Ports, AS2 and Acknowledgements

Chapter 5

The actual receipt and delivery of files is central to any solution. The most common approaches to receiving and sending data within the EDI Health Care space include SFTP, encrypted data over standard FTP, and secure communications over AS2. SFTP is a simple custom adapter setup and configuration. Dealing with encrypting and decrypting data over FTP requires custom code. Configuring AS2 for direct party-to-party communication requires certificates and complex configurations within BizTalk. This chapter details the set up and configuration for each of these methods, as well as how to successfully deliver various forms of EDI acknowledgements.

SFTP

SFTP is an excellent option for exchanging health care EDI documents, as it is HIPAA compliant and easy to implement. SFTP allows documents to be sent and received in plain text, as the protocol itself encrypts the information (with standard FTP, encrypting the file itself is required to be HIPAA compliant). There are several options for sending and receiving data via SFTP in BizTalk Server, but the recommended approach is to use the third-party bLogical BizTalk SFTP adapter (Blogical.Shared.Adapters.SFTP) available from CodePlex. This adapter is available free of charge and is very reliable. It can be downloaded, compiled, and made available within BizTalk Server within a few minutes.

Note The SFTP adapter automatically downloads the original host certificate from the party you are interacting with. However, if this certificate expires (which is common), the SFTP adapter won't automatically be able to download the new certificate. If you get an exception in the Windows Event Log that says "HostKey does not match previously retrieved HostKey" you will need to browse to the sftphostfiles.config file and delete the HostKey setting. The directory where this file is located is in the Local Settings of the host user that the SFTP adapter runs under. For example, if your BizTalk Host instance runs under DOMAIN\Host_User_Account, then you will browse to Host_User_Account\Local Settings. The config file is buried under a unique directory several levels below, so you need to run a search for it once you have located this directory.

Configuring the SFTP Adapter

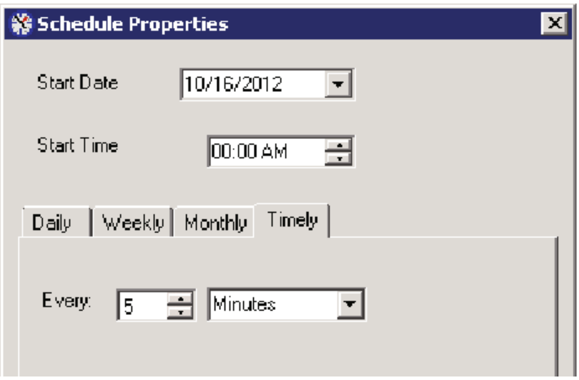
After the adapter has been installed, setting it up to receive and send data can be done by creating a new receive location or send port and setting the Type property to SFTP (or what it was named during the installation). You can then click Configure. Configure the SFTP adapter with the key fields shown in this section. In some cases, you will need to configure additional fields than what is shown here, but in most cases these are all that is required.

Some of the properties listed (such as the Schedule) are unique to the receiving of data. As you configure your SFTP send port or receive location, you'll be able to easily identify which properties apply.

Note Before you configure your SFTP adapter, be sure and test connectivity to the target SFTP site through a standard SFTP-compatible FTP utility (one excellent option is FileZilla, which has support for an array of FTP and SFTP connection types). There are a number of things that may require attention before you can connect successfully, and it is much easier to troubleshoot using a client utility than it is through the BizTalk SFTP adapter.

Schedule Property

This setting has some robust functionality for determining the schedule for querying the source SFTP site. Clicking the ellipsis on this property pops up an interface that allows for scheduling on Daily, Weekly, Monthly, or Timely intervals. You will most likely use the timely interval, every *x* number of minutes, for example. In Figure 5-1, you see the property set to poll the source SFTP site every five minutes. In most cases, you pull your EDI data on regular intervals throughout the day, but you need to coordinate with your trading partner to determine whether there are any scheduling windows that should be avoided.


Schedule Properties

Start Date: 10/16/2012

Start Time: 00:00 AM

Every: 5 Minutes

Figure 5-1. Setting the schedule property for Timely polling interval

After Get Property

This property defaults to Delete, which ensures that the file being retrieved is removed from the source SFTP site as soon as it has been successfully received by BizTalk. If there is an error in transmission, the file will remain on the server. In most cases, you will want to leave this set to Delete, but some trading partners provide archiving of data after a certain period of time, so you may want to leave the file on the server to take advantage of this. If this is not set to Delete, you will need to be sure your polling interval set in the Schedule property does not cause this same file to be retrieved multiple times before it is auto archived by the trading partner.

SSH Error Threshold Property

The SSH Error Threshold property can be used to control how many errors can be encountered before the adapter shuts down. It is fairly common to have connectivity issues with SFTP sites, and it would make sense to increase this error threshold to a sizeable amount to account for this. If left at a low number, the adapter may shut down if the source site cannot be reached over a certain period of time.

Note If the SFTP adapter encounters errors, the exceptions will be logged to the Windows Event Viewer. Be sure and monitor the state of your SFTP ports, as they will automatically shut down if the error thresholds are exceeded.

SSH Host Property

This property should be set with the actual SFTP server host address. This could be an IP or a named server. It should only contain the root server name, not any subfolders. It should also not contain sftp://. An example of this property set to an IP would be 192.168.0.1.

SSH Port Property

The default port for SFTP servers is 22. If you are interacting with an SFTP server that has a different value, you will need to set the appropriate port value here.

SSH Password Property

Set this to the password used for connecting to the SFTP server.

SSH Remote Path Property

If you are receiving data from a subdirectory of the SFTP site, you'll need to set the full path in this directory. Make sure and add a forward slash (/) before any path you enter in this property. The path is based off of the root server - so if your full path is 192.168.0.1/ChildOne/ChildTwo, you should enter /ChildOne/ChildTwo in this property, and enter 182.168.0.1 in the SSH Host property.

SSH Remote File Name Property

The filename can be set using any combination of plain text and BizTalk macros that you may need. Some of the most common macros are shown in Table 5-1. Macros can be combined—if, for example, you want to show the source filename and combine it with the current datetime, you could put a value of %SourceFileName%_%datetime% in the SSH Remote File Name property.

Table 5-1. Common BizTalk Macros

Macro	Description
%datetime%	Creates a string in the format of YYY-MM-DDThhmmss based on the current UTC time of the server. If you want to take into account the local time zone, you can use %datetime.tz%.
%Message_ID%	Setting your target filename with this macro included in it ensures that you will always have a uniquely named file. The Message_ID is the GUID (Globally Unique Identifier) of the message in the BizTalk Message box.
%SourceFileName%	Sets to the value available in the FILE.ReceivedFileName of the adapter picking up the original file. In some cases, you won't have access to the source filename in your send adapter—such as when the data is originating in an orchestration. This macro retains any file extensions that may have been present (such as .pgp or .txt).

There are more macros than are shown in this table, but there are some fairly severe limitations around what you can name files. If you find that the available BizTalk macros are not flexible enough to meet your requirements, you will have to develop a custom pipeline and pipeline component to create your filename. This pipeline can be added directly to the Send Pipeline on the SFTP Send Port.

SSH User Property

Set this to the username used for connecting to the SFTP server.

Trace Property

If you are running into exceptions when the SFTP adapter runs, you may want to set this property to True to log detailed information about what is happening.

Encrypted Data with Standard FTP

Using the Standard FTP adapter to send and receive data with BizTalk is a breeze but dealing with encrypting and decrypting data is not. This section outlines the standard properties used to configure an FTP adapter for sending or receiving data. Additionally, it discusses some of the challenges around custom pipeline and pipeline component development, and shows how to set up a custom pipeline on a Send Port and a Receive Location.

Note You can send encrypted data via the SFTP adapter or any other adapter, but FTP is the most typical protocol requiring encrypted data when dealing with health care data.

FTP Adapter Settings

If you are sending data over FTP, you can create a Send Port in BizTalk and set the Type to FTP. If you are receiving data over FTP, you can create a BizTalk Receive Location and set the Type to FTP. In either case, you will need to set the following key properties:

- *User Name:* The user with which you connect to the FTP site.
- *Password:* The password used for connections.
- *Server:* The FTP server. This should contain the IP or named server being connected to, and should not have the ftp:// prefix on it.
- *Port:* The specific port required for FTP connections.
- *Folder:* The remote folder that you are posting data to. It should not have a leading forward slash (/) on it.
- *Representation:* Binary or ASCII. In general, this should be set to binary, but some FTP servers don't handle binary data, so you may have to experiment with settings here.

With the FTP adapter settings configured properly, you need only to focus on the requirements of the send pipeline.

Pipelines and Pipeline Components

One of the most complex tasks in BizTalk is creating custom pipelines, as it is pure C# development. If you are using PGP for encryption and decryption, some pointers on how to develop this custom pipeline component are outlined in in this section. If you need to use an alternative encryption format, then you'll need to code something specific to the tools that are used for that format. In either case, you'll need someone who is familiar with C# development to be available to work on this.

There are two items that must be set up for both the send pipeline that encrypts data and the receive pipeline that decrypts data. These items are the custom pipeline and the custom pipeline component. The custom pipeline component should be developed first. Let's assume that you are going to be dealing with PGP encrypted data. There are several tools that you could use—one of the easiest to interact with is GNU Privacy Guard (www.gnupg.org). This utility allows for the generation and management of PGP keys, and provides a command line interface that can be communicated with via C# .NET code.

Calling the command line tool requires that you build out a .NET class to wrap the call so that the pipeline can pass parameters to the command line and execute it (using System.Diagnostics.ProcessStartInfo is one option to do this). Assuming you have built a wrapper class for the GNU Privacy Guard command line tool (generally located in the GNU/GnuPG/pub directory), then a sample of calling this command line tool from within a custom pipeline component to encode data is shown in Listing 5-1, while a sample of decoding data is shown in Listing 5-2.

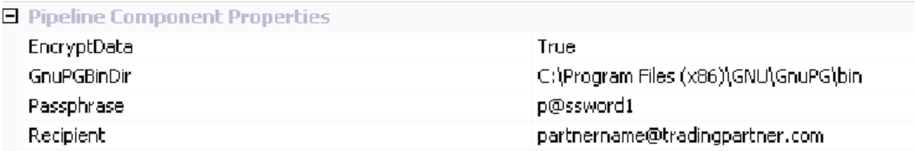
Listing 5-1. Calling a Class to Encode Data with Parameters

```
GnuPGWrapper GPG = new GnuPGWrapper(_gnupgbindir);
GnuPGComm
GPGCommand.Command = Commands.Encrypt;
GPGCommand.Recipient = _recipient; // this is the
recipient on the PGP key
GPGCommand.Passphrase = _passphrase ; // this is the
passphrase on the PGP key
GPGCommand.Armor = true;
GPGCommand.InputFile = inFile;
GPGCommand.OutputFile = outFile;
```

Listing 5-2. Calling a Class to Decode Data with Parameters

```
GnuPGWrapper GPG = new GnuPGWrapper(_gnupgbindir);
GnuPGCommand GPGCommand = GPG.Command;
GPGCommand.Command = Commands.Decrypt;
GPGCommand.InputFile = inFile;
GPGCommand.OutputFile = outFile;
GPGCommand.Passphrase = _passphrase; // this is the
passphrase of the PGP key
```

Creating the custom pipeline component takes some effort, and depends on the encryption and decryption requirements of your solution. You want to make a number of the fields configurable, so that you can use the send and receive pipelines on multiple trading partners. Figure 5-2 shows what these configurable properties could look like when they are set within the custom pipeline in Visual Studio.



⊟ Pipeline Component Properties	
EncryptData	True
GnuPGBinDir	C:\Program Files (x86)\GNU\GnuPG\bin
Passphrase	p@ssword1
Recipient	partnername@tradingpartner.com

Figure 5-2. Configurable parameters on the send pipeline component

The actual custom pipeline where you would be adding the custom pipeline components has to also be created. This is done within Visual Studio as a new BizTalk Pipeline project. An example of a send pipeline and what stage the custom pipeline component to encrypt should be added is shown in Figure 5-3. An example of a receive pipeline and the custom pipeline to decrypt is shown in Figure 5-4.

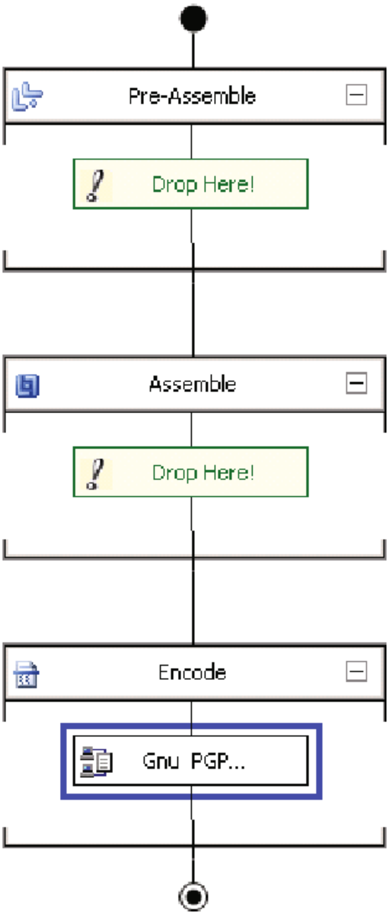


Figure 5-3. The send pipeline with encrypt component

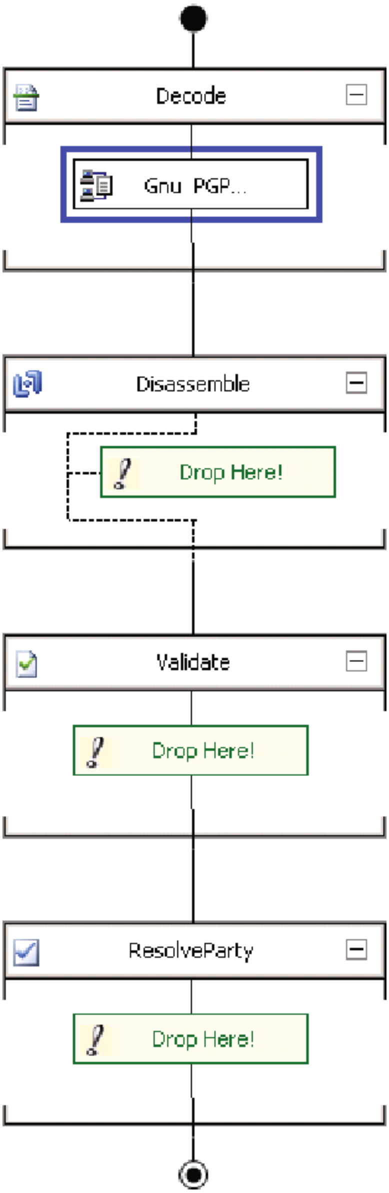
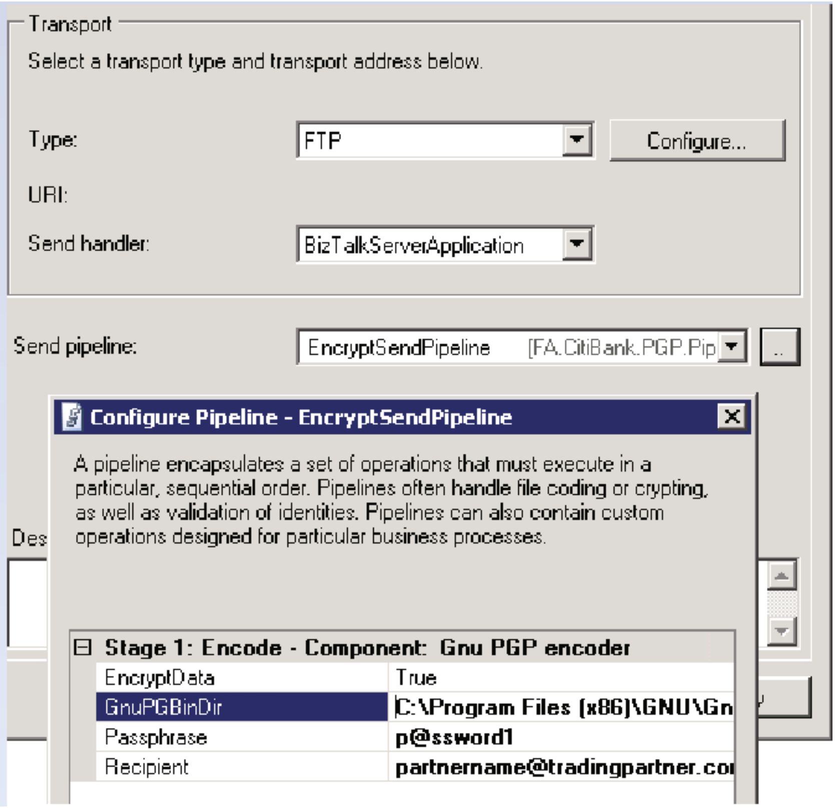


Figure 5-4. The receive pipeline with decrypt component

After you have created your pipelines and deployed them, they will be available to the Send Port and Receive Location where you have configured your FTP adapter. An example of a Receive Location with the decryption pipeline configured on it is shown in Figure 5-5.



Transport

Select a transport type and transport address below.

Type: FTP

URI:

Send handler: BizTalkServerApplication

Send pipeline: EncryptSendPipeline [FA.CitiBank.PGP.Pip

Configure Pipeline - EncryptSendPipeline

A pipeline encapsulates a set of operations that must execute in a particular, sequential order. Pipelines often handle file coding or crypting, as well as validation of identities. Pipelines can also contain custom operations designed for particular business processes.

Stage 1: Encode - Component: Gnu PGP encoder

EncryptData	True
GnuPGBinDir	C:\Program Files (x86)\GNU\Gn
Passphrase	p@ssword1
Recipient	partnername@tradingpartner.co

Figure 5-5. Configuring the pipeline on an FTP send port

AS2 Communications

Configuring BizTalk for AS2 communications can be a time-consuming and difficult task. The most complex aspect of it is dealing with certificates. Both you and your trading partner are required to exchange certificates and configure communications with one another with the same settings. Should your data be encrypted? Should your MDN be signed? Do you have the correct certificate for the development environment versus the production environment? Is your trading partner sending data in the expected format? The purpose of this section is to provide you with enough detail around configuring and testing AS2 so that you can avoid most of the pitfalls associated with setting this up.

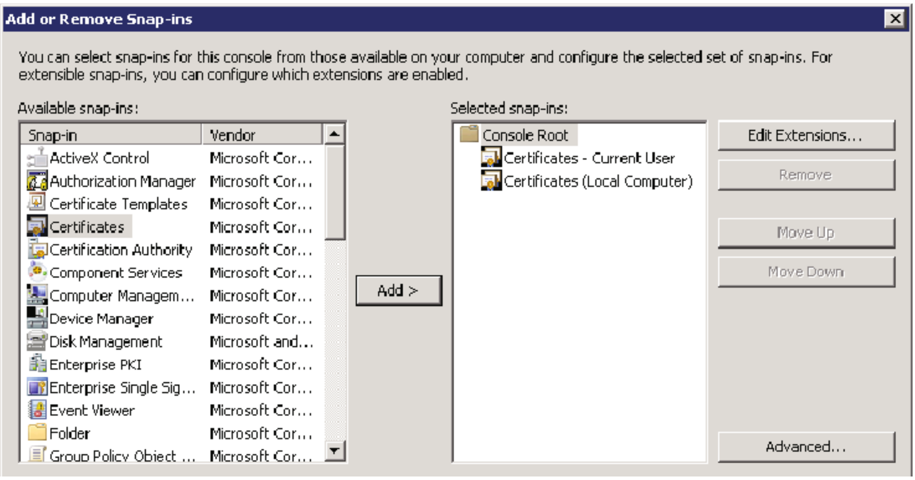
Certificates

The first thing you want to do is get your certificates set up. Begin by exchanging public keys with your trading partner. You should have a public and private key for your organization and a public key from the trading partner. After you have these, you can take the following steps to set up the certificates on the BizTalk server.

CERTIFICATE CONFIGURATION FOR AS2

This exercise demonstrates where to place and how to reference the certificates required in AS2 communications with BizTalk.

1. Log into the BizTalk server using a BizTalk service account.
2. Open the Certificate manager. From the Start menu, click Run and type **mmc**. Once this is open, click File and select Add/Remove Snap-in. Select Certificates and click Add. Select the My user account option and click Finish. Select Certificates again and click Add—this time, select the Computer account option and click Next. Select the Local Computer option and click Finish. You should now have two Certificate types, as shown in Figure 5-6. After this is complete, click OK.



Add or Remove Snap-ins

You can select snap-ins for this console from those available on your computer and configure the selected set of snap-ins. For extensible snap-ins, you can configure which extensions are enabled.

Available snap-ins:

Snap-in	Vendor
ActiveX Control	Microsoft Cor...
Authorization Manager	Microsoft Cor...
Certificate Templates	Microsoft Cor...
Certificates	Microsoft Cor...
Certification Authority	Microsoft Cor...
Component Services	Microsoft Cor...
Computer Managem...	Microsoft Cor...
Device Manager	Microsoft Cor...
Disk Management	Microsoft and...
Enterprise PKI	Microsoft Cor...
Enterprise Single Sig...	Microsoft Cor...
Event Viewer	Microsoft Cor...
Folder	Microsoft Cor...
Group Policy Object ...	Microsoft Cor...

Selected snap-ins:

- Console Root
  - Certificates - Current User
  - Certificates (Local Computer)

Figure 5-6. Configuring the certificate snap-in

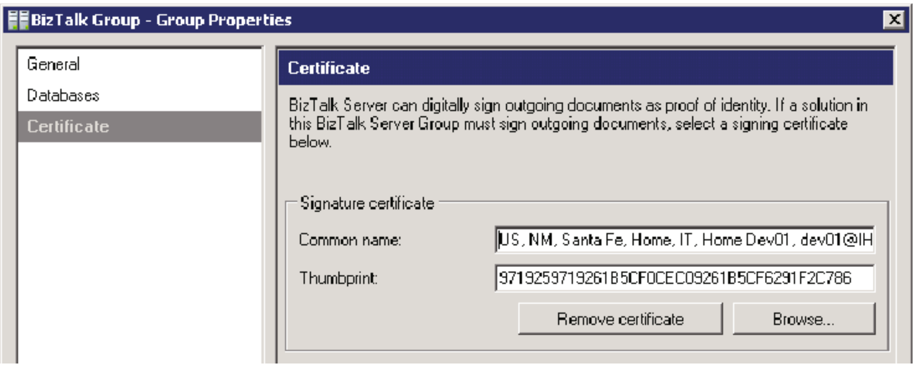
3. With the Certificate console open, expand Certificates Current User, and right-click Personal. Select Import and import the private key (.pfx) for your home organization.

4. Next, expand Certificates - Local Computer, and right-click Other People. Select Import and import the public key (.cer) for your trading partner's organization.

5. You should now see your certificates in several locations—the Personal and Other People folders of both the Local Computer and Current User. With these certificates installed, you now can reference them from the appropriate locations in BizTalk.

6. In the BizTalk Administration Console, right-click the BizTalk Group and select properties. Click the Certificate option and select Browse. Your home organization's certificate should appear, select it and click OK. Figure 5-7 shows the certificate set at this level. This is your primary certificate used to sign outbound data.

Note You can override this default certificate for specific parties, if needed, in the Certificate page of the AS2 properties for your trading partner. In most cases, you'll use a single certificate for everyone, but there may be times when you'll need to use a unique certificate for signing.



BizTalk Server can digitally sign outgoing documents as proof of identity. If a solution in this BizTalk Server Group must sign outgoing documents, select a signing certificate below.

Common name: US, NM, Santa Fe, Home, IT, Home Dev01, dev01@IH

Thumbprint: 9719259719261B5CF0CEC09261B5CF6291F2C786

Figure 5-7. Batching with multiple claims per ST/SE

7. Right-click your trading partner's BizTalk party in the BizTalk Admin Console's Parties folder and select Properties. Click the Certificate option and click Browse. Select the trading partner's certificate.

There are only two other locations that you may need to configure certificates for your AS2 communications with a single trading partner - on the "Signature Certificate" page of the AS2 agreement (which allows for overriding the default home organization certificate on outbound documents and MDNs) and on any Send Ports that you may be using. However, it is unlikely that you will need to do anything with either of these if you are engaging in standard AS2 communications.

IIS and the BizTalk HTTP Receive Location

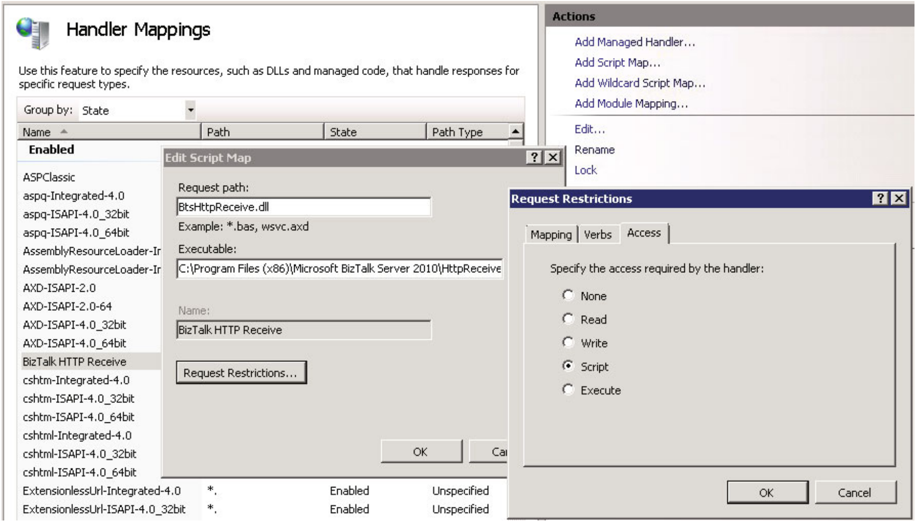
AS2 is communication over HTTP, so setting up a site within IIS on the BizTalk Server is a requirement. There are a number of ways this can be set up, but the most common is to create a virtual directory for a specific trading partner that maps inbound requests to the BTSHTTPReceive.dll (which then pushes the inbound data to BizTalk for processing). This is a fairly involved yet easy configuration, and the following exercise outlines how to set up the various components.

Note In some cases, your organization may not allow companies outside your network to post data directly via HTTP to BizTalk. In this case, you'll have to set up a proxy server to allow traffic to flow through your DMZ and hit the HTTP location in BizTalk. This is a separate area of expertise from BizTalk, and should be handled by a network administrator.

CONFIGURING IIS AND THE HTTP RECEIVE LOCATION

This exercise demonstrates how to create and configure the appropriate IIS components to handle inbound AS2 posts. It also shows how to set up the BizTalk Receive Location that receives these posts.

1. Log in to the BizTalk server using a BizTalk service account.
2. Open the IIS 7 manager, click the root server, and select the Handler Mappings option. In the Actions area on the right side of the screen, click Add Script Map. Set the Request path property to BtsHttpReceive.dll and set the Executable to the location of the BtsHttpReceive.dll (this is located in the HttpReceive folder in the root BizTalk Server directory). Set the Name field to BizTalk HTTP Receive and then click the Request Restrictions button. In the Request Restrictions box, on the Access tab, select Script and click OK.
3. Click OK on the Add Script Map window when this has all been completed. Right-click the BizTalk HTTP Receive item that was just created and select Edit Feature Permissions. In the window that opens, select the Read, Script, and Execute boxes and click OK. See Figure 5-8 for a view of the final configuration.



Handler Mappings

Use this feature to specify the resources, such as DLLs and managed code, that handle responses for specific request types.

Group by: State

Name	Path	State	Path Type
Enabled			
ASPClassic			
aspq-Integrated-4.0			
aspq-ISAPI-4.0_32bit			
aspq-ISAPI-4.0_64bit			
AssemblyResourceLoader-Ir			
AssemblyResourceLoader-Ir			
AXD-ISAPI-2.0			
AXD-ISAPI-2.0-64			
AXD-ISAPI-4.0_32bit			
AXD-ISAPI-4.0_64bit			
BizTalk HTTP Receive			
cshtm-Integrated-4.0			
cshtm-ISAPI-4.0_32bit			
cshtm-ISAPI-4.0_64bit			
cshtml-Integrated-4.0			
cshtml-ISAPI-4.0_32bit			
cshtml-ISAPI-4.0_64bit			
ExtensionlessUrl-Integrated-4.0	*.	Enabled	Unspecified
ExtensionlessUrl-ISAPI-4.0_32bit	*.	Enabled	Unspecified

Edit Script Map

Request path: BtsHttpReceive.dll

Example: *.bas, wsvc.axd

Executable: C:\Program Files (x86)\Microsoft BizTalk Server 2010\HttpReceive

Name: BizTalk HTTP Receive

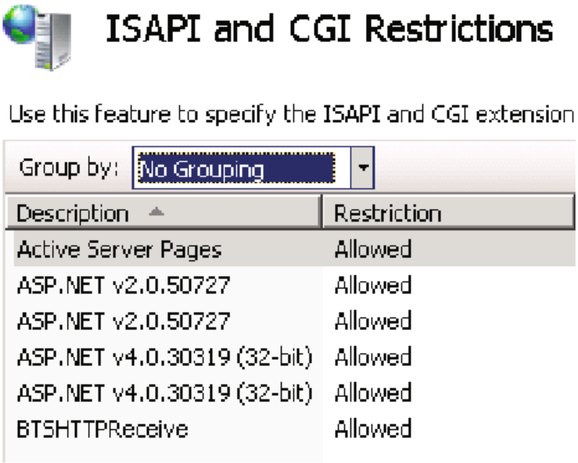
Request Restrictions

Specify the access required by the handler:

○ None
○ Read
○ Write
● Script
○ Execute

Figure 5-8. Configuring the HTTP receive handler map in IIS

4. Back on the root server in IIS, click the ISAPI and CGI Restrictions icon. In the window that opens, set the BTSHTTPReceive Restriction setting to Allowed, as shown in Figure 5-9.



ISAPI and CGI Restrictions

Use this feature to specify the ISAPI and CGI extension

Group by: No Grouping

Description	Restriction
Active Server Pages	Allowed
ASP.NET v2.0.50727	Allowed
ASP.NET v2.0.50727	Allowed
ASP.NET v4.0.30319 (32-bit)	Allowed
ASP.NET v4.0.30319 (32-bit)	Allowed
BTSHTTPReceive	Allowed

Figure 5-9. Configuring the HTTP receive handler map in IIS

5. Create a new Application Pool in IIS, and set the name to BizTalkAppPool (or similar). Set the .NET Framework version property to 4.0 (whichever specific version is available to you) and the Managed Pipeline mode to Integrated.

6. Create a new virtual directory (as an Application) under the Default Web Site. The name of this site should be specific to the trading partner that you will be receiving data from over HTTP—so in this case, name it TradingPartner. Set the Application Pool to the app pool you created in the previous step and select Test Connection to ensure you are able to connect.

Note Depending on your security setting, you may find that you also need to set the Physical Path Credentials to a specific account that has access to that directory. The easiest way to access this is to right-click the web Application you created and select Manage Application and then Advanced Settings.

7. Click the virtual directory you just created and select the Authentication icon. In Authentication window that opens, set Anonymous Authentication to Enabled.

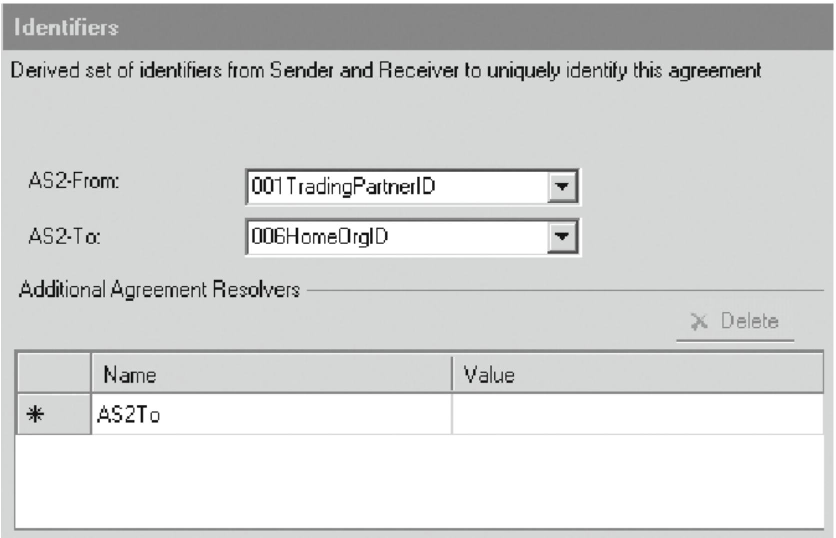
This completes the setup of all IIS related components for AS2. If additional trading partners need to be set up, create one additional virtual directory for each one.

Agreements and Party Settings

To specify how to handle the AS2 data and how to work with the underlying EDI document that is being sent via AS2, you will need to set up a BizTalk Party and two Agreements. One Agreement is for the AS2 messaging, and one Agreement is for dealing with the actual EDI data. The basic steps for setup are as follows for receiving AS2 data from a trading partner (sending data to a trading partner is very similar):

- Create a new BizTalk Party with the name of the trading partner you will be receiving data from.
- If you will be sending a 997 to the trading partner, specify the Send Port that the 997 will be sent out on.

- Create a new Agreement on this Party that will handle AS2 messaging (you can call it something like Agreement_AS2).
- On the General Tab, set the Protocol property to AS2, the First Party to the trading partner, and the Second Party to your home organization. Once you've set the General Tab this way, two additional tabs will appear, one for inbound data from the trading partner, and one for outbound data to the trading partner. If you are just receiving data from the trading partner and returning an MDN, you only need to configure the inbound trading partner tab.
- On the inbound trading partner tab, on the Identifiers tab, set the AS2-From and AS2-To properties to the appropriate values as defined in your trading partner agreement. These must match what is on the AS2 envelope being sent to you. Figure 5-10 shows an example of these settings.



Identifiers

Derived set of identifiers from Sender and Receiver to uniquely identify this agreement

AS2-From: 001TradingPartnerID

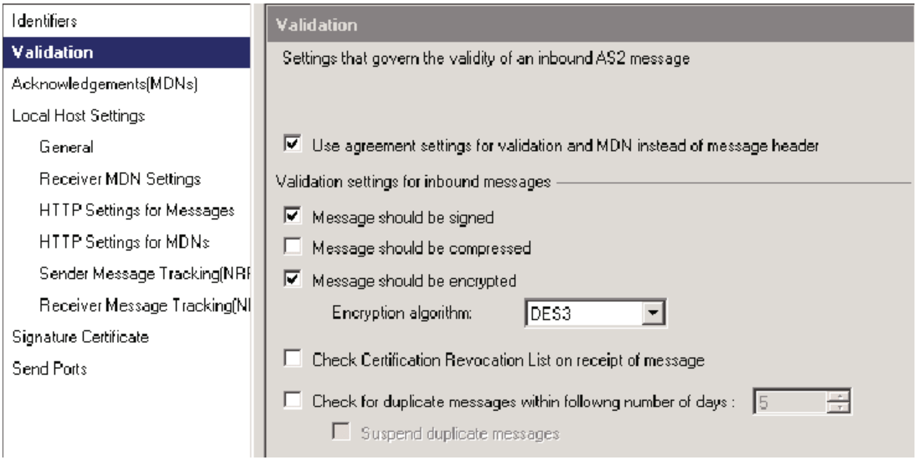
AS2-To: 006HomeOrgID

Additional Agreement Resolvers

	Name	Value
*	AS2To	

Figure 5-10. The Identifiers tab in the AS2 agreement

- On the Validation tab for the AS2 Agreement, you can set the appropriate values for validation of the data. For example, if you are receiving a signed and encrypted inbound post from a trading partner, then you would set the properties as shown in Figure 5-11.



Validation

Settings that govern the validity of an inbound AS2 message

☑ Use agreement settings for validation and MDN instead of message header

Validation settings for inbound messages

☑ Message should be signed

☐ Message should be compressed

☑ Message should be encrypted

Encryption algorithm: DES3

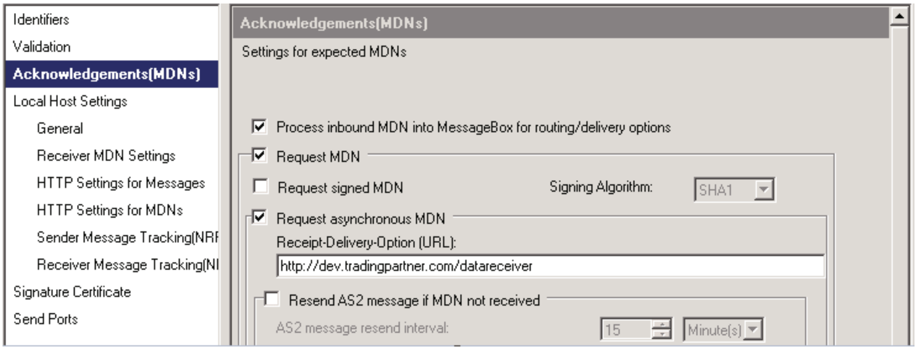
☐ Check Certification Revocation List on receipt of message

☐ Check for duplicate messages within following number of days : 5

☐ Suspend duplicate messages

Figure 5-11. The Validation tab in the AS2 agreement

- On the Acknowledgement tab of the AS2 Agreement, you can set the properties that pertain to the MDN response back to the trading partner. If you need to send an unsigned MDN, you can use the properties as shown in Figure 5-12. If you are sending a signed MDN, then the certificate specific in the Signature Certificate settings will be used (or, if none is specified, then the default certificate associated with the BizTalk Group will be used).



Acknowledgements(MDNs)

Settings for expected MDNs

☑ Process inbound MDN into MessageBox for routing/delivery options

☑ Request MDN

☐ Request signed MDN   Signing Algorithm: SHA1

☑ Request asynchronous MDN

Receipt-Delivery-Option (URL): http://dev.tradingpartner.com/datareceiver

☐ Resend AS2 message if MDN not received

AS2 message resend interval: 15 Minute(s)

Figure 5-12. The Acknowledgements tab in the AS2 agreement

There are some additional properties that will likely need to be set or adjusted on the other tabs. A few of the most common are noted here:

- On the Receive MDN Settings tab, enable the Sign requested MDN setting if you want to always send an MDN, regardless of what is noted on the inbound AS2 request from the trading partner.

- In the HTTP Settings for Messages, enable everything except for the Ignore SSL Certificate Name mismatch property.
- In the HTTP Settings for MDN, enable everything except for the Unfold HTTP headers property.
- On the Signature Certificate tab, set the certificate that you want to use for signing the outbound MDN. If nothing is selected here, the default certificate for the BizTalk Group will be used.
- Create a new Agreement for the EDI data that will be consumed. Setting this up will depend on the specifics of the EDI document type(s) that are being received over AS2 (you can see details of configuring Agreement settings in Chapters 2 and 3). What is important to know here is that you must have this additional Agreement in place so that BizTalk knows how to process the EDI data once the AS2 Agreement has successfully completed the data transfer.

The Generic MDN Send Port for Asynchronous Messages

The MDN is the acknowledgement for AS2 posts. There are two possible methods for postback of an MDN—synchronous and asynchronous. The synchronous response is posted back via the same open HTTP connection that the original document came in on, and does not require any additional BizTalk components (simply set the Request MDN checkbox in the BizTalk Agreement, and it will automatically post back). For asynchronous MDNs, a send port must be created. You can create a generic send port that will work for all parties by taking a few steps:

- Create a new Dynamic One-way Send Port and name it something such as SendAsyncronousMDNs.
- Set the Filter on the Send Port to EdiIntAS.IsAS2AsynchronousMdn == True.
- Set the Send pipeline to AS2Send.
- In the BizTalk Agreement for AS2, select the Request asynchronous MDN property and set the Receipt-Delivery-Option (URL) property to the URL that the trading partner is expecting data to be delivered on.

When the configuration is set like this, the moment a document is received from a trading partner, BizTalk will automatically create an MDN and drop it on the BizTalk Message Box. The SendAsynchronousMDNs send port subscribes to this document and sends it out to the URL specified in the Receipt-Deliver-Option (URL) property on whatever trading partner's Agreement was just used to receive the data.

Testing Your AS2 Configuration

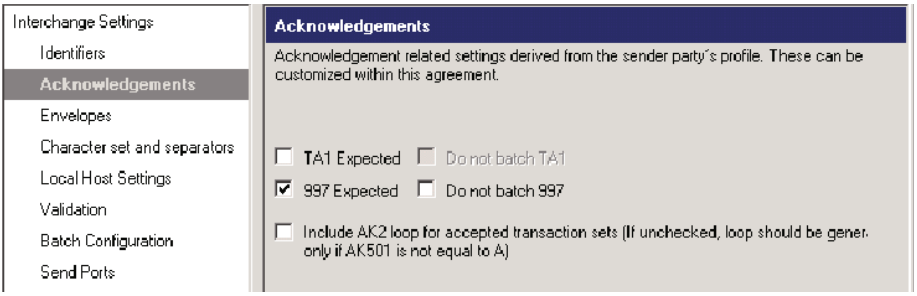
One of the most challenging (and frustrating) aspects of AS2 configuration is the actual trading partner testing. The best advice is to plan to set up your AS2 configuration in stages. Try to exchange plain text data (unencrypted and unsigned) first before dealing with the various settings requiring certificates. If you can get the plain, unencoded data to flow—and the MDN to return—successfully, then you can move into testing encryption and signing.

There are many things that can go wrong during testing, and the error messages are often very generic and cryptic. The errors could be on your side, or they could be on the trading partner's side. The more you can do to limit what is being tested at any given stage, the quicker you will be able to get to resolution and completion.

Sending 997/999 Acknowledgements

There are several types of acknowledgements that can be sent in response to EDI communications: Functional (997/999), Technical (TA), and MDNs (for AS2). Configuring and sending MDN acknowledgements was covered earlier in the AS2 section in this chapter. Technical Acknowledgements are rarely required, and are identical in setup to the Functional Let's look at sending the Functional acknowledgements for the EDI data itself. The steps to take are as follows:

- Open the BizTalk Party Agreement that relates to the documents and trading partner that you need to set up the Functional Acknowledgement for, click the Acknowledgements tab. Check 997 Expected, as shown in Figure 5-13.



Acknowledgements

Acknowledgement related settings derived from the sender party's profile. These can be customized within this agreement.

☐ TA1 Expected ☐ Do not batch TA1

☑ 997 Expected ☐ Do not batch 997

☐ Include AK2 loop for accepted transaction sets (If unchecked, loop should be gener. only if AK501 is not equal to A)

Figure 5-13. The Acknowledgements tab in the EDI agreement

- Set up one Send Port per trading partner. These send ports subscribe directly to the BizTalk Message Box, and filter on the specific trading partner required. The Send Port should have the following settings:
    - The transport type - FTP, SFTP, or other. Set the appropriate connection information for the actual adapter that will be used to connect to the trading partner.
    - The Send Pipeline should be set to EdiSend - or, if you are required to encrypt 997 data (which is uncommon), you will need to add your custom send pipeline to do the encryption.
    - Three filters, as follows:

      EDI.IsSystemGeneratedAck == true

      EDI.ST01 == 997 (or 999 if version 5010)

      EDI.ISA06 == [this should be the specific trading partner's ID that you are configuring for this specific Send Port - this ID can be retrieved from the Party settings of the BizTalk Agreement where you have configured 997s to be sent]

- Depending on your configuration, you may need to associate the 997/999 Send Port with your Agreement on the Send Ports tab.

After you have these settings configured, BizTalk automatically generates a 997/999 when the inbound EDI document is received and drops it on the BizTalk Message Box. Next, the Send Port picks it up and delivers it to the specified destination.

Conclusion

This chapter discussed the most common transport mechanisms for any BizTalk EDI health care implementation you may need to build out. It has also shown how to deal with acknowledgements—997/999s and MDNs. With the information related to AS2, SFTP, and encrypted data over FTP that has been covered, you can develop and interact with trading partners with ease.