

SYBASE®

Relay Server

Copyright © 2011 by Sybase, Inc. All rights reserved.

This publication pertains to Sybase software and to any subsequent release until otherwise indicated in new editions or technical notes. Information in this document is subject to change without notice. The software described herein is furnished under a license agreement, and it may be used or copied only in accordance with the terms of that agreement.

To order additional documents, U.S. and Canadian customers should call Customer Fulfillment at (800) 685-8225, fax (617) 229-9845.

Customers in other countries with a U.S. license agreement may contact Customer Fulfillment via the above fax number. All other international customers should contact their Sybase subsidiary or local distributor. Upgrades are provided only at regularly scheduled software release dates. No part of this publication may be reproduced, transmitted, or translated in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without the prior written permission of Sybase, Inc.

Sybase trademarks can be viewed at <http://www.sybase.com/detail?id=1011207>. Sybase and the marks listed are trademarks of Sybase, Inc. ® indicates registration in the United States of America.

Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Unicode and the Unicode Logo are registered trademarks of Unicode, Inc.

All other company and product names mentioned may be trademarks of the respective companies with which they are associated.

Use, duplication, or disclosure by the government is subject to the restrictions set forth in subparagraph (c)(1)(ii) of DFARS 52.227-7013 for the DOD and as set forth in FAR 52.227-19(a)-(d) for civilian agencies.

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Contents

About this book	v
Introduction to the Relay Server	1
Relay Server architecture	1
Deploying the Relay Server	7
Deploying the Relay Server components to Microsoft IIS 6.0 on Windows Server 2003	7
Deploying the Relay Server components to Microsoft IIS 7.0 or 7.5 on Windows Server 2008/Windows Server 2008 R2	10
Deploying the Relay Server components to Apache on Linux	13
Relay Server State Manager	17
Starting the Relay Server State Manager as a service	17
Starting the Relay Server State Manager automatically	18
Starting the Relay Server State Manager automatically with customized options	18
Relay Server State Manager command line syntax	19
Relay Server configuration file	21
Relay Server section	21
Backend farm section	22
Backend server section	23
Options section	25
Relay Server configuration file format	25
Outbound Enabler	27
Outbound Enabler as a service	32
Updating a Relay Server farm configuration	35

Updating a Relay Server configuration for Microsoft IIS on Windows	36
Updating a Relay Server configuration for Apache on Linux	36
Sybase Hosted Relay Service	37
Using the Sybase Hosted Relay Service	37
Using MobiLink with the Relay Server	39
Connecting a client to the Relay Server farm	39
Sample scenario	40
Index	43

About this book

This book describes how to set up and use the Relay Server, which enables secure communication between mobile devices and MobiLink, Afaria, and iAnywhere Mobile Office servers through a web server.

Introduction to the Relay Server

The Relay Server enables secure, load-balanced communication between mobile devices and back-end servers through a web server. Supported back-end servers include MobiLink, Unwired Server, Afaria, and Mobile Office. The Relay Server provides the following:

- A common communication architecture for mobile devices communicating with back-end servers.
- A mechanism to enable a load-balanced and fault-tolerant environment for back-end servers.
- A way to help communication between mobile devices and back-end servers in a way that integrates easily with existing corporate firewall configurations and policies.

Relay Server architecture

A Relay Server deployment consists of the following:

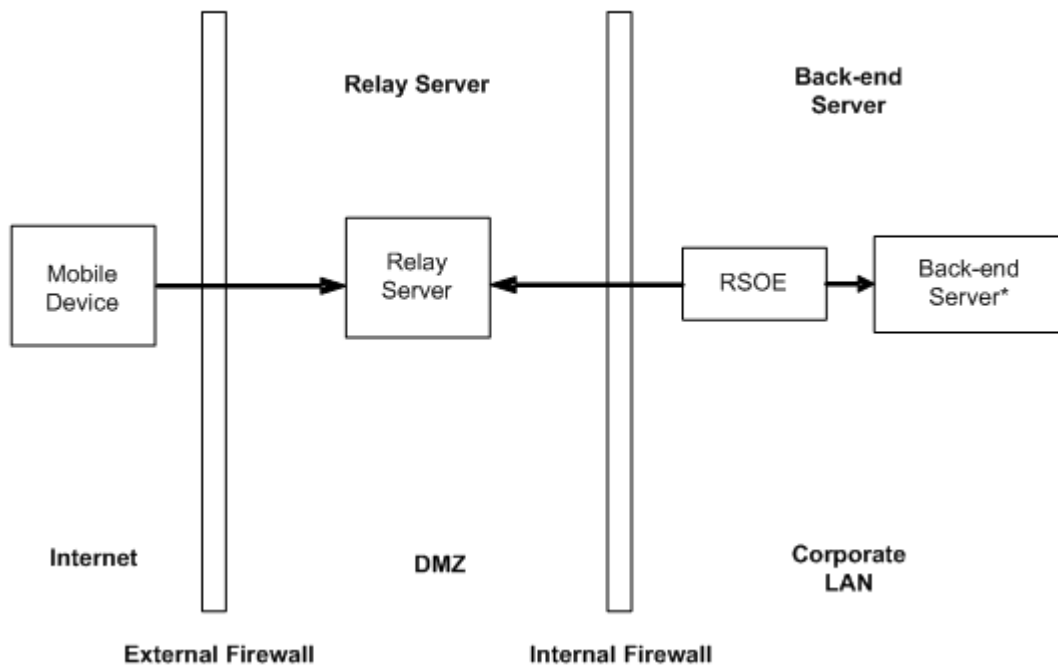
- Mobile devices running client applications and services that need to communicate with back-end servers running in a corporate LAN.
- Optional load balancer to direct requests from the mobile devices to a group of Relay Servers.
- One or more Relay Servers running in the corporate DMZ.
- At least one back-end server running in a corporate LAN that are responsible for servicing client requests. The following back-end servers are supported for use with the Relay Server:
 - MobiLink
 - Unwired Server
 - Afaria
 - Mobile Office
 - SQL Anywhere

Note

Refer to your license agreement or supported platforms matrix for information about which back-end servers are supported.

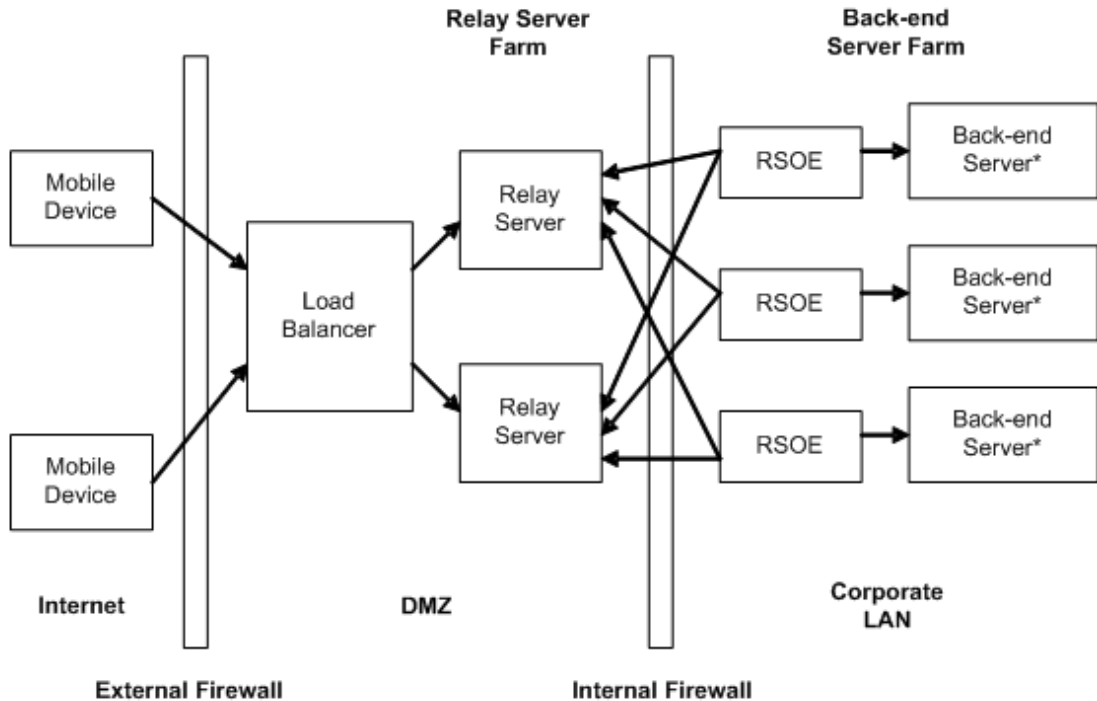
- One Relay Server Outbound Enabler (RSOE) per back-end server. The Outbound Enabler manages all communication between a back-end server and the Relay Server farm.

The following diagram shows the Relay Server architecture with a single Relay Server.



Note
Refer to your license agreement or supported platforms matrix for information about which back-end servers are supported.

The following diagram shows the Relay Server architecture for a more complex system with a Relay Server farm and a back-end server farm.

**Note**

Refer to your license agreement or supported platforms matrix for information about which back-end servers are supported.

The Relay Server consists of a set of web extensions, a background process for maintaining state information, and a web server.

Because the Relay Server is a web extension running in a web server, all communication is performed using HTTP or HTTPS. Using HTTP easily integrates with existing corporate firewall configurations and policies. The Relay Server requires that the connection from the corporate LAN to the Relay Server be initiated from inside the corporate LAN. This provides a more secure deployment environment because it does not require inbound connections from the DMZ into the corporate LAN.

The Relay Server contains two web extensions: a client extension and a server extension. The client extension handles client requests made from applications running on mobile devices. The server extension handles requests made by the Outbound Enabler on behalf of a back-end server.

The Relay Server farm

A Relay Server farm is any number of Relay Servers with a front-end load balancer. It is possible to set up a Relay Server farm with a single Relay Server, in which case a load balancer is not required. In this case, mobile devices can connect directly to the Relay Server.

Back-end server farm

A back-end server farm is a group of homogeneous back-end servers. A client making a request through the Relay Server farm must specify the back-end server farm it is targeting.

Load balancer

The load balancer directs requests from the mobile devices to a Relay Server running in the Relay Server farm. The load balancer is not required if there is only one Relay Server.

Relay Server Outbound Enabler

The Relay Server Outbound Enabler runs on the same computer as the back-end server. Its primary function is to initiate an outbound connection to all Relay Servers in the Relay Server farm on behalf of the back-end server. There is one Outbound Enabler per back-end server.

Relay Server security

The Relay Server has built-in security features, but also relies on the security features provided by the web server. In combination with the web server, the Relay Server provides the following features for secure communications:

- Server-side certificates
- Client-side certificates
- Backend server and farm configuration
- RSOE MAC address filtering and token authentication
- Client encryption technologies (Protocol-level encryption)

Server-side certificates

Using a server-side certificate, a client communicating with the Relay Server can verify the web server that is running the Relay Server is a trusted server. The client verifies the web server's public certificate with the root certificate stored on the client. If the certificates are verified, a key exchange occurs to establish the encrypted connection.

Client-side certificates

Using a client-side certificate, the web server can verify that a client communicating with the Relay Server is a trusted client. The web server verifies the client's public certificate with its root certificate that is stored in the certificate manager on the web server machine. If the certificates are verified, a key exchange occurs to establish the encrypted connection.

Backend server and farm configuration

The *rs.config* file is used by the Relay Server to define the peer list of Relay Servers, if running in a farm environment, including the backend farm and backend server configurations. Each Relay Server in the environments needs to maintain a copy of the *rs.config* file.

The configuration of the backend farm and backend servers ensures the Relay Server only communicates with machines with which it has been configured. Any attempted communication with machines for which the Relay Server has not been configured will be refused.

The backend farm can be configured to specify the level of communication security when accepting requests from the clients and the RSOE. There is a `client_security` and `backend_security` option that allows the backend farm to specify the type of communication that can be established. This option is specified as follows:

client_security=on|off **On** indicates the client must connect using HTTPS. **Off** indicates the client must connect using HTTP. This setting is optional. If no value is specified, the client can connect using HTTP or HTTPS.

backend_security=on|off **On** indicates the RSOE must connect using HTTPS. **Off** indicates the RSOE must connect using HTTP. This setting is optional. If no value is specified, the RSOE can connect using HTTP or HTTPS.

RSOE MAC address filtering and token authentication

The RSOE establishes the connection between the backend server and the Relay Server using three phases: 1) startup phase, 2) ready phase 3) working phase. For a detailed description of the RSOE startup, see:

- Windows IIS <http://www.sybase.com/detail?id=1059277>
- Apache <http://www.sybase.com/detail?id=1065869>

In the backend server section of the *rs.config* file, each server that exists in the backend farm is configured with an ID and associated farm name. The ID corresponds to the server name. The Relay Server has the ability to verify the MAC address of the machine running the RSOE to ensure the server communicating from within the internal firewall is trusted and allowed to establish a connection with the Relay Server. The MAC property is the MAC address of the network adapter used by the RSOE. The address is specified in IEEE 802 MAC-48 format.

The backend server section also allows for the configuration of a security token that is used by the Relay Server to authenticate the backend server connection. The token must be provided upon startup of the RSOE when establishing the connection with the Relay Server.

HTTP listening port of the outbound enabler

For security purposes, when using the standalone outbound enabler, the HTTP listening port of the backend server should be explicitly bound only to the loopback IP address (127.0.0.1).

Deploying the Relay Server

The Relay Server components are installed as part of the SQL Anywhere 12 installation. The install process automatically deploys all the necessary files on the machine that is going to run the Relay Server.

Deploying the Relay Server components to Microsoft IIS 6.0 on Windows Server 2003

The Relay Server for Windows consists of the following executables:

- *rs_client.dll*
- *rs_server.dll*
- *rs_monitor.dll*
- *rshost.exe*
- *dblgen12.dll*
- *dbsvc.exe*
- *dbfhide.exe*
- *dbtool12.dll*
- *dblib12.dll*
- *dbicu12.dll*
- *dbicudt12.dll*
- *dbsupport.exe*
- *dbghelp.dll*

For information about which versions of IIS are supported, see <http://www.sybase.com/detail?id=1061806>.

Setup scripts for Relay Server on IIS can be found in the *%SQLANY12%\RelayServer\IIS* directory.

Deploy the Relay Server files

1. Install the Relay Server components using the SQL Anywhere install. By default all files are installed to *%SQLANY12%* and are based on the bitness of the machine:
 - *%SQLANY12%\Bin32* and *%SQLANY12%\Bin64* are used for DLLs and executables for administration.
 - *%SQLANY12%\RelayServer\IIS\Bin32* and *%SQLANY12%\RelayServer\IIS\Bin64* are used for Relay Server specific files under the appropriate folder (for example, *Admin*, *Client*, *Monitor* or *Server*). The *Server* folder contains the *rshost.exe* and *rs.config* files.
2. Create a virtual directory called *rs* under the Default Web Site in the Microsoft IIS Manager for use by the Relay Server. The physical location of the virtual directory is *%SQLANY12%\RelayServer\IIS\BinXX* where *XX* is either 32 or 64 depending on the bitness of your IIS server.

3. Create the Relay Server configuration file *rs.config* using the following guidelines.
 - The file should have four sections:
 - Options section
 - Relay server section
 - Backend farm section
 - Backend server section
 - Each section starts with a section tag, made by enclosing a keyword that identifies the section name in square brackets.
 - Add the appropriate properties to each section. A property is defined by specifying the property name on the left side of an equal sign and its value on the right side of the equal sign. For example, property name = value.
 - The configuration file should contain only 7-bit ASCII characters.
4. Create an application pool:
 - a. Start Microsoft IIS Manager Console.
 - b. Right-click **Application Pools** and create a new application pool, for example RS_POOL.
 - c. Edit the properties for the application pool you created.
 - i. Click the **Recycling** tab and turn off all the recycling options.
 - ii. Click the **Performance** tab and do the following:
 - A. Turn off **Shutdown Worker Processes After Being Idle**.
 - B. Set the number of worker processes to the total number of processing cores. You can further adjust this number depending on your usage and performance preferences. See the Microsoft IIS performance notes about Web garden size for more information.
5. Set the Connection timeout property of the Default Web Site to a minimum of 60 seconds. By default this value should be 120 seconds, which is sufficient.
6. Edit the properties of rs and enable the Relay Server web extensions using the IIS Manager Console:
 - a. Click the **Directory** tab and do the following:
 - i. Set execute permissions to **Scripts And Executables**.
 - ii. Click **Create** under **Application Settings**. Select the application pool you created in step 4 as the associated application pool.
 - b. Click the **Directory Security** tab and do the following:
 - i. Click **Edit** in **Authentication and Access Control**.
 - ii. Enable anonymous access and fill in the user name and password for an account belonging to the Administrators group.

Alternatively, you may leave the setting as the built-in user **IUSR_%computername%** and execute the following command to grant permission to access the Microsoft IIS metabase.

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

- c. Under **Web Server Extensions** in the Microsoft IIS manager, add *rs_server.dll*, *rs_client.dll*, and *rs_monitor.dll* as a new Web service extension. The extension name should be ISAPI and the DLLs need to have the extension status set to Allowed.
7. Deploy the Relay Server configuration file by creating a Relay Server configuration file and copying it to the `%SQLANY12%\RelayServer\IIS\BinXX\server` directory.
8. Ensure optimum performance by reviewing the performance tips.
9. Start the Relay Server State Manager as a service using a command line similar to the following:

```
dbsvc -as -s auto -t rshost -w RelayServer "%SQLANY12%\RelayServer\IIS\BinXX\Server\rshost.exe" -q -qc -f "%SQLANY12%\RelayServer\IIS\BinXX\Server\rs.config" -o "c:\temp\ias_relay_server.log"
```

See:

Note

It is recommended that you start the State Manager as a service. However, it can also be started automatically by the Relay Server.

10. Update the Relay Server configuration for Microsoft IIS 6.0 on Windows:
 - a. For each computer that belongs to the Relay Server farm you are updating, copy the updated configuration file to the `%SQLANY12%\RelayServer\IIS\BinXX\Server` directory under the Relay Server web site home directory. The configuration file must be called *rs.config* if auto start is used.
 - b. From the `%SQLANY12%\RelayServer\IIS\BinXX\Server` directory, run the following command line to apply the configuration update:

```
rshost -u -f rs.config
```
 - c. Repeat the previous steps for each computer in the Relay Server farm that is being updated.

Note

After configuring the Relay Server with IIS, it is recommended that you restart the IIS server or the computer.

Performance tips

Keep the following in mind when deploying the Relay Server to Microsoft IIS on Windows:

- The Relay Server web extension does not rely on ASP.NET. Removing the ASP.NET ISAPI filter yields better performance. The filter gets turned on by default in a standard Microsoft IIS install. To turn off the filter, do the following:
 1. Start Microsoft IIS Manager Console.
 2. Edit the properties of **Default Web Site**.

3. Under the **ISAPI Filters** tab, remove the ASP.NET filter.
- For better performance, you can turn off the Microsoft IIS access log. To turn off the access log, do the following:
 1. Start Microsoft IIS Manager Console.
 2. Edit the properties of the *ias_relay_server* directory under **Default Web Site**.
 3. Under the **Directory** tab, clear the **Log Visits** selection.
 - In a production environment, Relay Server verbosity can be set to 0 via the Relay Server configuration file. This yields better performance under high loads.
 - 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 your machine has adequate resources, experiment with a higher number of processes if you suspect you are hitting a concurrency limit when the server is loaded with thousands of concurrent requests.

Deploying the Relay Server components to Microsoft IIS 7.0 or 7.5 on Windows Server 2008/Windows Server 2008 R2

The Relay Server for Windows consists of the following executables:

- *rs_client.dll*
- *rs_server.dll*
- *rs_monitor.dll*
- *rshost.exe*
- *dbngen12.dll*
- *dbsvc.exe*
- *dbfhide.exe*
- *dbtool12.dll*
- *dblib12.dll*
- *dbicu12.dll*
- *dbicudt12.dll*
- *dbsupport.exe*
- *dbghelp.dll*

For information about which versions of IIS are supported, see <http://www.sybase.com/detail?id=1061806>.

IIS 7 setup scripts for Relay Server can be found in the `%SQLANY12%\RelayServer\IIS\iis7_setup.txt` directory.

Deploy the Relay Server files

1. Make sure that the Microsoft IIS ISAPI Extensions feature is installed.
2. Install the Relay Server components using the SQL Anywhere install. By default all files are installed to `%SQLANY12%` and are based on the bitness of the machine:
 - `%SQLANY12%\Bin32` and `%SQLANY12%\Bin64` are used for DLLs and executables for administration.
 - `%SQLANY12%\RelayServer\IIS\Bin32` and `%SQLANY12%\RelayServer\IIS\Bin64` are used for Relay Server specific files under the appropriate folder (for example, *Admin*, *Client*, *Monitor* or *Server*). The *Server* folder contains the `rshost.exe` and `rs.config` files.
3. Backup the IIS configuration file `applicationHost.config` located in the `c:\Windows\System32\inetsrv\config` folder.
4. To add an application pool for the Relay Server, edit the `applicationHost.config` file to add the following code to the `<system.applicationHost>` » `<applicationPools>` section.

```
<add name="RelayServer" queueLength="65535" autoStart="true"
managedRuntimeVersion="" managedPipelineMode="Integrated">
  <processModel identityType="LocalSystem" idleTimeout="00:00:00"
maxProcesses="20" pingingEnabled="false"
  pingInterval="00:00:30" pingResponseTime="00:01:30" />
  <recycling disallowOverlappingRotation="true">
    <periodicRestart time="00:00:00">
      <schedule>
        <clear />
      </schedule>
    </periodicRestart>
  </recycling>
  <failure rapidFailProtection="false" />
  <cpu resetInterval="00:00:00" />
</add>
```

Note

The rest of the steps refer the `%SQLANY12%\RelayServer\IIS\BinXX` directory as `%rs_dir%` in the `applicationHost.config` file. However, environment variable expansion is not fully supported in every section in the IIS configuration file, so the `%rs_dir%` variable needs to be fully expanded when you add it into the `applicationHost.config` file.

5. To add the Relay Server application to the default site, edit the `applicationHost.config` file to add the following code to the `<system.applicationHost>` » `<applicationPools>` » `<sites>` » `<site name="Default Web Site">` section.

```
<application path="/rs" applicationPool="RelayServer">
  <virtualDirectory path="/" physicalPath="%rs_dir%"/>
</application>
```

6. To add the Relay Server ISAPI extensions, edit the `applicationHost.config` file to add the following code to the `<system.webServer>` » `<security>` » `<isapiCgiRestriction>` section.

```
<add path="%rs_dir%\Admin\rs_admin.dll" allowed="true" />
<add path="%rs_dir%\Client\rs_client.dll" allowed="true" />
```

```
<add path="%rs_dir%\Monitor\rs_monitor.dll" allowed="true" />
<add path="%rs_dir%\Server\rs_server.dll" allowed="true" />
```

7. To enable access to the Relay Server extensions, edit the *applicationHost.config* file to add the following code to the **<configuration>** section.

```
<location path="Default Web Site/rs">
  <system.webServer>
    <security>
      <authentication>
        <anonymousAuthentication userName="" />
      </authentication>
      <requestFiltering>
        <requestLimits maxAllowedContentLength="2147483647" />
      </requestFiltering>
    </security>
    <handlers accessPolicy="Execute, Script" />
  </system.webServer>
</location>
```

Note

The Relay Server is set up for anonymous access based on these instructions. Proper security needs to be configured for IIS and the Relay Server based on the business requirements.

8. To enforce HTTPS access to the Relay Server administration extension for security reasons, edit the *applicationHost.config* file to add the following code to the **<configuration>** section.

```
<location path="Default Web Site/rs/Admin">
  <system.webServer>
    <security>
      <access sslFlags="Ssl" />
    </security>
  </system.webServer>
</location>
```

9. Save these changes to the *applicationHost.config* file.
10. Set the Connection timeout property of the Default Web Site to a minimum of 60 seconds. By default this value should be 120 seconds, which is sufficient.
11. Create the Relay Server configuration file *rs.config* using the following guidelines.

- The file should have four sections:
 - Options section
 - Relay server section
 - Backend farm section
 - Backend server section
- Each section starts with a section tag, made by enclosing a keyword that identifies the section name in square brackets.
- Add the appropriate properties to each section. A property is defined by specifying the property name on the left side of an equal sign and its value on the right side of the equal sign. For example, property name = value.
- The configuration file should contain only 7-bit ASCII characters.

12. Copy the *rs.config* file to the `%SQLANY12%\RelayServer\IIS\BinXX\Server` directory.
13. Ensure optimum performance by reviewing the performance tips.
14. Start the Relay Server State Manager as a service using a command line similar to the following:

```
dbsvc -a <administrator> -p <password> -s auto -t rshost -w RelayServer  
"%rs_dir%\Server\rshost.exe" -q -qc -f "%rs_dir%\Server\rs.config" -o "c:  
\temp\ias_relay_server.log"
```

Note

It is recommended that you start the State Manager as a service. However, it can also be started automatically by the Relay Server.

15. Update the Relay Server configuration for Microsoft IIS on Windows:
 - a. For each computer that belongs to the Relay Server farm you are updating, copy the updated configuration file to the `%SQLANY12%\RelayServer\IIS\BinXX\Server` directory under the Relay Server web site home directory. The configuration file must be called *rs.config* if auto start is used.
 - b. From the `%SQLANY12%\RelayServer\IIS\BinXX\Server` directory, run the following command line to apply the configuration update:

```
rshost -u -f rs.config
```
 - c. Repeat the previous steps for each computer in the Relay Server farm that is being updated.

Deploying the Relay Server components to Apache on Linux

On Linux, the Relay Server files are installed to the `/opt/sqlanywhere12` as part of the SQL Anywhere installation. See the following topics for deployment information.

For information about which versions of Apache on Linux are supported, see <http://www.sybase.com/detail?id=1061806>.

Deploy the Relay Server files

1. Copy the following executables and shared objects from the `/opt/sqlanywhere12` directory to the `apache-install-dir\modules` directory:
 - `mod_rs_ap_client.so`
 - `mod_rs_ap_server.so`
 - `rshost`
 - `dblgen12.res`
 - `libdbtasks12.so`
 - `libdbicudt12.so`
 - `libdbicu12_r.so`
 - `libdblib12_r.so`
 - `dbsupport`
 - `dbfhide`
 - `libdblib12.so`
 - `mod_rs_ap_monitor.so`
 - `mod_rs_ap_admin.so`
2. Create the Relay Server configuration file `rs.config`
3. Copy `rs.config` into the `apache-install-dir\modules` directory. The server module expects the `rshost` executable to be in the same directory where you copied the `rs.config` file.
4. Edit the Relay Server configuration file `rs.config` using the following guidelines.
 - The file should have four sections:
 - Relay server section
 - Backend farm section
 - Backend server section
 - Options section
 - Each section starts with a section tag, made by enclosing a keyword that identifies the section name in square brackets.
 - Add the appropriate properties to each section. A property is defined by specifying the property name on the left side of an equal sign and its value on the right side of the equal sign. For example, `property name = value`.
 - The configuration file should contain only 7-bit ASCII characters.
5. Set the `PATH` and `LD_LIBRARY_PATH` environment variables to include the Apache `apache-install-dir\modules` directory.
6. Edit the Apache `conf/httpd.conf` file.
 - a. Add the following lines to load the Relay Server client and server modules:

```
LoadModule iarelayserver_client_module modules/mod_rs_ap_client.so
LoadModule iarelayserver_server_module modules/mod_rs_ap_server.so
```

Note

All modules are invoked using different URLs and all modules explicitly look for the string *iarelayserver* in the URL path. That part of the URL need not change.

- b. Add the following line to load the SQL Anywhere Monitor support module:

```
LoadModule iarelayserver_monitor_module modules/mod_rs_ap_monitor.so
```

- c. Add the following line to load the Remote Administration support module:

```
LoadModule iarelayserver_admin_module modules/mod_rs_ap_admin.so
```

- d. Add the following line to create a `<locationMatch>` section for the client module:

```
<LocationMatch /cli/iarelayserver/* >
    SetHandler iarelayserver-client-handler
</LocationMatch>
```

- e. Add the following line to create a `<location>` section for the server module:

```
<Location /srv/iarelayserver/* >
    SetHandler iarelayserver-server-handler
    RSConfigFile "/<apache-install>/modules/rs.config"
</Location>
```

Note

You must specify an `RSConfigFile` directive which specifies the location of the Relay Server configuration file, *rs.config*. The *rs.config* file must reside in the same directory where the *rshost* executable is deployed.

- f. Add the following line to create a `<location>` section for the SQL Anywhere Monitor module:

```
<Location /mon/iarelayserver/* >
    SetHandler iarelayserver-monitor-handler
</Location>
```

- g. Add the following line to create a `<location>` section for the Remote Administration module:

```
<Location /admin/iarelayserver/* >
    SetHandler iarelayserver-admin-handler
</Location>
```

- h. If the `TimeOut` directive is set, ensure it is set to at least 60 seconds.

7. On Linux, if any of the following environment variables are set globally when Apache spawns a process, then there is nothing further needed for the configuration of Apache: `$TMP`, `$TMPDIR` or `$TEMP`.

If any of the above environment variables are not set globally, or if you want the default Relay Server log file to go in a specific temporary directory (for example, when the State Manager is started automatically but without customizations), then edit the file `/<apache-dir>/bin/envvars` to set and then export `TMP`.

For example, to edit `$TMP` in the `envvars` file, do the following:

```
set TMP="/tmp"
export TMP
```

This sets the environment variable in the shell that Apache creates before it spawns its processes.

Note

The Apache user process must have write permissions to the specified *tmp* directory.

8. If you want to update the Relay Server configuration while it is started:
 - a. Copy the updated configuration file to the *apache-install-dir/modules* directory under the Apache install directory. The configuration file must be called *rs.config* if auto start is used.
 - b. From the */<Apache-install>/modules* directory, run the following command line to apply the configuration update:

```
rshost -u -f rs.config
```
 - c. If the Relay Server is set up as a farm with more than one server, repeat the previous steps for each computer in the Relay Server farm.

Relay Server State Manager

The Relay Server State Manager is a process that is responsible for maintaining Relay Server state information across client requests and Outbound Enabler sessions. The State Manager is also responsible for managing the log file used by the Relay Server. The State Manager can either be started automatically by the Relay Server or started as a service.

The default log file name is *ias_relay_server_host.log*. On Windows, this file is located in the directory specified by the TEMP environment variable. On Linux, the file is located in the directory specified by the TMP, TEMP, or TMPDIR environment variables. If none of those variables are set, a log file is created in the */tmp* directory.

Note

The Apache user process must have write permissions to the specified *tmp* directory.

On a graceful shutdown, the State Manager renames the log file to a file of the form *<yymmdd><nn>.log* where *<yymmdd>* represents the date on which the log file was renamed and *<nn>* is the sequential version number of the log file for that day.

Starting the State Manager as a service is the recommended method. Note that starting the State Manager manually on a command line is not supported.

It is possible to specify the options that are used by the Relay Server to start the State Manager. To change the options, set the **start** property in the options section of the Relay Server configuration file. For example:

```
[options]
start = "rshost -o c:\temp\myrshost.log"
```

Note that you must specify the name of the Relay Server State Manager executable (*rshost*) before the options.

Starting the Relay Server State Manager as a service

The State Manager can be started as a service by using the Service utility (*dbsvc*). The start property in the options section of the Relay Server configuration file should be set to **no**.

The Service utility is used to create, modify and delete services. For a full listing of usage information, run *dbsvc* without any options.

To set up an auto-started State Manager service named RelayServer on Windows

```
dbsvc -as -s auto -t rshost -w RelayServer "%SQLANY12%\RelayServer\IIS\BinXX\Server\rshost.exe" -q -qc -f "%SQLANY12%\RelayServer\IIS\BinXX\Server\r.s.config" -o "c:\temp\ias_relay_server.log"
```

To set up an auto-started State Manager service named RelayServer on Unix

```
dbshvc -y -a <apache-user> -t rshost -w RelayServer -q -qc -f /<your-  
director>/rs.config -os 100K -ot /tmp/rs.log
```

Remarks

The syntax of dbshvc on Windows is different than Unix. In Unix, you do not specify the full path of the executable as the first parameter after -w switch argument.

Use full paths only.

In Unix, use a user account (if possibly the same) so that Apache-user processes can attach to the State Manager shared memory and be able to read it and write to it.

Start the service

```
dbshvc.exe -u rs
```

To stop the service

```
dbshvc.exe -x rs
```

To uninstall the service

```
dbshvc.exe -d rs
```

Starting the Relay Server State Manager automatically

The State Manager process is started automatically when the first Outbound Enabler connects to the Relay Server. This is the default behavior when the start property in the options section of the Relay Server configuration file is not specified or is explicitly specified as auto. The default log file location is *%temp%\ias_relay_server_host.log*.

Starting the Relay Server State Manager automatically with customized options

When auto start is desired but you want to override some default behavior such as verbosity level or log file location, you can use the start property in the options section of the Relay Server configuration file to explicitly specify your State Manager command line. The -f option cannot be used in this case and the configuration file must be named *rs.config* and be placed in the same directory as the server extension.

Note

If you are using IIS, do not specify a log file location under the wwwroot directory. Microsoft IIS does not allow a worker process to create a file under the published tree.

Relay Server State Manager command line syntax

`rshost [option]+`

Parameters

Options The following options can be used to configure the State Manager. They are all optional.

rshost options	Description
-f <i>filename</i>	Indicates the name of the Relay Server configuration file.
-o <i>filename</i>	Indicates the name of the file to use for logging.
-os <i>size</i>	Controls the size of the log file and provides additional information in the log file banner. When <code>-os</code> is specified, the old log is renamed using the <code><yymmdd><nn>.olg</code> format. The log banner is rewritten to the new active log file, with the addition of the machine name, processor architecture, build target and operating system information.
-oq	Prevents a popup window if there is a startup error.
-q	Runs in a minimized window.
-qc	Closes the window on completion.
-u	Updates the configuration of a running Relay Server.
-ua	Archives the log file to <code><yymmdd><nn>.log</code> and truncates the file.

Relay Server configuration file

A Relay Server configuration file is used to define both a Relay Server farm and the back-end server farms connecting to the Relay Server farm.

Each section starts with a section tag. A section tag is formed by enclosing a keyword that identifies the section name in square brackets. For example, `[relay_server]` denotes the start of the Relay Server section.

The section tag is followed by several lines defining various properties related to the section being defined. A property is defined by specifying the property name on the left side of an equal sign and its value on the right side of the equal sign. For example, `property name = value`. All section and property names are case insensitive. Comments are marked with pound sign (#) character at the beginning of a line.

The configuration file should contain only 7-bit ASCII characters. The sections can be specified in any order.

Relay Server section

The Relay Server section is used to define a single Relay Server, so there must be a Relay Server section for each Relay Server in the farm. This section is identified by the `relay_server` keyword.

Relay Server section properties

The following properties can be specified in a Relay Server section:

- **enable** Specifies whether this Relay Server is to be included in the Relay Server farm. Possible values are:
 - **Yes** Indicates that this Relay Server is to be included in the Relay Server farm.
 - **No** Indicates that this Relay Server should not be included in the Relay Server farm.

The default is Yes. This property is optional.

- **host** The hostname or IP address that should be used by the Outbound Enabler to make a direct connection to the Relay Server.
- **http_port** The HTTP port that should be used by the Outbound Enabler to make a direct connection to the Relay Server. A value of **0** or **off** disables HTTP connections. By default, this property is enabled and set to 80.
 - **0 or off** Disable HTTP access from Outbound Enabler.
 - **1 to 65535** Enable HTTP at the specified port.

- **https_port** The HTTPS port that should be used by the Outbound Enabler to make a direct connection to the Relay Server. A value of **0** or **off** disables HTTPS connections. By default, this property is enabled and set to 443.
 - **0 or off** Disable HTTPS access from Outbound Enabler.
 - **1 to 65535** Enable HTTPS at the specified port.
- **description** Enter a custom description to a maximum of 2048 characters. This property is optional.

Backend farm section

The backend farm section specifies the properties of a back-end server farm. A back-end server farm is a group of homogeneous back-end servers. A client making a request through the Relay Server farm must specify the back-end server farm it is targeting. There is one backend farm section for each back-end server farm.

This section is identified by the `backend_farm` keyword.

Backend farm section properties

The following properties can be specified in a backend farm section:

- **active_cookie** Specifies whether or not a cookie is set to retain client-server affinity.
 - **yes** To maintain client-server session affinity, the Relay Server injects a standard HTTP set-cookie command with a proprietary cookie name in the response.
 - **no** An active cookie is not set. Use this option when the backend farm is serving a sessionless browser application. For example, when the backend farm is providing a sessionless SQL Anywhere web service.

For best results, set this control as follows:

Backend server type	active_cookie setting	active_header setting
MobiLink	no	yes
SQL Anywhere	no	no

- **active_header** Specifies whether or not a header is set to maintain client-server session affinity.
 - **yes** To maintain client-server session affinity, the Relay Server injects a proprietary header in the response in case intermediaries tamper with the `active_cookie`.
 - **no** A proprietary header is not set. Setting this option cuts down on traffic volume if the backend farm is serving only browser applications, or if the `active_cookie` is working well for all the clients of this backend farm.

- **backend_security** Specifies the level of security required of an Outbound Enabler in the back-end server farm to connect to the Relay Server farm. The possible values are:

- **on** Indicates that all connections from the back-end farm must be made using HTTPS.
- **off** Indicates that all connections from the back-end farm must be made using HTTP.

This property is optional. If no value is specified, either HTTP or HTTPS can be used to connect.

- **client_security** Specifies the level of security the back-end server farm requires of its clients. The possible values are:

- **on** Indicates that clients must connect using HTTPS.
- **off** Indicates that clients must connect using HTTP.

This property is optional. If no value is specified, clients can connect using either HTTP or HTTPS.

- **description** Enter a custom description to a maximum of 2048 characters. This property is optional.

- **enable** Specifies whether to allow connections from this back-end server farm. Possible values are:

- **Yes** Allow connections from this back-end server farm.
- **No** Disallow connections from this back-end server farm.

The default is Yes. This property is optional.

- **id** The name assigned to the back-end server farm, to a maximum of 2048 characters.

- **verbosity** You can set verbosity to the following levels:

- **0** Log errors only. Use this logging level for deployment. This is the default.
- **1** Request level logging. All HTTP requests are written to the log file.

Errors are displayed regardless of the log level specified, and warnings are displayed only if the log level is greater than 0.

Backend server section

The backend server section defines a back-end server connection. It specifies the information that is used by the Outbound Enabler when it connects to the Relay Server farm on behalf of a back-end server. There is a backend server section for each Outbound Enabler connecting to the Relay Server farm. The backend server section also assigns a back-end server to a back-end server farm.

The following back-end servers are supported for use with the Relay Server:

- MobiLink

- Unwired Server
- Afaria
- Mobile Office
- SQL Anywhere

Refer to your license agreement or supported platforms matrix for information about which back-end servers are supported.

This section is identified by the `backend_server` keyword.

Backend server section properties

The following properties can be specified in a backend server section:

- **description** Enter a custom description to a maximum of 2048 characters. This property is optional.
- **enable** Specifies whether to allow connections from this back-end server. Possible values are:
 - **Yes** Allows connections from this back-end server.
 - **No** Disallows connections from this back-end server.

The default is Yes. This property is optional.

- **farm** The name of the back-end server farm that this back-end server belongs to.
- **id** The name assigned to the back-end server connection, to a maximum of 2048 characters.
- **MAC** The MAC address of the network adapter used by the Outbound Enabler to communicate with the Relay Server. The address is specified using the IEEE 802 MAC-48 format. To get the MAC address in the correct format, look in the Relay Server Outbound Enabler console or log. This property is optional. If it is not specified, MAC address checking does not occur.
- **token** A security token that is used by the Relay Server to authenticate the back-end server connection, to a maximum of 2048 characters. This property is optional.
- **verbosity** You can set verbosity to the following levels:
 - **0** Log errors only. Use this logging level for deployment. This is the default.
 - **1** Request level logging. All HTTP requests are written to the log file.

Errors are displayed regardless of the log level specified, and warnings are displayed only if the log level is greater than 0.

Options section

The options section is used to specify properties that apply to each Relay Server in the farm. Only one options section is allowed.

This section is identified by the options keyword.

Options section properties

The following properties can be specified in an options section:

- **start** The method used to start the State Manager. The possible values are:
 - **auto** The State Manager is started automatically using the State Manager command line defaults.
 - **no** The State Manager is started externally as a Windows service.
 - **full path** Specify the full path to the State Manager executable (*rshost*).

The default is auto. This property is optional.

- **shared_mem** Specifies the maximum amount of shared memory that the Relay Server uses for state tracking. The default is 10 megabytes. This property is optional.
- **verbosity** You can set verbosity to the following levels:
 - **0** Log errors only. Use this logging level for deployment. This is the default.
 - **1** Request level logging. All HTTP requests are written to the log file.

Errors are displayed regardless of the log level specified, and warnings are displayed only if the log level is greater than 0.

Relay Server configuration file format

This is the basic format of a Relay Server configuration file:

```
#
# Options
#
[options]
# List of Relay Server properties that apply to all Relay Servers
option = value

#
# Define a Relay Server section, one for each
# Relay Server in the Relay Server farm
#
[relay_server]
# List of properties for the Relay Server
```

```
property = value

#
# Define a backend server farm section, one for each backend
# server farm
#
[backend_farm]
# List of properties for a backend server farm
property = value

#
# Define a backend server section, one for each
# Outbound Enabler connecting to the Relay Server farm
#
[backend_server]
# List of properties for the backend server connection
property = value
```

Outbound Enabler

The Outbound Enabler runs on the same computer as the back-end server. Its purpose is to:

- Open an outbound connection from the computer running in the corporate LAN to the Relay Server farm running in the DMZ.
- Forward client requests received from the Relay Server to the back-end server and forward back-end server responses back to the client via the Relay Server.

When the Outbound Enabler starts, it makes an HTTP request to retrieve the list of Relay Servers running in the farm. This is done using the server URL that maps to the web server extension component of the Relay Server. The server URL can map directly to a Relay Server or it can map to a load balancer. If the server URL maps to a load balancer, the load balancer forwards the request to one of the Relay Servers running in the farm. The Relay Server that receives the request from the Outbound Enabler returns the connection information for all Relay Servers in the farm. The Outbound Enabler then creates two outbound connections, called channels, to each Relay Server returned. One channel, called the up channel, is created using an HTTP request with an essentially infinite response. The response is a continuous stream of client requests from the Relay Server to the Outbound Enabler. The second channel, called the down channel, is created using an HTTP request with an essentially infinite content length. The request is formed by a continuous stream of server responses to client requests.

When the Outbound Enabler receives a client request on the up channel from one of the Relay Servers it has connected to, it forwards it to the back-end server that the Outbound Enabler is servicing. Once a response is received from the back-end server, it gets forwarded to the Relay Server from which it received the corresponding request using the down channel.

Note

The following back-end servers are supported for use with the Relay Server:

- MobiLink
- Unwired Server
- Afaria
- Mobile Office
- SQL Anywhere

Refer to your license agreement or supported platforms matrix for information about which back-end servers are supported.

Outbound Enabler syntax

rsoe [*option*]+

rsoe @ { *filename* | *environment-variable* } ...

Parameters

Options The following options can be used with the Outbound Enabler. Options that have defaults are optional. At a minimum, the Outbound Enabler needs to supply the connection string for the Relay Server (-cr), the farm (-f) and server (-id) names. If a security token is configured, it must also be specified (-t).

rsoe options	Description
<i>@data</i>	Reads options from the specified environment variable or configuration file. If you want to protect passwords or other information in the configuration file, you can use the File Hiding utility to obfuscate the contents of the configuration file.

rsoe options	Description
<p>-cr <i>"connection-string"</i></p>	<p>Specifies the Relay Server connection string. The format of the Relay Server connection string is a semicolon separated list of name-value pairs. The name-value pairs consist of the following:</p> <ul style="list-style-type: none"> ○ host IP address or hostname of the Relay Server. The default is localhost. ○ port The port the Relay Server is listening on. This is required. ○ http_userid Userid for authentication. Optional. You should consult your web server (or proxy) documentation to determine how to set up HTTP authentication. ○ http_password Password for authentication. Optional. You should consult your web server (or proxy) documentation to determine how to set up HTTP authentication. ○ http_proxy_userid Userid for proxy authentication. Optional. You should consult your web server (or proxy) documentation to determine how to set up HTTP authentication. ○ http_proxy_password Password for proxy authentication. Optional. You should consult your web server (or proxy) documentation to determine how to set up HTTP authentication. ○ proxy_host Specifies the host name or IP address of the proxy server. Optional. ○ proxy_port Specifies the port number of the proxy server. Optional. ○ url_suffix URL path to the server extension of the Relay Server. Optional. <p>By default, the rsoe requires the url_suffix to be specified.</p> <ul style="list-style-type: none"> ○ https 0 - HTTP (default) 1 - HTTPS <p>For https=1, the following options can also be specified:</p> <ul style="list-style-type: none"> ○ tls_type RSA ○ certificate_name Common name field of the certificate. ○ certificate_company Organization name field of the certificate.

rsoe options	Description
	<ul style="list-style-type: none"> ○ certificate_unit Organization unit field of the certificate. ○ trusted_certificate File containing a list of trusted root certificates.
-cs <i>"connection-string"</i>	<p>Sets the host and port used to connect to the back-end server. The default is "host=localhost;port=80".</p> <p>To enable periodic backend server status requests, add the status_url parameter to -cs. The status_url parameter is specified in the format status_url=/your-status-url.</p> <p>The following example shows how to specify status_url with -cs.</p> <pre style="color: red;">-cs "host=localhost;port=80;status_url=/getstatus/"</pre> <p>Use the -d option to specify the frequency of the backend server status requests.</p>
-d <i>seconds</i>	Sets the frequency of the backend server liveness ping and backend server status request. The default is 5 seconds.
-dl	Use this option to display log messages in the Relay Server Outbound Enabler console. By default, log messages are not displayed for verbosity levels 1 and 2.
-f <i>farm</i>	Specifies the name of the farm that the back-end server belongs to.
-id <i>id</i>	Specifies the name assigned to the back-end server.
-o <i>file</i>	Specifies the file to log output messages to.
-oq	Prevents the appearance of the error window when a startup error occurs.
-os	Sets the maximum size of the message log files. The minimum size limit is 10 KB.
-ot	Truncates the log file and logs messages to it.
-q	Run with a minimized window on startup.
-qc	Shuts down the window on completion.
-s	Stops the Outbound Enabler.
-t <i>token</i>	Sets the security token to be passed to the Relay Server.

rsoe options	Description
-uc	<p>Starts the rsoe in shell mode. This is the default. Applies to Unix and Mac OS X.</p> <p>You should only specify one of -uc, -ui, -um, or -ux. When you specify -uc, this starts the rsoe in the same manner as previous releases of the software.</p>
-ud	<p>Instructs the rsoe to run as a daemon. This option applies to Unix platforms only.</p>
-ui	<p>Starts the rsoe in shell mode if a usable display is not available. This option is for Linux with X window server support.</p> <p>When -ui is specified, the server attempts to find a usable display. If it cannot find one, for example because the X window server isn't running, then the rsoe starts in shell mode.</p>
-ux	<p>For Linux, opens the rsoe messages window where messages are displayed.</p> <p>When -ux is specified, the rsoe must be able to find a usable display. If it cannot find one, for example because the DISPLAY environment variable is not set or because the X window server is not running, the rsoe fails to start.</p> <p>To run the rsoe messages window in quiet mode, use -q.</p> <p>On Windows, the rsoe messages window appears automatically.</p>
-v level	<p>Set the verbosity level to use for logging. The <i>level</i> can be 0, 1, or 2:</p> <ul style="list-style-type: none"> ○ 0 Log errors only. Use this logging level for deployment. ○ 1 Session level logging. This is a higher level view of a synchronization session. ○ 2 Request level logging. Provides a more detailed view of HTTP requests within a synchronization session. <p>Levels 1 and 2 are only written to the log file and are not displayed. To have all log messages displayed, use the -dl switch.</p>

File Hiding utility (dbfhide)

The File Hiding utility (dbfhide) uses simple encryption to obfuscate the contents of configuration files and initialization files.

Syntax

dbfhide *original-configuration-file encrypted-configuration-file*

Option	Description
<i>original-configuration-file</i>	Specifies the name of the original file.
<i>encrypted-configuration-file</i>	Specifies a name for the new obfuscated file.

The Relay Server and Outbound Enabler detect that a configuration file has been obfuscated using dbfhide and process it.

This utility does not accept the @data parameter to read in options from a configuration file.

Deployment considerations

The following considerations should be noted when using the Outbound Enabler:

- **Outbound Enabler as a service** The Outbound Enabler may also be set up and maintained as a service using the Service utility.
- **Authentication** You cannot use simple or digest authentication. The *rsoe.exe* does not support simple or digest authentication with web servers, regardless of the web server type or operating system.

Outbound Enabler as a service

The Outbound Enabler can be started as a service by using the Service utility (*dbsvc*). The Service utility is used to create, modify and delete services. For a full listing of usage information, run *dbsvc* without any options.

To set up an auto-started RSOE service named oes (Outbound Enabler service) on Windows

```
dbsvc -as -s auto -t rsoe -w oes "%SQLANY12%\BinXX\rsoe.exe"
-cr "host=relayserver.sybase.com;port=80 " -cs "host=localhost;port=80 " -f
FarmName -id ServerName -ttoken
```

To set up an auto-started RSOE service named oes (Outbound Enabler service) on Unix

```
dbsvc -y -a <some-user-account> -t rsoe -w oes @/<full-dir-path>/oe.config
```

Remarks

The syntax of *dbsvc* on Windows is different than Unix. In Unix, you do not specify the full path of the executable as the first parameter after *-w* switch argument.

Use full paths only.

On Unix, specify the Outbound Enabler parameters in a command file only. Do not use command line switches in the setup *dbsvc* command.

Start the service

```
dbsvc.exe -u oes
```

To stop the service

```
dbsvc.exe -x oes
```

To uninstall the service

```
dbsvc.exe -d oes
```

Updating a Relay Server farm configuration

A Relay Server farm configuration is defined by the contents of the Relay Server configuration file. Each Relay Server in a Relay Server farm shares the same Relay Server configuration file, so when you update a Relay Server farm configuration you must update the Relay Server configuration file at each Relay Server in the farm. Updates include any of the following:

- Adding a new Relay Server to the Relay Server farm.
- Creating a new backend server farm and allowing it access to the Relay Server farm.
- Adding a new backend server to an existing backend server farm.
- Changing the properties of a Relay Server, backend server farm, or a backend server.
- Changing options.

One way to update a Relay Server configuration is to shutdown all Relay Servers, replace the Relay Server configuration file with the updated version, and restart all the Relay Servers. However, shutting down and restarting the Relay Servers means that users of the Relay Server may incur a service interruption.

The preferred method of updating a Relay Server configuration is to use the Relay Server State Manager to update the configuration while a Relay Server farm is running without interrupting service.

Updating a Relay Server configuration is done by launching a new instance of the Relay Server State Manager using the following command line format:

```
rshost -u -f filename
```

The `-u` option instructs the Relay Server State Manager to perform an update operation. The `-f` option specifies the name of the configuration file containing the updated configuration.

Below is an overview of the steps required to update a Relay Server farm configuration:

1. Make your changes to the master copy of the Relay Server configuration file.
2. On each computer running an instance of a Relay Server that belongs to the Relay Server farm being updated, do the following:
 - a. Replace the old configuration file with the updated configuration file.
 - b. Run the Relay Server State Manager with the updated configuration file.

Updating a Relay Server configuration for Microsoft IIS on Windows

Update a Relay Server configuration for Microsoft IIS on Windows

1. For each computer that belongs to the Relay Server farm you are updating, copy the updated configuration file to the `%SQLANY12%\RelayServer\IIS\BinXX\Server` directory under the Relay Server web site home directory. The configuration file must be called `rs.config` if auto start is used.
2. From the `%SQLANY12%\RelayServer\IIS\BinXX\Server` directory, run the following command line to apply the configuration update:

```
rshost -u -f rs.config
```

3. Repeat the previous steps for each computer in the Relay Server farm that is being updated.

Updating a Relay Server configuration for Apache on Linux

Update a Relay Server configuration for Apache on Linux

1. Copy the updated configuration file to the `/modules` directory under the Apache install directory. The configuration file must be called `rs.config` if auto start is used.
2. From the `/Apache-install/modules` directory, run the following command line to apply the configuration update:

```
rshost -u -f rs.config
```

3. Repeat the previous steps for each computer in the Relay Server farm that is being updated.

Sybase Hosted Relay Service

The Sybase Hosted Relay Service is a farm of Relay Servers hosted by Sybase. It is intended to ease the development of mobile applications that use MobiLink data synchronization and to simplify the evaluation process for developers, especially where data is sent using public wireless networks. You do not need to ask your IT department to install anything or open any holes in your corporate firewall. All communication between MobiLink and the hosting service uses HTTP(S) via an outbound connection initiated by MobiLink.

The Sybase Hosted Relay Service is not intended for production deployments. Before deploying your production application, you must first install the Relay Server in your own corporate infrastructure.

Using the Sybase Hosted Relay Service

Subscribing to the Sybase Hosted Relay Service

Subscribe to the Sybase Hosted Relay Service

To use the Sybase Hosted Relay Service you must first subscribe to it.

1. Go to <http://relayserver.sybase.com/account>. This takes you to the Sybase Hosted Relay Service home page.
2. Create an account by clicking **Register**.
3. You are asked to specify a **Subscription ID** (choose one that is unique to your organization) and **Password**, provide contact information for your self and your organization, and agree to the **Hosted Relay Service Terms of Service**. Click **Submit**.

Once you have successfully registered, an email is sent to you confirming your registration.

Logging in to the Sybase Hosted Relay Service

Log in to the Sybase Hosted Relay Service

1. Log in to your newly created account by clicking **Log In**.
2. Enter the **Subscription ID** and **Password** you entered during the registration process. Once logged in, you are taken to the **Account Information** page. The account information page allows you to modify subscriber information and specify the back-end server farm(s) that will be accessing this service.

Adding a server farm

Add a server farm

1. Click to select the type of farm you want to add. Choose from **Add New MobiLink Farm**, **Add New Afaria Farm**, and **Add New iAnywhere Mobile Office Farm**.

2. Enter a unique **Farm Name** to describe the server farm.
3. Provide a unique name for each server in the farm. You can specify a maximum of two servers.
4. Click **Create Farm**. A confirmation is displayed if the farm was successfully added.
5. Click **Configuration Instructions** to learn more about using the service. The instructions are based on the information you provided.
6. Click **Log Out** when you are done.

Using MobiLink with the Relay Server

The following sections provides information about using the Relay Server with MobiLink.

For information about which operating systems and browsers are supported for the Relay Server, see <http://www.sybase.com/detail?id=1002288>.

Connecting a client to the Relay Server farm

Once a Relay Server farm has been properly configured, a client connects to the Relay Server farm using the following URL:

`http://<Relay Server client extension URL>/<farmname>`

Options

Option	Description
<code><Relay Server client extension URL></code>	For Microsoft IIS on Windows, <code><domain name><relayserver.sybase.com>/ias_relay_server/client/rs_client</code> For Apache on Linux, <code><domain name><relayserver.sybase.com>/cli/iarelayserver</code> Use <code>relayserver.sybase.com</code> as the <code><domain name></code> if you are using the publicly available Sybase Hosted Relay Service.
<code><farmname></code>	Identifies the back-end farm (a group of back-end servers) that Relay Server forwards the client request to.

SQL Anywhere MobiLink client connection example

A SQL Anywhere MobiLink client should specify the following options to connect to server farm **F1**:

```
-e "ctp=http;  
    adr='host=relayserver.sybase.com;  
        url_suffix=url_suffix=/rs/client/rs_client.dll/F1' "
```

For HTTPS, change `http` to `https`.

UltraLite/UltraLiteJ MobiLink client connection example

An UltraLite/UltraLiteJ MobiLink client should set the following properties in the `ULSyncParams` class to connect to server farm **F1**:

- Set the stream type to `HTTPS`.
- Set the stream parameters to the following:

```
"host=relayserver.sybase.com:url_suffix=/rs/client/rs_client.dll/F1"
```

QAnywhere client connection example

A QAnywhere client should specify the following options to connect to server farm **F1**:

```
-x "http(host=relayserver.sybase.com:url_suffix=/rs/client/rs_client.dll/F1"
```

Sample scenario

Suppose company ABC has developed a mobile application and now wants to set up the deployment runtime to service the mobile application. Initially, the mobile deployment consists of 10000 devices and grows in the future. The customer therefore wants a fault tolerant and load-balanced environment that is able to handle the load today and be easily extended to handle more mobile deployments in the future. Based on the data synchronization characteristics of the mobile application, the customer has determined that the following configuration is needed:

- 2 MobiLink servers
- 2 Relay Servers
- 1 load balancer

Since the company uses Microsoft IIS as its web server, the Microsoft IIS version of the Relay Server is used.

Notes

- Each Relay Server is deployed on its own computer. Two computers, with host names **rs1.abc.com** and **rs2.abc.com** are used.
- Each MobiLink server is deployed on its own computer. The two MobiLink servers are assigned names **ml1** and **ml2** and belong to the back-end server farm called **abc.mobilink**.
- The load balancer is addressable using the host name **www.abc.com**.
- For maximum security, HTTPS is used by all clients and Outbound Enablers connecting to the Relay Servers. It is assumed that all web servers are equipped with a certificate from a well known Certificate Authority (CA), and the back-end server computers all have the corresponding trusted root certificates in their standard certificate store.

Set up the Relay Server farm

1. The first step is to create the Relay Server configuration file.

The filename containing the configuration must be called **rs.config**. For this particular scenario, the following configuration file is used:

```
#  
# Options  
#  
[options]
```

```

verbosity = 1

#
# Define the Relay Server farm
#
[relay_server]
host = rs1.abc.com

[relay_server]
host = rs2.abc.com

#
# Define the MobiLink backend server farm
#
[backend_farm]
id = abc.mobilink
client_security = on
backend_security = on

#
# List MobiLink servers that are connecting to the Relay Server farm
#
[backend_server]
farm = abc.mobilink
id = ml1
token = mltoken1

[backend_server]
farm = abc.mobilink
id = ml2
token=mltoken2

```

2. Deploy the configuration file *rs.config* along with the Relay Server components to the two computers that are running the Relay Server.
3. Start MobiLink server on the two computers that are running the MobiLink servers using the Integrated Outbound Enabler.

On the computer running MobiLink server with id ml1:

```
mlsrv12 -x oe<config=oe1.txt> -zs ml1 <other ML options>
```

where oe1.txt = -f abc.mobilink -id ml1 -t mltoken1 -cr
"host=www.abc.com;port=443;https=1"

On the computer running MobiLink server with id ml2:

```
mlsrv12 -x oe<config=oe2.txt> -zs ml2 <other ML options>
```

where oe2.txt = -f abc.mobilink -id ml2 -t mltoken2 -cr
"host=www.abc.com;port=443;https=1"

4. Once all servers and Outbound Enablers are running, MobiLink clients are able to connect to the farm using the following connection information:
 - **HTTPS** protocol
 - **host** www.abc.com

- **url_suffix** /rs/client/rs_client.dll/abc.mobilink

Index

A

- Apache
 - deploying the Relay Server, 13
- application pool
 - creating, 7
- architectures
 - Relay Server, 1

B

- back-end server farm
 - Relay Server, 4
- backend farm section
 - Relay Server configuration file, 22
- backend server section
 - Relay Server configuration file, 23

C

- client
 - connecting to a Relay Server farm, 39
- configuration files
 - Relay Server, 21
 - Relay Server format, 25

D

- deploying
 - Relay Server, 7

H

- hosted Relay Server
 - adding a server farm, 37
 - logging in, 37
 - subscribing, 37
- HTTP load balancer
 - Relay Server, 4

L

- load balancer
 - HTTP, 4

M

- Microsoft IIS
 - Relay Server, performance tips, 9
- Microsoft IIS 6.0

- deploying the Relay Server, 7
- Microsoft IIS 7.x
 - deploying the Relay Server, 10
- mobile device
 - connecting to a Relay Server farm, 39
- MobiLink
 - using the Relay Server, 39

O

- options section
 - Relay Server configuration file, 25
- Outbound Enabler
 - about, 27
 - deployment considerations, 32
 - Relay Server farm, 4
 - syntax, 27

R

- Relay Server
 - about, 1
 - architecture, 1
 - back-end server farm, 4
 - configuration file, 21
 - deployment, 7
 - hosted service, 37
 - Outbound Enabler, 27
 - Outbound Enabler deployment, 32
 - Outbound Enabler syntax, 27
 - Relay Server farm, 4
 - rshost syntax, 19
 - rsoe syntax, 27
 - sample scenario for MobiLink, 40
 - State Manager, 17
 - state manager command line syntax, 19
 - synchronizing through a web server, 1
 - updating configuration, 35
 - using MobiLink, 39
- Relay Server configuration file
 - about, 21
 - backend farm section, 22
 - backend server section, 23
 - farm configuration, 35
 - format, 25
 - options section, 25
 - procedure to update, Apache, 36
 - procedure to update, Microsoft IIS, 36
 - Relay Server section, 21

- updating, 35
- Relay Server deployment
 - Apache on Linux, 13
 - application pool, Microsoft IIS 6.0, 7
 - files for Linux, 13
 - files for Windows IIS 6.0, 7
 - Microsoft IIS 6.0 on Windows, 7
 - Microsoft IIS 7.x on Windows, 10
 - web server extensions, Apache, 13
 - web server extensions, Microsoft IIS 6.0, 7
 - web server extensions, Microsoft IIS 7.x, 11
- Relay Server farm
 - connecting a client, 39
 - connecting a mobile device, 39
- Relay Server farm configuration
 - updating, 35
- Relay Server hosting service
 - (*see also* Sybase hosted relay service)
- Relay Server Outbound Enabler (*see* Outbound Enabler)
- Relay Server performance tips
 - Microsoft IIS, 9
- Relay Server section
 - Relay Server configuration file, 21
- Relay Server State Manager
 - about, 17
 - command line syntax, 19
 - Relay Server deployment, Apache, 13
 - Relay Server deployment, Microsoft IIS 6.0, 7
 - Relay Server deployment, Microsoft IIS 7.x, 11
 - starting as a Windows service, 17
 - starting automatically, 18
 - starting automatically with custom options, 18
- Relay Server web extensions
 - about, 1
 - deploying, Microsoft IIS 6.0, 7
 - deploying, Microsoft IIS 7.x, 11
- relaysrv (*see* Relay Server)
- rshost (*see* Relay Server State Manager)
- rsoe (*see* Outbound Enabler)

S

- server farm
 - Relay Server, 4
- services
 - running the State Manager as a service, 17
- state management

- Relay Server, 17
- State Manager
 - command line syntax, 19
 - options, 19
- Sybase hosted relay service
 - about, 37
 - adding a server farm, 37
 - logging in, 37
 - subscribing, 37
- synchronizing through a web server
 - Relay Server, 1
- syntax
 - Outbound Enabler, 27

W

- web extensions
 - Relay Server, 1