May 27, 2003

DevNote #102 – Testing HTTPS with the SIFWorks® Enterprise ZIS

The Schools Interoperability Framework 1.1 standard requires that all agents be able to communicate with zone integration servers over the HTTPS transport protocol. The SIFWorks® Agent Developer Kit (ADK™) for Java offers both HTTP and HTTPS implementations that are easily selected by changing an agent's properties at runtime. This Developer Note shows how to set up an HTTPS test environment with the SIFWorks Enterprise ZIS and the GetZoneStatus example agent.

The following software is assumed:

- Java 2 Standard Edition SDK version 1.4 or later
- SIFWorks Enterprise ZIS 0.9.19 or later
- SIFWorks ADK 1.0.0 Beta 8 or later

HTTPS Certificates

1. Creating a Keystore and an HTTPS Certificate for the ZIS

Agents that connect to the ZIS over an HTTPS connection expect a valid certificate to be presented by the server. This is done as part of the normal handshaking process performed when the agent and server initiate a connection. When SIFWorks establishes an HTTPS socket to listen for incoming traffic, it reads its key material and certificate from a Java keystore file located in the SIFWorks `\certs` directory named `Server.ks`. By default this file does not exist, so it must be created and a valid certificate imported into it.

There are several ways to obtain a certificate. You can purchase one from a third-party Certificate Authority (CA) such as VeriSign; you can establish your own Certificate Authority using software from Microsoft, Netscape, and others; or you can create a "self-signed" certificate using the `keytool` program included with Java. This Developer Note assumes you will be creating your own self-signed certificate. Follow the steps below to create a keystore with a private key and self-signed public key certificate for the zone integration server.

1. Open a command prompt to the SIFWorks `\certs` directory

2. Execute the following line to create a private key in a new keystore file named `Server.ks`

```
keytool –genkey –v –keystore Server.ks –alias SIFWorks –keyalg RSA –keysize 1024
```

3. When prompted for a password, enter "changeit"

```
Enter keystore password: changeit
```

4. When prompted for First and Last name, enter the IP address of the SIFWorks server:

```
What is your first and last name?
  [unknown]: localhost
```

5. Fill in the remainder of the fields:

```
What is the name of your organizational unit?
  [unknown]: SIFWorks ZIS
```

```
What is the name of your organization?
  [unknown]: Edustructures
What is the name of your City or Locality?
  [unknown]: Salt Lake City
What is the name of your State or Province?
  [unknown]: UT
What is the two-letter country code for this unit?
  [unknown]: US
```

6. When prompted for a password, press Enter to use the same password as the keystore

7. The `Server.ks` keystore should now be created and should contain a private key and self-signed public key certificate. You can view the contents of the keystore by executing the command "`keytool –list –v –keystore Server.ks`".

8. Next, export the public key certificate to a file by executing the following command:

```
keytool –export –keystore Server.ks –alias SIFWorks –file ZIS.cer
```

At this point we have successfully created the `Server.ks` keystore with a private key and self-signed public key certificate for the ZIS, and have exported the certificate to a file named `ZIS.cer`. In Step 3 below we'll import this certificate into the SIFWorks keystore so it will be presented to agents that establish HTTPS connections with the server.

2. Creating an HTTPS Certificate for the GetZoneStatus Agent

Just as SIFWorks must present a valid certificate to agents that connect to it, agents must also present a valid certificate to the server. Follow the steps below to create a private key and self-signed public key certificate for the ADK's GetZoneStatus example agent. The key material will be stored in a file named `Agent.ks` in the agent's directory.

1. Open a command prompt to the ADK's `\examples\GetZoneStatus` directory

2. Execute the following line to create a private key and self-signed certificate in the `Agent.ks` keystore:

```
keytool –genkey –v –keystore Agent.ks –alias GetZoneStatus –keyalg RSA –keysize 1024
```

3. When prompted for a password, enter "changeit"

```
Enter keystore password: changeit
```

4. When prompted for First and Last name, enter the IP address of the computer where the agent will run:

```
What is your first and last name?
  [unknown]: localhost
```

5. Fill in the remainder of the fields:

```
What is the name of your organizational unit?
  [unknown]: GetZoneStatus Agent
What is the name of your organization?
  [unknown]: Edustructures
What is the name of your City or Locality?
  [unknown]: Salt Lake City
What is the name of your State or Province?
  [unknown]: UT
What is the two-letter country code for this unit?
  [unknown]: US
```

6. When prompted for a password, press Enter to use the same password as the keystore

7. The `Agent.ks` keystore should now be created and should contain a private key and self-signed public key certificate. You can view the contents of the keystore by executing the command "`keytool –list –v –keystore Agent.ks`".

8. Next, export the public key certificate to a file by executing the following command:

```
keytool –export –keystore Agent.ks –alias GetZoneStatus –file GetZoneStatus.cer
```

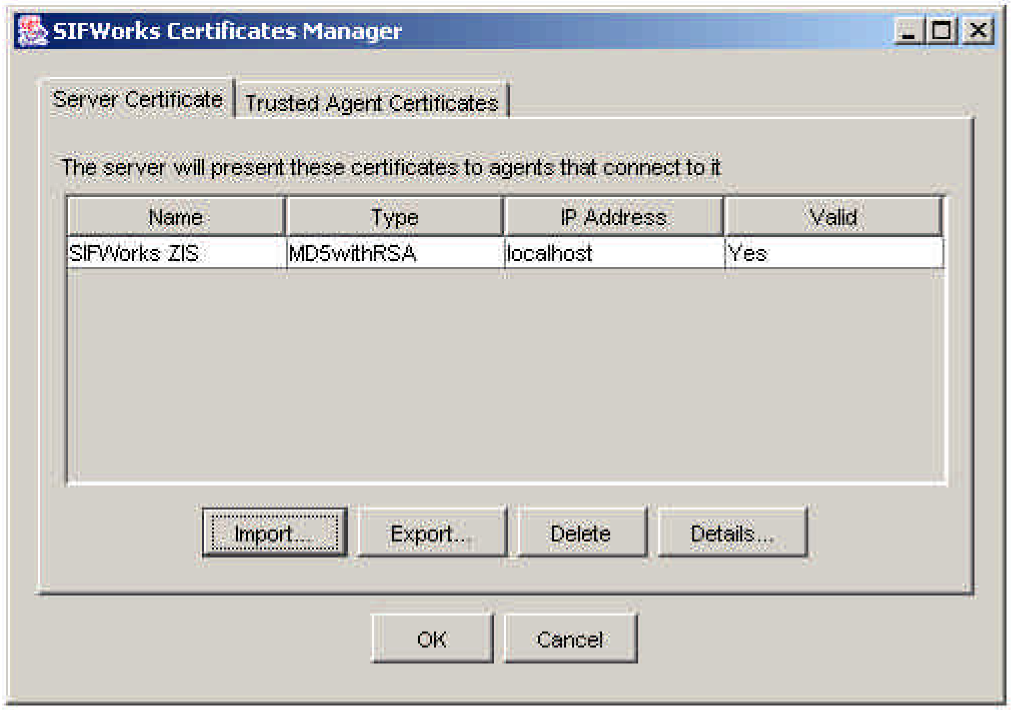
9. At this point we have successfully created the `Agent.ks` keystore with a private key and self-signed public key certificate for the GetZoneStatus agent, and have exported the certificate to a file named `GetZoneStatus.cer`. In Step 4 below we'll import this certificate into the SIFWorks list of trusted certificates so it will be trusted by the ZIS when the agent establishes a connection.

3. Import the ZIS Certificate into the SIFWorks Keystore

The SIFWorks keystore file—`Server.ks`—was created in Step 1 with a private key and a self-signed public key certificate was exported to the `ZIS.cer` file. In this step we'll use the SIFWorks Certificates Manager to import that file into the server's keystore as a *trusted* certificate. Note if you obtained your certificate from a third-party CA, you would follow this same procedure to import it into the SIFWorks keystore. Agents that connect to the zone integration server will be presented with this certificate.

1. Run the `certsmgr.cmd` batch file from the SIFWorks directory

2. Click the Import button and browse to the `ZIS.cer` file created in Step 1.



The server will present these certificates to agents that connect to it

Name	Type	IP Address	Valid
SIFWorks ZIS	MD5withRSA	localhost	Yes

Leave the Certificates Manager running as we'll use it in the next step.

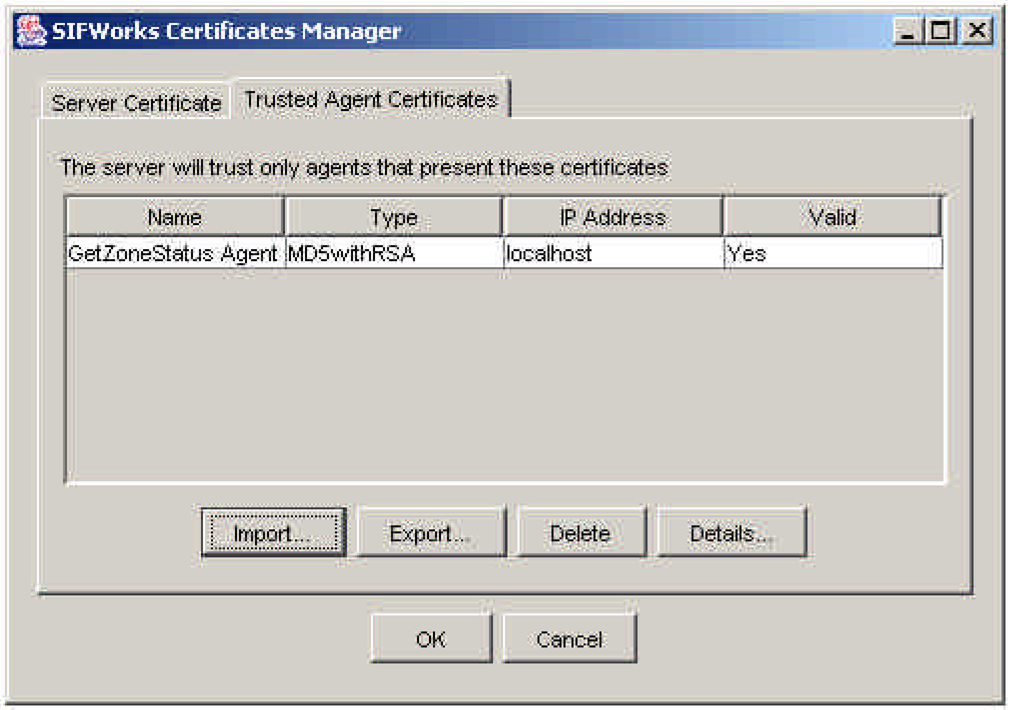
4. Import the Agent Certificate into the SIFWorks Truststore

Now that both the ZIS and agent have a private key and self-signed public key certificate, it's time to configure each to *trust* one another's certificates. With SIFWorks, this is done by using the Certificates Manager to import the agent's certificate into the `Trusted.ks` "truststore" located in the `\certs` directory (the file is automatically created by the Certificates Manager if it doesn't exist.)

1. Run the `certsmgr.cmd` batch file from the SIFWorks directory if not already running

2. Click on the second tab, "Trusted Agent Certificates"

3. Click the Import button and browse to the `GetZoneStatus.cer` file created in Step 2.



The server will trust only agents that present these certificates

Name	Type	IP Address	Valid
GetZoneStatus Agent	MD5withRSA	localhost	Yes

4. Click OK to close the Certificates Manager

5. Import the ZIS Certificate into the Agent's Truststore

In the previous step, we configured SIFWorks to *trust* the agent's certificate. It's now time to configure the agent to trust SIFWorks:

1. Copy the `ZIS.cer` file to the ADK's `\examples\GetZoneStatus` directory

2. Open a command prompt to the ADK's `\examples\GetZoneStatus` directory

3. Execute the following command to create a truststore named `Trusted.ks` and import the `ZIS.cer` certificate into that truststore:

```
keytool –import –v –keystore Trusted.ks –alias SIFWorks –file ZIS.cer
```

4. When prompted, enter a password of "changeit"

```
Enter keystore password: changeit
```

5. When asked if the certificate should be trusted, type "yes"

```
Trust this certificate? [no]: yes
```

Running SIFWorks & GetZoneStatus

1. Configure and Start the SIFWorks ZIS

Follow these steps to configure and start the SIFWorks ZIS:

1. If SIFWorks if already running, stop it

2. SIFWorks 0.9.19 ships with a configuration file that does not reference the same keystore and truststore filenames as the Certificates Manager used in this document. Before proceeding, SIFWorks's `conf\zis.xml` file needs to be updated so the names of these files are correct. Open the file in a text editor such as Notepad and search for the text "`Trusted.cer`". Change the file extension from "`.cer`" to "`.ks`". Similarly, search for `Server.cer` and change its extension to "`.ks`".

3. Start SIFWorks

4. Open a web browser and navigate to the Console at `http://localhost:7003/index.jsp`

5. Login to the SIFWorks Console by typing your login name and password

6. Navigate to the `Settings > Transports > HTTPS` node in the tree. Verify that the HTTPS transport is enabled on port 7443 and is listed as Active. (NOTE: If the word "Active" is displayed in red, one or more errors have occurred. Click the text to view the error messages.)

Once you've verified that an HTTPS transport is enabled, create a new zone for testing:

7. Create a new zone named "`TEST`".

8. Click on the Zone Settings tab. Under Messaging Options, enable "Can Request SIF_ZoneStatus". The GetZoneStatus example agent requires this permission.

9. Add the `GetZoneStatus` agent to the `TEST` zone.

2. Compile and Run the GetZoneStatus Example Agent

Follow these steps to compile and run the ADK's GetZoneStatus example agent:

1. Open a command prompt to the ADK's `\examples\GetZoneStatus` directory

2. Run the `compile.cmd` batch file to compile the agent

3. Create a response file from which the agent will read its command-line arguments. This is necessary when running the example agents in HTTPS mode because including each of the necessary arguments on a single command-line would exceed the maximum of 9 arguments that the `run.bat` batch file is prepared to accept. By using a response file, you can include an unlimited number of the arguments to the GetZoneStatus program. (An added benefit is that you don't have to retype the command-line each time you run the agent.)

   Use Notepad to create a file named `https.txt` with the following lines

   ```
   /zone TEST
   /url https://localhost:7443/TEST
   /https
   /port 15000
   /keystore Agent.ks
   /truststore Trusted.ks
   ```

4. Run the agent by executing the following command:

   ```
   run https.txt
   ```

5. You should see output similar to the following:

   ```
   Reading command-line from https.txt:
   /zone TEST /url https://localhost:7443/TEST /https /port 15000 /keystore Agent.ks
   /truststore Trusted.ks

   Initializing agent...
   Connecting to zone "TEST" (http://localhost:7443/TEST)...
   ```

```
2003-05-27 12:04:58.045 [ADK.Agent$TEST] Using keystore: path\Agent.ks
2003-05-27 12:04:58.045 [ADK.Agent$TEST] Using default Java keystore password
'changeit'
2003-05-27 12:04:58.045 [ADK.Agent$TEST] Using truststore: path\Trusted.ks
2003-05-27 12:04:58.045 [ADK.Agent$TEST] Using default Java truststore password
'changeit'
Requesting SIF_ZoneStatus object...
-The agent displays a SIF_ZoneStatus object here-
```

How to Enable HTTPS in an ADK Agent

Because HTTP is so much easier to use during development and initial deployment of SIF Agent software, the default transport protocol enabled by the ADK is HTTP instead of HTTPS. Turning on HTTPS support is as easy as setting a few agent properties in your code at agent startup time. In a commercial agent, these properties are usually read from a configuration file or a database.

Enabling HTTPS for a Push mode agent involves setting up HTTPS-specific properties in your agent's `AgentProperties` object. For an example, take a look at the `parseCL` function of the `ADKExamples.java` source included with any of the ADK example agents like GetZoneStatus. This should be done at agent startup time before the agent connects to any zones.

Follow these steps in the startup code of your agent to enable HTTPS:

1. Obtain the default `HttpsProperties` object from your `Agent` object:

   ```
   // Get the Agent's default HttpsProperties object
   HttpsProperties props = myAgent.getDefaultHttpsProperties();
   ```

2. Call any of the following `HttpsProperties` class methods to set HTTPS properties:

   ```
   void setKeyStore( String file );
   void setKeyStorePassword( String pwd );
   void setTrustStore( String file );
   void setTrustStorePassword( String pwd );
   void setRequireClientAuth( boolean enable );
   ```

   For example,

   ```
   // Set the keystore and truststore properties
   props.setKeyStore( "Agent.ks" );
   props.setKeyStorePassword( "changeit" );
   props.setTrustStore( "Trusted.ks" );
   props.setTrustStorePassword( "changeit" );

   // Require client authentication
   props.setRequireClientAuth( true );
   ```

   Note if you do not explicitly set the keystore and truststore properties, the ADK uses the Java defaults. Refer to the Java JSSE documentation for more information on the location and contents of the default keystore and truststore.

3. Call the `HttpsProperties.setPort` method to assign a port number for the listening socket. This is the port your Push-mode agent will listen on for incoming traffic from the ZIS. If the agent will run in Pull mode, this step is not necessary.

   ```
   props.setPort( 15000 );
   ```

4. Finally, call the `AgentProperties.setTransportProtocol` method to inform the agent that it should use HTTPS instead of HTTP as its transport protocol:

   ```
   props.setTransportProtocol( "https" );
   ```

That's all there is to it. When you start up your agent it will establish an HTTPS listening socket instead of an HTTP socket.