

   /* (non-Javadoc)
    * @see javax.ejb.SessionBean#ejbPassivate()
    */
   public void ejbPassivate() throws EJBException, RemoteException {
        // TODO Auto-generated method stub

   /* (non-Javadoc)
    * @see javax.ejb.SessionBean#ejbRemove()
    */
   public void ejbRemove() throws EJBException, RemoteException {
        // TODO Auto-generated method stub

   /* (non-Javadoc)
    * @see javax.ejb.SessionBean#setSessionContext(javax.ejb.SessionContext)
    */
   public void setSessionContext(SessionContext arg0) throws EJBException,
              RemoteException {
        // TODO Auto-generated method stub


   /* (non-Javadoc)
    * @see javax.ejb.TimedObject#ejbTimeout(javax.ejb.Timer)
    */
   //public void ejbTimeout(Timer arg0) {
        // TODO Auto-generated method stub


   public void ejbCreate() throws javax.ejb.CreateException {

   public ModuleData process(ModuleContext moduleContext,ModuleData
inputModuleData)throws ModuleException
        try
        {   Hashtable mp =(Hashtable)
inputModuleData.getSupplementalData("module.parameters");
              String fileName = (String) mp.get("FileName");
              Message msg = (Message)inputModuleData.getPrincipalData();
              AuditMessageKey amk = new AuditMessageKey(msg.getMessageId(),
AuditDirection.INBOUND);
              Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, "XIScan: Module
called");
              Payload payload = msg.getDocument();
```

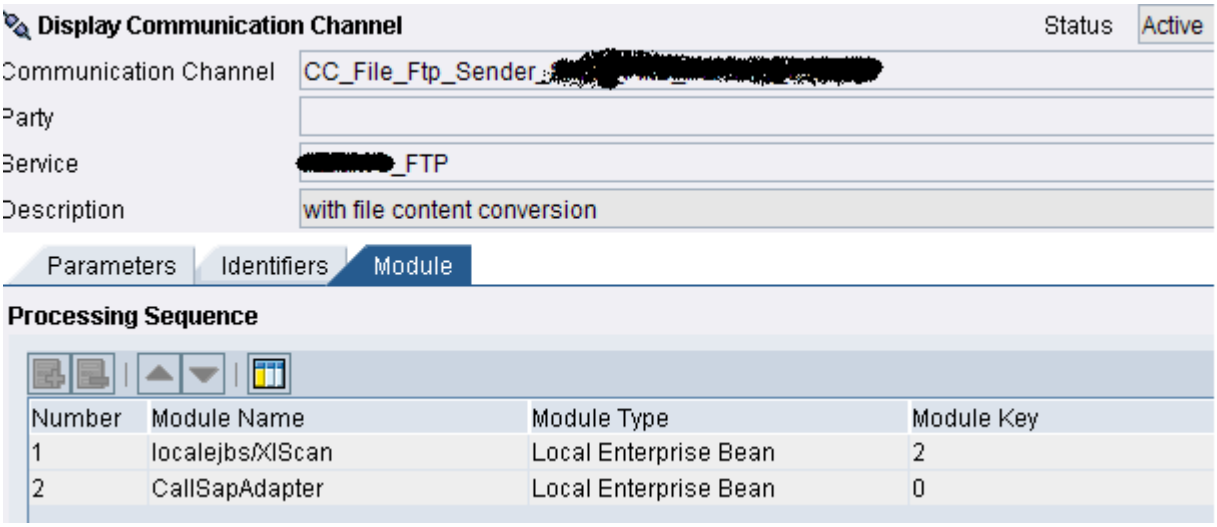
```
                byte[] scancontent = payload.getContent();
                //TextPayload txtPayload=msg.getDocument();
                //String txt_str=txtPayload.getText();
                //Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, "Payload
Conversion to String: Done");
                //byte[] scaninputbytes = txt_str.getBytes();
                //Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, "Bytes
calculation to the payload: Done");
                Context ctx = new InitialContext();
                VSIService vsiService = (VSIService)ctx.lookup(VSIService.JNDI_NAME);
                if (vsiService != null)
                 { Instance vsiInstance = vsiService.getInstance("Z_MCAFEE");
                   Instance vsiInstance1 = vsiService.getInstance("Z_MCAFEE");
                   if (vsiInstance != null || vsiInstance1 !=null)
                   {Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, "VSI
Instances are Created: Done");
                  try
                        {
                        Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, "XISBM
Scan Started");
                     boolean b1=vsiInstance.scanBytes(scancontent);
                     if(b1==true)
                     {Audit.addAuditLogEntry(amk, AuditLogStatus.SUCCESS, "No VIRUS
Found");

                 catch (VirusInfectionException vie)
                              { //Infection[] virusInfections =
vie.getInfections();
                                    //if (virusInfections.length!=0)
                                    {  //System.exit(3);
                                          Audit.addAuditLogEntry(amk,
AuditLogStatus.ERROR, "VIRUS Found"+vie.getLastErrorRC());
XIDeliveryException adpterexp = new XIDeliveryException("Virus found: Exiting Module,
Please DELETE the FILE Named:"+fileName);
                                          throw adpterexp;
                                    //boolean b2=vsiInstance1.cleanFile(fileName);

                                    //if(b2!=true)
                                //{
                                   //Audit.addAuditLogEntry(amk,
AuditLogStatus.ERROR, "VIRUS Found"+vie.getLastErrorRC());
                  finally {
                        vsiService.releaseInstance(vsiInstance);
                        vsiService.releaseInstance(vsiInstance1);
                Message msg1 = (Message)inputModuleData.getPrincipalData();
                inputModuleData.setPrincipalData(msg1);
                return inputModuleData;
          }
                catch(Exception e)
                 {
                ModuleException me = new ModuleException(e);
                throw me;
                }
                  finally {
                              System.gc();
```

Testing Scenario

Once the Module is deployed in the J2EE Engine, use the same in the adapter as below-



For testing SAP has provided one sample virus – EICAR. Once the virus file is picked up by the SAP PI it will throw the error at adapter level and the message processing will stop.

Below the error it will show in the adapter –

Related Content

Adapter Module Development

Avira Antivirus Specification

Avira Manual

For more information, visit the SOA Management homepage

Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.