Configuring the Relay Server with Microsoft IIS using SSL – 2nd Edition

A whitepaper from Sybase iAnywhere

Author: Joshua Savill, Product Manager

Date: April 3rd, 2009

This whitepaper was written in the context of SQL Anywhere 11.
CONTENTS

Introduction 3
Software Required 3
The Relay Server 3
  Relay Server Architecture 3
    The Outbound Enabler 4
Starting the Relay Server 4
Configuring the Relay Server with Microsoft Internet Information Services 7
  Deploying the Relay Server Web Extensions 7
  Creating an Application Pool 8
  Enabling the Relay Server Web Extensions 8
Performance Tips 9
SQL Anywhere 11 Documentation on Relay Server for Microsoft IIS 9
Understanding the Relay Server Configuration File and Starting the Relay Server as a Windows
  Service 9
    Options Section 10
    Relay Server Section 11
    Backend Farm Section 12
    Backend Server Section 12
    Applying the Relay Server Configuration File to the Relay Server State Manager 13
Using the Outbound Enabler 14
Configuring Microsoft IIS FOR Testing using SSL 14
Summary 20

FIGURES

Figure 1 – Relay Server Infrastructure with MobiLink Synchronization ........................................... 4
Figure 2 – Startup Phase of the Relay Server ............................................................................. 5
Figure 3 – Ready Phase of the Relay Server ............................................................................... 6
Figure 4 – Working Phase of the Relay Server ............................................................................ 7
Figure 5 – Certificate Created in Microsoft IIS for SSL Communications ................................. 16
Figure 6 – Process and Install the Certificate Created by the Authority Certificate .................... 18
Figure 7 – Certificate Summary from Microsoft IIS ................................................................. 19
INTRODUCTION

Secure data transmission and data availability are integral components of corporate information systems in today's wireless and occasionally connected environments. The need for data security is at an all time high as data breaches and sensitive information thefts are becoming a common occurrence and combined with the demand for always available data, corporate data systems are being pushed to new limits.

MobiLink already provides a highly scalable session-based synchronization system that allows bi-directional data transfer between a main database, called the consolidated database, and many remote databases. The consolidated database can be one of several ODBC-compliant databases, including SQL Anywhere, Sybase Adaptive Server Enterprise, Oracle, Microsoft SQL Server, MySQL and IBM DB2 (Mainframe).

As new demands are being placed on IT and Web Infrastructure, SQL Anywhere's MobiLink synchronization technology is more flexible than ever in developing a highly secure and always available solution for mobilizing enterprise applications to front line employees. With SQL Anywhere 11, Sybase iAnywhere has introduced the relay server, a new feature that eases integration with existing IT and Web Infrastructure to provide a secure gateway for data synchronization. Using the relay server in a multi-tenant environment is a cost effective way to secure a wide spectrum of services required for enterprise business.

SOFTWARE REQUIRED

- Microsoft Windows 2003 Server
  - Microsoft Internet Information Service 6.0
- SQL Anywhere with MobiLink 11.0.x, including the RSA encryption option

THE RELAY SERVER

RELAY SERVER ARCHITECTURE

The relay server is a set web extensions for Microsoft Internet Information Services (IIS) and Apache web servers that enables secure communications for web traffic. The relay server provides multi-tenant load balancing and failover, and can be easily integrated with Sybase iAnywhere products.

The relay server can be easily integrated into existing IT and Web infrastructure without requiring changes to existing corporate firewalls and IT policies. It is designed to handle incoming HTTP and HTTPS requests from MobiLink, UltraLite, QAnywhere and other Sybase iAnywhere HTTP/S clients through multiple connector pieces called the outbound enabler. Multiple backend HTTP services are provided when using the relay server, including MobiLink, Afaria, and Mobile Office. This configuration does not require new ports to be enabled on either the internal or external firewall if the relay server is located in the corporate DMZ. Figure 1 shows the typical setup using a single relay server and MobiLink server. The relay server is located in the corporate DMZ.
THE OUTBOUND ENABLER

The outbound enabler is a new feature of SQL Anywhere 11 that is designed specifically for use as the connector between the backend server and the relay server. Figure 1 above shows the MobiLink server as the backend server communicating with the outbound enabler. The outbound enabler acts as a double connection piece that establishes on-demand connections with the MobiLink server, and a pair of up/down channel connections with the relay server in the DMZ.

The relay server forwards client requests to the outbound enabler using the established up channel. The outbound enabler then relays the client request to the backend server, in this case the MobiLink server. Once the outbound enabler receives the response from the MobiLink server, it forwards the response back to the relay server using the established down channel. The relay server then relays the response to the client.

The outbound enabler and clients are serviced by the relay server using two different web extensions. The relay server client extension is used for client connections and the server extension is used for the outbound enabler.

If a relay server farm exists, the outbound enabler can establish an HTTP or HTTPS connection with the relay server to retrieve a list of all available relay servers. This is done by supplying the server with a URL that maps to the relay server server extension. If there is a load balancer handling a relay server farm, the server URL would map through the load balancer to the relay server server extension.

The outbound enabler, after retrieving information regarding the available relay servers, creates the up/down channel pairs with each relay server. The communication with the backend MobiLink server is limited to HTTP only. There is no HTTPS communication, because the outbound enabler is designed to be run on the same machine as the MobiLink server. As a result, encryption for this communication is not necessary. Connect, transfer, and disconnect activities of the outbound enabler are driven on demand by the clients and backend servers.

STARTING THE RELAY SERVER

The relay server architecture is broken down into three phases:
- startup phase
- ready phase
- working phase

The startup phase, shown in figure 2, is initiated when the outbound enabler establishes a connection with the web server running the relay server (or load balancer, if multiple web servers exist). The outbound enabler then retrieves a list of all available relay servers in the relay server farm.

Figure 1 – Relay Server Infrastructure with MobiLink Synchronization
The ready phase, shown in figure 3, is initiated when the outbound enabler has the list of all available relay servers, at which point the outbound enabler detects if the backend server is available and then establishes a direct connection with each relay server, bypassing the load balancer if one exists.
The working phase, shown in figure 4, is initiated when a MobiLink, UltraLite, QAnywhere, or other Sybase iAnywhere client establishes a connection with the web server. In figure 4, the connection is drawn with a hashed mark to represent a stateless HTTP/S connection. All communication between the relay server and client is through the web server, or load balancer if one exists.

**Performance tip:** If a load balancer is being used to handle multiple relay servers in a farm environment, the load balancer does not need to maintain HTTP session affinity between the client and the relay server. Individual HTTP requests within the same HTTP session may go through different relay servers in the farm. Significant load balancer performance gain can be obtained by turning off the HTTP affinity feature.
Figure 4 – Working Phase of the Relay Server

If a MobiLink server farm is being used, all three phases (startup, ready, and working) occur regardless of the number of MobiLink servers.

CONFIGURING THE RELAY SERVER WITH MICROSOFT INTERNET INFORMATION SERVICES

DEPLOYING THE RELAY SERVER WEB EXTENSIONS

The relay server requires the following executables, configuration file, and DLLs:

- rs_client.dll
- rs_server.dll
- rhost.exe
- rs.config
- dbigendll.dll
- dbsvc.exe
- dbhld.exe
- dbtool.dll
- dbicu11.dll
- dbicud11.dll
- dbsupport.exe
- dbghelp.dll
- dblib11.dll

In Windows Explorer:
1. Create the following directories under the **Default Web Site** (typically C:\Inetpub\wwwroot) home directory for the relay server:
   - ias_relay_server
   - ias_relay_server|client
• ias_relay_server\server
2. Copy rs_client.dll to ias_relay_server\client directory.
3. Copy rs_server.dll, rshost.exe and rs.config to ias_relay_server\server directory.
4. The remainder of the DLLs and EXEs are copied into the ias_relay_server\server directory.
5. In the Microsoft Internet and Information Services (IIS) Manager, right click on the Default Web Site and ensure the Connection timeout is set to at least 60 seconds. By default this value should be 120 seconds and is sufficient.

Note: The DLLs in the \ias_relay_server\server directory need to be accessible by the Network Service for incoming client and outbound enabler connections. Therefore, the \ias_relay_server\server directory needs to be accessible in the Windows PATH. If the \ias_relay_server\server directory cannot be defined in the Windows PATH, then the DLLs for step 4 should be located in the system32 folder or in a directory that is defined in the Windows PATH.

CREATING AN APPLICATION POOL

In the Microsoft Internet Information Services (IIS) Manager:
1. Right click on Application Pools → New → Application Pool...
2. Call the new Application Pool RS_POOL.
4. Under the Performance tab, deselect Shutdown worker processes after being idle for (time in minutes) and set the Maximum number of worker processes to the total number of processing cores in the machine. For more information on the number of worker processes see Microsoft IIS performance documentation on Web garden.

ENABLING THE RELAY SERVER WEB EXTENSIONS

In the Microsoft IIS Manager:
2. In the Directory tab, select Create. This enables the Application pool section. Set the Application pool to RS_POOL. Click OK.
3. In the Directory Security tab select Edit in the Authentication and access control section. Enable anonymous access and fill in the user name and password for an account belonging to the Administrators group. Alternatively, the build-in user IUSR_%computername% can be used but the following command must be executed to grant permission to access the IIS metabase for the user:

   C:\Windows\Microsoft.Net\Framework\<Version>\aspnet_regiis.exe -ga IUSR_%computername%

4. Under the Web Service Extensions folder, select Add a new Web service extension. Add both rs_server.dll and rs_client.dll as new web extensions. The Extension name should be called ISAPI. Set the extension status to Allowed.

Note: The Default Web Site and relay server state manager (rshost.exe) need to be stopped and restarted for the above changes to take affect.
PERFORMANCE TIPS

The relay server does not rely on ASP.NET. Removing ASP.NET ISAPI filter yields better performance. The filter gets turned on by default in a standard Microsoft IIS installation. To turn off the filter:

1. In the Microsoft IIS Manager, right click on the Default Web Site and select Properties.
2. Under the ISAPI Filters tab, remove the ASP.NET filter.

Further performance gains can be achieved by turning off the IIS access log. To turn off the IIS access log:

1. In the Microsoft IIS Manager, under Web Sites right click on the ias_relay_server directory and select Properties.
2. In the Directory tab, deselect the Log Visits selection.

The relay server does not impose restrictions on the Web garden size. One worker process may serve requests from all outbound enablers as well as from all the clients. However, the number of threads that can be created in the process is limited by the process heap space left available for thread creation. The thread created by Microsoft IIS has a 256k stack size. If the machine has adequate resources, experiment with a higher number of processes if there is suspicion the machine is hitting a concurrency limit when the server is loaded with thousands of concurrent requests. To modify the Web garden size:

1. In the Microsoft IIS Manager, under Application Pools right click on the RS_POOL and select Properties.
2. Under the Performance tab, the Maximum number of worker processes can be adjusted.

In a production environment, the relay server should be run with minimal to no verbosity. This is set via the relay server configuration file (rs.config). The verbosity level should be set to 0.

SQL ANYWHERE 11 DOCUMENTATION ON RELAY SERVER FOR MICROSOFT IIS

For more information about deploying the relay server to Microsoft IIS, see the following SQL Anywhere 11 documentation:

MobilLink - Server Administration »
Using MobilLink Server Technology »
The Relay Server »
Deploying the Relay Server »
Deploying the Relay Server components to IIS on Windows

Online:

UNDERSTANDING THE RELAY SERVER CONFIGURATION FILE AND STARTING THE RELAY SERVER AS A WINDOWS SERVICE

The relay server configuration file is used to define all relay servers, MobilLink servers, and the MobilLink server farms in the environment. The configuration file, rs.config, is divided into four sections:

- Options
- Relay server
- Backend farm
• Backend server

**Note:** After changing the *rs.config* file, the relay server state manager (*rshost.exe*) needs to be updated of the changes. This is done by executing *rshost --u --rs.config* on the command line.

**OPTIONS SECTION**

**Tip:** It is considered best practice to start the relay server as a Windows service. This section describes how to configure the relay server to start as a Windows service.

The options section is used to specify properties that apply to each relay server and determine how the relay server is started. A sample of the section is shown below:

```
#-------------------------------------
# Relay server options
#-------------------------------------
[options]
start = no
shared_mem = 10M
verbosity = 2
```

The following command sets up the relay server as an auto start Windows service:

```bash
dbsvc -y -as -s Automatic -w rs "c:\Inetpub\wwwroot\ias_relay_server\server\rshost.exe" -q -qc -f "c:\Inetpub\wwwroot\ias_relay_server\server\rs.config" -o "c:\temp\myrshost.log"
```

The options above automatically start *rshost.exe* with a shared memory cache of 10 megabytes (default). The verbosity is set to request level logging 2, with the logging file located in the `%TEMP%\ias_relay_server_host.log` file by default.

The following command starts the Windows service using the `c:\Inetpub\wwwroot\ias_relay_server\server\rs.config` file as specified:

```bash
dbsvc -u rs
```

The following command stops the Windows service:

```bash
dbsvc -x rs
```

The logging for *rshost.exe* is directed to the *myrshost.log* file located in the `c:\temp` directory.
Note: The www services need to be started and stopped independently of the relay server service.

For more information about the Options Section, see the following SQL Anywhere 11 documentation:

MobiLink - Server Administration »
Using MobiLink Server Technology »
The Relay Server »
Relay Server configuration file »
Options section

Online:

RELAY SERVER SECTION

The relay server section is used to define each relay server that exists in the environment. Each entry in the relay server section identifies a unique relay server.

#-------------------
# Relay server peers
#-------------------
[relay_server]
enable = yes
host = RelayHost1.com
http_port = 80
https_port = 443
description = Relay Server Definition 1

[relay_server]
enable = yes
host = RelayHost2.com
http_port = 80
https_port = 443
description = Relay Server Definition 2

The above example shows there are two relay servers in the farm: RelayHost1.com and RelayHost2.com. Each is defined to listen on port 80 for HTTP requests and port 443 for HTTPS requests.

For more information about the relay server section, see the following SQL Anywhere 11 documentation:

MobiLink - Server Administration »
Using MobiLink Server Technology »
The Relay Server »
Relay Server configuration file »
Relay Server section

Online:

BACKEND FARM SECTION

The backend farm section specifies the properties of the MobiLink server farm. If multiple MobiLink server farms exist in the environment, there will be multiple entries in this section. A sample of the section is shown below:

#-----------------
# Backend farms
#-----------------
[backend_farm]
enable          = yes
id              = MobiLink.Server
client_security = on
backend_security= on
description     = The is the MobiLink server farm entry

The above example shows there is only one MobiLink server farm, which is identified by the ID MobiLink.Server.

For more information about the backend farm section, see the following SQL Anywhere 11 documentation:

MobiLink - Server Administration »
Using MobiLink Server Technology »
The Relay Server »
Relay Server configuration file »

backend farm section

Online:

BACKEND SERVER SECTION

The backend server section defines a backend server connection to the machine running the MobiLink server. Each entry in the backend server section defines a single MobiLink server. The definition information is used by the outbound enabler when it connects to the relay server. A sample of the section is shown below:

#-----------------
# Backend servers
#-----------------
[backend_server]
enable   = yes
farm     = MobiLink.Server
id       = MLServer1
mac      = 00-0C-29-7A-C2-AB
token    = 7b2493b0-d0d4-464f-b0de-24643e1e0feb

[backend_server]
enable   = yes
farm     = MobiLink.Server
id       = MLServer2
mac      = 00-0C-29-7A-C2-AB
token    = 9a2633e0-a1b5-321b-d0ac-2667f1a2abc

The above example shows that there are two MobiLink servers in the MobiLink.Server server farm. Each MobiLink server runs on the same machine, identified by a unique id and the MAC address of the machine’s network adapter. There is also a unique security token to authenticate backend server connections with the relay server.

For more information about the backend server section, see the following SQL Anywhere 11 documentation:

MobiLink - Server Administration »
Using MobiLink Server Technology »
The Relay Server »
   Relay Server configuration file »
Backend server section

Online:

APPLYING THE RELAY SERVER CONFIGURATION FILE TO THE RELAY SERVER STATE MANAGER

The changes to the rs.config file need to be applied to the relay server state manager. This can be accomplished by running the following:

"c:\Inetpub\wwwroot\ias_relay_server\server\rshost.exe" -qc -u -f rs.config

- rshost.exe is the relay server state manager
- -qc shutdown the relay server state manager window upon completion
- -u updates the configuration of the running relay server
- -f rs.config specifies the configuration file

Note: Given the nature of the Windows operating system and Microsoft Internet Information Services, it may be best to reboot the system or at minimum restart the relay server state manager, World Wide Publishing, and IIS Admin Services. This will ensure all PATH changes and system configuration are correct in the Windows system. Any unexpected behaviour with the relay server after making the PATH and system configuration should be resolved after a restart of the system or services.
USING THE OUTBOUND ENABLER

The outbound enabler is designed like a double connector between the relay server and the MobiLink server. The outbound enabler establishes a connection with the relay server using HTTP or HTTPS, and the MobiLink server using only HTTP. The outbound enabler facilitates the communication and requests between the relay server and MobiLink server.

The outbound enabler also reports back to the relay server if it cannot establish a connection with the MobiLink server. At that time, the outbound enabler terminates the connection with the relay server and does not accept any new requests from the relay server until the MobiLink server is back online.

A sample outbound enabler command line is shown below:

```
"%SQLANY11%\bin32\rsoe.exe" -cr
"host=relayserver.sybase.com;port=80" -cs
"host=localhost;port=80" -f MobiLink.Server -id MLServer1 -t
7b2493b0-d0d4-464f-b0de-24643e1e0feb
```

- `rsoe.exe` is the outbound enabler utility
- `-cr <connection string>` is the connection to the relay server (HTTP or HTTPS)
- `-cs <connection string>` is the connection to the MobiLink server (HTTP)
- `-f MobiLink.Server` is the farm name for the backend server
- `-id MLServer1` is the name assigned for the MobiLink server
- `-t 7b2493b0-d0d4-464f-b0de-24643e1e0feb` is the security token passed to the relay server

**Note:** The `–cs` switch defaults to `host=localhost;port=80` if it is not specified on the outbound enabler command line.

There are additional switches that can be provided to the outbound enabler. For more information about the outbound enabler, see the following SQL Anywhere 11 documentation:

- **MobiLink - Server Administration**
  - **Using MobiLink Server Technology**
  - **The Relay Server**
  - **Outbound Enabler**

Online:


CONFIGURING MICROSOFT IIS FOR TESTING USING SSL

The first requirement for configuring Microsoft IIS using SSL encryption is to create a Certificate Authority’s certificate, also known as a root certificate. To create the root certificate, use the `createcert` utility provided with the RSA encryption option.
Note: For more information on the createcert utility, see:
http://www.sybase.com/detail?id=1055179

1. Run the following:

   C:\>"%SQLANY11%\win32\createcert"
   SQL Anywhere X.509 Certificate Generator Version 11.0.1.2044
   Choose encryption type ((R)SA or (E)CC): RSA
   Enter RSA key length (512-16384): 1024
   Generating key pair...
   Country Code: CA
   State/Province: ON
   Locality: Waterloo
   Organization: Sybase
   Organizational Unit: Sybase
   Common Name: Sybase
   Enter file path of signer's certificate: <enter>
   Certificate will be a self-signed root
   Serial number [generate GUID]: <enter>
   Generated serial number: de9cd2c1270f4af980f525ca4a779b4a
   Certificate valid for how many years (1-100): 10
   Certificate Authority (Y/N) [N]: Y
   1. Digital Signature
   2. Nonrepudiation
   3. Key Encipherment
   4. Data Encipherment
   5. Key Agreement
   6. Certificate Signing
   7. CRL Signing
   8. Encipher Only
   9. Decipher Only
   Key Usage [6,7]: <enter>
   Enter file path to save certificate: rsa_root.crt
   Enter file path to save private key: rsa_key.key
   Enter password to protect private key: pwd
   Enter file path to save identity: id.pem

2. The Microsoft IIS web server needs to be configured for secure communications. This requires
   the creation of a web server certificate that needs to be signed by the Authority Certificate.
   a. In the Internet Information Services (IIS) Manager, select the Default Web Site → Properties.
   b. Under the Web Site tab, set the SSL port to 443.
   c. Under the Directory Security tab → Server Certificates → Create New Certificate, select to Prepare the request now, but send it later.
   d. Use the Name Default Web Site and Bit length 1024.
e. The Organization and Organizational unit are both Sybase.
f. The Common name for the new certificate is jsavill-iis.
g. The Country is CA, Province is ON, and City is Waterloo.
h. Save the certreq.txt file.

Figure 5 – Certificate Created in Microsoft IIS for SSL Communications

3. The certreq.txt file created by Microsoft IIS needs to be signed by the Authority Certificate. Run the following:

C:\>"%SQLANY11%\bin32\createcert" -s certreq.txt
SQL Anywhere X.509 Certificate Generator Version 11.0.1.2044
Enter file path of signer's certificate: rsa_root.crt
Enter file path of signer's private key: rsa_key.key
Enter password for signer's private key: pwd
Serial number [generate GUID]: <enter>
Generated serial number: c79a26d921ed411b91bf565ae0c30f28
Certificate valid for how many years (1-100): 10
Certificate Authority (Y/N) [N]: <enter>
1. Digital Signature
2. Nonrepudiation
3. Key Encipherment
4. Data Encipherment
5. Key Agreement
6. Certificate Signing
7. CRL Signing
8. Encipher Only
9. Decipher Only

Key Usage [3,4,5]: <enter>

Enter file path to save certificate: rsa_iis.crt

4. To verify the certificate, run the following:

C:\>"%SQLANY11%\bin32\viewcert" rsa_iis.crt

SQL Anywhere X.509 Certificate Viewer Version 11.0.1.2044

X.509 Certificate

Common Name: jsavill-iis
Organizational Unit: Sybase
Organization: Sybase
Locality: Waterloo
State/Province: ON
Country Code: CA
Issuer: Sybase
Serial Number: c79a26d921ed411b91bf565ae0c30f28
Issued: Apr 9, 2008 14:08:00
Expires: Apr 10, 2018 14:08:00
Signature Algorithm: RSA, SHA1
Key Type: RSA
Key Size: 1024 bits
Basic Constraints: Is not a certificate authority
Key Usage: Key Encipherment, Data Encipherment, Key Agreement

5. After signing the certreq.txt file from Microsoft IIS, the createcert utility has created a certificate for SSL. The certificate has been saved as rsa_iis.crt. The certificate needs to be processed and installed on Microsoft IIS. In Microsoft IIS, select the Default Web Site → Properties.


7. Browse to the location of the rsa_iis.crt.
Figure 6 – Process and Install the Certificate Created by the Authority Certificate

8. Set the SSL port to 443 and Finish the wizard.
9. Verify the HTTPS connection to IIS
   a. Run an Internet browser.
   b. In the Address field enter https://servername and press Enter.
   c. Click Continue to this website. The Under Construction will appear indicating the certificate installed correctly.

**Note:** For production systems, server certificates should be purchased from a reputable Certificate Authority. Below are the certificate considerations for a relay server farm:

1. Load balancer with standard certificate. The relay servers require individual standard server certificates.
2. Load balancer without SSL. The relay servers require standard wildcard certificates. This setup works for clients that do not verify common names of the certificate against the domain name. For example, this works for UltraLite, but not UltraLiteJ.
3. Load balancer with standard certificates but no certificates for the relay servers. This setup does not support HTTPS on the up/down channels between the outbound enabler and relay servers. Clients can still use HTTPS for communication.

The types of certificates required for security are completely dependent upon business needs and the requirements for encryption.
SUMMARY

Sybase iAnywhere’s new relay server feature for SQL Anywhere 11 eases integration with existing IT and Web Infrastructure to provide a secure gateway for data synchronization. Using the relay server in a multi-tenant environment is a cost effective way to secure a wide spectrum of services required for enterprise business.

This document has outlined how to configure the relay server using Microsoft IIS to complete secure MobiLink synchronizations. After finishing this document you should have the understanding to setup a complete synchronization system using SSL communication between the MobiLink, UltraLite, QAnywhere, or other Sybase iAnywhere clients and Microsoft IIS using the new relay server feature.