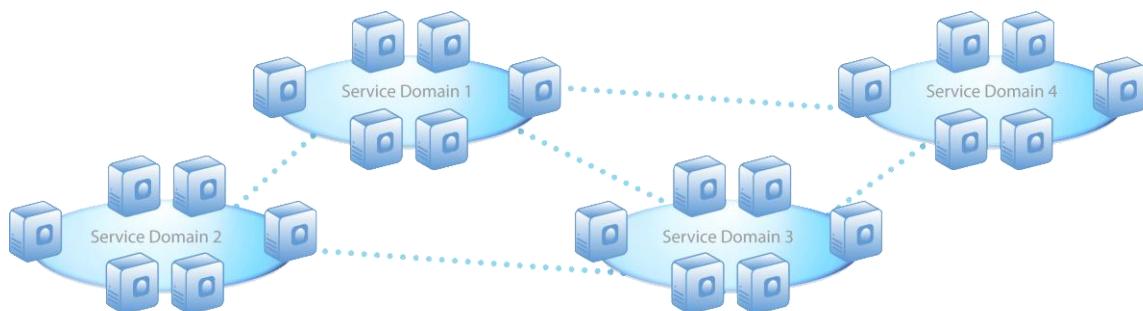


.NET StockTrader 5.0

Installation and Configuration

*Reconfiguring .NET StockTrader and Configuration Service 5.0
Overview*

6/9/2011
© Microsoft Corporation 2011



THIS IS NOT A PRODUCT SPECIFICATION.

This document and related sample code supports Windows Server® 2008 R2, Windows Azure, SQL Azure, Azure AppFabric and the Microsoft .NET Framework 4.0 as a redistributable sample application kit.

The information contained in this document represents the current view of Microsoft Corp. on the issues disclosed as of the date of publication. Because Microsoft must respond to changing market conditions, this document should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented. This document is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS DOCUMENT.

Information in this document, including URL and other Internet Web site references, is subject to change without notice. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Microsoft grants you the right to reproduce this guide, in whole or in part.

Microsoft may have patents, patent applications, trademarks, copyrights or other intellectual property rights covering subject matter in this document, except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights or other intellectual property.

© 2011 Microsoft Corp. All rights reserved.

Microsoft, Windows Server, Windows Azure, SQL Azure, SQL Server, the Windows logo, Windows, Active Directory, Windows Vista, Visual Studio, Internet Explorer, Windows Server System, Windows NT, Windows Mobile, Windows Media, Win32, WinFX, Windows PowerShell, Hyper-V, and MSDN are trademarks of the Microsoft group of companies.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Contents

Introduction to ConfigWeb	3
The Basic Deployment Options.....	3
Remote Modes and Advanced Web Service Security.....	5
Securing Business Services and Order Processor with Service Certificate and Custom Username Validation.....	5
Securing Order Processor with Service Certificate and Custom Client Certificate Validation.....	5
Remote Modes and Interoperability	5
Initial ConfigWeb Login and Reconfiguration Steps	6
The Service Map View.....	25
Back to StockTrader as a User.....	27
The Business Service Self Host Program and TCP/Binary Service Endpoints.....	28
Different Hosts—Different Configuration Repositories.....	39
Monitoring Exception and Trace Logs	40
Azure StockTrader: Re-Configuring for Remote Modes	42
Adding Scale-Out, Load Balanced Nodes	72
Azure Security Discussion and Changing the Security Configuration for WCF Endpoints	75
User Name Credentials	75
Where Does the User Name Come from and How is it Authenticated?	76
How the User Name Gets Passed and Authenticated on Requests.....	82
Changing Passwords so others Cannot Access Our Services	83
Client Certificate Credentials	85
How the Client Certificate Gets Validated, and What Certificate is Valid as a Client Credential for the StockTrader Business Services?	95
Changing the StockTrader Web Application to Supply a Client Certificate to the StockTrader Business Services.	97
Changing the Samples to Use Different Certificates vs. Default Install Certificates.....	110
Configuring the MSMQ Durable Message Queue Endpoint for Order Processing On-Premise StockTrader	110
Activating and Using the Order Processor Transacted MSMQ Endpoint	110
New Hosts and Configuration Service Dynamic URI Management	127
Re-Configuring StockTrader to Work with Oracle 11G	127

Separation of Implementation from Schema and Contract	129
Why A Configuration Service and ConfigWeb?.....	129
Starting Point for Implementing the Configuration Service	130

Introduction to ConfigWeb

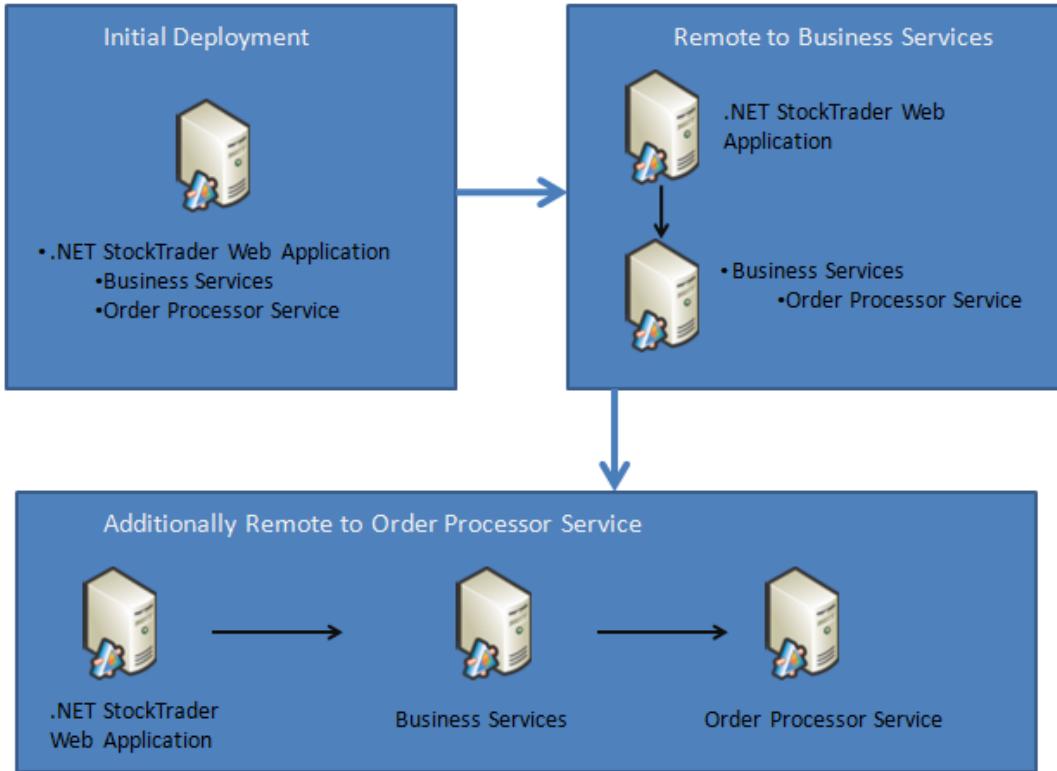
This document provides the basic steps used to reconfigure the .NET StockTrader 5 using the ConfigWeb management site. ConfigWeb is a separate Web application installed with .NET StockTrader that can remotely attach to any service or application that implements the Configuration Service 5.0. To access ConfigWeb, simply click on the ConfigWeb menu item within the .NET StockTrader Web application (<http://localhost/trade>), or the StockTrader Windows Start menu group. The configuration system utilized by StockTrader is contained in separate base classes and assemblies such that any developer, if desired, can implement the system in their own applications or services to help manage distributed application configuration data, and achieve load balancing and failover at the service operation level. A Visual Studio template and wizard is provided, and you can read the related document *Using the Configuration Service 5.0 Visual Studio Template*.

The Basic Deployment Options

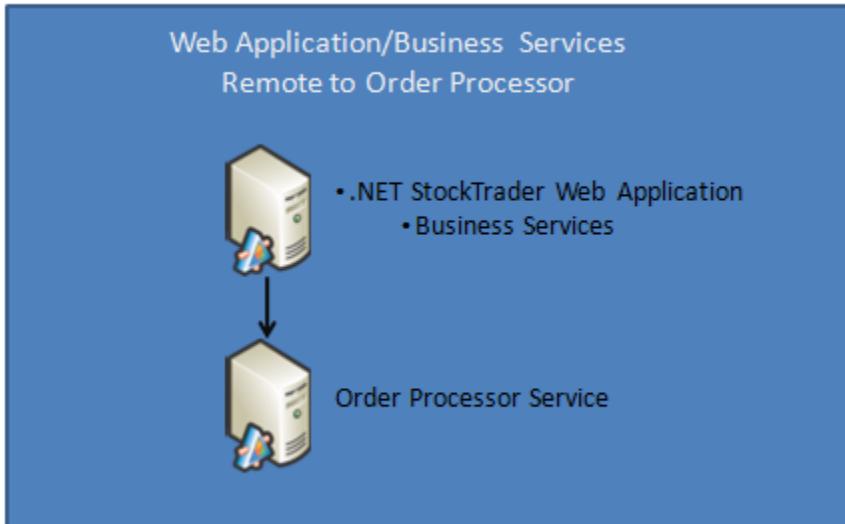
When logging into ConfigWeb, you can directly specify an address to any of the core StockTrader components that are running, including:

- .NET StockTrader Web application
- .NET StockTrader Business Services (hosted in IIS or in a .NET self-host program, accessible via the Windows Start menu)
- .NET StockTrader Order Processor Service (hosted in a .NET self-host program, accessible via the Windows Start menu)

These elements are initially configured on a default install to **run within a single ASP.NET process, without utilizing any remote services**, although the tiers are still logically partitioned within the host ASP.NET worker process. The relationships that can be changed look as follows (you can explore all of these options using a single computer, as each service host is configured to utilize separate ports):



Additionally, the following deployment option is possible:



Remote Modes and Advanced Web Service Security

In .NET StockTrader 5.0, two optional modes demonstrate WS-* security. Activating and using both modes are illustrated in step-by-step procedures in a different document, see <http://msdn.microsoft.com/stocktrader> for details. Neither of these modes is active by default for an on-premise application. However, for the Azure deployment, one of these secure modes is always used.

Securing Business Services and Order Processor with Service Certificate and Custom Username Validation

The first option demonstrates using an X.509 service certificate over a ws2007HttpBinding or netTcpBinding with a security mode of TransportWithMessageCredentials. Client credentials are set to Username (no certificate is required on the client; messages are encrypted within the transport layer), and this mode demonstrates authenticating users using a custom username store, in this case directly using the Configuration Service Users table to authenticate users connecting to the business service layer (BSL). A base class provided within the Configuration Service does the validation, using the standard .NET overrides (System.Security) to perform custom validation of the user name/password. By default, this base class validates the incoming credentials against the Configuration Service Users table (which is a convenient way to store such credentials for authentication); however this can easily be replaced with other logic to use other credential stores such as Active Directory, custom LDAP stores, etc.

Securing Order Processor with Service Certificate and Custom Client Certificate Validation

The second option illustrates client credentials set to Certificate (service and client each require their own X.509 certificate; client certificate is used for identification/authorization to the remote services). The client certificate is validated using a custom certificate validator that checks the thumbprint of the certificate, only allowing access to the secured endpoint for specific client certificates specified in the validator class. The validator class, like the UserName validator, is provided as a base class with the Configuration Service, using the standard .NET X.509 framework classes to do the validation.

Remote Modes and Interoperability

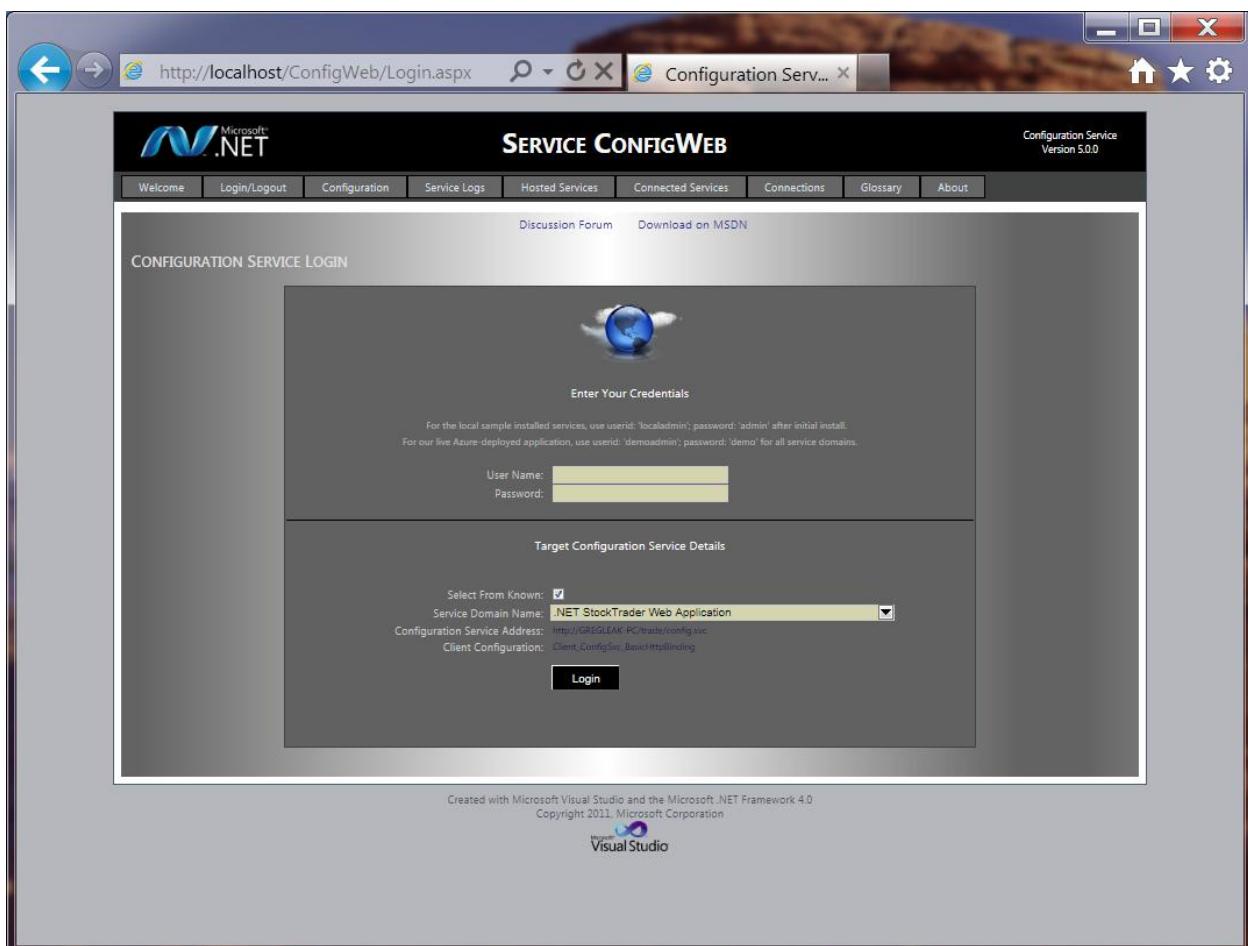
Several additional Connected Service definitions ship with the default install of .NET StockTrader 5.0 that demonstrate interoperability with non-Microsoft platforms. Third-party elements are required for these modes to operate:

- .NET StockTrader ASP.NET Web Application and/or .NET StockTrader smart client to IBM WebSphere 7.x Trade Business Service Layer over SOAP. This mode requires installation of IBM WebSphere 7 with the Trade 7 sample installed. Note that the IBM WebSphere Trade 7 Java Server Pages Web application can also be used seamlessly with the .NET middle tier services. The Java-based Trade 7 application is included in the StockTrader download.

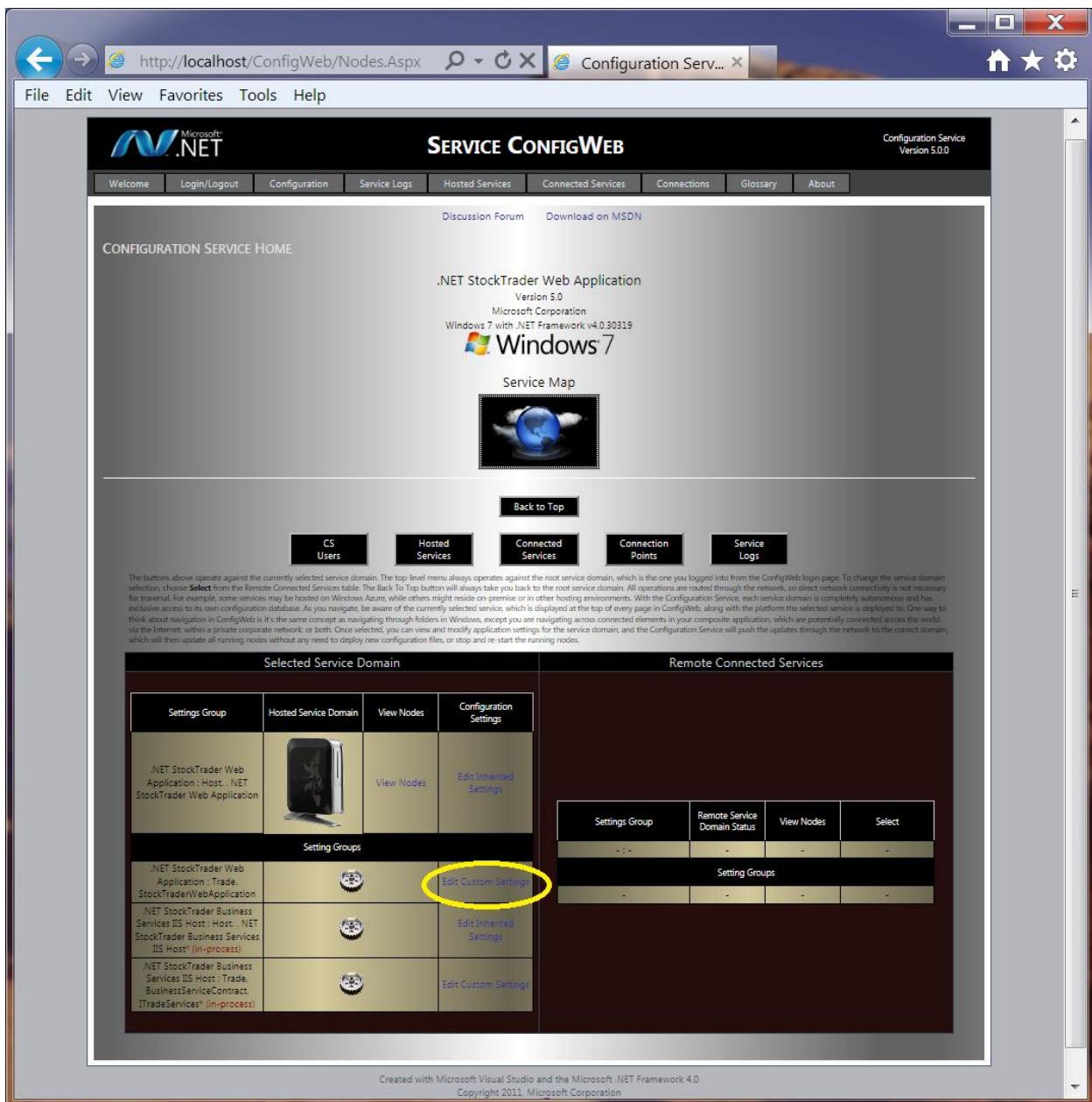
Initial ConfigWeb Login and Reconfiguration Steps

To get started, we will use StockTrader to provide an overview of using the Configuration Service and ConfigWeb for basic changes to the modes in which the composite application operate. Later, we will reconfigure services for the advanced security modes. Note that after the initial install of .NET StockTrader, the application is fully functional (operating in a no-services mode as a standalone Web application); so you can already run the .NET StockTrader Web application at <http://localhost/trade> before completing any steps in this guide. You can also run the smart client, accessible via the Windows Start menu.

To reconfigure .NET StockTrader to use a remote service tier for the business service layer, log in to ConfigWeb (<http://localhost/configweb>) by pointing to the **StockTrader Web application configuration service**. You can login as username '**localadmin**', password '**admin**'. The drop-down selection (which can be customized) shows configuration addresses for the StockTrader services and other samples included with the download, and the first address listed is pre-specified to point to the .NET StockTrader Web application Configuration Service. You can change the server name by deselecting **Select from Known List**, otherwise by default the server hosting ConfigWeb will be listed.



On login you will see the configuration home page, as follows:



In the following steps, we will reconfigure the StockTrader Web application to connect to remote Business Services, hosted in IIS. To do so, click on the Edit Configuration link for the StockTrader Web application circled in the image above.

This will bring you to the main configuration settings for the StockTrader Web application. These are depicted below. Settings are separated into three groupings: Basic, Detailed and Advanced.

<http://localhost/ConfigWeb/Config.aspx?n=1>

Configuration Service Version 5.0.0

Discussion Forum [Download on MSDN](#)

CONFIGURATION KEYS

.NET StockTrader Web Application
Version 5.0
Microsoft Corporation
Windows 7 with .NET Framework v4.0.30319


[Back to Top](#)

Inherited settings are simply application global settings inherited from a base Configuration Service settings class. Custom settings are application-specific settings the developer has defined for the specific application. These can be any settings you might otherwise have defined in the <appSettings> section of a .NET Web.config or App.Config file. While stored in the central configuration database, these settings are cached locally in-memory, and used just like they came from a config file (although no code is necessary to populate them into local variables). The in-memory cache means access is as fast as accessing a local static variable. But, Configuration Service keeps all nodes in sync when configuration updates are made on one system—without having to deploy new application configuration files across nodes, and stop/re-start running nodes.

[Basic](#) [Detailed](#) [Advanced](#)

[Define Keys](#)

Target Settings Group: Trade.StockTraderWebApplication
Custom Application Settings

Setting Name	Current Value	Description	Change Value
AccessMode	In-Process Activation	This setting determines how the Trade Web application makes calls to the Business Logic Layer (BSL) via .NET Remoting. It allows for a wide variety of settings. First, there are three distinct BSL layers hosting service endpoints: 1) Windows Azure running in a Web Role; 2) An on-premise IIS-based service; 3) An on-premise EXE application that ships with the sample. For each, there are a variety of transports and security models illustrated. The Web application will only use one service endpoint at a time; so this setting is a switch that can for example be changed to have the Web application start communicating to our on-premise BSL vs. the Azure-hosted BSL with a simple update from ConfigWeb.	Change Value
Always Check Order Alerts	false	Whether to run the closed orders check on every visit to an authenticated page to display the Order Alert message, or to use the check frequency setting value.	Change Value
Check Order Alert Frequency	15	Time in seconds to check for closed orders; only used if CheckOrderAlertsOnEveryRequest is false	Change Value
DisplayOrdersDefault	10	This setting determines the default number of orders to display in the Account.aspx page.	Change Value
DisplayOrdersMax	100	This setting determines the maximum number of orders to display in the Account.aspx page.	Change Value

[Return to Home Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation


Click on the **Change Value** link circled above for the AccessMode setting. You will next be able to update the configuration to one of several different ways you can connect to remote Business Services. The AccessMode setting determines which connection types to make active and utilize while users are using the StockTrader Web application (<http://localhost/trade>).

For now, we will change the setting to IIS-Hosted .NET 4.0 services, corresponding to a setting of **On-Premise IIS Host BSL Basic Http SOAP**, which you should select as depicted. Note that note all modes are initially active. Also note that if you deploy the StockTrader Business Services to Windows Azure, this is where you could link your on-premise StockTrader Web application to cloud-hosted services on Azure.

.NET StockTrader Web Application
Version 5.0
Microsoft Corporation
Windows 7 with .NET Framework v4.0.30319
Windows 7

[Back to Top](#)

You can change a setting's value here, and all running nodes will automatically receive the new setting and be updated. The setting update will be pushed through the network to the correct service domain automatically. Once updated, new values for that configuration always start up with the update value since it has been pushed into the configuration store, which nodes read on startup. Running nodes are updated live via the node service and reflection. **AccessMode** is provided on purpose, but developers can easily add their own server-side validation logic for custom logic to perform custom actions across all nodes when specific settings change, if they desire. For StockTrader, a key customer setting for the Web application is **AccessMode**, which determines which Business Service domain to utilize, and over which network/encoding/security standard. The **OrderMode** setting for the StockTrader Business Service domain serves a similar purpose, specifying which remote Order Processor Service domain to use, and via what type of communication channel.

Setting Name	Current Value	Setting Description	Valid Values	Last Updated
AccessMode	In-Process Activation	This setting determines how the Trade Web application makes calls to the Business Service Layer (BSL) over WCF. As a sample application, there are a wide variety of settings. First, there are three distinct BSL layers hosting service endpoints: 1) Windows Azure running in a Web Role; 2) An on-premise IIS-based service; 3) An on-premise EXE application that ships with the sample. For each, there are a variety of transports and security modes illustrated. The Web application will only use one service endpoint at a time, so this setting is a switch that can or cannot be changed to have the Web application start communicating to our on-premise BSL vs. the Azure-hosted BSL with a simple update from ConfigWeb.	In-Process Activation, Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate Azure BSL wsHttp security = TransportWithMessageCredential: UserName Azure BSL netTcp security = TransportWithMessageCredential: ClientCertificate Azure BSL netTcp security = TransportWithMessageCredential: UserName On-Premise Self-Host BSL http On-Premise Self-Host BSL netTcp On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: UserName On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: UserName On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate On-Premise IIS Host BSL wsHttp security = Message: ClientCertificate On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: UserName On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: UserName Interop: Basic Http SOAP BSL	5/31/2011 8:12:45 AM

Enter/Select New Value:

- In-Process Activation
- Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate
- Azure BSL wsHttp security = TransportWithMessageCredential: UserName
- Azure BSL netTcp security = TransportWithMessageCredential: ClientCertificate
- Azure BSL netTcp security = TransportWithMessageCredential: UserName
- On-Premise Self-Host BSL http
- On-Premise Self-Host BSL netTcp
- On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate
- On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: UserName
- On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate
- On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: UserName
- On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate
- On-Premise IIS Host BSL wsHttp security = Message: ClientCertificate
- On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate
- On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: UserName
- On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate
- On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: UserName
- Interop: Basic Http SOAP BSL

[Update](#)

[Return to Configuration Page](#)

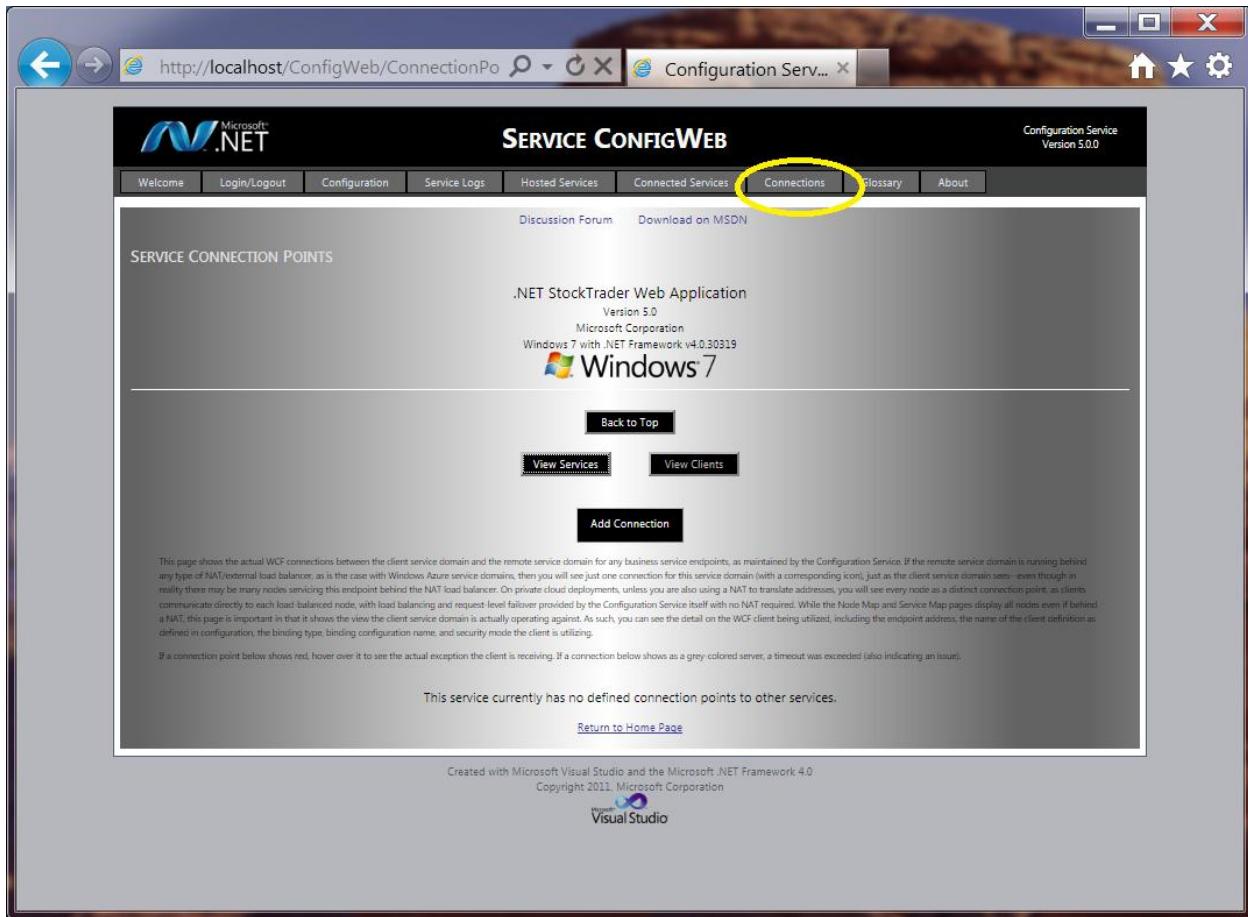
Click on the **Update** button to make the setting active. If your StockTrader Web application has been deployed to many nodes, or is scaled out on Windows Azure, all nodes will be immediately synchronized with the new setting.

Next, click on the menu item **Configuration** to return to the main ConfigWeb page:

The screenshot shows a web browser window with the URL <http://localhost/ConfigWeb/Nodes.aspx>. The title bar reads "Configuration Serv...". The main content area is titled "SERVICE CONFIGWEB" and displays the "CONFIGURATION SERVICE HOME". A yellow circle highlights the "Configuration" tab in the top navigation bar. Below the tabs, there are links for "Discussion Forum" and "Download on MSDN". The central part of the page shows ".NET StockTrader Web Application Version 5.0 Microsoft Corporation Windows 7 with .NET Framework v4.0.30319". It features a "Service Map" icon and several navigation links: "Back to Top", "CS Users", "Hosted Services", "Connected Services", "Connection Points", and "Service Logs". A note at the bottom explains the navigation system. The main table, "Selected Service Domain", lists two entries: ".NET StockTrader Web Application : Host .NET StockTrader Web Application" and ".NET StockTrader Web Application : Trade .StockTraderWebApplication". The "Remote Connected Services" table is mostly empty, with only a header row visible.

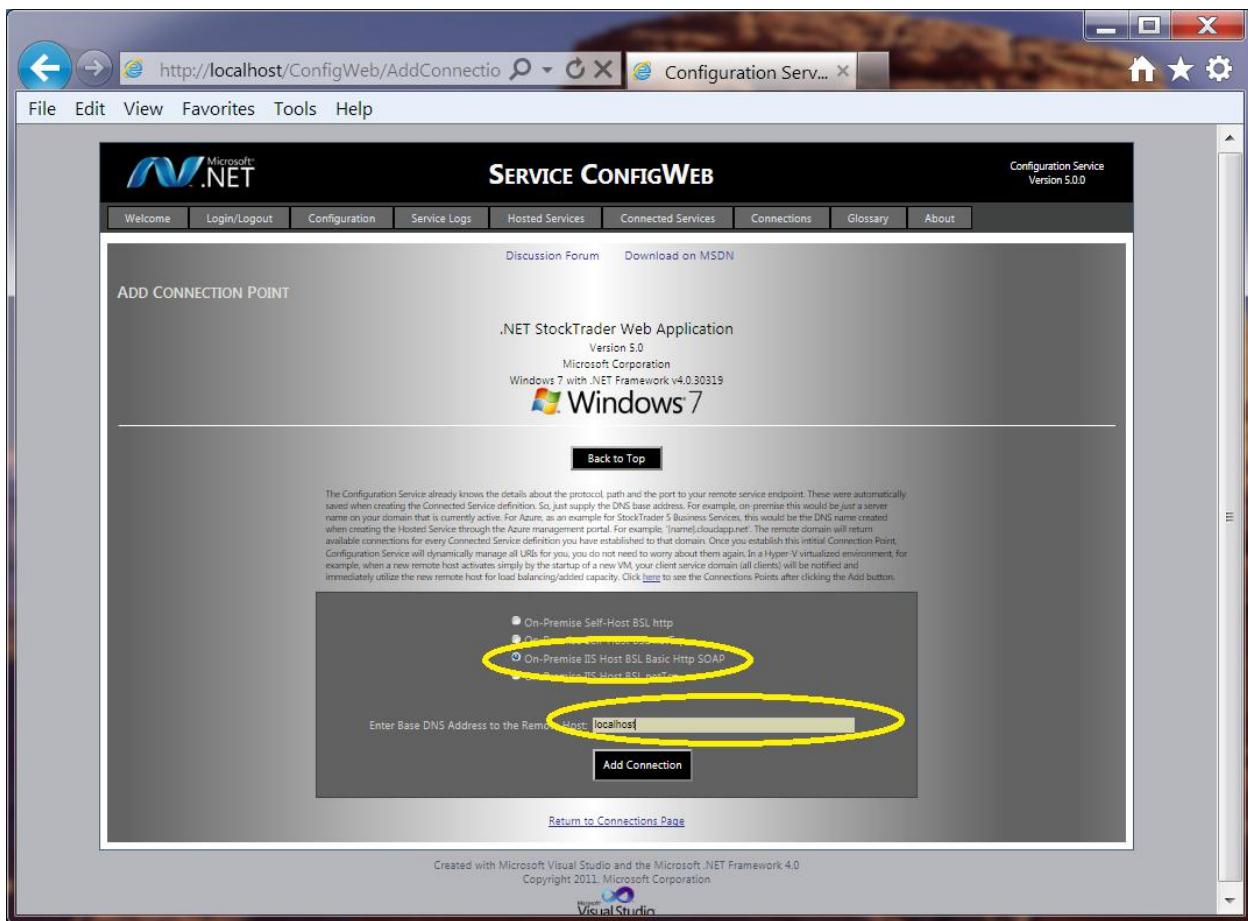
You will notice that we now have Business Services (and Order Processor Service) no longer showing up under the root StockTrader Web application, because all business services (and all connectivity to the StockTraderDB database) are now set to be remotely activated by a separate business service tier running in a different **physical** tier. However, there are no 'Remote Connected Services' showing up either. The .NET StockTrader Web application needs to be connected to a running instance of the

remote Business Services. To do so, you will click on the **Connection** menu item to add an initial connection point:



The screenshot shows a Microsoft .NET Configuration Service window titled "SERVICE CONFIGWEB". The top navigation bar includes links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, **Connections** (which is circled in yellow), Glossary, and About. Below the navigation bar, there are links for Discussion Forum and Download on MSDN. The main content area is titled "SERVICE CONNECTION POINTS" and displays information for ".NET StockTrader Web Application" (Version 5.0, Microsoft Corporation, Windows 7 with .NET Framework v4.0.30319). It also shows the Windows 7 logo. At the bottom of this section are buttons for Back to Top, View Services, View Clients, and Add Connection. A note below the application details states: "This page shows the actual WCF connections between the client service domain and the remote service domain for any business service endpoints, as maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/external load balancer, as is the case with Windows Azure service domains, then you will see just one connection for this service domain (with a corresponding icon), just as the client service domain sees—even though in reality there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly to each load-balanced node, with load balancing and request-level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all nodes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the WCF client being utilized, including the endpoint address, the name of the client definition as defined in configuration, the binding type, binding configuration name, and security mode the client is utilizing." It also notes: "If a connection point below shows red, hover over it to see the actual exception the client is receiving. If a connection below shows as a grey colored server, a timeout was exceeded (also indicating an issue)." At the very bottom, it says "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and features the Microsoft Visual Studio logo.

Click on **Add Connection**, and you will be presented with a list of options of the different Business Services you can connect to (these options are available because they have already been defined as Connected Services in the StockTrader Web application repository):



Select the option **.NET StockTrader Business Services IIS Host Http/SOAP**.¹

Next, simply type in the name or address of any machine you installed StockTrader on. The repository already stores the logical URI (path, port) information, so you just need the server name or IP address here. If remote business services (the /TradeWebBSL Web application) are running on multiple physical servers (nodes), the Configuration Service will be aware of this, but you still need to initially point to a single node running IIS-hosted business services to establish the initial dynamic clustering relationship.

After clicking **Add Connection**, click on the **Return** button to go back to the **Connections** page:

¹ Note that the .NET StockTrader Business Services IIS Host Tcp/Binary endpoint will only be viable on installations of Business Services on IIS 7.0, as IIS 6 (Windows Server 2003) and IIS 5 (Windows XP/SP2) only support http-based endpoints. The StockTrader setup only activates tcp endpoints for IIS when installing on Windows Vista/above or Windows Server 2008/above, accordingly, although the connection options will still appear in this menu. The self-hosted tcp endpoint (.NET StockTrader Business Service Tcp/Binary) will, however, be viable on Windows Server 2003 and Windows XP/SP2, as the self-host program (not hosted in IIS) can host endpoints on non-http endpoints.

The screenshot shows a Microsoft .NET Configuration Service page titled "SERVICE CONFIGWEB". The top navigation bar includes links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Glossary, and About. A "Discussion Forum" and "Download on MSDN" link are also present.

The main content area displays "SERVICE CONNECTION POINTS" for ".NET StockTrader Web Application". It shows the application version (5.0), developer (Microsoft Corporation), and operating system (Windows 7 with .NET Framework v4.0.30319). The Windows 7 logo is displayed.

Below this, there are buttons for "Back to Top", "View Services", "View Clients", and "Add Connection". A note explains the page's purpose: "This page shows the actual WCF connections between the client service domain and the remote service domain for any business service endpoints, as maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/external load balancer, as is the case with Windows Azure service domains, then you will see just one connection for this service domain (with a corresponding icon), just as the client service domain does—even though in reality there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly to each load balanced node, with load balancing and request-level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all nodes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the WCF client being utilized, including the endpoint address, the name of the client definition as defined in configuration, the binding type, binding configuration name, and security mode the client is utilizing."

A note below states: "If a connection point below shows red, hover over it to see the actual exception the client is receiving. If a connection below shows as a grey colored server, a timeout was exceeded (also indicating an issue)."

The "Connections To Host: .NET StockTrader Business Services IIS Host" section lists two connection points:

Remote Address	Client Details	Service Status	Remove
http://gregleak-pc/tradewebsl/tradebsl.svc	<p>On-Premise IIS Host BSL Basic Http SOAP</p> <p>Client Configuration: Client_BasicHttpsBinding Service Contract: TradeBusinessServiceContract.ITradeServices Binding Configuration: Client_BasicHttpBinding Binding Type: basicHttpBinding Security Mode: None Endpoint Behavior: None Assigned</p>		Delete
net.tcp://gregleak-pc/tradewebsl/tradebsl.svc	<p>On-Premise IIS Host BSL netTcp</p> <p>Client Configuration: Client_TcpBinding Service Contract: TradeBusinessServiceContract.ITradeServices Binding Configuration: Client_TcpBinding Binding Type: netTcpBinding Security Mode: None Endpoint Behavior: None Assigned</p>		Delete

At the bottom, there is a "Return to Home Page" link and a note: "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0 Copyright 2011, Microsoft Corporation ".

Note that on IIS7 (Windows Vista or Window Server 2008), two connections will be added. This is because the StockTrader Web Application has two Connected Service definitions in its repository, both hosted by the same IIS TradeWebBSL host application: one is the HTTP/SOAP endpoint, the other is a TCP/binary endpoint. Since it is subscribed to both, the Configuration Service will automatically create Connection Points to both, even though only one will be active at a time, based on the AccessMode setting, per the design of the StockTrader Web application. If you are running on an earlier version of IIS (IIS 5 on Windows XP or IIS 6 on Windows Server 2003), you will only see the HTTP/SOAP Connection Point, since these earlier versions of IIS are not capable of hosting non-Http service endpoints. The active Connection Points will be highlighted in green.

You can now login to the actual StockTrader Web application at <http://localhost/trade> (vs. ConfigWeb) as a registered site user, and now instead of running business services within the same process as the ASP.NET Trade Web application (the user interface tier), all activations are happening remotely to the separate IIS-hosted TradeWebSL Web application via WCF/Web Services. Of course, you will not notice a difference as a StockTrader user in the StockTrader Web application itself. In this way, business services could be configured, for example, to run behind a second firewall, with all business logic and all database connectivity protected behind this firewall.

The next (optional) step to demonstrate further physical partitioning into distinct service layers will be to configure the Business Services to also remotely invoke, on a third tier, order processing to occur via a separate service via a service-to-service interaction. To configure how orders are processed, you will be configuring the StockTrader Business Service IIS Host, not the StockTrader Web application since the Web application is blind to anything beyond the Business Service tier. This is where the connected nature of the Configuration Service between service domains comes into play, using Linked Services. Instead of logging out, you can select the IIS Business Service tier directly, since the '**compositeadmin**' user is also registered as an administrator of this remote service within its own repository.

Next, go back to the main page by clicking on the **Configuration** menu item and select the Business Service IIS tier as the *top node* within ConfigWeb, by clicking as indicated below:

The screenshot shows a Microsoft Internet Explorer window displaying the 'SERVICE CONFIGWEB' interface at <http://localhost/ConfigWeb/Nodes.aspx>. The title bar includes the Microsoft .NET logo, the page title 'Configuration Serv...', and standard browser controls.

The main content area is titled 'CONFIGURATION SERVICE HOME'. It displays information about the '.NET StockTrader Web Application' (Version 5.0, Microsoft Corporation, Windows 7 with .NET Framework v4.0.30319). A 'Service Map' section features a globe icon. Below the map are navigation links: 'Back to Top', 'CS Users', 'Hosted Services', 'Connected Services', 'Connection Points', and 'Service Logs'. A note below these links explains the 'Select' button's function: it allows navigating to the Configuration Service for the currently selected service, which is automatically updated across the network.

The central part of the screen is a grid-based navigation interface. It has two main sections: 'Selected Service Domain' (left) and 'Remote Connected Services' (right). The 'Selected Service Domain' section contains a single item: '.NET StockTrader Web Application : Host : .NET StockTrader Web Application'. The 'Remote Connected Services' section lists two items under 'Setting Groups': '.NET StockTrader Business Services IIS Host : Host : .NET StockTrader Business Services IIS Host' and '.NET StockTrader Business Services IIS Host : Trade : BusinessServiceContract : ITTradeServices'. The 'Select' button for the first item is highlighted with a yellow circle.

At the bottom of the page, a footer notes: 'Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0. Copyright 2011, Microsoft Corporation. Microsoft Visual Studio'

At this point, you are looking at the application from the Business Service tier down, as follows (the top node is always on the left, with its remote services shown on the right, much like Windows folder navigation):

The screenshot shows the Microsoft .NET Service ConfigWeb interface. At the top, there's a navigation bar with links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Glossary, and About. Below the navigation bar, the title "SERVICE CONFIGWEB" is displayed, along with "Configuration Service Version 5.0.0".

The main content area is titled "CONFIGURATION SERVICE HOME" and features a banner for ".NET StockTrader Business Services IIS Host" with "Version 5.0" and "Microsoft Corporation". It also mentions "Windows 7 with .NET Framework v4.0.30319". A "Service Map" section shows a globe icon.

Below the map are several navigation links: "CS Users", "Hosted Services", "Connected Services", "Connection Points", and "Service Logs".

A note below these links explains the navigation context: "The buttons above operate against the currently selected service domain. The top-level menu always operates against the root service domain, which is the one you logged into from the ConfigWeb login page. To change the service domain selection, choose Select from the Remote Connected Services table. The Back To Top button will always take you back to the root service domain. All operations are routed through the network, so direct network connectivity is not necessary for traversal. For example, some services may be hosted on Windows Azure, while others might reside on-premise or in other hosting environments. With the Configuration Service, each service domain is completely autonomous and has exclusive access to its own configuration database. As you navigate, be aware of the currently selected service, which is displayed at the top of every page in ConfigWeb, along with the platform the selected service is deployed to. One way to think about navigation in ConfigWeb is it's the same concept as navigating through folders in Windows, except you are navigating across connected elements in your composite application, which are potentially connected across the world via the Internet, within a private corporate network, or both. Once selected, you can view and modify application settings for the service domain, and the Configuration Service will push the updates through the network to the correct domain, which will then update all running nodes without any need to deploy new configuration files, or stop and re-start the running nodes."

The main table, titled "Selected Service Domain" and "Remote Connected Services", lists service domains. The first row shows ".NET StockTrader Business Services IIS Host" with a gear icon and "Edit Inherited Settings" link. The second row shows ".NET StockTrader Business Services IIS Host : Host : .NET StockTrader Business Services IIS Host" with a gear icon and "Edit Custom Settings" link, which is circled in yellow. The third row shows ".NET StockTrader Order Processor Service - Host : .NET StockTrader Order Processor Service (in-process)" with a gear icon and "Edit Inherited Settings" link. The fourth row shows ".NET StockTrader Order Processor Service - Trade : OrderProcessorContract : IOrderProcessor (in-process)" with a gear icon and "Edit Custom Settings" link.

At the bottom of the page, a footer notes: "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0 Copyright 2011, Microsoft Corporation".

Now simply follow the same process completed for re-configuring the business services to be remotely activated by the Web application, except this time you will be configuring the IIS Business Service host to remotely activate the Order Processor Service. This is done via the **OrderMode** configuration setting for business services. Click on the **Edit Configuration** link as circled above. You can now change the OrderMode setting for business services:

Configuration Keys

.NET StockTrader Business Services IIS Host
Version 5.0
Microsoft Corporation
Windows 7 with .NET Framework v4.0.30319

Windows 7

[Back to Top](#)

Inherited settings are simply application global settings inherited from a base Configuration Service settings class. Custom settings are application-specific settings the developer has defined for the specific application. These can be any settings you might otherwise have defined in the <appSettings> section of a .NET Web config or App.Config file. While stored in the central configuration database, these settings are cached locally in-memory, and used just like they came from a config file (although no code is necessary to populate them into local variables). The in-memory cache means access is as fast as accessing a local static variable. But, Configuration Service keeps all nodes in sync when configuration updates are made on one system--without having to deploy new application configuration files across nodes, and stop/re-start running nodes.

[Basic](#) [Detailed](#) [Advanced](#)

[Define Keys](#)

Target Settings Group: Trade.BusinessServiceContract.ITradeServices
Custom Application Settings

Setting Name	Current Value	Description	Change Value
OrderMode	In-Process Activation	The setting is similar to the Access Mode setting for the StockTrader Web application. It defines how the BSL (Business Logic Layer) calls into the Order Processor Service. As a sample application, there are a variety of settings. For example, to two distinct service domains hosting the OPS-on-premise and Windows Azure. And several different WCF bindings with different transports and security models are selectable. The BSL will only use one endpoint at a time, so this setting is a switch that tells the BSL which host, and how to invoke the Order Processor. As a side-note, calls over WCF are asynchronous (one-way), we do not need to wait for a response from the service, we just need to submit the order successfully. This can improve performance. In-Process Activation is a special setting that tells the BSL not to invoke a remote OPS service, but process the order directly within the BSL process itself.	Change Value
Encrypt Account Passwords in DB: Use Salted Hash	true	If true (default), StockTrader user passwords for registered users of the StockTrader application will be stored as one-way encrypted using a salted hash. If false, passwords will be stored as plain text. For compatibility with non-.NET implementations of business services (such as WSC2) against the same physical SQL Server databases, set to false. Note, however, the SQL Database Loader must be used to appropriately re-load a data set that matches this setting.	Change Value
LoginDisplayLogins	true	In the service host console, whether to display logins (for monitoring activity/demos). If set to true, please keep the corresponding setting for number of iterations to display above 100 if doing benchmarks, as these operations are somewhat expensive.	Change Value
LoginIterations for Display in Console	1	In the service host console, the number of logins before doing a console.write (for monitoring activity/demos). Please make sure to keep this above 100 if doing benchmarks, as these operations are somewhat expensive.	Change Value

[Return to Home Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

Click on the **Change Value** link as circled above. Business services also has several different ways to connect to the Order Processor Service. We will connect here via asynchronous net.tcp messages.

Later, you can also activate the transacted MSMQ endpoint using ConfigWeb, and use it instead. See the notes later in this section about MSMQ, an endpoint that is not active by default but can be activated in just a few clicks via ConfigWeb. You need to make sure you have installed the base MSMQ service as a Windows feature, before you do this. The queues are created automatically by the OPS host when it starts (as a console application, windows host, or NT Service: all three host types are provided).

Setting Name	Current Value	Setting Description	Valid Values	Last Updated
OrderMode	In-Process Activation	<p>The setting is similar to the Access Mode setting for the StockTrader Web application. It indicates how the BSL is invoking calls to the Order Processor Service. In a single instance application, there is a variety of settings. For example, to two distinct service domains hosting the OPS—on-premise and Windows Azure. And several different WCF bindings will be used. These security modes are selectable. The BSL will only use one endpoint at a time so this setting is a switch that tells the BSL which host and how to invoke the Order Processor. As a side-note, calls over WCF are asynchronous (one-way), we do not need to wait for a response from the service, we just need to send the order and it will improve performance. In-Process Activation is a special setting that tells the BSL not to invoke a remote OPS service, but process the order directly within the BSL process itself.</p>	<pre>In-Process Activation: Azure OPS.netTcp security = TransportWithMessageCredential: ClientCertificate Azure OPS.netTcp security = TransportWithMessageCredential: ClientUserName Azure OPS.netTcp security = TransportWithMessageCredential: ClientUserPassword Azure OPS.wsHttp security = TransportWithMessageCredential: ClientCertificate Azure OPS.wsHttp security = TransportWithMessageCredential: ClientUserName On-Premise Self-Host OPS.netcp security = TransportWithMessageCredential: ClientCertificate On-Premise Self-Host OPS.netcp security = TransportWithMessageCredential: ClientUserName On-Premise Self-Host OPS.wsHttp security = TransportWithMessageCredential: ClientCertificate On-Premise Self-Host OPS.wsHttp security = TransportWithMessageCredential: ClientUserName On-Premise Self-Host OPS.wsHttp security = TransportWithMessageCredential: ClientUserPassword On-Premise Self-Host OPS.wsHttp security = TransportWithMessageCredential: ClientUserPassword On-Premise Self-Host OPS.wsmq Transacted Queue security = TransportWithMessageCredential: ClientUserName On-Premise Self-Host OPS.wsmq Transacted Queue security = TransportWithMessageCredential: ClientUserPassword On-Premise Self-Host OPS.netTcp security = TransportWithMessageCredential: ClientCertificate On-Premise Self-Host OPS.netTcp security = TransportWithMessageCredential: ClientUserName On-Premise Self-Host OPS.wsmq Transacted Queue security = TransportWithMessageCredential: ClientUserPassword On-Premise Self-Host OPS.wsmq Transacted Queue security = TransportWithMessageCredential: ClientUserPassword On-Premise Self-Host OPS On-Premise Self-Host OPS SOAP OPS On-Premise Self-Host OPS On-Premise Self-Host OPS MSMQ Transacted Queue Interop: Basic Http SOAP OPS</pre>	6/9/2011 8:17:27 PM

Enter/Select New Value:

- In-Process Activation
- Azure OPS.netTcp security = TransportWithMessageCredential: ClientCertificate
- Azure OPS.netTcp security = TransportWithMessageCredential: ClientUserName
- Azure OPS.netTcp security = TransportWithMessageCredential: ClientUserPassword
- Azure OPS.wsHttp security = TransportWithMessageCredential: ClientCertificate
- Azure OPS.wsHttp security = TransportWithMessageCredential: ClientUserName
- On-Premise Self-Host OPS.netcp security = TransportWithMessageCredential: ClientCertificate
- On-Premise Self-Host OPS.netcp security = TransportWithMessageCredential: ClientUserName
- On-Premise Self-Host OPS.wsHttp security = TransportWithMessageCredential: ClientCertificate
- On-Premise Self-Host OPS.wsHttp security = TransportWithMessageCredential: ClientUserName
- On-Premise Self-Host OPS.wsmq Transacted Queue security = TransportWithMessageCredential: ClientUserName
- On-Premise Self-Host OPS.wsmq Transacted Queue security = TransportWithMessageCredential: ClientUserPassword
- On-Premise Self-Host OPS.netTcp security = TransportWithMessageCredential: ClientCertificate
- On-Premise Self-Host OPS.netTcp security = TransportWithMessageCredential: ClientUserName
- On-Premise Self-Host OPS.wsmq Transacted Queue security = TransportWithMessageCredential: ClientUserPassword
- On-Premise Self-Host OPS.wsmq Transacted Queue security = TransportWithMessageCredential: ClientUserPassword
- On-Premise Self-Host OPS On-Premise Self-Host OPS SOAP OPS
- On-Premise Self-Host OPS On-Premise Self-Host OPS MSMQ Transacted Queue Interop: Basic Http SOAP OPS

[Return to Configuration Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation


- Choose the On-Premise Self-Host OPS netTcp mode.
- Click the **Update** button to change the configuration to this remote mode.

The Order Processor Service is hosted in a separate .NET host program (or “self-hosted”):

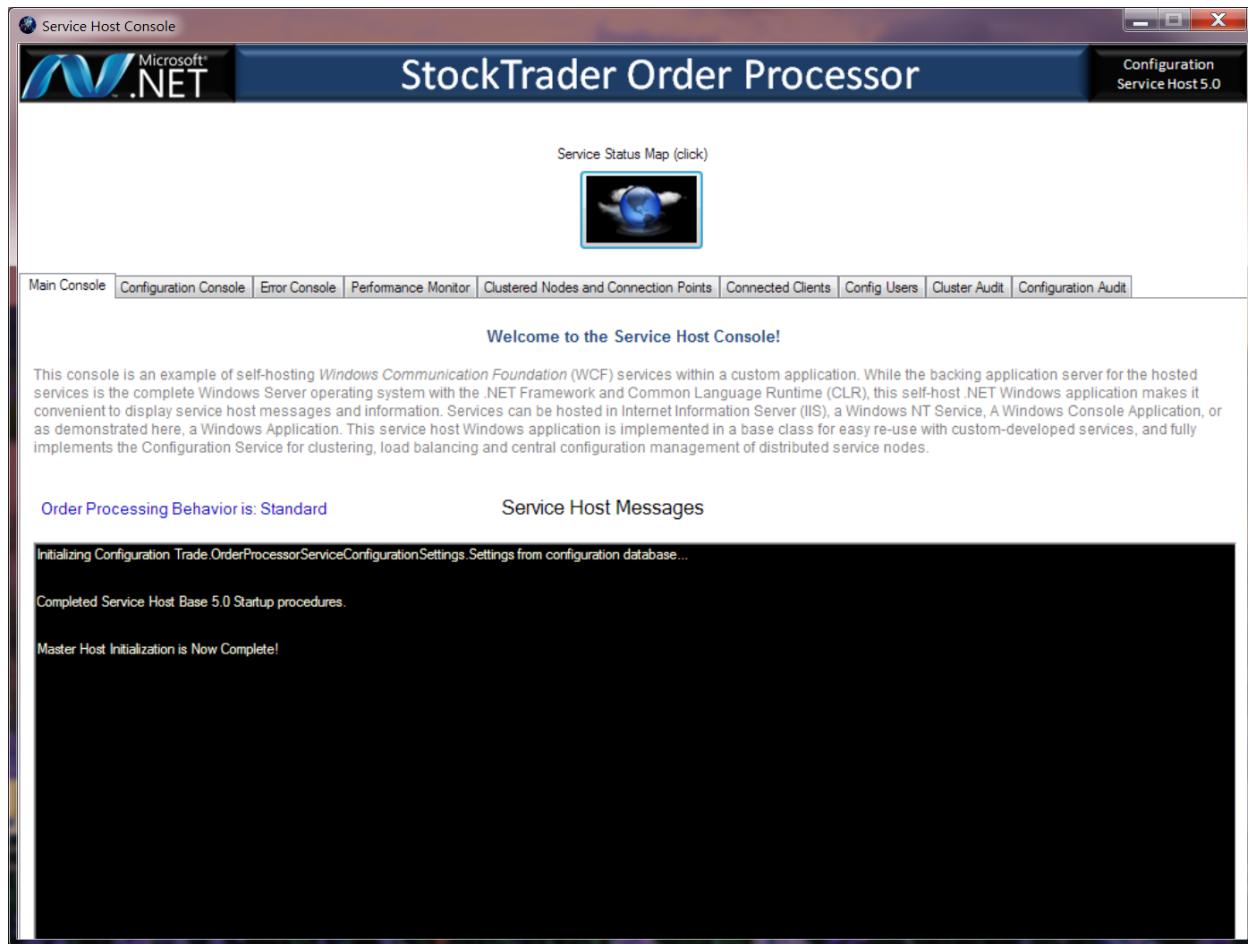
Trade.OrderProcessorServiceHost.exe, accessible via the StockTrader Start menu group in Windows.

Order Processor Services **are not hosted in IIS** (although they could be hosted in IIS 7 via WAS hosting).

But, as with the Business Services self-host, a Windows NT Service host type is provided as well, if you want to explore hosting services within an NT Service. The following interchangeable host programs are provided:

- Windows Host (inherits from base Config Service .NET Windows Forms class)
- Console Host
- NT Service Host (run \stocktrader\install\dir\setup\util\installOPSService.cmd from a CMD prompt with -i to install and -u to install/uninstall as an NT Service. It will install in auto-start mode; so do not forget it will be active on re-boots unless you change this.)

As before with the Web application, you will need to add an initial connection within business services to the Order Processor Service so that orders can actually be placed by users in the StockTrader Web application. First, however, you will need to startup the Order Processor Service host program. This can be found in the **Windows Start** menu items for .NET StockTrader, under **.NET StockTrader Self Hosts**. Start **Trade.OrderProcessorServiceHost.exe**:² Make sure to choose “Run As Administrator”, or else WCF will not have the correct permissions to register and host its endpoints.



At this point, you are ready to add an initial connection from the Business Services-IIS Host layer to the running remote Order Processor Service. To do so, in ConfigWeb you will go back to the Connections page:

² Also, all executable programs are installed, pre-compiled, in [stocktrader install directory]\Builds. You can also optionally start the console host implementation, but for this example we will start the Windows host. Note this Windows host program is itself a re-usable base class that is part of the Configuration Service, and can be easily used in your own host programs without writing any Windows .NET code.

SERVICE CONFIGWEB

Configuration Service Version 5.0.0

Welcome Login/Logout Configuration Service Logs Hosted Services Connected Services Connections Glossary About Discussion Forum Download on MSDN

SERVICE CONNECTION POINTS

.NET StockTrader Web Application
Version 5.0
Microsoft Corporation
Windows 7 with .NET Framework v4.0.30319

Windows 7

[Back to Top](#)

[View Services](#) [View Clients](#)

[Add Connection](#)

This page shows the actual WCF connections between the client service domain and the remote service domain for any business service endpoints, as maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/external load balancer, as is the case with Windows Azure service domains, then you will see just one connection for this service domain (with a corresponding icon), just as the client service domain does—even though in reality there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly to each load balanced node, with load balancing and request-level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all nodes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the WCF-client being utilized, including the endpoint address, the name of the client definition as defined in configuration, the binding type, binding configuration name, and security mode the client is utilizing.

If a connection point below shows red, hover over it to see the actual exception the client is receiving. If a connection below shows as a grey colored server, a timeout was exceeded (also indicating an issue).

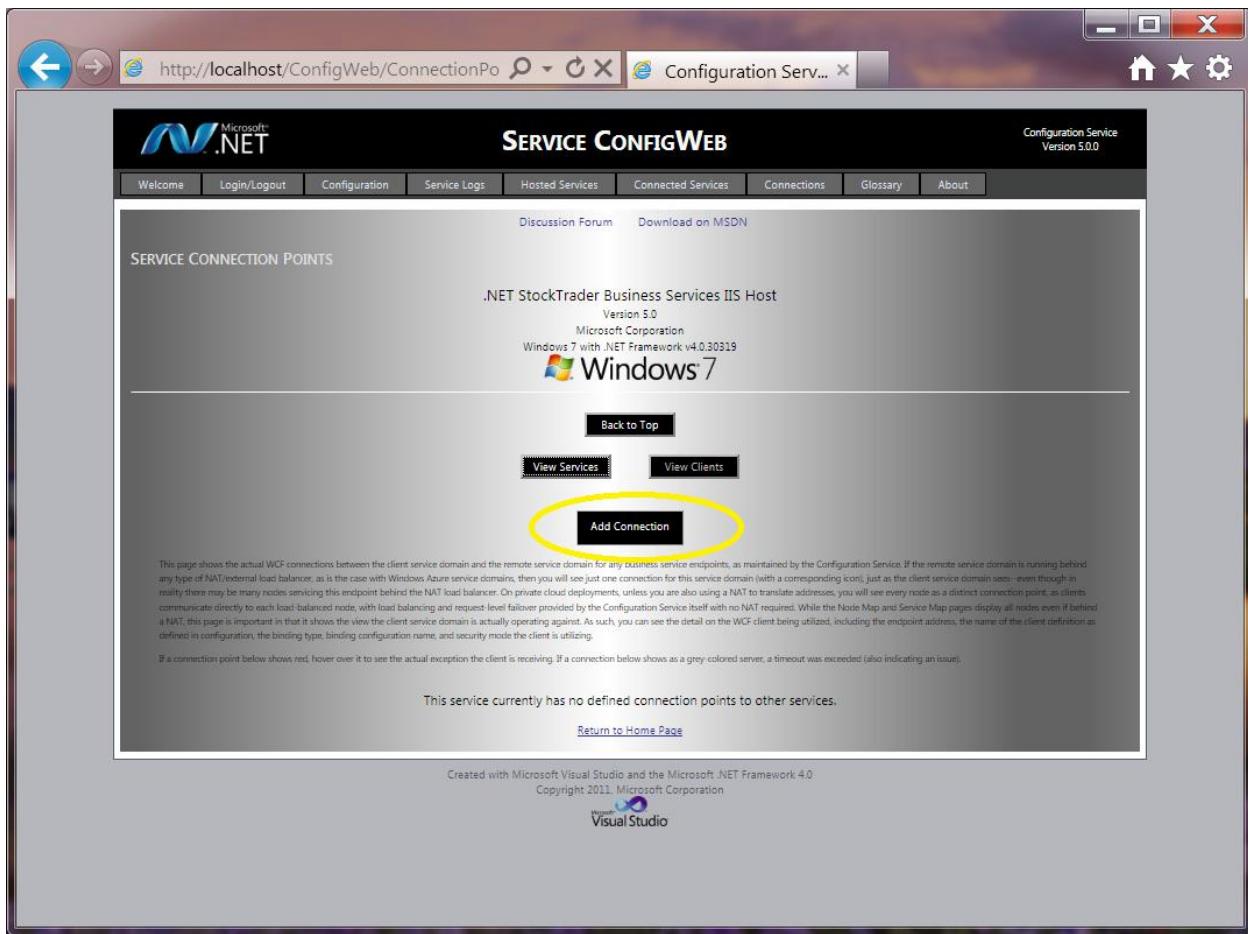
Connections To Host: .NET StockTrader Business Services IIS Host

Remote Address	Client Details	Service Status	Remove
http://gregleak-pc/tradewebsl/tradebsl.svc	<p>On-Premise IIS Host BSL Basic Http SOAP</p> <p>Client Configuration: Client_BasicHttpsBinding Service Contract: TradeBusinessServiceContract.ITradeServices Binding Configuration: Client_BasicHttpBinding Binding Type: basicHttpBinding Security Mode: None Endpoint Behavior: None Assigned</p> <p>On-Premise IIS Host BSL netTcp</p> <p>Client Configuration: Client_TcpBinding Service Contract: TradeBusinessServiceContract.ITradeServices Binding Configuration: Client_TcpBinding Binding Type: netTcpBinding Security Mode: None Endpoint Behavior: None Assigned</p>		Delete
net.tcp://gregleak-pc/tradewebsl/tradebsl.svc			Delete

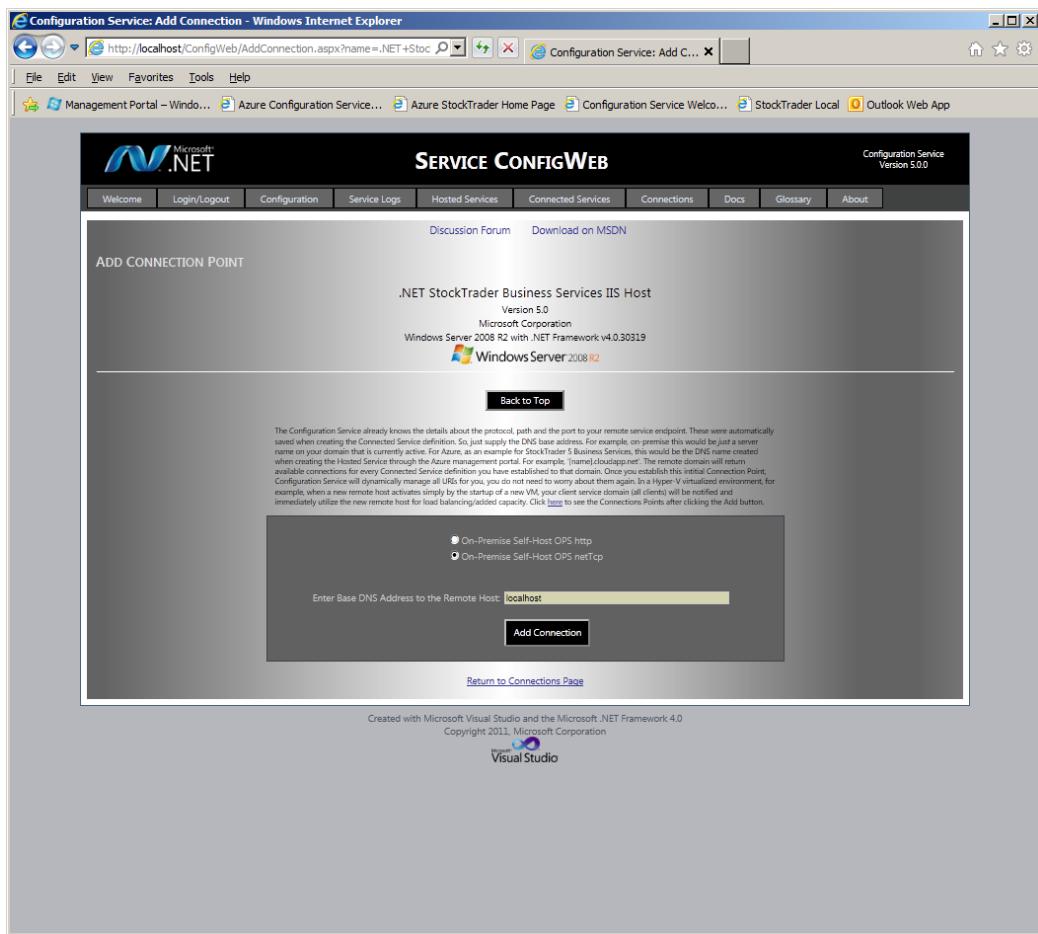
[Return to Home Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

You are now at the Connections page for the Web application, but via the inter-connected nature of the configuration services, **we can directly drill-down into the .NET StockTrader Business Services IIS Host** connections from here, by clicking on the link as circled above.



Click on the **Add Connection** button in this page, and you will see the connection options for business services to connect to the remote Order Processor Service. Choose the **On-Premise Self-Host OPS netTcp option**.



Enter the name of the server running the Order Processor Service (or **localhost** for the local computer). Again, you just need a server name here, since the Business Service IIS Host repository already has a Connected Service Definition to this endpoint that holds the logical URI information (path and port).

Click the **Add Connection** button.

You can now click on the **Return** button to go back to see the active connections. Note if you click on the Connections menu tab instead, you will be positioned back at the StockTrader Web application and will need to drill down again to see Business Services IIS Host connections. You will notice more than one connection has been returned. That is because the Order Processor Host is hosting the Order Processor Service on two different protocols simultaneously, and the Business Services IIS Host is subscribed to both types of endpoints; these Connected Service definitions are pre-set on install. In both cases, orders are not only remotely processed, but they will be placed asynchronously from business services, because of the way the Order Processor Service contract is defined (it is defined as a **one-way** contract).

The screenshot shows a Microsoft Internet Explorer window displaying the Configuration Service Connection Points page. The URL is <http://localhost/ConfigWeb/ConnectionPoint.aspx?name=.NET Stock>. The page title is "Configuration Service: Conn...". The main content area is titled "SERVICE CONFIGWEB" and "Microsoft .NET". It displays information about ".NET StockTrader Business Services IIS Host Version 5.0" running on "Windows Server 2008 R2 with .NET Framework v4.0.30319". Below this, there are links for "Back to Top", "View Services", and "View Clients". A "Add Connection" button is also present. The main content area is titled "SERVICE CONNECTION POINTS" and contains a detailed table of connection points.

Client Details		Service Status	Remove
On-Premise Self-Host OPS http Client Configuration: Client_BasicHttpBinding Service Contract: TradeBusinessServiceContractITradeServices Binding Configuration: Client_BasicHttpBinding Binding Type: basicHttpBinding Security Mode: None Endpoint Behavior: None Assigned			Delete
On-Premise Self-Host OPS netTcp Client Configuration: Client_TcpBinding Service Contract: TradeBusinessServiceContractITradeServices Binding Configuration: Client_TcpBinding Binding Type: netTcpBinding Security Mode: None Endpoint Behavior: None Assigned			Delete

At the bottom of the page, there is a note: "This page shows the actual (WCF) connections between the client service domain and the remote service domain for any business service endpoints, as maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/Universal load balancer, as is the case with Windows Azure service domains, then you will see just one connection for this service domain (with a corresponding icon), just as the client service domain sees – even though in reality there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly to each load-balanced mode, with load balancing and request-level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all modes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the WCF client being utilized, including the endpoint address, the name of the client definition as defined in configuration, the binding type, binding configuration name, and security mode the client is utilizing." There is also a note: "If a connection point below shows red, hover over it to see the actual exception the client is receiving. If a connection below shows as a grey-colored server, a timeout was exceeded (also indicating an issue)."

Finally, click on the **Configuration** menu item to go back to the main ConfigWeb home page. Initially, you will be re-positioned with the Web Application as the top node.

The screenshot shows the Service ConfigWeb interface running on a Windows 7 system. The top navigation bar includes links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Glossary, and About. The main content area displays the Configuration Service Home page, featuring the Microsoft .NET logo and the title "SERVICE CONFIGWEB". Below this, there are links for Discussion Forum and Download on MSDN.

The "Selected Service Domain" table contains the following data:

Settings Group	Hosted Service Domain	View Nodes	Configuration Settings
.NET StockTrader Web Application : Host .NET StockTrader Web Application		View Nodes	Edit Inherited Settings
Setting Groups			
.NET StockTrader Web Application : Trade .StockTraderWebApplication		Edit Custom Settings	

The "Remote Connected Services" table contains the following data:

Settings Group	Remote Service Domain Status	View Nodes	Select
.NET StockTrader Business Services IIS Host : Host .NET StockTrader Business Services IIS Host		View Nodes	Select
Setting Groups			
.NET StockTrader Business Services IIS Host : Trade .BusinessServiceContract .ITradeServices			

A yellow circle highlights the "Select" button in the "Remote Connected Services" table under the first setting group. The footer of the page indicates it was created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0, Copyright 2011, Microsoft Corporation.

You can now select business services as the root node, by clicking where circled above. Business services will be shown on the left, with its newly added connections to the Order Processor Service.

.NET StockTrader Business Services IIS Host
Version 5.0
Microsoft Corporation
Windows 7 with .NET Framework v4.0.30319
Windows 7

Service Map

Selected Service Domain

Settings Group	Hosted Service Domain	View Nodes	Configuration Settings
... Selected Service Domain		View Nodes	Edit Inherited Settings
... Setting Groups		Edit Custom Settings	
... Setting Groups			

Remote Connected Services

Settings Group	Remote Service Domain Status	View Nodes	Select
... Remote Connected Services		View Nodes	Select
... Setting Groups			

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

Similarly, you could now optionally ‘drill-down’ one layer further, and position Order Processor Service as the root node, to configure its settings.

The Service Map View

The Service Map will display the entire physical topology of connected services making up an application, including all clustered servers each **virtual service host** is running on. You can bring up the Service Map by clicking on the button as shown below:

Configuration Service
Version 5.0.0

[Discussion Forum](#) [Download on MSDN](#)

CONFIGURATION SERVICE HOME

.NET StockTrader Business Services IIS Host
Version 5.0
Microsoft Corporation
Windows 7 with .NET Framework v4.0.30319

Windows 7

Service Map

[Back to Top](#)

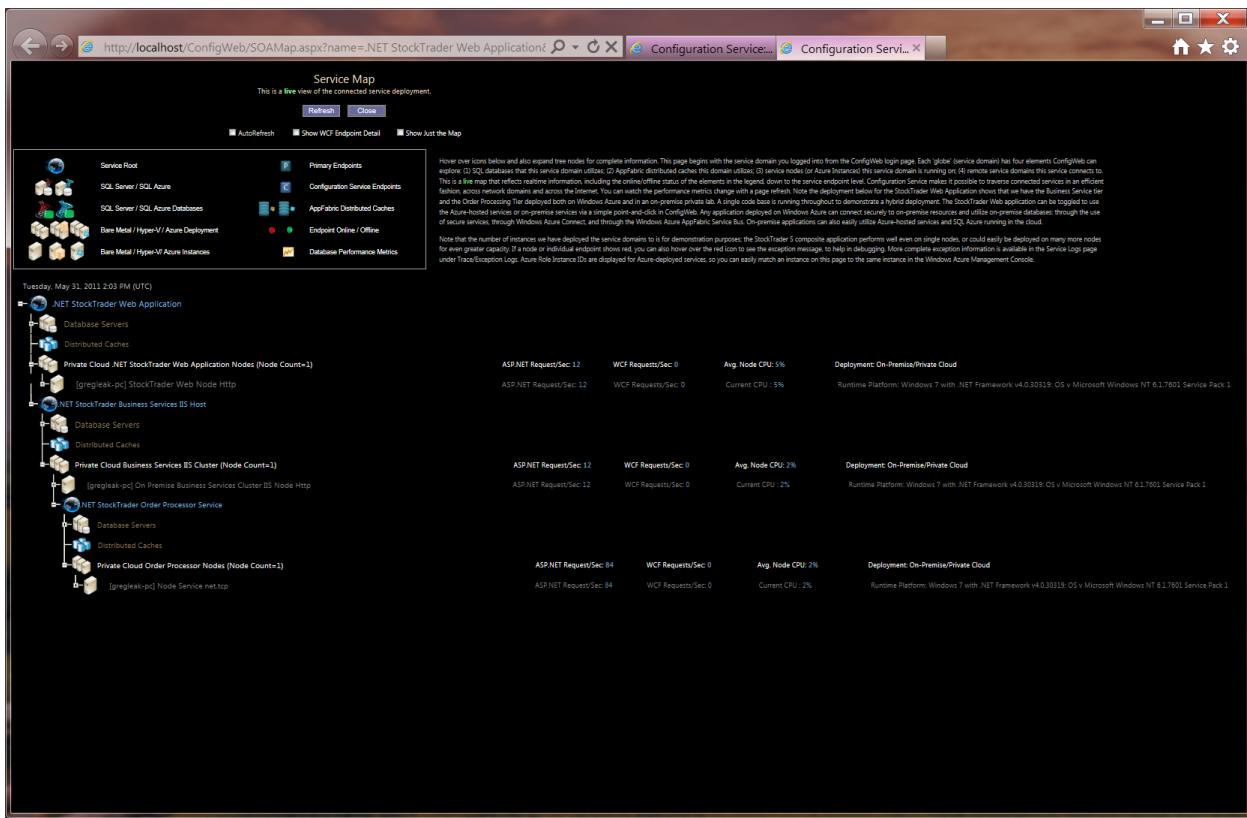
[CS Users](#) [Hosted Services](#) [Connected Services](#) [Connection Points](#) [Service Logs](#)

The buttons above operate against the current selected service domain. The top-level menu always operates against the root service domain, which is the one you logged into from the ConfigWeb login page. To change the service domain settings, choose **Select** from the Remote Connected Services table. The **Back to Top** button will always take you back to the root service domain. All operations are carried through the network. As far as the Configuration Service is concerned, each service domain is completely autonomous and has exclusive access to its own configuration database. As you navigate, be aware of the currently selected service, which is displayed at the top of every page in ConfigWeb, along with the platform the selected service is displayed to. One way to think about navigation in ConfigWeb is it's the same concept as navigating through folders in Windows, except you are navigating across connected elements in your composite application, which are potentially connected across the world via the Internet, within a private corporate network, or both. Once selected, you can view and modify application settings for the service domain, and the Configuration Service will push the updates through the network to the correct domain, which will then update all running nodes without any need to deploy new configuration files, or stop and re-start the running nodes.

Selected Service Domain				Remote Connected Services			
Settings Group	Hosted Service Domain	View Nodes	Configuration Settings	Settings Group	Remote Service Domain Status	View Nodes	Select
.NET StockTrader Business Services IIS Host : Host..NET StockTrader Business Services IIS Host		View Nodes	Edit Inherited Settings	.NET StockTrader Order Processor Service : Host..NET StockTrader Order Processor Service		View Nodes	Select
Setting Groups				Setting Groups			
.NET StockTrader Business Services IIS Host : Trade. BusinessServiceContract. ITradeServices			Edit Custom Settings				

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

You will see the following:



You can see the three-tiered nature of the deployment. Note you can expand any node to view its individual endpoints. Also, hover over icons to see additional information the Configuration Service gathers. This view gives a near-instantaneous read on the online/offline status of clustered servers and their individual service endpoints, for an entire composite application. Green icons indicate online, red, offline. If an icon shows as red, hover over it to view the exception message. If other clustered service nodes are started (for any tier in the application: Web; IIS Host Business Services; Order Processor Service), they will be populated in the view above as new clustered nodes for the virtualized service. The Configuration Service is automatically providing this clustering/load balancing.

In the diagram above, only one server node shows for each tier, since each tier is only running on one server (the same local PC). You can use **AutoRefresh** to poll every few seconds for updates automatically; and **Endpoint Detail** to see the actual URI information vs. the established “friendly names” stored/defined in the configuration repository for each host.

Back to StockTrader as a User

Now you can log into the StockTrader Web application as a user of the composite application at <http://localhost/trade> (vs. administrator using ConfigWeb), and try placing a buy order. This will now be processed by the remote OPS service, asynchronously.

Later in this document there are optional steps to use the MSMQ endpoint instead for placing orders with a durable, transacted message queue.

Transaction Flow If You Activate and Use the MSMQ Endpoint for Order Processing

If you choose to use the MSMQ endpoint (not active by default, see steps later in this document), the flow of the buy order will be:

1. The Web application places a synchronous http Web-service submit to a remote Business Service Layer (IIS-hosted). This layer may be virtualized across many load-balanced servers.
2. Business Services running in IIS creates an order header in the database, but places an asynchronous submission of the order body as a message to the Order Processor Service, which will be responsible for processing the order.
3. The message is actually sent not to the Trade.OrderProcessorServiceHost.exe service host; but rather WCF automatically (by nature of an MSMQ ‘loosely-coupled’ binding) sends it to the MSMQ message queue on the remote server. The Order Processor Service and its associated message queue may also be virtualized across many clustered, load balanced servers, based on the dynamic clustering built into the Configuration Service.
4. WCF infrastructure will automatically notify the host program (Trade.OrderServiceHost.exe) and trigger its service contract method to process an order when it arrives in the message queue. Under heavier load, five orders will be processed at a time based on the TransactedBatching Service Behavior set on the Msmq service endpoint.
5. The order is read via a transacted queue by the Order Processor Service, and processed to the database. The user in the Web application will get a notification when the order is processed.
6. If the database part of the processing logic fails, the distributed transaction will automatically rollback and the order will not be lost, it will stay in the queue. Since the transacted queue is set to enforce ordering, it will automatically be retried n-times on a delay (based on WCF service binding settings in TraderOrderProcessorHost.exe.config); and if it still cannot be processed, it will automatically be sent to a Poison Message queue so it is not lost, but does not interrupt normal processing.
7. You can trigger such a distributed transaction test by getting a quote for symbol **ACIDBUY**, and attempting to buy the stock. While the order will be successfully submitted to the MSMQ transacted message queue; the Order Processor Service (with the acid test logic built in) will trigger a simulated exception processing the order. WCF infrastructure logic will retry the operation, but quickly trigger a fault exception as a detected poison message. At this point, the Order Processor service, with a custom service fault handler, will move the order to a custom-created **poison message queue**; perhaps for later auditing or processing (which is not implemented; for now the order will simply remain pending for the user, and stay in the poison message queue unless you implement such additional logic to process poison messages).

The Business Service Self Host Program and TCP/Binary Service Endpoints

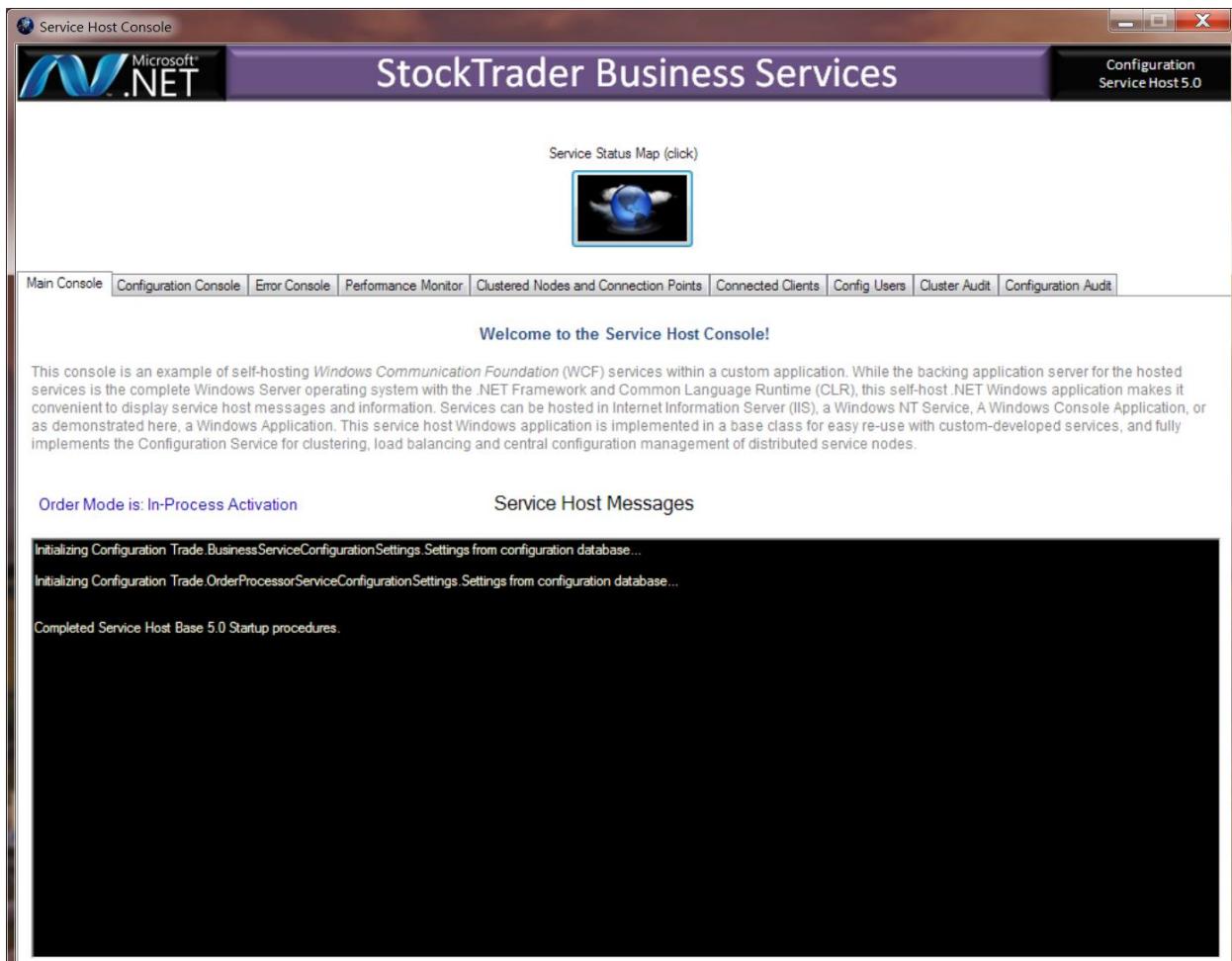
While the above instructions showed how to connect to the IIS-hosted Business Service layer from the Web application, another option is to run the business service self-host application, and connect to it instead. These are the Web application AccessMode settings named with “On-Premise Self-Host xxx”:

You set these as you did for connecting to the IIS Hosted Business Services. However, as with the Order Processor Service, make sure to start the Business Service Self Host program first, from the .NET

StockTrader Windows Start menu items, under the StockTrader Self-Hosts subfolder; or you can also run from the \builds directory. There are three self-host environments available, all share the same configuration repository and only one can be active per node at a time (however, they are interchangeable and you can run different host types on different nodes if you want):

- Windows Host (inherits from base Config Service .NET Windows Forms class)
- Console Host
- NT Service Host (run \stocktrader\install\dir\setup\util\installBSLService.cmd from a CMD prompt with -i to install and -u to install/uninstall as an NT Service. It will install in auto-start mode; so do not forget it will be active on re-boots unless you change this.)

To get started, launch the **Trade.BusinessServiceHost.exe** Windows self-host program on your PC. Make sure to choose “Run As Administrator”.



With the self-host program, you have two options: you can connect via HTTP/XML-encoding, or via TCP/Binary-encoding. TCP/Binary can offer performance advantages vs. HTTP-XML encoding; given lower serialization costs and network. Multiple clients can connect simultaneously over all the bindings a service is supporting.

Note that IIS 6.0 can only host HTTP-based bindings; however this is not a limitation with IIS 7.0, so you will see possible TCP endpoints for IIS-Hosted services with StockTrader 5.0 as well (not just for the self-host version of the BSL); however IIS-hosted Tcp/binary endpoints will only be active/working if running on IIS 7.x (Windows Vista/above or Windows Server 2008/above), as Setup deactivates these endpoints on Windows XP and Windows Server 2003 automatically since IIS 6 will not recognize them.

To change the StockTrader Web Application to use the self-hosted business services in the Windows host program, go back to the home page of ConfigWeb by selecting the **Configuration** menu item.

First, we will delete all connections to the IIS Host business services. This step is not required, as the Configuration Service can track/maintain relationships to any number of service hosts hosting any number of service endpoints from a client application/service. However, it will make this tutorial a bit clearer for first time users, and less likely to accidentally configure the wrong remote host in later steps in this tutorial.



Select the **Connections** menu item.

The screenshot shows the Microsoft .NET Configuration Service web interface at <http://localhost/ConfigWeb/ConnectionPc>. The title bar reads "Configuration Serv...". The main page displays "SERVICE CONFIGWEB" and "Configuration Service Version 5.0.0". It includes links for "Welcome", "Login/Logout", "Configuration", "Service Logs", "Hosted Services", "Connected Services", "Connections", "Glossary", and "About". Below these are links for "Discussion Forum" and "Download on MSDN". The main content area is titled "SERVICE CONNECTION POINTS" and shows a ".NET StockTrader Web Application" running on "Version 5.0" of "Windows 7" with ".NET Framework v4.0.30319". There are buttons for "Back to Top", "View Services", "View Clients", and "Add Connection". A note below states: "This page shows the actual WCF connections between the client service domain and the remote service domain for any business service endpoints, as maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/external load balancer, as is the case with Windows Azure service domains, then you will see just one connection for this service domain (with a corresponding icon), just as the client service domain sees - even though in reality there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly to each load-balanced node, with load balancing and request level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all nodes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the WCF client being utilized, including the endpoint address, the name of the client definition as defined in configuration, the binding type, binding configuration name, and security mode the client is utilizing." A note at the bottom says: "If a connection point below shows red, hover over it to see the actual exception the client is receiving. If a connection below shows as a grey-colored server, a timeout was exceeded (also indicating an issue)."

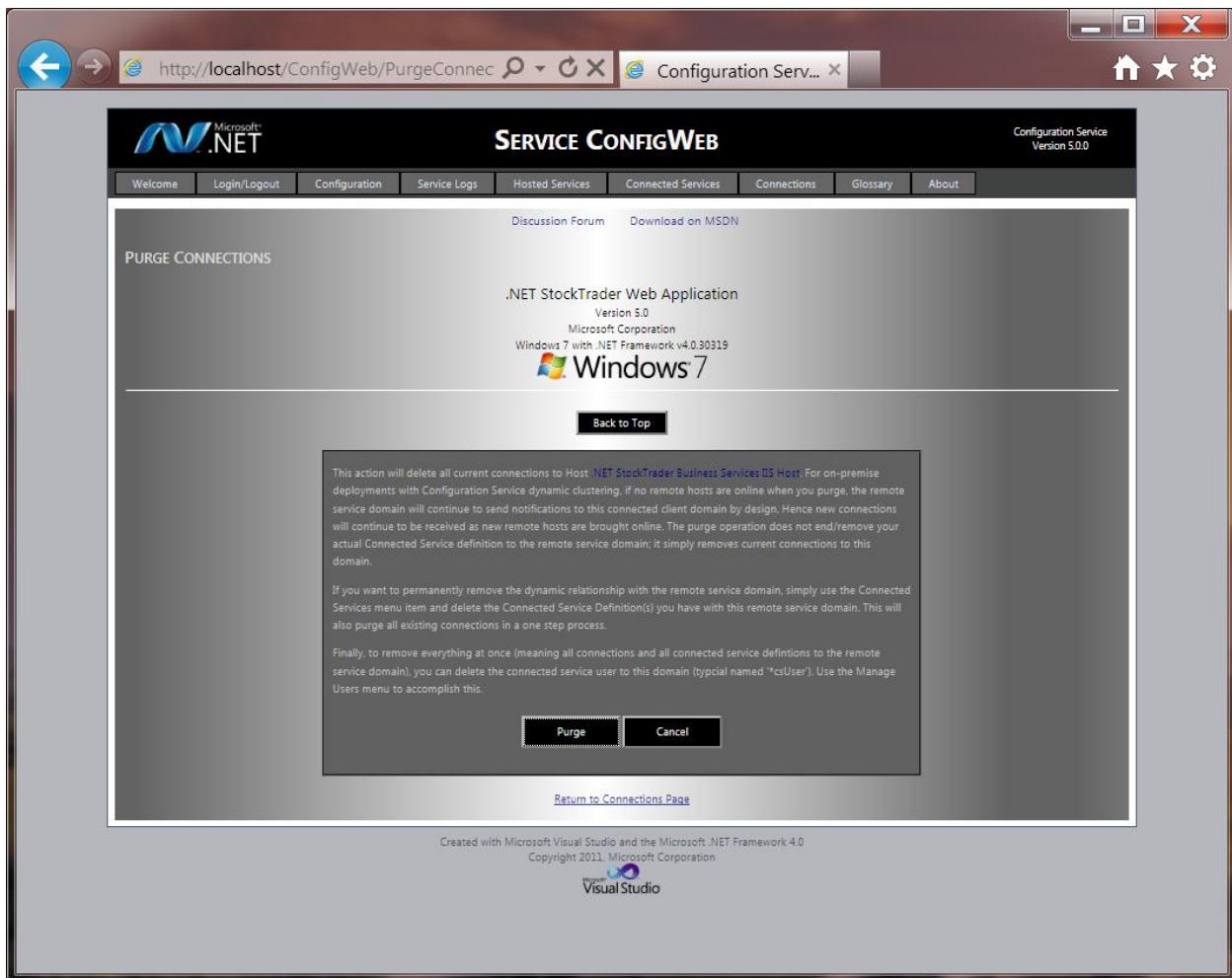
Connections To Host: .NET StockTrader Business Services IIS Host

Remote Address	Client Details	Service Status	Remove
http://gregleak-pc/tradewebsl/tradebsl.svc	<p>On-Premise IIS Host BSL Basic Http SOAP</p> <p>Client Configuration: Client_BasicHttpBinding Service Contract: Trade BusinessServiceContract.ITradeServices Binding Configuration: Client_BasicHttpBinding Binding Type: basicHttpBinding Security Mode: None Endpoint Behavior: None Assigned</p> <p>On-Premise IIS Host BSL netTcp</p> <p>Client Configuration: Client_TcpBinding Service Contract: Trade BusinessServiceContract.ITradeServices Binding Configuration: Client_TcpBinding Binding Type: netTcpBinding Security Mode: None Endpoint Behavior: None Assigned</p>		Delete
net.tcp://gregleak-pc/tradewebsl/tradebsl.svc			Delete

[Return to Home Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

Now select **Purge All Connections to This Service Host**.



Click on the **Purge** button. After the connections have been purged, click on the main **Configuration** menu item to go back to the ConfigWeb home page:

The screenshot shows the Service ConfigWeb interface at <http://localhost/ConfigWeb/Nodes.aspx>. The top navigation bar includes links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Glossary, and About. The Configuration Service Version 5.0.0 is displayed in the top right.

The main content area displays the .NET StockTrader Web Application, Version 5.0, Microsoft Corporation, Windows 7 with .NET Framework v4.0.30319. A "Service Map" section features a globe icon. Below it are buttons for Back to Top, CS Users, Hosted Services, Connected Services, Connection Points, and Service Logs.

A note below the buttons explains the navigation context: "The buttons above operate against the currently selected service domain. The top-level menu always operates against the root service domain, which is the one you logged into from the ConfigWeb login page. To change the service domain selection, choose Select from the Remote Connected Services table. The Back To Top button will always take you back to the root service domain. All operations are routed through the network, so direct network connectivity is not necessary for traversal. For more information to be displayed on this screen, the application may reside on a host in other host service environments. With the Configuration Service, each service domain is completely autonomous and has exclusive access to its own configuration database. As you navigate, be aware of the current service domain, which is displayed at the top of every page in ConfigWeb, along with the platform this selected service is deployed to. One way to think about navigation in ConfigWeb is it's the same concept as navigating through folders in Windows, except you are navigating across connected elements in your composite application, which are potentially connected across the world, which will then update all running nodes without any need to deploy new configuration files, or stop and re-start the running nodes."

The "Selected Service Domain" table lists two entries:

Settings Group	Hosted Service Domain	View Nodes	Configuration Settings
.NET StockTrader Web Application : Host: .NET StockTrader Web Application		View Nodes	Edit Inherited Settings
Setting Groups			
.NET StockTrader Web Application : Trade: StockTraderWebApplication		Edit Custom Settings	

The "Edit Custom Settings" link for the second entry is highlighted with a yellow circle.

The "Remote Connected Services" table is mostly empty, showing placeholder rows for "Setting Groups".

At the bottom, it says "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation".

From here, we will now select **Edit Configuration** above to change the StockTrader Web application AccessMode yet again.

Configuration Keys

.NET StockTrader Web Application
Version 5.0
Microsoft Corporation
Windows 7 with .NET Framework v4.0.30319

Windows 7

[Back to Top](#)

Inherited settings are simply application global settings inherited from a base Configuration Service settings class. Custom settings are application-specific settings the developer has defined for the specific application. These can be any settings you might otherwise have defined in the <appSettings> section of a .NET Web.config or App.Config file. While stored in the central configuration database, these settings are cached locally in-memory, and used just like they came from a config file (although no code is necessary to populate them into local variables). The in-memory cache means access is as fast as accessing a local static variable. But, Configuration Service keeps all nodes in sync when configuration updates are made on live systems - without having to deploy new application configuration files across nodes, and stop/re-start running nodes.

[Basic](#) [Detailed](#) [Advanced](#)

[Define Keys](#)

Target Settings Group: Trade StockTraderWebApplication
Custom Application Settings

Setting Name	Current Value	Description	Change Value
AccessMode	On-Premise IIS Host BSL Basic Http SOAP	This setting determines how the Trade Web application makes calls to the Business Service Layer (BSL) over WCF. As a sample application, there are a variety of settings. First, there are three distinct BSL layers hosting service endpoints: 1) Windows Azure running in a Web Role; 2) An on-premise IIS-based service; 3) An on-premise EXE application that ships with the sample. For each, there are a variety of transports and security modes illustrated. The Web application will only use one service endpoint at a time, so this setting is a switch that can for example be changed to have the Web application start communicating to our on-premise BSL vs. the Azure-hosted BSL with a simple update from ConfigWeb.	Change Value
Always Check Order Alerts	false	Whether to run the closed orders check on every visit to an authenticated page to display the Order Alert message, or to use the check frequency setting value.	Change Value
Check Order Alert Frequency	15	Time in seconds to check for closed orders; only used if CheckOrderAlertsOnEveryRequest is false	Change Value
DisplayOrdersDefault	10	This setting determines the default number of orders to display in the Account.aspx page.	Change Value
DisplayOrdersMax	100	This setting determines the maximum number of orders to display in the Account.aspx page.	Change Value

[Return to Home Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

Select **Change Value** as indicated above.

The screenshot shows a Windows-based web application window titled "Configuration Serv...". The URL is <http://localhost/ConfigWeb/ConfigUpdate>. The page displays a table of configuration settings, specifically focusing on the "AccessMode" setting.

Setting Name	Current Value	Setting Description	Valid Values	Last Updated
AccessMode	On-Premise Self-Host BSL netTcp	This setting determines how the Trade Web application makes calls to the Business Service Layer (BSL) over WCF. As a sample application, there are a wide variety of settings. First, there are three distinct BSL layers hosting service endpoints: 1) Windows Azure running in a Web Role; 2) An on-premise IIS-based service; 3) An on-premise EXE application that ships with the sample. For each, there are a variety of transport and security modes illustrated. The Web application will only use one service endpoint at a time, so this setting is a switch that can be used to change to have the Web application start communicating to our on-premise BSL vs. the Azure-hosted BSL with a simple update from ConfigWeb.	In-Process Activation; Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate; Azure BSL wsHttp security = TransportWithMessageCredential: UserName; Azure BSL netTcp security = TransportWithMessageCredential: ClientCertificate; Azure BSL netTcp security = TransportWithMessageCredential: UserName; On-Premise Self-Host BSL http; On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate: UserName; On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate: UserName; On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate: UserName; On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate: UserName; On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate: UserName; On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate: UserName; On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate: UserName; On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate: UserName; On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate; On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate; On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate; On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate; On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate; On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate; On-Premise IIS Host BSL Basic Http SOAP; On-Premise IIS Host BSL netTcp; Interop: Basic Http SOAP BSL	5/31/2011 6:27:20 AM

Below the table, a dropdown menu lists various security options, with "On-Premise Self-Host BSL http" selected and highlighted with a yellow oval. A "Update" button is located below the dropdown.

A message box at the bottom states: "The configuration key was successfully updated."

[Return to Configuration Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

Select **Tcp_On-Premise Self-host BSL netTcp** and click **Update**. This equates to activating remote business services via TCP/Binary encoding vs. Http with XML encoding. Next, we need to establish an initial connection to the running instance of the self-hosted business service program, `Trade.BusinessServiceHost.exe`.

First, go back to the home page for ConfigWeb by selecting the **Configuration** menu item. Now choose the **Connections** menu item again:

The screenshot shows the Microsoft .NET Configuration Service (Version 5.0) running on a Windows 7 system. The browser address bar shows the URL <http://localhost/ConfigWeb/Nodes.aspx>. The main navigation bar includes links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, **Connections** (which is circled in yellow), Glossary, and About.

CONFIGURATION SERVICE HOME

.NET StockTrader Web Application
Version 5.0
Microsoft Corporation
Windows 7 with .NET Framework v4.0.30319

Windows 7

Service Map

Back to Top

CS Users **Hosted Services** **Connected Services** **Connection Points** **Service Logs**

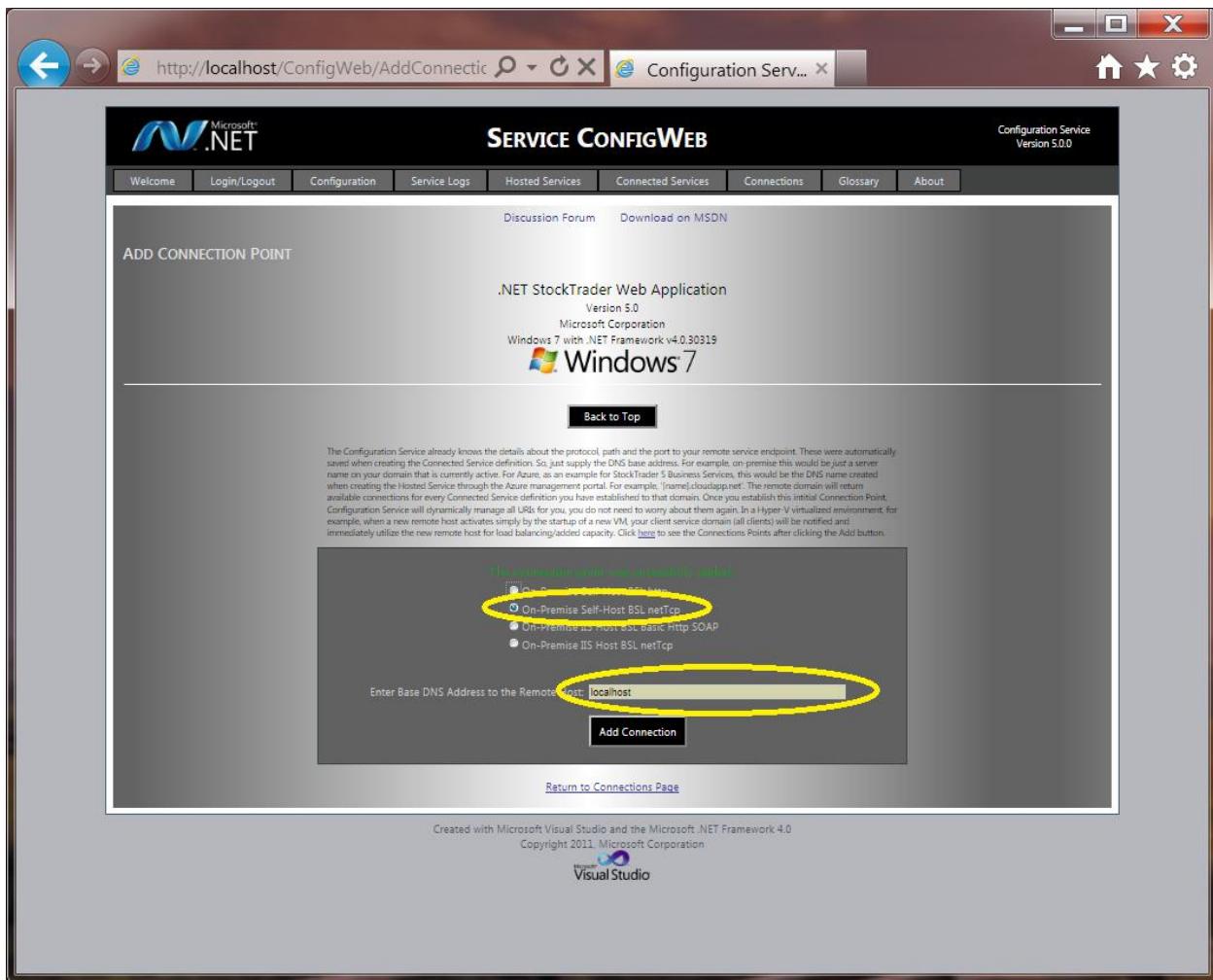
The **Selected Service Domain** table contains two entries:

- .NET StockTrader Web Application : Host : .NET StockTrader Web Application** (with a server icon)
- .NET StockTrader Web Application : Trade : StockTraderWebApplication** (with a gear icon)

The **Remote Connected Services** table is currently empty.

At the bottom, it says "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation".

Select Add Connection, and then choose the **.NET StockTrader Business Services Tcp/Binary** endpoint.



Make sure to type in the server name hosting the running Business Service self host program, and click Add Connection. Next, click on the main **Configuration** menu item.

The screenshot shows the Microsoft .NET Configuration Service (ConfigWeb) interface. At the top, there's a navigation bar with links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Glossary, and About. The main content area is titled "SERVICE CONFIGWEB". It displays the ".NET StockTrader Web Application" version 5.0, running on Windows 7 with .NET Framework v4.0.30319. Below this, there's a "Service Map" section featuring a globe icon. A "Back to Top" button is located above a row of navigation buttons: CS Users, Hosted Services, Connected Services, Connection Points, and Service Logs. A note below these buttons explains the functionality of the "Select" button in the "Connected Services" section. The main pane shows a table titled "Selected Service Domain" and "Remote Connected Services". The "Selected Service Domain" table has columns for Settings Group, Hosted Service Domain, View Nodes, and Configuration Settings. It lists ".NET StockTrader Web Application : Host: .NET StockTrader Web Application" and ".NET StockTrader Web Application : Trade. StockTraderWebApplication". The "Remote Connected Services" table has columns for Settings Group, Remote Service Domain Status, View Nodes, and Select. It lists ".NET StockTrader Business Services Self Host : Host. NET StockTrader Business Services Self Host" and ".NET StockTrader Business Services Self Host : Trade. BusinessServiceContract. ITradeServices". The ".NET StockTrader Business Services Self Host : Host. NET StockTrader Business Services Self Host" entry is circled in yellow. At the bottom, it says "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0 Copyright 2011, Microsoft Corporation" and features the Microsoft Visual Studio logo.

As with IIS hosted services, we now see the self-hosted business services being depicted in the right hand pane.

You can log into the <http://localhost/trade> StockTrader Web application as a site user. The StockTrader Web application is again using a remote service tier, but this time is connected to a different physical host process- the Windows self-host program.

Different Hosts—Different Configuration Repositories

It is very important to keep in mind that the configuration repositories **are specific to the service host** hosting the services. Hence, while Business Services are hosted in both IIS and in the separate self-host

Trade.BusinessServiceHost.exe program, each maintains its own repository. Making changes to settings for one service host will not affect the other service host—they are physically distinct. When you are running business services in-process (AccessMode=InProcess) with the Web application, and you edit the business services configuration directly, you will be effecting the IIS-Hosted Business Service Repository only, based on the way the in-process mode is initially configured. Hence, if you now change your AccessMode to Http_WebService or Tcp_WebService from the Web application, the Trade.BusinessService.exe self host program will not yet be configured for **Msmq OrderMode**. You can optionally do so, or leave OrderMode as **InProcess**. If you change it, make sure you also follow the procedure as before to add an initial Connection Point from the Business Services self-host program to the Order Processor Service.

Monitoring Exception and Trace Logs

Configuration 5.0 allows any application or service to log exception and/or trace information to a logging database. You can then use ConfigWeb to view these logs.

The screenshot shows the Microsoft .NET Configuration Service (ConfigWeb) audit.aspx page. At the top, there's a navigation bar with links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Glossary, and About. Below the navigation bar, there's a banner for the .NET StockTrader Web Application, Version 5.0, running on Windows 7 with .NET Framework v4.0.30319. A yellow circle highlights the "Trace and Error Logs" button in the "Error/Event Log" section.

Source	Severity	Message	Time
.NET StockTrader Order Processor Service	Warning	Rejected operation getServiceConfiguration for user: businessservice\user. User credentials (userid and/or password) failed authentication.	Tuesday, May 31, 2011 2:19 PM (UTC)
.NET StockTrader Order Processor Service	Warning	Rejected operation getServiceConfiguration for user: businessservice\user. User credentials (userid and/or password) failed authentication.	5/31/2011 7:15:08 AM
.NET StockTrader Business Services Self Host	Warning	Completed Service Host base 5.0 Startup procedures.	5/31/2011 7:05:42 AM
.NET StockTrader Business Services Self Host	Information	Calling PostInitProcedure...	5/31/2011 7:05:42 AM
.NET StockTrader Business Services Self Host	Information	Master Host Initialization is Now Complete!	5/31/2011 7:05:42 AM
.NET StockTrader Order Processor Service	Information	Initializing Configuration Trade.OrderProcessorServiceConfigurationSettings.Settings from configuration database..	5/31/2011 7:05:42 AM
.NET StockTrader Order Processor Service	Information	Successfully Loaded All Settings from configuration database..	5/31/2011 7:05:42 AM
.NET StockTrader Order Processor Service	Information	Initializing Configuration Trade.OrderProcessorServiceConfigurationSettings.Settings from configuration database..	5/31/2011 7:05:42 AM
.NET StockTrader Order Processor Service	Information	Successfully Loaded All Settings from configuration database..	5/31/2011 7:05:42 AM
.NET StockTrader Order Processor Service	Information	Successfully Loaded All Settings from configuration database..	5/31/2011 7:05:42 AM
.NET StockTrader Business Services Self Host	Information	Initializing Connection Point WCF Client Channels.	5/31/2011 7:05:42 AM
.NET StockTrader Business Services Self Host	Information	Successfully Loaded All Settings from configuration database..	5/31/2011 7:05:42 AM

Please note that information is also sent to each node's Event Log as setup in the ConfigWeb as an application-level setting. The log is very important for understanding issues, including exceptions your application might have. For WCF services, a Fault Behavior is automatically applied, so you do not actually have to write code to log exceptions, it all happens automatically. However, you can also write to the log using the ConfigurationUtility writeConfigConsoleMessage method.

You can view rich tracing information for Configuration Service 5.0 by turning on Detailed Logging and/or Log Node Notifications. This is a setting for each service domain in ConfigWeb. For performance reasons, you should keep detailed logging off for production applications.

Azure StockTrader: Re-Configuring for Remote Modes

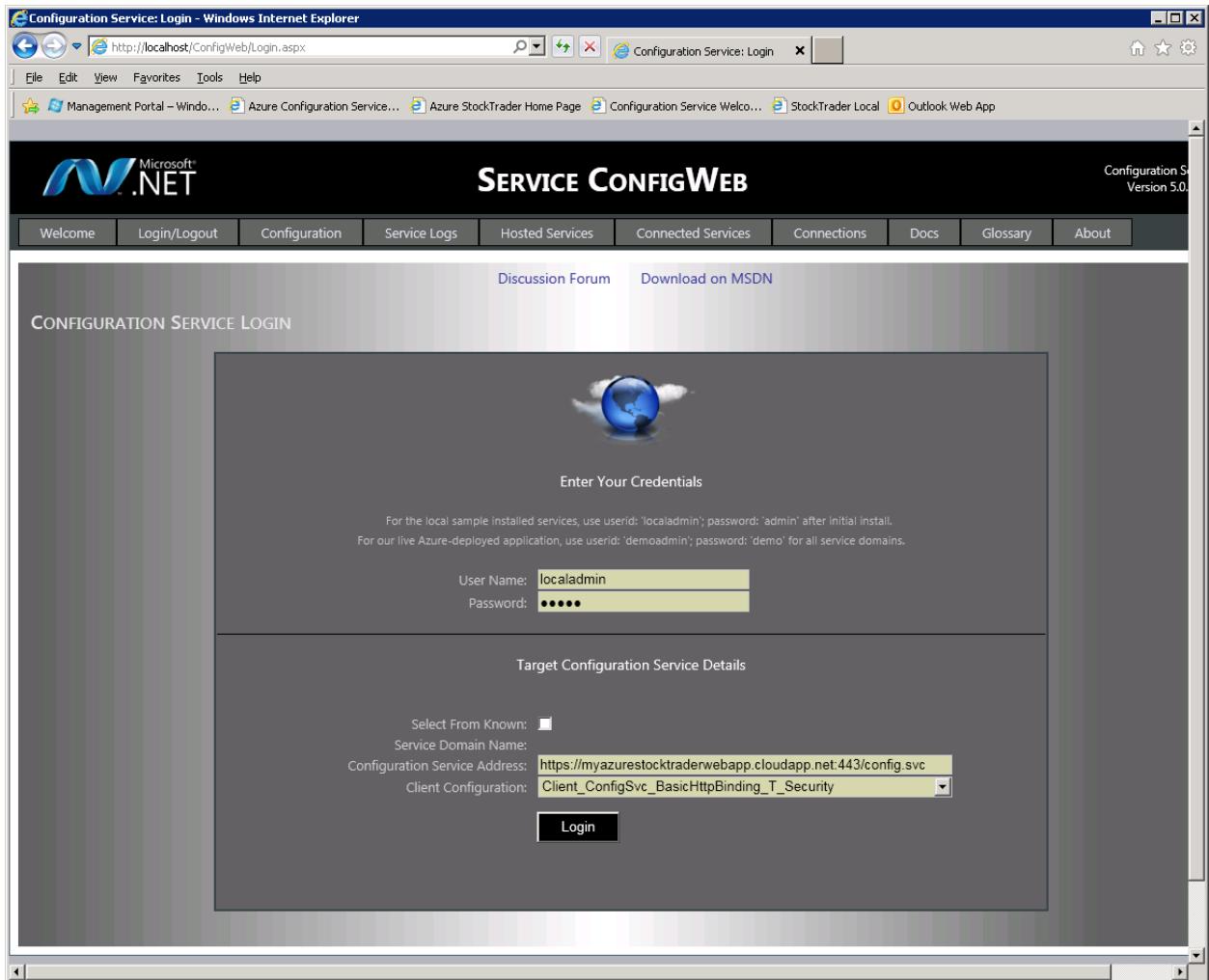
By default, just like the on-premise version of .NET StockTrader, the Azure StockTrader Web Application will be operating as a standalone ASP.NET Web application, and not using the remote modes. The steps to reconfigure for remote modes are almost the same as those above for on-premise .NET StockTrader, but with a couple of additional steps. For Azure StockTrader, the default install does not create Connected Service Definitions to Business Services or the Order Processor Service. So, you will need to do this first before adding connections from the Web application to the Business Services; and from the Business Services to the Order Processor Service. Here is how.

- Compile and deploy the three Visual Azure StockTrader Solutions per the AzureReadme file in the Windows Start menu. Make sure you deploy each to the correct Hosted Service via the Windows Azure Management Console.
- Bring up ConfigWeb. You can either use your own on-premise ConfigWeb for these steps (<http://localhost/configweb>), deploy AzureConfigWeb to your own Azure subscription as a Hosted Service, or, for convenience, use our Azure ConfigWeb at:
<https://azureconfigweb.cloudapp.net>.

Note

The login page for Azure ConfigWeb (as provided in your download and also as used at the link above) has several preset logins that are selectable. These Azure addresses point to OUR (Microsoft's) hosted StockTrader application service domains. As you complete the following sections, you will want to make sure to enter the addresses to YOUR StockTrader Hosted Services. You can edit the Web.Config of the AzureConfigWeb Visual Studio project (and/or your on-premise ConfigWeb application), to change the default addresses that appear in this drop-down list, so they point to your service domains for the various elements of your Azure StockTrader deployment vs. to our hosted services. Then you will not have to enter addresses on every ConfigWeb login, or accidentally login to our services (which will only let you in as a demo administrator with no change privileges). So, now might be a good time to do this, using the base DNS addresses of your Azure service domains for the StockTrader Web Application, StockTrader Business Services and Order Processor Service (the three separate Azure Hosted Services/deployments). Near the top of the Web.Config you will see the list, and you can substitute your DNS addresses for ours for each of the three services. The ports and the client configurations will remain the same. Change the information carefully.

- Login to the Azure StockTrader Web application. Deselect “Select from Known” on the login page and make sure to enter the address to YOUR StockTrader hosted service as follows. **Note you can always use the Azure Management Portal to find the base DNS address for any of your Hosted Services if you forgot the address.**



- Above, you will replace '**myazurestocktraderwebapp.cloudapp.net**' with the address you created when you first created your Hosted Service (in the Windows Azure Management Console) via the steps in the AzureReadme.html. You have a unique base DNS address for your own site that is different than the one used above.
- Make sure you go over https, and select the **Client_ConfigSvc_BasicHttpBinding_T_Security** Client Configuration.

The screenshot shows the Configuration Service Home page in Windows Internet Explorer. The title bar reads "Configuration Service: Home - Windows Internet Explorer". The address bar shows the URL "http://localhost/ConfigWeb/Nodes.aspx". The page itself is titled "SERVICE CONFIGWEB" and features the Microsoft .NET logo. It displays the "Azure StockTrader Web Application" under "Version 5.0" and "Microsoft Corporation". Below this, it says "Windows Azure" and "Windows Server 2008 R2 with .NET Framework v4.0.30319". A "Service Map" section contains a globe icon. Navigation links include "ES Users", "Hosted Services", "Connected Services", "Connection Points", and "Service Logs". A note at the bottom left explains the service domain context. The main content area is titled "Selected Service Domain" and lists several service groups with icons and edit buttons. To the right, a "Remote Connected Services" section shows a similar grid. At the bottom, it credits "Microsoft Visual Studio" and "Copyright 2011, Microsoft Corporation".

From here, you will notice that the application is running in Collapsed “in-process” mode, as a standalone Web application. But, you cannot use Add Connection Point just yet, this time (unlike for the on-premise steps before), you will first need to create initial Connected Service Definitions via ConfigWeb.

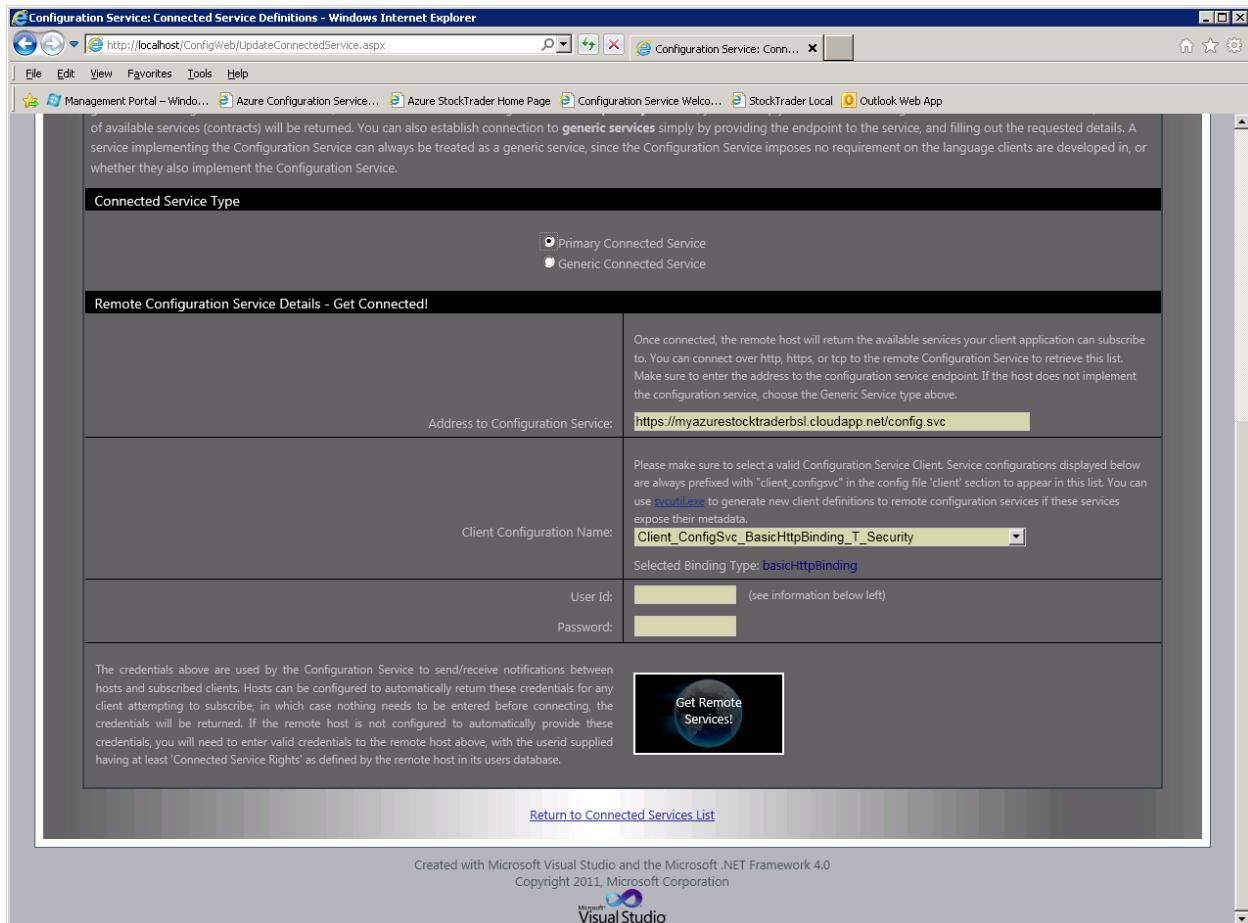
- Click on the Connected Services button in ConfigWeb.

The screenshot shows a Microsoft Internet Explorer window with the title bar "Configuration Service: Connected Services - Windows Internet Explorer" and the URL "http://localhost/ConfigWeb/CServices.aspx?name=Az". The page itself is titled "SERVICE CONFIGWEB" and features the Microsoft .NET logo. It displays information about the "Azure StockTrader Web Application" (Version 5.0, Microsoft Corporation, Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319). Below this, there's a "Connected Services" section with a table header:

Virtual Host ID	Service Name	Service Type	Service Contract	Client Configuration	Edit/ Delete
-----------------	--------------	--------------	------------------	----------------------	--------------

At the bottom of the page, there's a note about the creation: "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation" followed by the Microsoft Visual Studio logo.

- Next, click **Add Service**.



- Now, you will enter the address to your Azure StockTrader Business Services configuration service endpoint. This will be:

[https://\[myazur stocktraderbsl.cloudapp.net\]/config.svc](https://[myazur stocktraderbsl.cloudapp.net]/config.svc)

But you will replace [myazur stocktraderbsl.cloudapp.net] with the base DNS address you created for your Azure StockTrader Business Service Hosted Service in the Windows Azure Management portal.

- Make sure you select the https/secure Client Configuration, or you will not be able to connect. Select **Client_ConfigSvc_BasicHttpBinding_T_Security** (the 'T' stands for *transport* security, just a naming convention Configuration Service uses).
- Click **Get Remote Services!**

Configuration Service: Connected Service Definitions - Windows Internet Explorer
 http://localhost/ConfigWeb/UpdateConnectedService.aspx

File Edit View Favorites Tools Help

Management Portal – Windo... Azure Configuration Service... Azure StockTrader Home Page Configuration Service Welco... StockTrader Local Outlook Web App

services

credentials will be returned. If the remote host is not configured to automatically provide these credentials, you will need to enter valid credentials to the remote host above, with the user id supplied having at least 'Connected Service Rights' as defined by the remote host in its users database.

Please Select the Primary Service

Host Name Identifier:	Azure StockTrader Business Services
Select From Available Remote Endpoints:	Azure BSL wsHttp security = TransportWithMessageCredential: UserName Service Contract: TradeBusinessServiceContract.ITradeServices Service Binding Type: ws2007HttpBinding Security Mode: TransportWithMessageCredential Service Virtual Path: tradebsl.svc Service Port: 443 Uses Https: True Connected Configuration Service ID: Assigned CS UriId:
Select Client Contract to Use:	You must select from the client contracts available based on those you supplied in your initialization logic. Trade BusinessServiceContract.ITradeServices
Select Client Configuration to Use:	This list has been automatically trimmed to client configurations found in config that 1) Match the service contract selected above; 2) Use a compatible binding type and security mode; and 3) match the Configuration Service naming pattern. Select the appropriate available client configuration to use. For example, as generated by scutline via WS Metadata Exchange (mex). If you do not see the configuration you want, you need to either add it to your web.config or .exe.config file, or correct issues stemming from the matching pattern. The matching pattern prevents many runtime exceptions before they happen. Client_Ws2007HttpBinding_T_Security_MCredential_UserName_BSL Selected Binding Type: ws2007HttpBinding
Online Check Method:	isOnline
Online Check Parameters (no entry equates to a null parameter array):	None
Apply Default UserName Client Credentials:	<input checked="" type="checkbox"/> Yes, use a specific user ID defined in this configuration database to supply per request to this service. See documentation, you can also supply different credentials dynamically at runtime if desired based on other authentication sources.
Choose User for Default Client UserName Credentials (if checked above). Only users with 'Service Operation Rights' will appear in this list.	azurebsloperationuser
<input type="button" value="Add"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>	

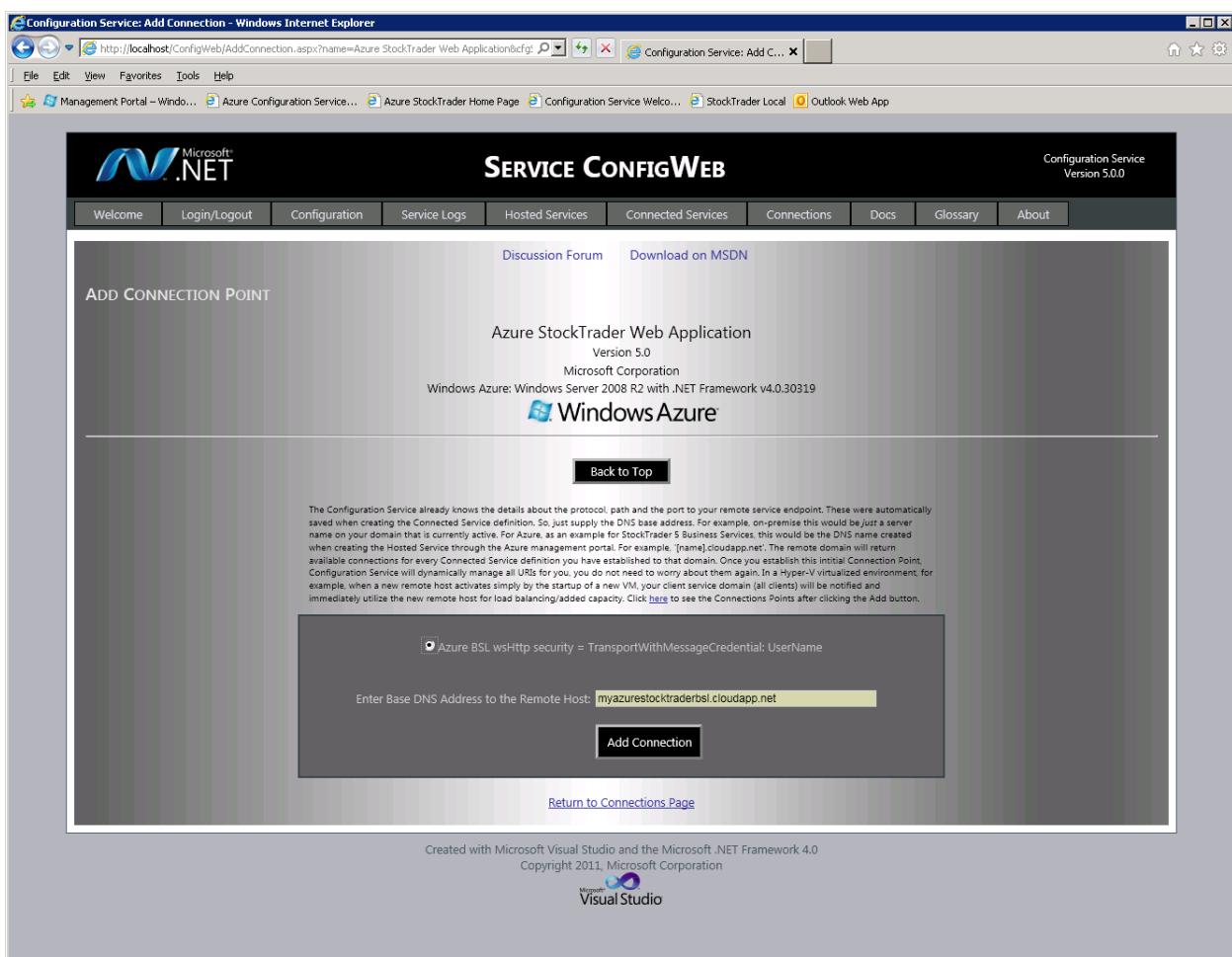
- Look carefully at the entries above. You will be establishing a Connected Service Definition to a BSL endpoint that operates with TransportWithMessageCredential security, using a username/password for authentication to the remote services. Any client not supplying this, will be rejected automatically by the StockTrader BSL. The StockTrader Web application is configured to supply a single username/password combination, much like an ADO.NET connection string works for an application.

Note

Later you can explore using ConfigWeb to change the endpoint to be the currently de-activated endpoint using x.509 client certificates for client credentials, instead of username/password. Both are just as secure, as long as going over https. These steps are documented later in this document.

- The configuration repository for the StockTrader Web application is pre-loaded with a User definition for the StockTrader BSL with the correct user id and password. So you just need to check off Apply Default UserName Client Credentials as shown above.

- Next, choose the **azurebsloperationuser** from the dropdown list.
- Now simply click the **Add** button.
- Next, click on the **Connections** top-level menu item at the top of the current ConfigWeb page.
- From here click the **Add Connection** button.
- You will then see the following:



- Note that YOUR DNS address is already supplied, since the Configuration Service now already knows it. So you do not have to do anything except click **Add Connection**; the address in the textbox is already correct.
- After the Connection Point is added, click on the **Connections** top-level menu item to see the following page:

The screenshot shows a Microsoft Configuration Service Connection Points page in Internet Explorer. The title bar reads "Configuration Service: Connection Points - Windows Internet Explorer". The address bar shows "http://localhost/ConfigWeb/ConnectionPoint.aspx". The page header includes the Microsoft .NET logo, the title "SERVICE CONFIGWEB", and a "Configuration Service Version 5.0.0" link. A navigation menu at the top includes Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Docs, Glossary, and About.

The main content area displays the "Azure StockTrader Web Application" version 5.0, running on Windows Azure. It shows a "Windows Azure" logo and a "Back to Top" button. Below this are "View Services" and "View Clients" buttons, and an "Add Connection" button.

A detailed description of the page's purpose is provided:

This page shows the actual WCF connections between the client service domain and the remote service domain for any business service endpoints, as maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/external load balancer, as is the case with Windows Azure service domains, then you will see just one connection for this service domain (with a corresponding icon), just as the client service domain sees - even though in reality there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly to each load-balanced node, with load balancing and request-level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all nodes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the WCF client being utilized, including the endpoint address, the name of the client definition as defined in configuration, the binding type, binding configuration name, and security mode the client is utilizing.

If a connection point below shows red, hover over it to see the actual exception the client is receiving. If a connection below shows as a grey-colored server, a timeout was exceeded (also indicating an issue).

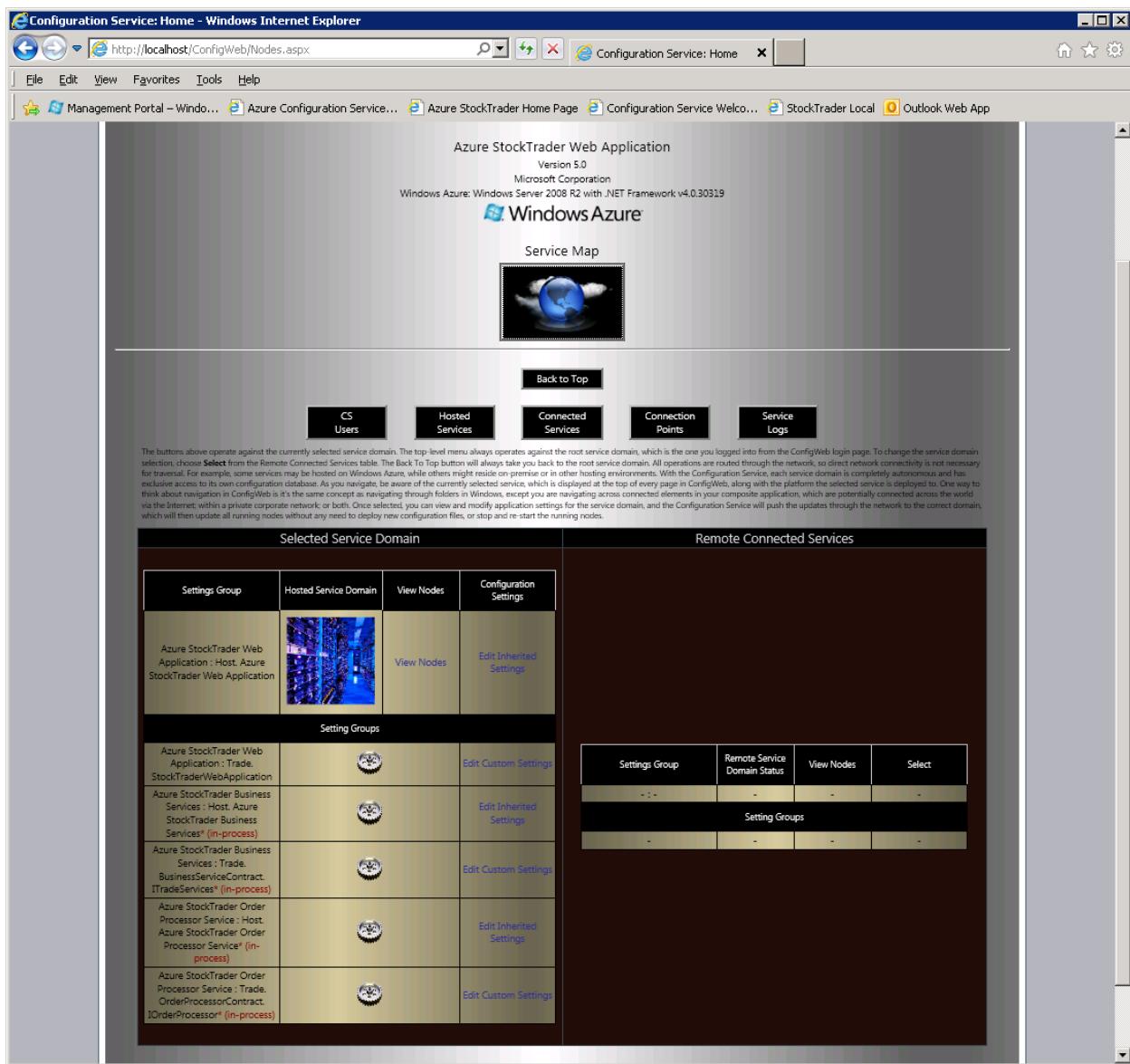
The "Connections To Host: Azure StockTrader Business Services" section contains a table with the following data:

Connections To Host: Azure StockTrader Business Services			
View/Configure This Service Host's Connection Points Purge All Connections to This Service Host			
Remote Address	Client Details	Service Status	Remove
https://myazur stocktraderbsl.cloudapp.net:443/tradebsl.svc	Azure BSL wsHttp security = TransportWithMessageCredential; UserName Client Configuration: Client_Ws2007HttpBinding_T_Security_MCredential_UserName_BSL Service Contract: TradeBusinessServiceContract_ITradeServices Binding Configuration: Client_Ws2007HttpBinding_T_Security_MCredential_UserName Binding Type: ws2007HttpBinding Security Mode: TransportWithMessageCredential Endpoint Behavior: USERNAME_CLIENT_CREDENTIAL_BEHAVIOR_BSL		Delete

At the bottom of the page, there is a "Return to Home Page" link and a copyright notice: "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0. Copyright 2011, Microsoft Corporation. Microsoft Visual Studio".

You will note the text with the friendly name of the service is in black. For StockTrader, with multiple alternate selectable service endpoints with different security modes/transport to connect to, a Connection Point currently in use will be green. This is tied to the AccessMode setting for the StockTrader Web application, and the OrderMode setting for the StockTrader Business Services. So, we need to change the AccessMode for the Web app to use this remote service, instead of running in standalone 'collapsed' mode. Now, you have already done this before for the on-premise application in the first part of the walkthrough. The steps are exactly the same, as below:

- Click on the **Configuration** menu item to go back to the home page of ConfigWeb.



- Click on **Edit Custom Settings** (the second link for the Azure StockTrader Web Application).

The screenshot shows a Microsoft Internet Explorer window titled "Configuration Service: Configuration Keys - Windows Internet Explorer". The address bar displays the URL <http://localhost/ConfigWeb/Config.aspx?name=Azure StockTrader Web>. The page header includes the Microsoft .NET logo and the title "SERVICE CONFIGWEB". The top navigation bar has links for "Welcome", "Login/Logout", "Configuration", "Service Logs", "Hosted Services", "Connected Services", "Connections", "Docs", "Glossary", and "About". Below the navigation bar, there are links for "Discussion Forum" and "Download on MSDN". The main content area is titled "CONFIGURATION KEYS" and shows the "Azure StockTrader Web Application" configuration. It includes the version information "Version 5.0" and "Microsoft Corporation" along with the note "Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319". A "Windows Azure" logo is present. Below this, there are four buttons: "Basic", "Detailed", "Advanced", and "Define Keys". The "Advanced" button is highlighted. The table below shows the "Custom Application Settings" for the target setting group "Trade.StockTraderWebApplication". The table has columns for "Setting Name", "Current Value", "Description", and "Change Value". The rows are:

Setting Name	Current Value	Description	Change Value
AccessMode	In-Process Activation	This setting determines how the Trade Web application makes calls to the Business Service Layer (BSL) over WCF. As a sample application, there are a wide variety of settings. First, there are three distinct BSL layers hosting service endpoints: 1) Windows Azure running in a Web Role; 2) An on-premise IIS-based service; 3) An on-premise EXE application that ships with the sample. For each, there are a variety of transports and security modes illustrated. The Web application will only use one service endpoint at a time; so this setting is a switch that can for example be changed to have the Web application start communicating to our on-premise BSL vs. the Azure-hosted BSL with a simple update from ConfigWeb.	Change Value
Always Check Order Alerts	false	Whether to run the closed orders check on every visit to an authenticated page to display the Order Alert message, or to use the check frequency setting value.	Change Value
Check Order Alert Frequency	15	Time in seconds to check for closed orders: only used if CheckOrderAlertsOnEveryRequest is false	Change Value
DisplayOrdersDefault	10	This setting determines the default number of orders to display in the Account.aspx page.	Change Value
DisplayOrdersMax	100	This setting determines the maximum number of orders to display in the Account.aspx page.	Change Value

At the bottom of the table, there is a link "Return to Home Page". The footer of the page includes the text "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation", followed by the Microsoft Visual Studio logo.

- Select the Access Mode Setting by clicking on Change Value.

Configuration Service: Setting Update - Windows Internet Explorer

File Edit View Favorites Tools Help

Management Portal – Windo... Azure Configuration Service... Azure StockTrader Home Page Configuration Service Welco... StockTrader Local Outlook Web App

AccessMode	In-Process Activation	<p>First, there are three distinct BSL layers hosting service endpoints: 1) Windows Azure running in a Web Role; 2) An on-premise IIS-based service; 3) An on-premise .EXE application that ships with the sample. For each, there are a variety of transports and security modes illustrated. The Web application will only use one service endpoint at a time; so this setting is a switch that can for example be changed to have the Web application start communicating to our on-premise BSL vs. the Azure-hosted BSL with a simple update from ConfigWeb.</p> <p>Premise Self-Host BSL http; On-Premise Self-Host BSL netTcp; On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate; On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: UserName; On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate; On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: UserName; On-Premise IIS Host BSL wsHttp security = Message: ClientCertificate; On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate; On-Premise IIS Host BSL Basic Http SOAP; On-Premise IIS Host BSL netTcp; Interop: Basic Http SOAP BSL</p>	4/14/2011 8:19:37 PM
------------	-----------------------	---	-------------------------

Enter/Select New Value:

In-Process Activation

Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate

Azure BSL wsHttp security = TransportWithMessageCredential: UserName

Azure BSL netTcp security = TransportWithMessageCredential: ClientCertificate

Azure On-Premise Self-Host BSL http

On-Premise Self-Host BSL netTcp

On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate

On-Premise Self-Host BSL netTcp security = TransportWithMessageCredential: UserName

On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate

On-Premise Self-Host BSL wsHttp security = TransportWithMessageCredential: UserName

On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate

On-Premise IIS Host BSL wsHttp security = Message: ClientCertificate

On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: UserName

On-Premise IIS Host BSL netTcp security = TransportWithMessageCredential: ClientCertificate

On-Premise IIS Host BSL Basic Http SOAP

On-Premise IIS Host BSL netTcp

Interop: Basic Http SOAP BSL

Update

[Return to Configuration Page](#)

- You will select the **AccessMode** as shown above:

Azure BSL wsHttp security = TransportWithMessageCredential: UserName

Note

Remember that most of the endpoints shown above are not active by default, but they are loaded into the configuration repositories so they can be activated later optionally, and the settings (AccessModes) pre-configured. And not all endpoints can be hosted simultaneously by the same service host at the same time: for example, if the security settings for the same protocol are different, one endpoint might have to be de-activated first before activating the desired endpoint; and, a different Service Behavior applied to the service host hosting the endpoint. All of this can be accomplished via ConfigWeb without programming, or redeployment, as shown later.

- Click the **Update** button.
- Go back to the **Connections Page**:

Remote Address	Client Details	Service Status	Remove
https://myazurestraderbsi.cloudapp.net:443/tradebsi.svc	Azure BSL wsHttp security = TransportWithMessageCredential:UserName Client Configuration: Client_Ws2007HttpBinding_T_Security_MCredential_UserName_BSL Service Contract: TradeBusinessServiceContractITradeServices Binding Configuration: Client_Ws2007HttpBinding_T_Security_MCredential_UserName Binding Type: ws2007HttpBinding Security Mode: TransportWithMessageCredential Endpoint Behavior: USERNAME_CLIENT_CREDENTIAL_BEHAVIOR_BSL		Delete

- You will notice the name is now showing in **green** text; meaning this connection is the active connection being used based on the AccessMode setting. Remember, you might have connections to many different hosts, and different service endpoints for each host. ConfigWeb/Configuration Service can handle this, ala StockTrader. There is no limitation to the

number of services a client can subscribe to, etc.

- If you use the StockTrader Web application as a user now, and login and visit some pages, you will be going remote to the StockTrader Business Services tier (the user never knows).
- Login to the StockTrader application as a user (perhaps as user uid:0), and click around a few pages. Place two orders as buy orders.

The screenshot shows a Microsoft Internet Explorer window displaying the StockTrader application. The title bar reads "Account Summary - Windows Internet Explorer". The main content area has a blue header with the "Windows Azure" logo, the "STOCKTRADER" logo, and the text "NET StockTrader Version 5.0.0". Below the header is a navigation menu with links like Welcome, Login/Logout, Trade Home, Account, Portfolio, Quotes, Docs, Glossary, ConfigWeb, and About. There are also links for Discussion Forum and Download on MSDN.

ACCOUNT SUMMARY

A message box displays "Trade Alert: The following orders have completed." followed by a table of completed orders:

Order ID	Order Status	Creation Date	Completion Date	Txn Fee	Type	Symbol	Quantity
7500	completed	6/9/2011 8:57:44 PM	6/9/2011 8:57:44 PM	\$15.95	buy	s103	100.00
7501	completed	6/9/2011 8:57:48 PM	6/9/2011 8:57:48 PM	\$15.95	buy	s103	100.00

Below the table are summary statistics: Subtotal Buys (\$120,454.40), Subtotal Sells (\$0.00), Subtotal Fees (\$154.40), Net Impact (\$120,608.80), and Cash Balance (\$120,608.80).

User Profile

The profile section shows account details for "uid:0":

Full Name: Full Name 0	Email Address: user0@company.com
Address: 111 First Street, Redmond, WA 98033	Password: [redacted]
Credit Card: 489023-0320	Confirm Password: [redacted]

Buttons for "Update" and "Cancel" are present.

Below the profile are account statistics:

Account ID: 0	Account Created: Sunday, April 17, 2011 5:47 PM
User ID: uid0	Last Login: Thursday, June 09, 2011 8:57 PM
Opening Balance: \$200,000.00	Total Logins: 1
Cash Balance: \$79,668.10	Total Logouts: 0

At the bottom of the page, there is a footer with the text "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation". The Visual Studio logo is also present.

- Now, as the Administrator, goto the Service Map for Azure StockTrader Web Application, and hover over the BSL service domain as shown below:

Thursday, June 09, 2011 9:23 PM (UTC)

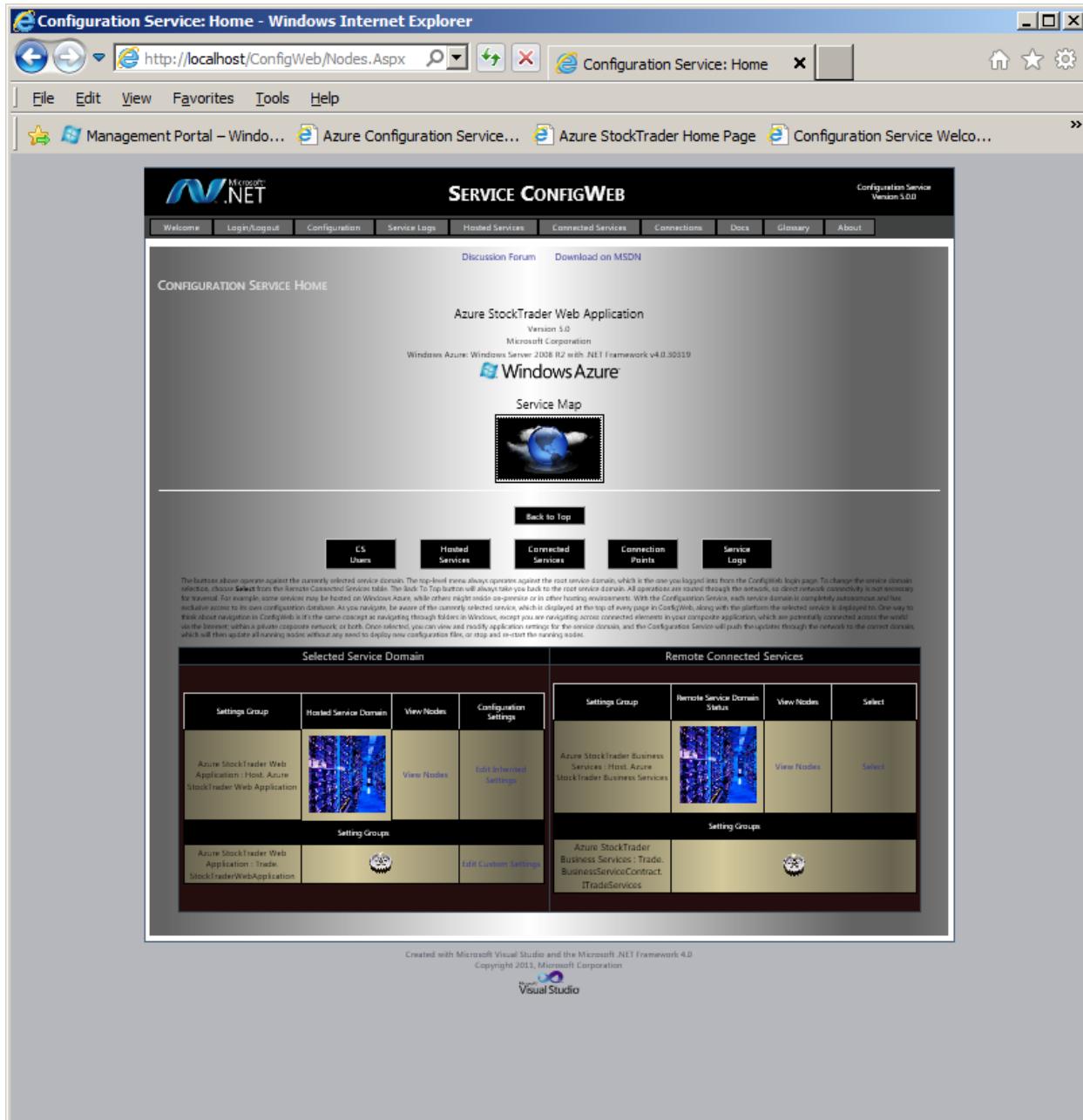
Component	ASP.NET Request/Sec	WCF Requests/Sec
Azure StockTrader Web Application	0	0
Database Servers	0	0
Distributed Caches	0	0
Azure StockTrader Web Application (WebRole) Azure Instances (Node Count=2)	0	0
[10.26.25.3] StockTrader Web Node Http (Trade_WebRole_IN_0)	0	0
[10.26.16.250] StockTrader Web Node Http (Trade_WebRole_IN_1)	0	0
Azure StockTrader Business Services	0	0
Database Servers	0	0
Distributed Caches	0	0
Azure StockTrader BSL (Web Role) Azure Instances (Node Count=2)	1	1
Windows Azure Service Domain		
Total Primary Business Service Requests Since Active: 81		
Primary Business Service Requests per Day: 81		
[10.26.16.25] StockTrader BSL Web Node Http (AzureBusinessServiceRole_IN_1)	0	0
[10.26.16.25] StockTrader BSL Web Node Http (AzureBusinessServiceRole_IN_0)	0	0

When you hover over the domain/virtual host (or an individual node), you will see how many service requests have been serviced. These are set to persist for each node on 60 second intervals, to keep accurate over time even as nodes are added, subtracted, taken offline, restarted, etc. The 60 second persist interval can be adjusted in ConfigWeb, it's another Configuration Setting managed by the Configuration Service.

Note by default the Configuration Service is counting all activations of the primary service class; this count is accurate since the service is set to activate per call. Also note that visiting one StockTrader Web page actually involves multiple remote service calls, not just one per page. So as you click around a few pages as a user in the StockTrader web application itself, you will see many more service requests by the BSL service than pages you actually visited. Each individual request is being tracked. For the Web application, we have the request tracker set to just track **valid logins**, not all page visits.

Now that the StockTrader Web application is set for remote mode to (securely) utilize the StockTrader Business Services domain; the next step is to bring the Order Processor Service into the equation. Currently, the Business Service layer is directly placing orders. So, to demonstrate the use of a service for this processing (potentially hosted by another company, or in another data center, on-premise, or in the public cloud), we will now switch the Business Service layer to send buy and sell orders to a remote service, the Order Processor Service (OPS).

- Close the Service Map. Click on the Configuration menu item to make sure you are at the home page with the root StockTrader Web application selected.



- You will now click on the '**Select**' link for the Azure StockTrader Business Services domain.

The screenshot shows the Microsoft Configuration Service: Home - Windows Internet Explorer interface. At the top, there's a navigation bar with File, Edit, View, Favorites, Tools, Help, and a toolbar with icons for Back, Forward, Stop, Refresh, and Home. Below the toolbar, there are links to Management Portal, Azure Configuration Service, Azure StockTrader Home Page, and Configuration Service Welcome.

The main content area has a Microsoft .NET logo and the title "SERVICE CONFIGWEB". It displays "Azure StockTrader Business Services Version 5.0" and "Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30519". There's a "Service Map" section with a globe icon.

Below the map, there are several buttons: "Back to Top", "CS Domains", "Hosted Services", "Connected Services" (which is circled in yellow), "Connection Points", and "Service Logs". A note below these buttons explains the function of each.

The main table is titled "Selected Service Domain" and "Remote Connected Services". It lists service domains and their settings. The "Connected Services" table lists remote service domains and their status.

At the bottom, it says "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2013, Microsoft Corporation".

- Now, with Azure StockTrader Business Services selected, click on the Connected Services Button (not the top-level menu item).

The screenshot shows a Microsoft Internet Explorer window with the title bar "Configuration Service: Connected Services - Windows Internet Explorer". The address bar displays the URL "http://localhost/ConfigWeb/CSServices.aspx?name=Azur". The page content is titled "SERVICE CONFIGWEB" and features the Microsoft .NET logo. A navigation menu at the top includes links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Docs, Glossary, and About. Below the menu, there are links for Discussion Forum and Download on MSDN. The main content area is titled "CONNECTED SERVICES" and displays information about "Azure StockTrader Business Services" (Version 5.0, Microsoft Corporation) running on "Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319". A "Back to Top" button is located near the bottom left of the content area. At the very bottom, it says "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation" with the Microsoft Visual Studio logo.

- Now click **Add Service**.

Instructions

This page enables you to establish connections to remote services your application will utilize. A primary service implements the Configuration Service; a generic service does not. For example, a generic service might be one written in Java, PHP or .NET. When establishing connections to **primary services**, you will simply connect to the configuration service of the remote host, and a list of available services (contracts) will be returned. You can also establish connection to **generic services** simply by providing the endpoint to the service, and filling out the requested details. A service implementing the Configuration Service can always be treated as a generic service, since the Configuration Service imposes no requirement on the language clients are developed in, or whether they also implement the Configuration Service.

Connected Service Type

Primary Connected Service
 Generic Connected Service

Remote Configuration Service Details - Get Connected!

Address to Configuration Service:	Once connected, the remote host will return the available services your client application can subscribe to. You can connect over http, https, or tcp to the remote Configuration Service to retrieve this list. Make sure to enter the address to the configuration service endpoint. If the host does not implement the configuration service, choose the Generic Service type above. net.tcp://myazur stocktraderops.cloudapp.net:10003/orders/config
Client Configuration Name:	Please make sure to select a valid Configuration Service Client. Service configurations displayed below are always prefixed with 'client_configsvc' in the config file 'client' section to appear in this list. You can use wsutil.exe to generate new client definitions to remote configuration services if these services expose their metadata. Client_ConfigSvc_TcpBinding_T_Security_OPS
User Id:	(see information below left)
Password:	

The credentials above are used by the Configuration Service to send/receive notifications between hosts and subscribed clients. Hosts can be configured to automatically return these credentials for any client attempting to subscribe, in which case nothing needs to be entered before connecting, the credentials will be returned. If the remote host is not configured to automatically provide these credentials, you will need to enter valid credentials to the remote host above, with the user id supplied having at least 'Connected Service Rights' as defined by the remote host in its users database.

Get Remote Services!

- You will now type in the address of your StockTrader Business Service domain (Azure Hosted Service). Shown above is [net.tcp://\[myazur stocktraderops.cloudapp.net\]:10003/orders/config](http://net.tcp://myazur stocktraderops.cloudapp.net:10003/orders/config). You will replace [myazur stocktraderops.cloudapp.net] with the address to your service domain, followed by :10003/orders/config (the net.tcp endpoint for the OPS Azure Worker Role). Make sure you start the address with net.tcp:// since the Order processor exclusively uses net.tcp, and not https, for communication. (The net.tcp endpoint also is secure/encrypted).
- Make sure you choose the **Client_ConfigSvc_TcpBinding_T_Security_OPS** client configuration (which uses net.tcp and an expected identity for the OPS based on its service certificate (StockTraderOPSService.Com)).
- Click **Get Remote Services!**

Configuration Service: Connected Service Definitions - Windows Internet Explorer

http://localhost/ConfigWeb/UpdateConnectedService.aspx Configuration Service: Conn... [Close]

File Edit View Favorites Tools Help

Management Portal – Windo... Azure Configuration Service... Azure StockTrader Home Page Configuration Service Welco... StockTrader Local Outlook Web App

Please Select the Primary Service

Host Name Identifier:	Azure StockTrader Order Processor Service																
Azure OPS netTcp security = TransportWithMessageCredential: ClientUser																	
Select From Available Remote Endpoints:	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>Service Contract:</td><td>Trade.OrderProcessorContract.IOrderProcessor</td></tr> <tr><td>Service Binding Type:</td><td>netTcpBinding</td></tr> <tr><td>Security Mode:</td><td>TransportWithMessageCredential</td></tr> <tr><td>Service Virtual Path:</td><td>orders</td></tr> <tr><td>Service Port:</td><td>10001</td></tr> <tr><td>Uses Https:</td><td>False</td></tr> <tr><td>Connected Configuration Service ID:</td><td></td></tr> <tr><td>Assigned CS Userid:</td><td></td></tr> </table>	Service Contract:	Trade.OrderProcessorContract.IOrderProcessor	Service Binding Type:	netTcpBinding	Security Mode:	TransportWithMessageCredential	Service Virtual Path:	orders	Service Port:	10001	Uses Https:	False	Connected Configuration Service ID:		Assigned CS Userid:	
Service Contract:	Trade.OrderProcessorContract.IOrderProcessor																
Service Binding Type:	netTcpBinding																
Security Mode:	TransportWithMessageCredential																
Service Virtual Path:	orders																
Service Port:	10001																
Uses Https:	False																
Connected Configuration Service ID:																	
Assigned CS Userid:																	
You must select from the client contracts available based on those you supplied in your initialization logic.																	
Select Client Contract to Use:	Trade.OrderProcessorContract.IOrderProcessor																
This list has been automatically trimmed to client configurations found in config that 1) Match the service contract selected above; 2) Use a compatible binding type and security mode; and 3) match the Configuration Service naming pattern. Select the appropriate available client configuration to use. For example, as generated by gacutil.exe via WS Metadata Exchange (. mex). If you do not see the configuration you want, you need to either add it to your web.config or .exe.config file, or correct issues stemming from the matching pattern. The matching pattern prevents many runtime exceptions before they happen.																	
Select Client Configuration to Use:	Client_TcpBinding_T_Security_MCredential_UserName_OPSSelect Selected Binding Type: netTcpBinding																
Online Check Method:	isOnline																
Online Check Parameters (no entry equates to a null parameter array):	None																
Apply Default UserName Client Credentials:	<input checked="" type="checkbox"/> Yes, use a specific user ID defined in this configuration database to supply per request to this service. See documentation, you can also supply different credentials dynamically at runtime if desired based on other authentication sources.																
Choose User for Default Client UserName Credentials (if checked above). Only users with 'Service Operation Rights' will appear in this list.	azureopsoperationuser																
<input type="button" value="Add"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>																	

[Return to Connected Services List](#)

- Make sure to select **Apply Default UserName Client Credentials**, and then use the dropdown to select the **azureopsoperationuser** (this is pre-configured with the correct default password for the OPS service).
- Click **Add**.
- After the Service Definition has been added, click on the **Connections** top-level menu link in ConfigWeb at the top of the current page.

The screenshot shows the Microsoft Configuration Service Connection Points interface. At the top, it displays the URL <http://localhost/ConfigWeb/ConnectionPoint.aspx>. The main header includes the Microsoft .NET logo and the title "SERVICE CONFIGWEB". The top navigation bar has links for Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Docs, Glossary, and About. Below the navigation, there are links for Discussion Forum and Download on MSDN.

The main content area is titled "SERVICE CONNECTION POINTS" and shows the "Azure StockTrader Web Application" details: Version 5.0, Microsoft Corporation, and Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319. It also features a "Windows Azure" logo.

Below this, there are buttons for Back to Top, View Services, View Clients, and Add Connection. A note explains the page's purpose: "This page shows the actual WCF connections between the client service domain and the remote service domain for any business service endpoints, as maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/external load balancer, as is the case with Windows Azure service domains, then you will see just one connection for this service domain (with a corresponding icon), just as the client service domain sees – even though in reality there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using NAT to translate addresses, you will see every node as a distinct connection point, as clients connect directly to each load balanced node, with load balancing and request-level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all nodes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the binding, configuration name, and security mode the client is utilizing."

The "Connections To Host: Azure StockTrader Business Services" section contains a table with the following data:

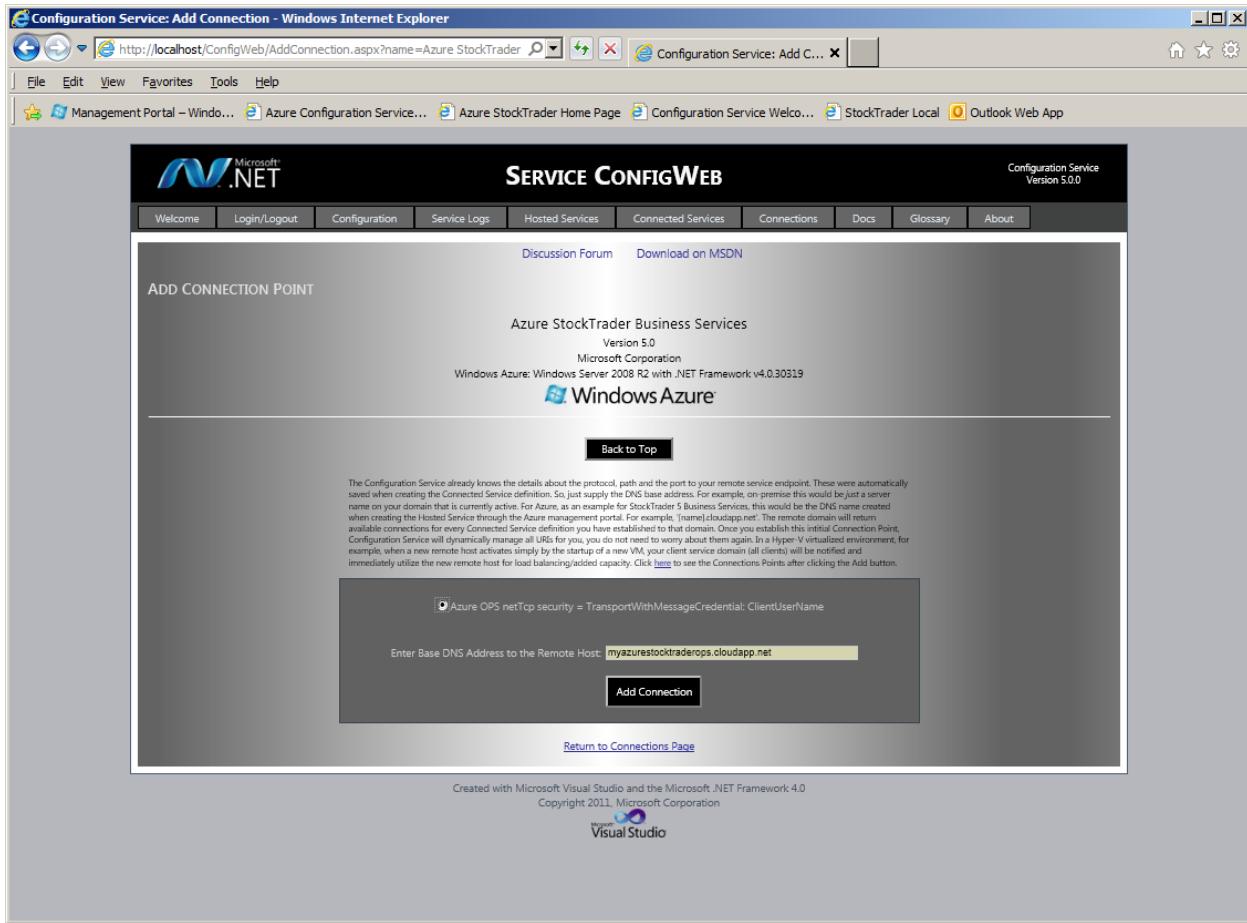
Remote Address	Client Details	Service Status	Remove
https://myazuresocktraderbsi.cloudapp.net:443/tradebsi.svc	Azure BSL wsHttp security = TransportWithMessageCredential: UserName Client Configuration: Client_Wi2007HttpBinding_T_Security_MCredential_UserName_BSL Service Contract: TradeBusinessServiceContractTradeServices Binding Configuration: Client_Wi2007HttpBinding_T_Security_MCredential_UserName Binding Type: ws2007HttpBinding Security Mode: TransportWithMessageCredential Endpoint Behavior: USERNAME_CLIENT_CREDENTIAL_BEHAVIOR_BSL		Delete

At the bottom of the table, there is a "Purge All Cache" button. Below the table, there is a "Return to Home Page" link and a note: "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0".

- At this point, you are looking at the StockTrader Web Application's (root node) connections to the Business Service tier. We need to drill down across service domains to view and configure the Business Service tier's Connection Points (to add one to the Order Processor).
- Click on **View/Configure This Service Host's Connection Points** as circled above.

The screenshot shows a Microsoft Internet Explorer window titled "Configuration Service: Connection Points - Windows Internet Explorer". The URL in the address bar is <http://localhost/ConfigWeb/ConnectionPoint.aspx?name=Azure StockTrade>. The page itself is titled "SERVICE CONFIGWEB" and features the ".NET" logo. It displays the "SERVICE CONNECTION POINTS" for the "Azure StockTrader Business Services" service. The service is listed as "Version 5.0" from "Microsoft Corporation" and is running on "Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319". There are buttons for "Back to Top", "View Services", "View Clients", and "Add Connection". A note at the bottom states: "This service currently has no defined connection points to other services." The page is footered with "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation" along with the Visual Studio logo.

- You can see at the top of the page, we are now looking at the Business Service domain's connection points to other services (and there are none, yet).
- Click **Add Connection**.



- You do not have to change the address, this is the correct address to your Order Processor Service hosted service domain, which the Configuration Service already knows.
- Simply click **Add Connection**. This might take a few seconds to complete.
- Now, click on the Connections Menu item again at the top-level menu.
- Click on **View/Configure This Service Host's Connection Points** as you did before, to drill down into the Business Service tier connection points.
- Click on **View/Configure This Service Host's Connection Points** as circled above.

Connections To Host: Azure StockTrader Order Processor Service

Remote Address	Client Details	Service Status	Remove
net.tcp://myazurestocktraderops.cloudapp.net:10001/orders	Azure OPS netTcp security = TransportWithMessageCredential; ClientUserName Client Configuration: Client_TcpBinding_T_Security_MCredential_UserName_OPS Service Contract: Trade.OrderProcessorContract.OrderProcessor Binding Configuration: Client_TcpBinding_T_Security_MCredential_UserName Binding Type: netTcpBinding Security Mode: TransportWithMessageCredential Endpoint Behavior: USERNAME_CLIENT_CREDENTIAL_BEHAVIOR_OPS		Delete

- Make sure you are looking at the right Service (BSL); and its Connection Points (to the Order Processor Service as circled above).
- Again, the Connection Point is added, but it will not be used (orders sent by the BSL to the remote Order Processor Service) until we change the Business Services's **OrderMode** setting.
- To do so, click **Back To Top** to go back to the root home page for the StockTrader Web application.

The screenshot shows the Configuration Service Home page in Microsoft Internet Explorer. The title bar reads "Configuration Service: Home - Windows Internet Explorer". The address bar shows the URL "http://localhost/ConfigWeb/Nodes.aspx". The page header features the Microsoft .NET logo and the title "SERVICE CONFIGWEB". The top right corner indicates "Configuration Service Version 5.0.0". The main content area is titled "CONFIGURATION SERVICE HOME" and displays the "Azure StockTrader Web Application" (Version 5.0) running on "Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319". Below this, there is a "Service Map" section with a globe icon. Navigation links include "Discussion Forum" and "Download on MSDN". A note at the bottom left explains the "Back To Top" button's function. The "Remote Connected Services" table has two sections: "Selected Service Domain" and "Remote Connected Services". The "Selected Service Domain" section shows "Azure StockTrader Web Application : Host: Azure StockTrader Web Application" with "View Nodes" and "Edit Inherited Settings" options. The "Remote Connected Services" section shows "Azure StockTrader Business Services : Host: Azure StockTrader Business Services" with "View Nodes" and "Select" options. A note at the bottom right of the table states: "The buttons above operate against the currently selected service domain. The top-level menu always operates against the root service domain, which is the one you logged into from the ConfigWeb login page. To change the service domain selection, choose Select from the Remote Connected Services table. The Back To Top button will always take you back to the root service domain. All operations are routed through the network; so direct network connectivity is not necessary for traversal. For example, some services may be hosted on Windows Azure, while others might reside on-premise or in other hosting environments. With the Configuration Service, each service domain is completely autonomous and has exclusive access to its own configuration database. As you navigate, be aware of the currently selected service, which is displayed at the top of every page in ConfigWeb, along with the platform the selected service is deployed to. One way to think about navigation in ConfigWeb is it's the same concept as navigating through folders in Windows, except you are navigating across connected elements in your computer application, which are potentially connected across the world via the Internet, within a private corporate network, or both. Once selected, you can view and modify application settings for the service domain, and the Configuration Service will push the update through the network to the correct domain, which will then update all running nodes without any need to deploy new configuration files, or stop and re-start the running nodes."

- Now click Select to select the remote Business Services domain.

Configuration Service: Home - Windows Internet Explorer

File Edit View Favorites Tools Help

Management Portal – Windo... Azure Configuration Service... Azure StockTrader Home Page Configuration Service Welco... StockTrader Local Outlook Web App

Azure StockTrader Business Services
Version 5.0
Microsoft Corporation
Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319
Windows Azure

Service Map

Back to Top

CS Users Hosted Services Connected Services Connection Points Service Logs

The buttons above operate against the currently selected service domain. The top-level button always operates against the root service domain, which is the one you logged into from the ConfigWeb login page. To change the service domain selection, choose Select from the Remote Connected Services table. The Back To Top button will always take you back to the root service domain. All operations are routed through the network, so direct network connectivity is not necessary for traversal. For example, some services may be hosted on Windows Azure, while others might reside on-premise or in other hosting environments. With the Configuration Service, each service domain is completely autonomous and has exclusive access to its own configuration database. As you navigate, be aware of the currently selected service, which is displayed at the top of every page in ConfigWeb, along with the platform the selected service is deployed to. One way to think about navigation in ConfigWeb is it's the same as navigating between different windows; except, changes are made across connected elements in your computer application, which are potentially connected across the world via the Internet, within a private corporate network, or both. Once selected, you can view and modify application settings for the service domain, and the Configuration Service will push the updates through the network to the correct domain, which will then update all running nodes without any need to deploy new configuration files, or stop and re-start the running nodes.

Selected Service Domain				Remote Connected Services			
Settings Group	Hosted Service Domain	View Nodes	Configuration Settings	Settings Group	Remote Service Domain Status	View Nodes	Select
Azure StockTrader Business Services : Host: Azure StockTrader Business Services		View Nodes	Edit Inherited Settings	-	-	-	-
Azure StockTrader Business Services : Trade, BusinessServiceContract, ITradeServices		Edit Custom Settings	-	-	-	-	-
Azure StockTrader Order Processor Service : Host: Azure StockTrader Order Processor Service* (in-process)		Edit Inherited Settings	-	-	-	-	-
Azure StockTrader Order Processor Service : Trade, OrderProcessorContract, IOrderProcessor* (in-process)		Edit Custom Settings	-	-	-	-	-

- Now click **Edit Custom Settings** (the second link down).

The screenshot shows a Microsoft .NET Configuration Service interface. At the top, it displays 'Configuration Service: Configuration Keys - Windows Internet Explorer' and the URL 'http://localhost/ConfigWeb/Config.aspx?name=Azure StockTrader Business'. The main content area is titled 'SERVICE CONFIGWEB' and shows 'Azure StockTrader Business Services' with 'Version 5.0' and 'Microsoft Corporation'. Below this, it says 'Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319'. A 'Windows Azure' logo is present. The page includes a 'Back to Top' button and tabs for 'Basic', 'Detailed', 'Advanced', and 'Define Keys'. A note about inherited settings is visible. The 'Custom Application Settings' table has three rows:

Setting Name	Current Value	Description	Change Value
OrderMode	In-Process Activation	The setting is similar to the Access Mode setting for the StockTrader Web application. It indicates how the BSL is invoking calls to the Order Processor Service. As a service application, there are a variety of settings. For example, to two endpoints to simultaneously invoke Ops services, or to one endpoint to Azure. And several different WCF bindings with different transports and security models are selectable. The BSL will only use one endpoint at a time, to do so, this setting is a switch that tells the BSL which host, and how to invoke the Order Processor. As a side-note, calls over WCF are asynchronous (one-way), we do not need to wait for a response from the service, we just need to submit the order successfully. This can improve performance. In-Process Activation is a special setting that tells the BSL not to invoke a remote OPS service, but process the order directly within the BSL process itself.	Change Value
Encrypt Account Passwords in DB: Use Salted Hash	true	If true (default), StockTrader user passwords for registered users of the StockTrader application will be stored as one-way encrypted using a salted hash. If false, passwords will be stored as plain text. For compatibility with non-.NET implementations of business services (such as WS02 against the same physical SQL Server databases), set to false. Note, however, the SQL Database Loader must be used to appropriately re-load a data set that matches this setting.	Change Value
LoginDisplayLogins	false	In the service host console, whether to display logins (for monitoring activity/demo). If set to true, please keep the corresponding setting for number of iterations above 100 if doing benchmarks, as these operations are somewhat expensive.	Change Value

- Click **OrderMode** setting's **Change Value** link.

Configuration Service: Setting Update - Windows Internet Explorer

http://localhost/ConfigWeb/ConfigUpdate.aspx?name=Azure StockTrader B Configuration Service: Setting Update

OrderMode	In-Process Activation	<p>According to Company, two distinct service domains hosting the OPS--on-premise and Windows Azure. And, several different WCF bindings with different transports and security modes are selectable. The BSL will only use one endpoint at a time, so this setting is a switch that tells the BSL which host, and how to invoke the Order Processor. As a side-note, calls over WCF are asynchronous (one-way), we do not need to wait for a response from the service, we just need to submit the order successfully. This can improve performance. In-Process Activation is a special setting that tells the BSL not to invoke a remote OPS service, but process the order directly within the BSL process itself.</p> <p>TransportWithMessageCredential: ClientUserName; Azure OPS wsHttp security = TransportWithMessageCredential: ClientCertificate; Azure OPS wsHttp security = TransportWithMessageCredential: ClientUserName; On-Premise Self-Host OPS netTcp security = TransportWithMessageCredential: ClientCertificate; On-Premise Self-Host OPS netTcp security = TransportWithMessageCredential: ClientUserName; On-Premise Self-Host OPS wsHttp security = TransportWithMessageCredential: ClientCertificate; On-Premise Self-Host OPS wsHttp security = TransportWithMessageCredential: ClientUserName; On-Premise Self-Host OPS netTcp; On-Premise Self-Host OPS http; On-Premise Self-Host OPS MSMQ Transacted Queue; Interop: Basic Http SOAP OPS</p>	4/14/2011 8:49:45 PM
-----------	-----------------------	--	-------------------------

Enter/Select New Value:

In-Process Activation

Azure OPS netTcp security = TransportWithMessageCredential: ClientCertificate

Azure OPS netTcp security = TransportWithMessageCredential: ClientUserName

Azure OPS wsHttp security = TransportWithMessageCredential: ClientCertificate

Azure OPS wsHttp security = TransportWithMessageCredential: ClientUserName

On-Premise Self-Host OPS netTcp security = TransportWithMessageCredential: ClientCertificate

On-Premise Self-Host OPS netTcp security = TransportWithMessageCredential: ClientUserName

On-Premise Self-Host OPS wsHttp security = TransportWithMessageCredential: ClientCertificate

On-Premise Self-Host OPS wsHttp security = TransportWithMessageCredential: ClientUserName

On-Premise Self-Host OPS netTcp

On-Premise Self-Host OPS http

On-Premise Self-Host OPS MSMQ Transacted Queue

Interop: Basic Http SOAP OPS

Update

[Return to Configuration Page](#)

- Select the active endpoint as shown. This is:

Azure OPS netTcp security = TransportWithMessageCredential: ClientUserName

- Click the **Update** button.

- Click on the top-level Configuration menu item (or Back To Top).

The screenshot shows a Microsoft Internet Explorer window with the title bar "Configuration Service: Home - Windows Internet Explorer" and the URL "http://localhost/ConfigWeb/Nodes.aspx". The page itself is titled "SERVICE CONFIGWEB" and features the Microsoft .NET logo. It displays the "Azure StockTrader Web Application" version 5.0 from Microsoft Corporation, running on Windows Azure. A "Service Map" section shows a globe icon. Below it, there are five navigation links: "CS Users", "Hosted Services", "Connected Services", "Connection Points", and "Service Logs". A note at the bottom left explains the "Selected Service Domain" and "Remote Connected Services" sections. In the "Remote Connected Services" section, the "Select" button for the "Azure StockTrader Business Services" row is highlighted with a yellow circle.

Selected Service Domain

Settings Group	Hosted Service Domain	View Nodes	Configuration Settings
Azure StockTrader Web Application : Host. Azure StockTrader Web Application		View Nodes	Edit Inherited Settings
Setting Groups			
Azure StockTrader Web Application: Trade. StockTraderWebApplication		Edit Custom Settings	

Remote Connected Services

Settings Group	Remote Service Domain Status	View Nodes	Select
Azure StockTrader Business Services : Host. Azure StockTrader Business Services		View Nodes	Select
Setting Groups			
Azure StockTrader Business Services : Trade. BusinessServiceContract. ITradeServices			

- Now choose Select to select the Business Service tier.

Configuration Service: Home - Windows Internet Explorer

File Edit View Favorites Tools Help

Management Portal – Windo... Azure Configuration Service... Azure StockTrader Home Page Configuration Service Welco... StockTrader Local Outlook Web App

Microsoft .NET SERVICE CONFIGWEB Configuration Service Version 5.0

Welcome Login/Logout Configuration Service Logs Hosted Services Connected Services Connections Docs Glossary About Discussion Forum Download on MSDN

CONFIGURATION SERVICE HOME

Azure StockTrader Business Services
Version 5.0
Microsoft Corporation
Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319

Windows Azure

Service Map

Back to Top

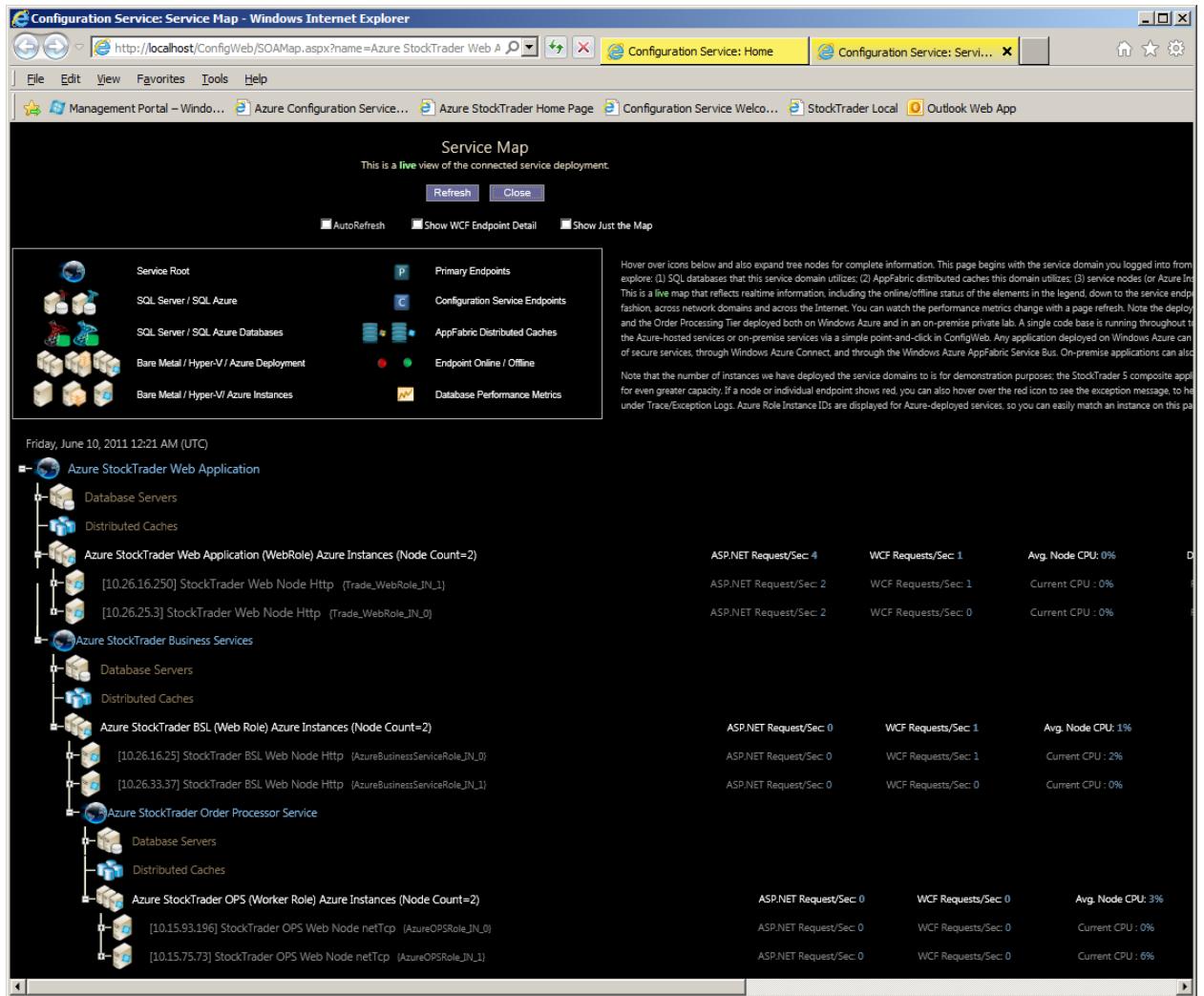
CS Users Hosted Services Connected Services Connection Points Service Logs

The bottom-level navigation against the currently selected service domain. The top-level menu always operates against the root service domain, which is the one you logged into from ConfigWeb login page. To change the service domain for traversal, choose **Select** from the Remote Connected Services table. The **Select** button enables you to back to the current service domain configuration page directly. In addition, the Configuration Service, each service domain is completely autonomous and has exclusive access to its own configuration database. As you navigate, be aware of the currently selected service, which is displayed at the top of every page in ConfigWeb, along with the platform the selected service is deployed to. One way to think about navigation in ConfigWeb is it's the same concept as navigating through folders in Windows, except you are navigating across connected elements in your composite application, which are potentially connected across the world via the Internet, within a private corporate network, or both. Once selected, you can view and modify application settings for the service domain, and the Configuration Service will push the updates through the network to the correct domain, which will then update all running nodes without any need to deploy new configuration files, or stop and re-start the running nodes.

Selected Service Domain				Remote Connected Services			
Settings Group	Hosted Service Domain	View Nodes	Configuration Settings	Settings Group	Remote Service Domain Status	View Nodes	Select
Azure StockTrader Business Services : Host: Azure StockTrader Business Services		View Nodes	Edit Inherited Settings	Azure StockTrader Order Processor Service : Host: Azure StockTrader Order Processor Service		View Nodes	Select
Setting Groups				Setting Groups			
Azure StockTrader Business Services : Trade, BusinessServiceContract, ITradeServices		Edit Custom Settings		Azure StockTrader Order Processor Service : Trade, OrderProcessorContract, IOrderProcessor			

- Now the StockTrader Business Service Tier is selected, and from here you can traverse the network another layer by clicking on Select for the Order Processor Service if you want.

- Click on the Service Map button.



- You can now see the complete composite application deployment using all three tiers (plus SQL Azure database) of the application.
- If you login to the StockTrader Web application now, and place some sell and buy orders; they will be sent to the remote OPS tier for processing. You can do so, then refresh the Service Map page above to see the nodes for the OPS that processed the requests (hover over the nodes for the OPS above).

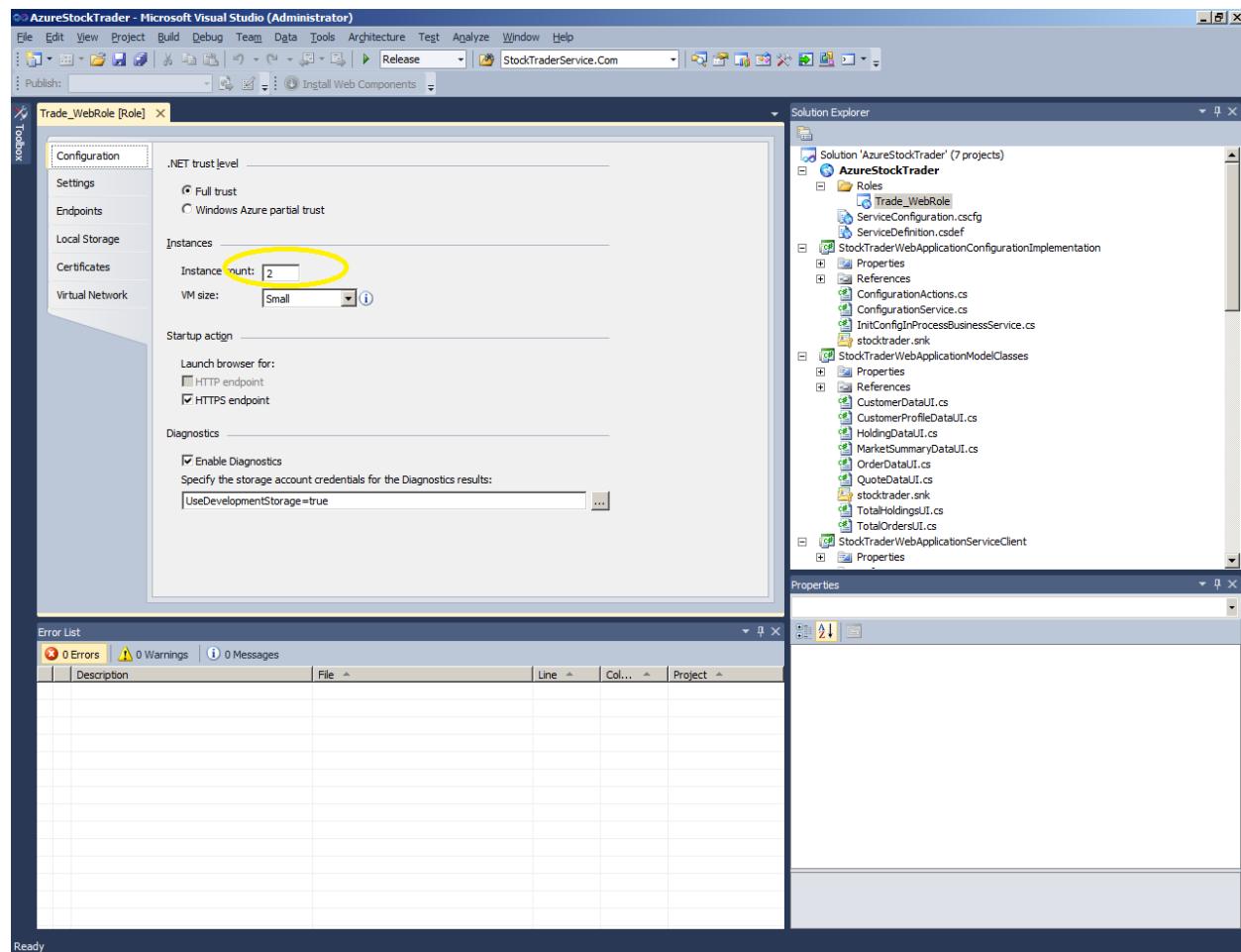
Adding Scale-Out, Load Balanced Nodes

For **on-premise deployments**, to add a new node, simply install StockTrader on a new computer and indicate during install that the databases have already been created. Type in the address to the database server; the new installation will ensure your services share their single central

configuration repository. Hence, their configurations will be kept in sync. Simply startup the service hosts on the new computers, and the configuration system will add them as nodes; and notify existing clients of the presence of the new nodes. Closing the host program (or stopping the IIS service for IIS-hosted services) will similarly cause the node to remove itself from the system, and notify clients. The Web application and business service clients detect failed service nodes, and make a new request to a different node if more than one node is online. They will stop directing traffic to failed nodes altogether after a configurable number of consecutive failed online check requests (6 by default).

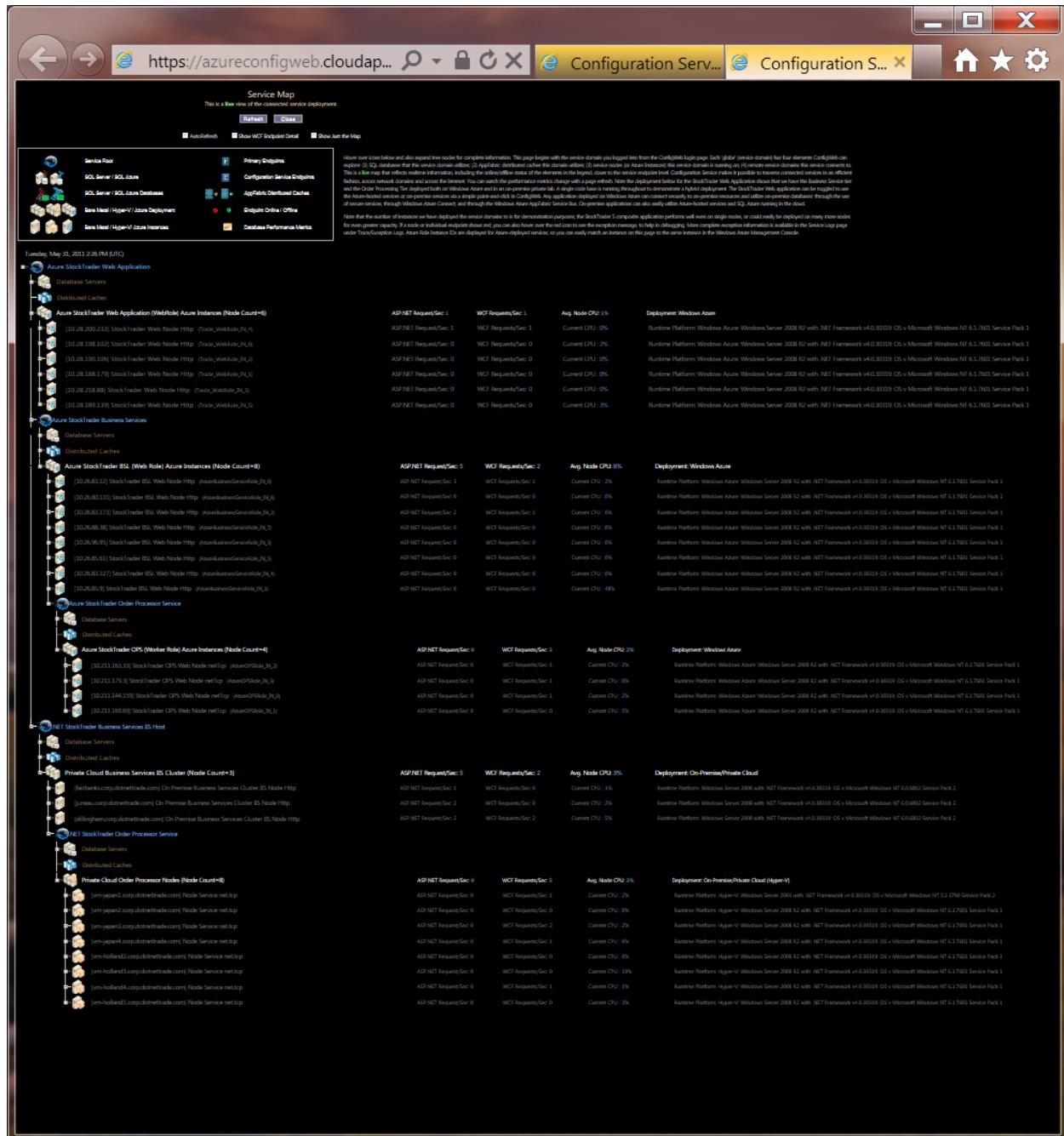
For Azure deployments, you simply use Visual Studio to change the number of nodes (and re-publish/upgrade the application). See the circled item below.

Or, you can also edit the configuration file from the Azure Management Portal to change the nodes the hosted service domain scales out across. Windows Azure automatically load balances requests against the nodes, which live behind the NAT/Azure Load Balancer (Azure Fabric Controller).



Below, you can see in our Azure/Private Cloud hybrid deployment we have scaled each tier to add more capacity, both on Windows Azure and in our private cloud running on Hyper-V/with System

Center VMM 2012. On Azure, Having at least two nodes active is important, as it ensures high availability and makes you eligible for the Windows Azure compute SLA for uptime.



For **on-premise**, you can also load balance/cluster using Windows Network Load Balancing (aka NLB; a feature integrated into Windows Server), or hardware load balancing. This can be accomplished on any tier. For example, the Web application, which is accessed by an unknown number of anonymous users via a browser, would be load balanced using NLB or a hardware load balancer, and its address hence virtualized to the outside world. Also, NLB or hardware load balancers such as F5 BigIP can be used to accomplish load balancing and failover for the Business

Services and the Order Processor Service (or any WCF service) in the same manner. *If you do so, you have to mark your Primary and Configuration Service endpoints with the load-balanced IP address:port or DNS address:port using ConfigWeb in the hosted services pages.

Azure Security Discussion and Changing the Security Configuration for WCF Endpoints

User Name Credentials

On install the StockTrader Web application, Business Services and Order Processor use no security for the on-premise installation. However, if you deploy the Azure projects, security is enabled by default, using User Name client credentials. The on-premise elements can be reconfigured to these same secure modes as well, optionally, via ConfigWeb. In this section we will discuss this topic, and show how it works. Later we will walk through some steps for the Azure hosted version to change the security from User Name credentials to Client Certificate credentials.

WCF supports a wide variety of security modes. Azure StockTrader uses https/ssl for the Web application to protect http transmissions, since these over-the-wire transmissions include the users' personal stock data, passwords for sign on, etc. While the StockTrader Web application authenticates the user logins via a custom Account table in the StockTrader database, the StockTrader Businesses services and Order Processor Service do not implement their own independent authentication (although they could be changed to authenticate every request with custom logic). So, to lock-down these internet-exposed service endpoints, we are using WCF security. This ensures that only the StockTrader Web application itself can access the BSL tier, and only the BSL tier can access the OPS tier to submit orders. We are doing so with WCF security set to TransportWithMessageCredentials. So, all communication is over https/ssl to protect the content of the messages over-the-wire; and additionally the client must send client credentials that the WCF infrastructure automatically examines before it allows a service operation to be invoked. A Service Certificate (X.509) is used (and configured on the WCF services) in this configuration, both for the Business Service tier which operates over https/ssl; and for the Order Processor which operates over net.tcp.

By default, the client credentials are configured as User Name client credentials, which means on each service request to the BSL, for example, from the Web application, the Web application has to send a userid/password that WCF then authenticates on the service side. This is independent of how the StockTrader Web application uses ASP.NET Forms authentication to authenticate individual registered users of the application via the login page. Rather, the StockTrader Web application uses a global username/password much like applications use a username password (in the ADO.NET connection string) to connect to a database.

Where Does the User Name Come from and How is it Authenticated?

The username/password the StockTrader Web application supplies on each request to the Business Services tier is stored in the Configuration database, and can be configured via ConfigWeb. When establishing Connected Service Definitions, checking off “Supply Default Username Credentials” allows you to specify which username/password is used. You can create new user ids, or change the passwords on the ones already loaded using the Users menu item in ConfigWeb. If you have everything working on Azure in remote modes, you can test the security by using the Users page to change the password for the ‘**azurebusinessserviceopsuser**’ user (here ops means service operations: ‘ops’). If you change the password for this user, which is defined in both the StockTrader configuration database and the Business Service database, on either tier so they do not match, all requests will be rejected by the BSL that come from the StockTrader Web application (until you make them match again). So you can test this out now using ConfigWeb as follows:

- Login to the StockTrader Web Application configuration service via ConfigWeb.

The screenshot shows the Service ConfigWeb interface running in a web browser. The title bar reads "Configuration Serv...". The main content area is titled "SERVICE CONFIGWEB" and displays the "Configuration Service Version 5.0". Below this, the "CONFIGURATION SERVICE HOME" section includes links for "Discussion Forum" and "Download on MSDN". The "Azure StockTrader Web Application" section shows its version (Version 5.0) and deployment details ("Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319"). A "Service Map" button is present. The bottom of the page contains navigation links for "CS Users", "Hosted Services", "Connected Services", "Connection Points", and "Service Logs". A note at the bottom explains the navigation behavior. The "Selected Service Domain" and "Remote Connected Services" tables are shown, listing service domains and their settings. The "Selected Service Domain" table has two rows: "Azure StockTrader Web Application : Host. Azure StockTrader Web Application" and "Azure StockTrader Web Application : Trade. StockTraderWebApplication". The "Remote Connected Services" table has two rows: "Azure StockTrader Business Services : Host. Azure StockTrader Business Services" and "Azure StockTrader Business Services : Trade. BusinessServiceContract. ITradeServices". The footer indicates the page was created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0, Copyright 2011, Microsoft Corporation, and features the Microsoft Visual Studio logo.

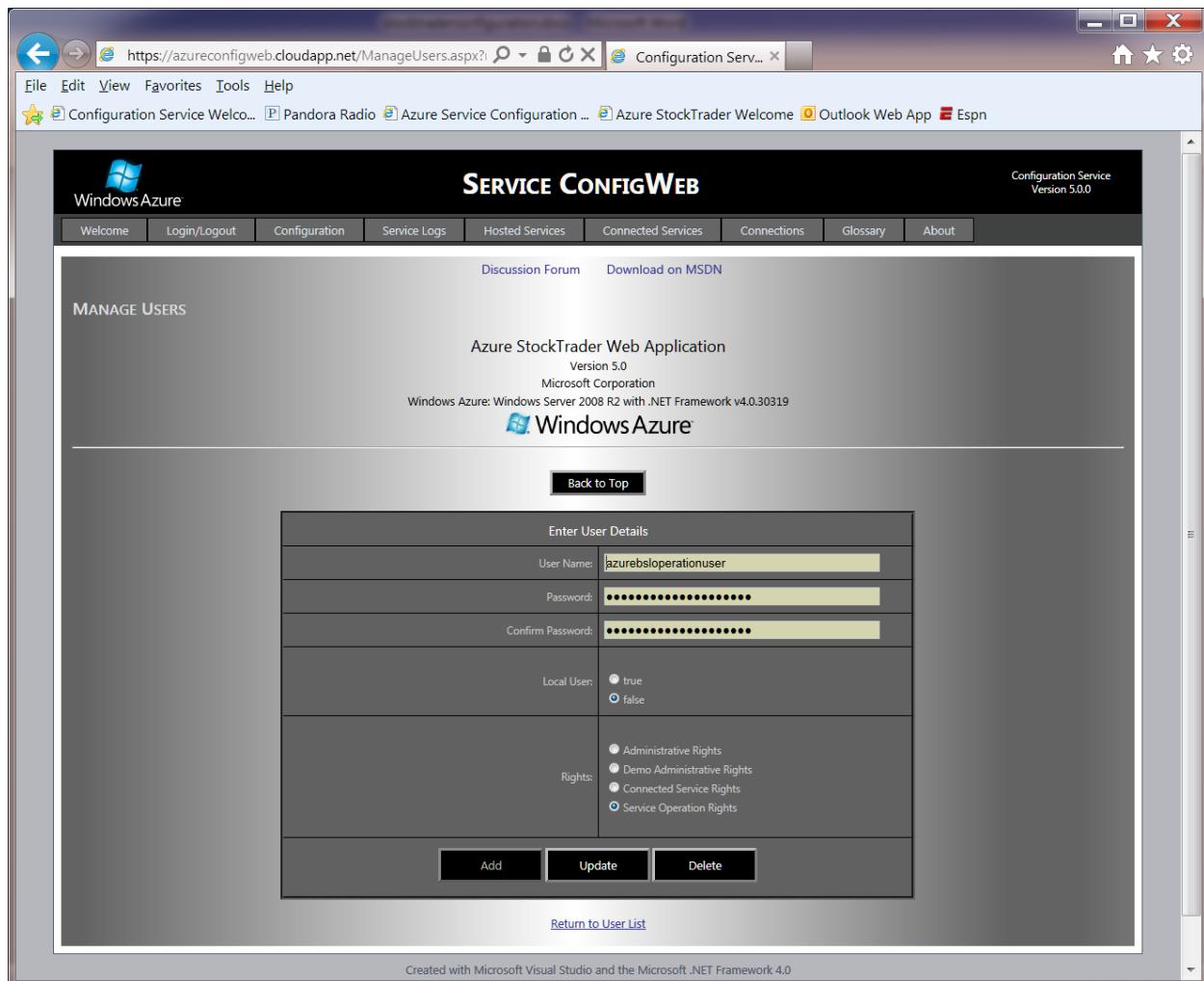
- Click on the **CS Users** button.

The screenshot shows a Windows Azure application window titled "SERVICE CONFIGWEB". The main content area displays the "Azure StockTrader Web Application" version 5.0, Microsoft Corporation, and the Windows Azure logo. Below this, there are two buttons: "Back to Top" and "Add User". A table lists users with columns: User Key, User Name, Is Local User, Rights, and Edit. The table contains the following data:

User Key	User Name	Is Local User	Rights	Edit
1	localadmin	true	Administrative Rights	Edit
2	webcsuser	true	Connected Service Rights	Edit
42	bsloperationuser	false	Service Operation Rights	Edit
67	compositeadmin	true	Administrative Rights	Edit
145	demoadmin	true	Demo Administrative Rights	Edit
147	azurebsloperationuser	false	Service Operation Rights	Edit
148	azurebslcuser	false	Connected Service Rights	Edit
150	bsliishostcsuser	false	Connected Service Rights	Edit

At the bottom of the page, there is a link "Return to Home Page" and a Microsoft Visual Studio footer.

- Select the Edit link for the '*azurebsloperationuser*' user.



- Change the password to '**test**'. The BSL service host will still expect '**azurebsluser#1**' as the password. So this will break the Azure deployment by creating a security violation (on purpose!) after changing the password above and clicking Update.
- Click on the **Connections** menu item.

The screenshot shows the Service Config Web interface at <https://azureconfigweb.cloudapp.net/ConnectionPoint.aspx>. The main title bar says "Configuration Serv...". The page header includes "Windows Azure", "SERVICE CONFIGWEB", "Configuration Service Version 5.0", and links for "Welcome", "Login/Logout", "Configuration", "Service Logs", "Hosted Services", "Connected Services", "Connections", "Glossary", and "About". Below the header is a "Discussion Forum" and a "Download on MSDN" link.

The main content area is titled "SERVICE CONNECTION POINTS" and displays "Azure StockTrader Web Application Version 5.0 Microsoft Corporation Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319". It features a "Back to Top" button, "View Services", "View Clients", and an "Add Connection" button.

A note below the clients section states: "This page shows the actual WCF connections between the client service domain and the remote service domain for any business service endpoints, as maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/internal load balancer, as is the case with Windows Azure service domains, then you will see just one connection for this service domain (with a corresponding local IP), just as the client service domain sees – even though in reality there may be many nodes serving this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly with each node. Note that the Configuration Service itself uses its NAT settings. Unlike the Hosted Web Pages display all routes even if defined a NAT. This page is important in that it allows you to view the client service domain is actually operating against. As such, you can use the details on this page to troubleshoot security issues involving the endpoint address, the name of the event endpoint as defined in configuration, the binding type/binding configuration name, and security mode the client is utilizing."

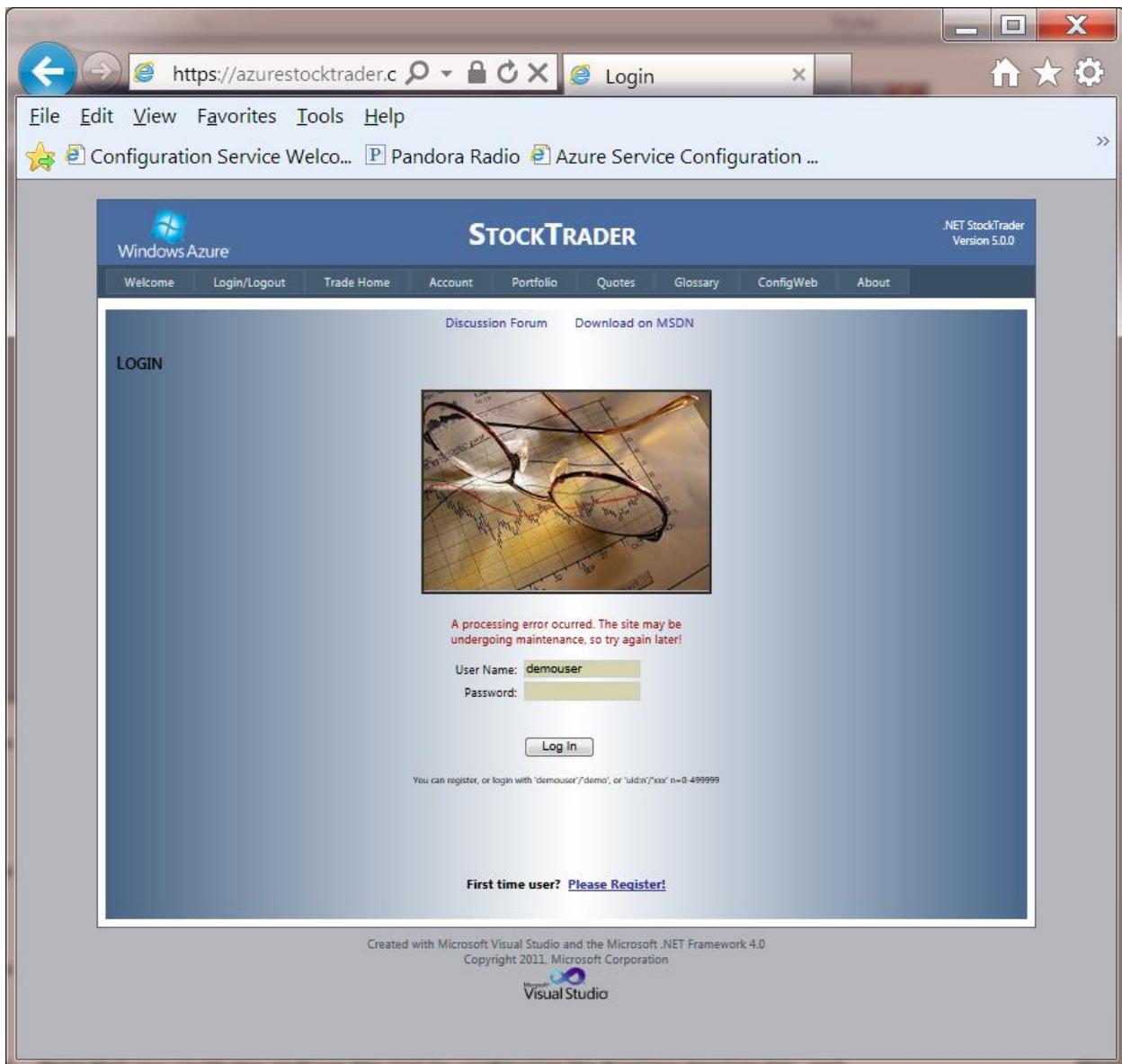
The "Connections To Host: Azure StockTrader Business Services" table lists a single connection:

Remote Address	Client Details	Service Status	Remove
https://azurestocktraderbsl.cloudapp.net:443/tradebsl.svc	Azure BSL web service - TradeBusinessServiceContract, Windows Azure Service Bus Credential: UserName Client Configuration: Client_Ws2007HttpBinding_T_Security_MICredential, UserName_B6 Service Contract: TradeBusinessServiceContractTradeServices Binding Configuration: Client_Ws2007HttpBinding_T_Security_MICredential, UserName Binding Type: ws2007HttpBinding Security Mode: MessageSecurityCredential Endpoint Behavior: USERNAME_CLIENT_CREDENTIAL_BEHAVIOR_B6L	Green (Up)	Delete

A message box on the right side of the table says: "NAT Node: Exception has been thrown by the target of an invocation. An unsecured or incorrectly secured fault was received from the other party. See the inner FaultException for the fault code and detail..."

At the bottom of the page are links for "Return to Home Page", "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0", "Copyright 2012 Microsoft Corporation", and the "Visual Studio" logo.

- Immediately you will notice the connection shows up as **red**. You can hover over it to see the security exception message. More detailed exception information is always logged and viewable in the **Service Logs** page.
- Try to login as an end-user to the StockTrader Web application (make sure you are running in remote mode as configured in the previous section). You will not be able to.



- You can repeat this test for the Business Service tier with respect to its client credentials supplied to the Order Processor Service. If you change these so they do not match what is expected by the OPS (which uses the same mechanism for authentication), you will be able to log into the StockTrader Web application as a registered user, but not place trade orders.
- Also if you were to change the Connected Service Definition (for either the StockTrader Web application-to-BSL definition or the Business Service tier-to Order Processor Service) to uncheck the checkbox '**Apply Default UserName Client Credentials**', the StockTrader Web application will be unable to make remote service requests to the BSL (or BSL to OPS, depending on which one you change for the respective service domain).

The screenshot shows the 'Primary Service Details' page for the service 'Azure StockTrader Business Services'. Key configuration details include:

- Host Name Identifier:** Azure StockTrader Business Services
- Service friendly name as assigned by the Service Hoster:** Azure_BSL and its proxy C:\Temp\WIF\MessageCredential_UserName
- Service Contract:** Trade.BusinessServiceContract.ITradeServices
- Service Binding Type:** ws2007HttpBinding
- Security Mode:** TransportWithMessageCredential
- Service Virtual Path:** Determined Dynamically
- Protocol:** http
- UseHttps:** False
- Connected Configuration Service ID:** 1
- Assigned CS User:** 146_aurebuser

If, in ConfigWeb, you now go back to the Service Logs page (Trace and Error Logs), you will see the following exception after the login attempt above:

The screenshot shows the 'Error/Event Log' section of the Service Logs page. A single log entry is displayed:

```

System.Reflection.TargetInvocationException: Exception has been thrown by the target of an invocation. ---> System.ServiceModel.Security.MessageSecurityException: An unsecured or incorrectly secured message was received from the other party. See the inner FaultException for more details. ---> System.ServiceModel.FaultException<FaultCode>: All security tokens in the message could not be validated. — End of inner exception stack trace —> Stack trace at System.ServiceModel.Channels.SecurityChannelFactory<TService>.RequestChannel.ProcessReply(Message reply, TimeSpan timeout) at System.ServiceModel.Channels.SecurityChannelFactory<TService>.RequestChannel<T>(Message message, TimeSpan timeout) at System.ServiceModel.Channels.TransportChannel<T>.Request(Message message, TimeSpan timeout) at System.ServiceModel.Channels.TransportChannel<T>.Invoke(Message message, TimeSpan timeout) at System.ServiceModel.Channels.TransportChannel<T>.Invoke<T>(Object[] args, Object[] outs, TimeSpan timeout) at System.ServiceModel.Channels.ServiceChannelProxy.InvokeService<T>(MethodCall methodCall) at ProxyOperationRuntime<T>.Invoke<T>(Object[] args, Object[] outs, TimeSpan timeout) at System.ServiceModel.Channels.TransportChannel<T>.Invoke<T>(Message message, TimeSpan timeout) at System.Runtime.Remoting.Proxies.RealProxy.PrivateInvoke(MessageData& msgData, Int32 type) at Trade.BusinessServiceContract.ITradeServices<T>.End<T>(IAsyncResult result) — End of inner exception stack trace —> ClientAccessException: Client.Client.get_Channel()<2>[1] (String server) in C:\stocktrader\StockTraderWebApplication\StockTraderWebApplicationClients\BusinessServiceClient.csline 1127 at ConfigService.LoadBalancingClient.Client.get_Channel()<2>[1] (String server) in C:\stocktrader\StockTraderWebApplication\StockTraderWebApplicationClients\BusinessServiceClient.csline 2379 at TradeStockTraderWebApplicationServiceClient.BusinessServiceClient.Login(String userId, String password) in C:\stocktrader\StockTraderWebApplication\StockTraderWebApplication\StockTraderWebApplicationClients\BusinessServiceClient.csline 81 at TradeStockTraderWebApplicationServiceClient.BSLClient.Login(String userId, String password) in C:\stocktrader\StockTraderWebApplication\StockTraderWebApplication\StockTraderWebApplicationClients\BSLClient.csline 88

```

- Now in ConfigWeb, change the password back to the default of '`azurebsluser#1`'. Refresh the connections page, and try the Login for the StockTrader Web application again. All will be well again.
- Note that you could also perform this test by configuring the Business Services tier, and changing the password for this same user id via ConfigWeb for this tier, as opposed to the client Web application tier. The client credentials must match BSL-authenticated credentials.

How the User Name Gets Passed and Authenticated on Requests

The base client class for Configuration Service will automatically supply the client username credentials selected when you first create the Service Definition: *but only if the client configuration name selected includes 'T_Security' indicating transport security is applied on the binding, see the Configuration Service Technical Guide for details. You would never want to pass username/passwords over unencrypted channels.

The Business Services tier has a special WCF behavior applied to the Primary Service Host, named '`BSL_M_Security_UserName_Behavior`'. This service behavior tells the service to require username credentials on each service request. In turn, the behavior definition, which is contained in the Web.config file, tells WCF to use a custom username validator class to authenticate the usernames on each request. In the behavior definition within Web.config, the configuration tells WCF the name of the assembly, and the class to use within this assembly as follows:

```
<userNameAuthentication userNamePasswordValidationMode="Custom" cacheLogonTokens="true"
customUserNamePasswordValidatorType="Trade.BusinessServiceImplementation.CustomUserNameValidator,
Trade.BusinessServiceImplementation"/>
```

This class is the CustomUserNameValidator class in the **BSLCustomCertValidator.cs** file within the **BusinessServiceImplementation** project (it's used whether running on-premise or on Azure when configured for this mode). If you open this class and look at this class (which also contains the custom certificate validator logic, discussed later), you will find the following class definition:

```
public class CustomUserNameValidator : ConfigService.CustomValidators.CustomUserNameValidator
{
    public override void Validate(string userName, string password)
    {
        base.Validate(userName, password);
    }
}
```

It's very simple, because it inherits from a base class the Configuration Service 5.0 provides. This base class uses the standard .NET framework System.Security assembly to perform the validation on the service-side. The logic provided in the base class by Configuration Service 5 uses the **Users** table within the service's configuration database, looking for a valid matching user that has at least 'Service Operation Rights'. This method is automatically invoked on service requests as they come in from clients. The Configuration Service can perform the validation very quickly, since it maintains an in-

memory cache of its username/passwords from the Users table, and this is always kept synchronized with the database as long as updates flow through the Configuration Service itself (which ConfigWeb always uses). The base class looks like this, and is contained in the \configuration\ CustomValidators project within the Configuration Service Visual Studio solution:

```
/// <summary>
/// Note how this class is tied in via a ServiceBehavior, defined in config, to override default
Windows auth validation.
/// </summary>
public abstract class CustomUserNameValidator : UserNamePasswordValidator
{
    /// <summary>
    /// Overrides to instead validate the username/password against the Configuration DB Users table.
    /// </summary>
    /// <param name="userName">User id coming in as UserName credentials from client.</param>
    /// <param name="password">Password coming in as UserName credentials from client.</param>
    public override void Validate(string userName, string password)
    {
        ServiceConfigHelper configHelper = new
ServiceConfigHelper(ServiceConfigHelper.MasterServiceSelfHost._settingsInstance);
        ServiceUsers csUser = new ServiceUsers();
        csUser.UserId = userName;
        csUser.Password = password;
        //All we need to do is make this call, which validates the user for operation rights.
        if (userName.ToLower().Equals("demoadmin") || !configHelper.setServiceRights(ref csUser,
ConfigUtility.CONFIG_SERVICE_OPERATION_RIGHTS) >= ConfigUtility.CONFIG_SERVICE_OPERATION_RIGHTS)
            throw new SecurityTokenValidationException("The operation failed authentication with the
username/password provided.");
    }
}
```

Note we have a special user '**demoadmin**' setup that lets people browse ConfigWeb for our Azure deployment, but not change any configuration for the deployment. We explicitly disallow this user from using this ID to invoke service operations on the BSL and OPS tiers.

By throwing a security exception if the username/password supplied by the client does not match the required rights for the service, the service operation is automatically rejected by WCF. You could replace this base class implementation (or use the override method in the Business Service implementation) to do your own validation: for example against an Active Directory or other directory service, or against your own users database. Just remember that using the Configuration Service will be fast, since no database or network call needs to take place on validation of each service operation from the client (which, under load, might be coming in at thousands of times per second).

Changing Passwords so others Cannot Access Our Services

Locking down your specific deployment of StockTrader is pretty straightforward. You simply need to use the ConfigWeb Users menu to change all the passwords to the various accounts (users) defined there. Just remember the following rules:

Linked services will break if the root login credentials for ConfigWeb do not match the service you are trying to "Select" in the right hand connected services table in ConfigWeb home page. The link will simply show "Not Selectable." So, for example, if you change the 'localadmin' password to the Web

application, and then login to ConfigWeb when running in remote mode (**AccessMode** set to anything but ‘In-Process Activation’), you will not be able to “Select” and configure the Business Services from that login session. Instead, you would have to login to the Business Services tier directly as a different ConfigWeb login, using its old ‘localadmin’ password, to configure it. The ‘localadmin’ password is the same on install for all StockTrader service domains; so linked services just work; but really each service domain should have its own unique password.

Think of the ‘localadmin’ user akin to an ‘sa’ user on SQL Server. It always has to be there and cannot be deleted, as it’s the master administrator for that service domain. So you might want to make unique passwords for each domain. And if you do, there is another administrator ID setup named ‘**Compositeadmin**’, and this one is set up to use for linked services. If you make each service domain’s ‘localadmin’ password unique, but keep the same password for ‘Compositeadmin’ across service domains, then when you login to ConfigWeb as ‘Compositeadmin’, you will see and be able to select and configure remote services; but when logging in as ‘localadmin’, you will only be able to configure the root service you logged into.

Also, while you cannot ‘Select’ remote connected services in ConfigWeb if the credentials do not match, you should note that every service operation is authenticated by the remote service always, so even if you could ‘select’ it (you could modify the browser URL, for example, to make a request, or do so programmatically as a web service call); you will not get anything back from the remote service, since your request will fail authentication.

So in summary, to lock down your Azure-hosted (or on-premise) install of StockTrader and all its service domains, simply change all the passwords for each user, keeping the following in mind:

- csUser credentials must match. Whenever you see a user with ‘cs’ in the username, this is either a local Connected Service (hence ‘cs’) user for the hosted service, which it doles out to subscribing client to subscribe to notifications; or it’s a non-local user that a client application is using to subscribe to host notifications. So you should make sure that username has the same password everywhere it appears; otherwise, like ADO.NET, if you change the password on the service (akin to changing a login password on SQL Server); then client apps using these credentials will fail authentication because they are using the old password (like an ADO.NET connection string using an old password—the application can no longer connect).
- The Service Operation user credentials also must match, as shown in the previous section, between client and the remote service it is passing these credentials to for authentication.
- For linked services to work, you must create some administrator credentials that are shared between client and remote connected service. You are free to create other admin usernames; but ‘CompositeAdmin’ is setup for this purpose.

Client Certificate Credentials

The StockTrader sample (both for Azure and .NET-on-premise deployments) also has support for using **Client Certificate Credentials** instead of the default Client User Name Credentials discussed above. In this case, we have the configuration service repositories already loaded with the appropriate endpoint definitions, defined using the Hosted Services pages in ConfigWeb. However, the Client Certificate credentials endpoint is marked as non-activated by default, since we are using User Name credentials. In this section we will show how to change the Business Service tier to require a Client (X.509) Certificate as client credentials from the Web application. The processing logic is the same pattern as for the User Name credentials above: a custom certificate validator class in the BSL (and the OPS) is configured via a WCF Service Behavior in web.config (or app.config for the self-hosts). This class inherits from a base class supplied by the Configuration Service. The base class has the .NET System.Security logic to perform certificate authentication via the **System.Security.Cryptography.X509Certificates** .NET Framework assembly.

The Business Service tier can only select one mechanism for authentication at a time over a given network scheme (in this case https). So we will need to **deactivate** the endpoint using User Name credentials first, before we **activate** the defined endpoint using Client Certificate credentials.

- Make sure you are logged into ConfigWeb against the **StockTrader Web** application Azure Hosted Service domain, as in the section above.

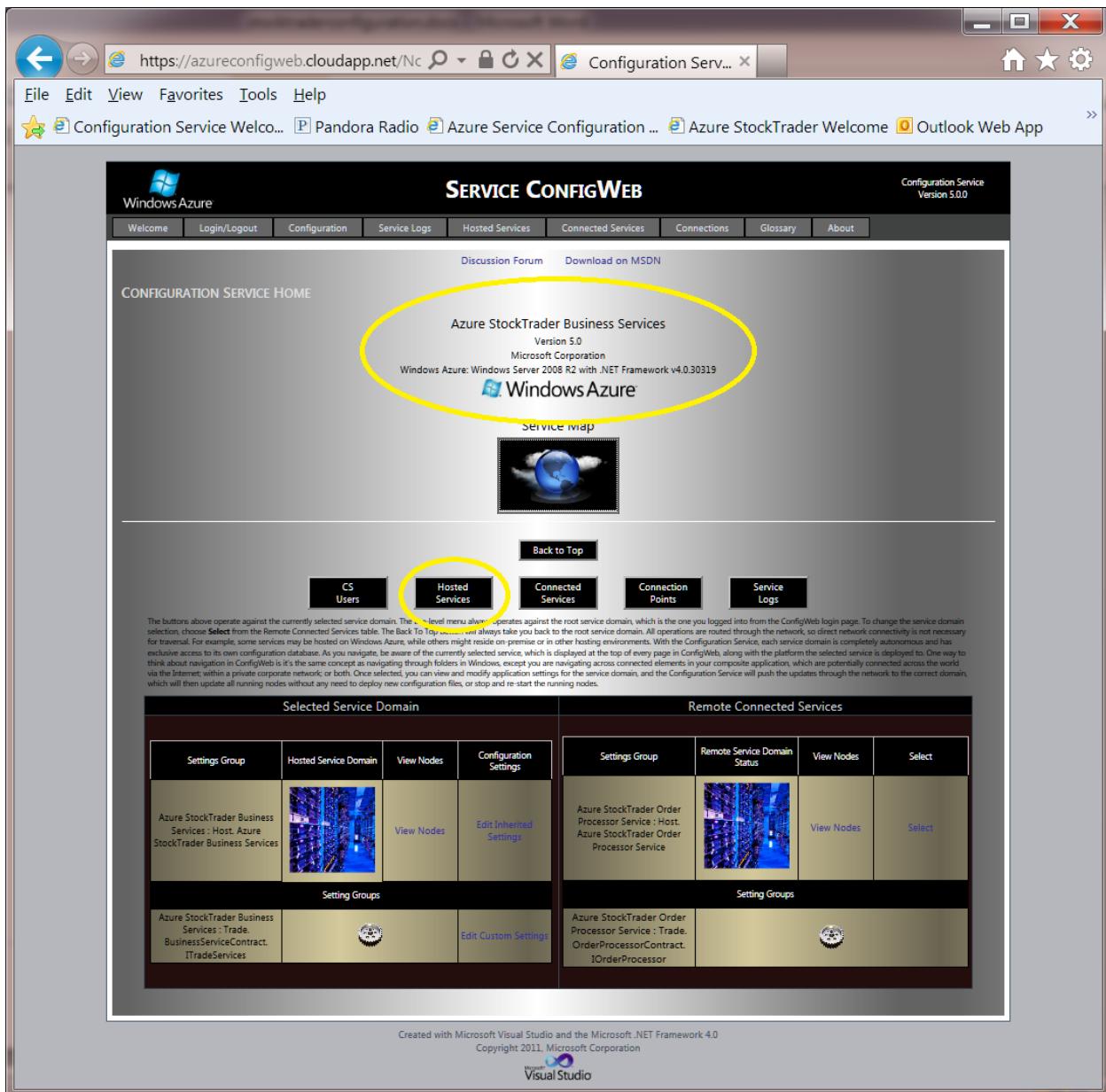
The screenshot shows the Service ConfigWeb interface. At the top, there's a browser header with the URL <https://azureconfigweb.cloudapp.net/Nc>. Below the header, the main menu includes File, Edit, View, Favorites, Tools, Help, and several links like Configuration Service Welcome, Pandora Radio, Azure Service Configuration, Azure StockTrader Welcome, and Outlook Web App.

The main content area is titled "SERVICE CONFIGWEB" and "Windows Azure". It features a "Configuration Service Version 5.0" banner. Below this, there are links for Discussion Forum and Download on MSDN. The central part is the "CONFIGURATION SERVICE HOME" section, which displays the "Azure StockTrader Web Application" details: Version 5.0, Microsoft Corporation, Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319, and a "Service Map" icon.

Below the application details, there are five navigation buttons: CS Users, Hosted Services, Connected Services, Connection Points, and Service Logs. A note explains that these buttons operate against the currently selected service domain. The "Selected Service Domain" table lists the "Azure StockTrader Web Application" with its host information. The "Remote Connected Services" table lists the "Azure StockTrader Business Services" with its host information. In both tables, the "Select" link for the Business Services row is highlighted with a yellow circle.

At the bottom, it says "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation".

- From this home page, click on the **Select** link for the Business Services tier.



- With the Business Services service domain selected, click the **Hosted Services** button.

The screenshot shows the Service ConfigWeb interface for Windows Azure. The main title is "SERVICE CONFIGWEB". The top navigation bar includes links for Configuration Service Welcome, Pandora Radio, Azure Service Configuration, Azure StockTrader Welcome, Outlook Web App, and ESPN. The page header also displays "Configuration Service Version 5.0.0".

The main content area is titled "VIRTUAL SERVICE HOSTS". It displays information about the "Azure StockTrader Business Services" version 5.0, developed by Microsoft Corporation, running on Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319.

Below this, there are two buttons: "Back to Top" and "Add Virtual Host".

The central part of the page shows a table of "Virtual Service Hosts". The table has columns: Service Host ID, Service Host Type, Service Host Name / Implementation Class, Is a Workflow Host, Number of Service Endpoints, Modify Services for this Virtual Host, and Edit This Virtual Host.

Service Host ID	Service Host Type	Service Host Name / Implementation Class	Is a Workflow Host	Number of Service Endpoints	Modify Services for this Virtual Host	Edit This Virtual Host
2	Configuration Service	Trade.BusinessServiceHostConfigurationImplementation.ConfigurationService	No	2	Select	Edit
1	Node Service	ConfigServiceServiceNodeCommunicationImplementation.NodeCommunication	No	1	Select	Edit
3	Primary Service	Trade.BusinessServiceImplementation.TradeServiceBSI	No	4	Select	Edit

At the bottom of the page, there is a "Return to Home Page" link, a copyright notice ("Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0. Copyright 2011, Microsoft Corporation"), and the Microsoft Visual Studio logo.

- Click the Select link for the **Primary Service** as shown above.

Note

Did you know you can create as many Primary Service hosts as you want with Configuration Service for a given service domain? The StockTrader Business Services and Order Processing Service only define one, but in custom implementations/other applications, you can create many, if desired, to host several different business logic implementation classes each with their own endpoints/security, etc. Service hosts can either be based on Web Services with Ws-* bindings, or be RESTful service hosts.

The screenshot shows a Microsoft Internet Explorer window displaying the Azure StockTrader Business Services configuration page. The URL is https://azureconfigweb.cloudapp.net/HSServices.aspx. The page title is "Configuration Service Welco...". The main content area displays the Windows Azure logo and the text "Windows Azure". Below this, there is a "Defined Endpoints for:" section with a "Back to Top" button and an "Add Service Endpoint" button. A note below the table states: "This page shows endpoints defined for a service host. A WCF ServiceHost can host multiple endpoints at once, potentially using different network schemes (http, https, net.tcp, etc.), different security configurations, and the like. Configuration Service lets you add, remove, edit, activate endpoints dynamically, without having to write new code. The options shown below are basically point and click options chosen in the hosted service definition page. Choose Edit to view/modify existing endpoints as defined. Keep in mind that if changing an endpoint, WCF requires that the service host be closed and then re-created with the new description. Configuration Service does this automatically across all active nodes." The main feature is a table listing four service endpoints:

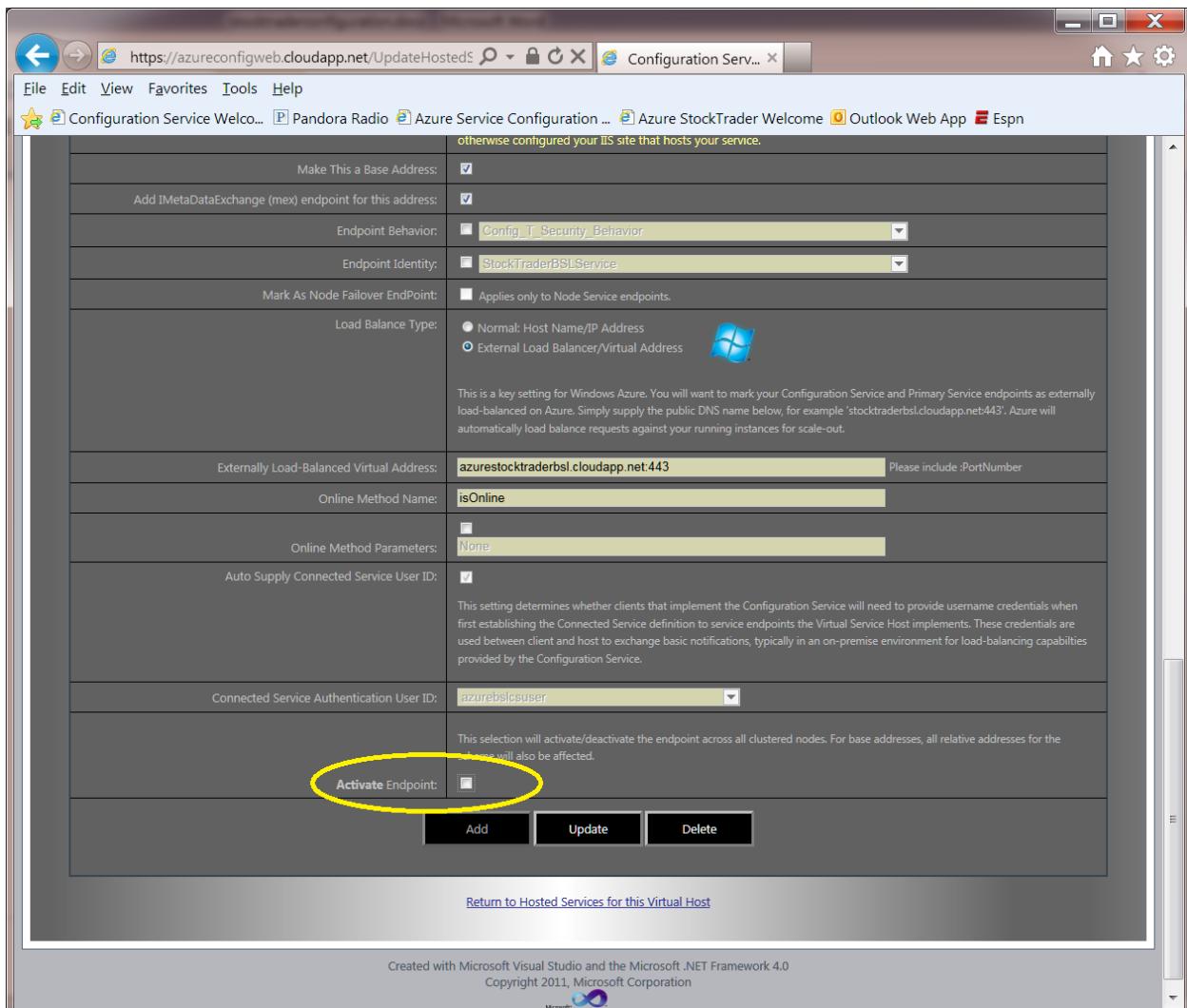
Hosted Service ID	Service Assigned Name For the Endpoint	Service Contract	Virtual Path and Port	Base Address	Binding Configuration	Activated	Edit
5	Azure BSL wsHttp security = TransportWithMessageCredential: UserName	Trade.BusinessServiceContract, ITradeServices	Port: 443 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:443	yes	Host_Ws2007HttpBinding_T_Security_MCredential_UserName Binding Type: ws2007HttpBinding UseHttps: True	True	Edit
6	Azure BSL Basic Http SOAP	Trade.BusinessServiceContract, ITradeServices	Port: 80 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:80	yes	Host_BasicHttpBinding Binding Type: basicHttpBinding UseHttps: False	False	Edit
7	Azure BSL wsHttp security = Message with ClientCredential: ClientCertificate	Trade.BusinessServiceContract, ITradeServices	Port: 80 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:80	yes	Host_Ws2007HttpBinding_M_Security_MCredential_ClientCert Binding Type: ws2007HttpBinding UseHttps: False	False	Edit
4	Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate	Trade.BusinessServiceContract, ITradeServices	Port: 443 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:443	yes	Host_Ws2007HttpBinding_T_Security_MCredential_ClientCert Binding Type: ws2007HttpBinding UseHttps: True	False	Edit

[Return to Virtual Host List](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

Microsoft Visual Studio

- You will notice the only endpoint currently active is the one using User Name client credentials, although several other endpoints are already defined in the configuration repository, they are in-active.
- Click on the **Edit** link for the only activated endpoint (with User Name credentials over https with a ws2007HttpBinding being used).



- We will deactivate this endpoint by de-selecting the **Activate Endpoint** checkbox.
- Click the **Update** button to process the change.
- Click on the **Return to Hosted Services for this Virtual Host** link at the bottom of the page.

Azure StockTrader Business Services
Version 5.0
Microsoft Corporation
Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319
Windows Azure

Defined Endpoints for:
Trade.BusinessServiceImplementation.TradeServiceBSL

Add Service Endpoint

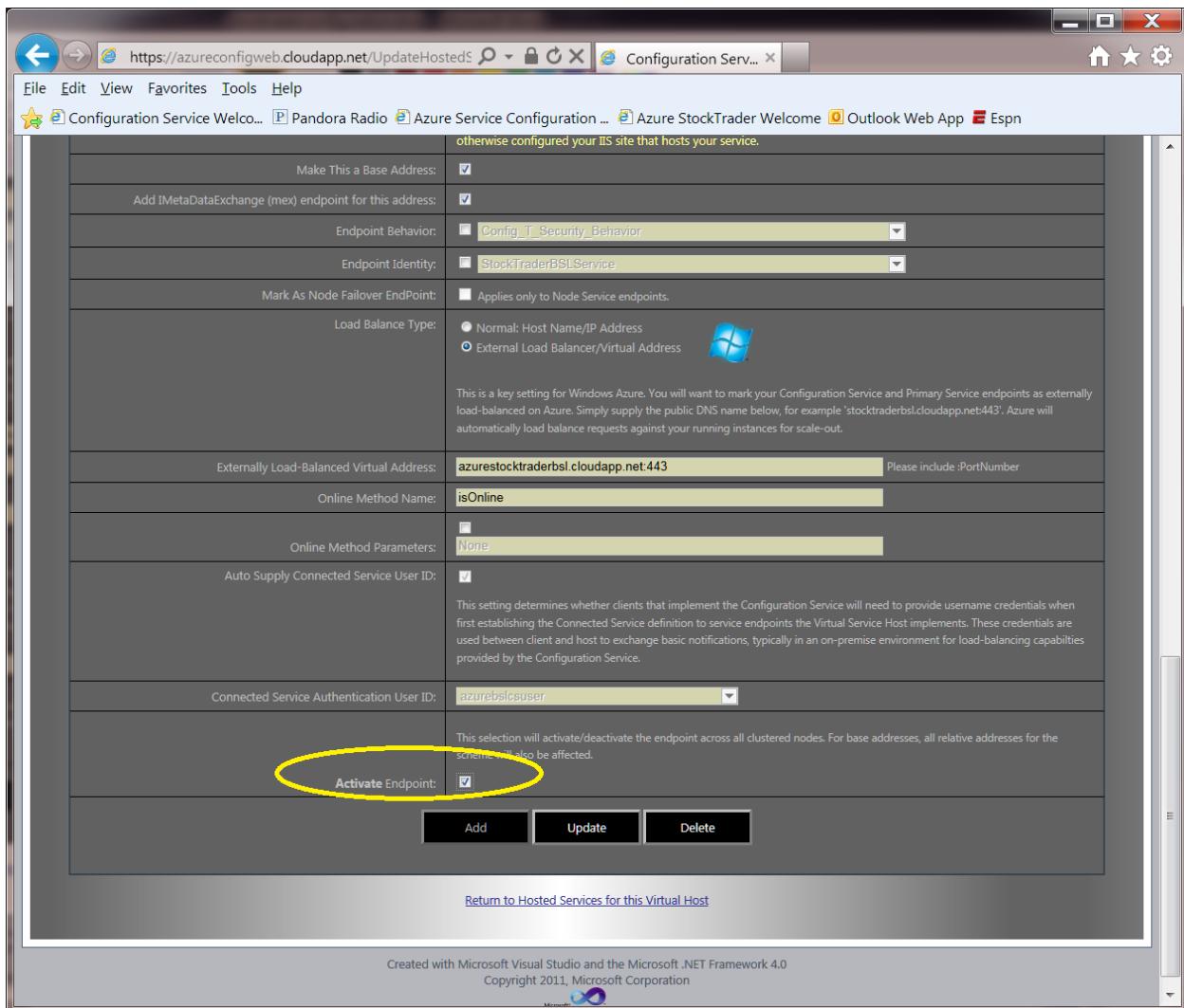
This page shows endpoints defined for a service host. A WCF ServiceHost can host multiple endpoints at once, potentially using different network schemes (http, https, net.tcp, etc.), different security configurations, and the like. Configuration Service lets you add, remove, edit, activate endpoints dynamically, without having to write new code. The options shown below are basically point and click options chosen in the hosted service definition page. Choose Edit to view/modify existing endpoints as defined. Keep in mind that if changing an endpoint, WCF requires that the service host be closed and then re-created with the new description. Configuration Service does this automatically across all active nodes.

Hosted Service ID	Service Assigned Name For the Endpoint	Service Contract	Virtual Path and Port	Base Address	Binding Configuration	Activated	Edit
6	Azure BSL Basic Http SOAP	Trade.BusinessServiceContract. ITradeServices	Port: 80 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:80	yes	Host_BasicHttpBinding Binding Type: basicHttpBinding UseHttps: False	False	Edit
7	Azure BSL wsHttp security = Message with ClientCredential; ClientCertificate	Trade.BusinessServiceContract. ITradeServices	Port: 80 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:80	yes	Host_Ws2007HttpBinding_M_Security_MCredential_ClientCert Binding Type: ws2007HttpBinding UseHttps: False	False	Edit
4	Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate	Trade.BusinessServiceContract. ITradeServices	Port: 443 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:443	yes	Host_Ws2007HttpBinding_T_Security_MCredential_ClientCert Binding Type: ws2007HttpBinding UseHttps: True	False	Edit
5	Azure BSL wsHttp security = TransportWithMessageCredential: UserName	Trade.BusinessServiceContract. ITradeServices	Port: 443 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:443	yes	Host_Ws2007HttpBinding_T_Security_MCredential_UserName Binding Type: ws2007HttpBinding UseHttps: True	False	Edit

[Return to Virtual Host List](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation
Microsoft Visual Studio

- Select the Edit link for the hosted service endpoint named: **Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate**.



- Check the **Activate Endpoint** checkbox. Click the **Update** button.

For the StockTrader Business Services, we are not quite done yet. Right now, the new endpoint is activated, which requires client certificate credentials as we want. However, we need to configure the Service Host hosting this Primary Service endpoint to use the correct Service Behavior, which tells the application a client certificate will now be required, and what validator to use to validate the client certificate supplied on requests.

- Click on the **Return to Hosted Services** link at the bottom of the page above.

Azure StockTrader Business Services
Version 5.0
Microsoft Corporation
Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319

Defined Endpoints for:
Trade.BusinessServiceImplementation.TradeServiceBSL

Add Service Endpoint

This page shows endpoints defined for a service host. A WCF ServiceHost can host multiple endpoints at once, potentially using different network schemes (http, https, net.tcp, etc.), different security configurations, and the like. Configuration Service lets you add, remove, edit, activate endpoints dynamically, without having to write new code. The options shown below are basically point and click options chosen in the hosted service definition page. Choose Edit to view/modify existing endpoints as defined. Keep in mind that if changing an endpoint, WCF requires that the service host be closed and then re-created with the new description. Configuration Service does this automatically across all active nodes.

Hosted Service ID	Service Assigned Name For the Endpoint	Service Contract	Virtual Path and Port	Base Address	Binding Configuration	Activated	Edit
6	Azure BSL Basic Http SOAP	Trade.BusinessServiceContract. ITradeServices	Port: 80 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:80	yes	Host_BasicHttpBinding Binding Type: basicHttpBinding UseHttps: False	False	Edit
7	Azure BSL wsHttp security = Message with ClientCredential; ClientCertificate	Trade.BusinessServiceContract. ITradeServices	Port: 80 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:80	yes	Host_Ws2007HttpBinding_M_Security_MCredential_ClientCert Binding Type: ws2007HttpBinding UseHttps: False	False	Edit
4	Azure BSL wsHttp security = TransportWithMessageCredential; ClientCertificate	Trade.BusinessServiceContract. ITradeServices	Port: 443 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:443	yes	Host_Ws2007HttpBinding_T_Security_MCredential_ClientCert Binding Type: ws2007HttpBinding UseHttps: True	False	Edit
5	Azure BSL wsHttp security = TransportWithMessageCredential; UserName	Trade.BusinessServiceContract. ITradeServices	Port: 443 Path: tradebsl.svc NAT Address: azurestocktradervsl.cloudapp.net:443	yes	Host_Ws2007HttpBinding_T_Security_MCredential_UserName Binding Type: ws2007HttpBinding UseHttps: True	False	Edit

[Return to Virtual Host List](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

Microsoft Visual Studio

- Now click on the **Return to Virtual Host list** link as shown above.

SERVICE CONFIGWEB

Windows Azure Configuration Service Version 5.0.0

VIRTUAL SERVICE HOSTS

Azure StockTrader Business Services
Version 5.0
Microsoft Corporation
Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319

Service Host ID	Service Host Type	Service Host Name / Implementation Class	Is a Workflow Host	Number of Service Endpoints	Modify Services for this Virtual Host	Edit This Virtual Host
2	Configuration Service	Trade.BusinessServiceHostConfigurationImplementation.ConfigurationService	No	2	Select	Edit
1	Node Service	ConfigService.ServiceNodeCommunicationImplementation.NodeCommunication	No	1	Select	Edit
3	Primary Service	Trade.BusinessServiceImplementation.TradeServiceBSL	No	4	Select	Edit

[Return to Home Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

Microsoft Visual Studio

- Now click on the Edit link as shown, for the Primary Service Host itself.

Azure StockTrader Business Services
Version 5.0
Microsoft Corporation
Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319

Instructions

You may create as many **Primary** Virtual Hosts as you like, these host your custom business services, for which you supply the implementation logic. The Configuration Service will manage their lifecycle automatically. Every service implementing Configuration Service will always have one **Configuration** Service host, and one **Node** Service host, which are created automatically. Configuration Service endpoints are external, and should be secured with transport security. All service operations are authenticated. The Node Service will typically be configured with private endpoints, such as internal endpoints on Windows Azure. The Node Service automatically keeps configurations synchronized between running nodes. You are free to modify WCF properties (behaviors, endpoints, bindings) for Configuration Service and Node hosts.

Enter Virtual Host Details

Service Type:	<input checked="" type="radio"/> Primary Business Service <input type="radio"/> Node Custom Cache Service
Service Implementation Class:	TradeBusinessServiceImplementation.TradeServiceBSL
Service Behavior Configuration:	BSL_M_Security_ClientCert_Behavior
Is WorkflowServiceHost:	<input type="checkbox"/> This is a WorkflowServiceHost
Service Host Environment:	<input checked="" type="radio"/> Internet Information Services (IIS) or Azure Web Role Service <input type="radio"/> .NET Windows Service (NT Service) or Azure Worker Role Service <input type="radio"/> .NET Windows Application <input type="radio"/> .NET Console Application
Azure Http Role Endpoint Name or IIS Http Host Header Name:	 HttpsEndpoint
Azure Tcp Role Endpoint Name or IIS Tcp Host Header Name:	 Host Headers Not Used

- Now, in the second dropdown (Service Behavior Configuration), select the **BSL_M_Security_ClientCert_Behavior**, as shown above.
- Click the **Update** button.

How the Client Certificate Gets Validated, and What Certificate is Valid as a Client Credential for the StockTrader Business Services?

Like the User Name validator discussed in the previous sections, Configuration Service has a base class that is a Custom Certificate Validator. This is tied to the ServiceBehavior in the Web.config for the StockTraderBSL Azure project (and also the on-premise project): it's in the \configuration\CustomValidators project within the Configuration Service Visual Studio Solution.

This base class is configured to validate based on several mechanisms, including the thumbprint of the certificate supplied by the client. We override a method in the StockTrader Business Service

implementation that inherits from the base class to provide the list of valid certificate thumbprints we will allow to use our BSL service operations. The class that does this is in the BSLCustomCertValidator.cs file in the BusinessServiceImplementation project. The code looks as follows:

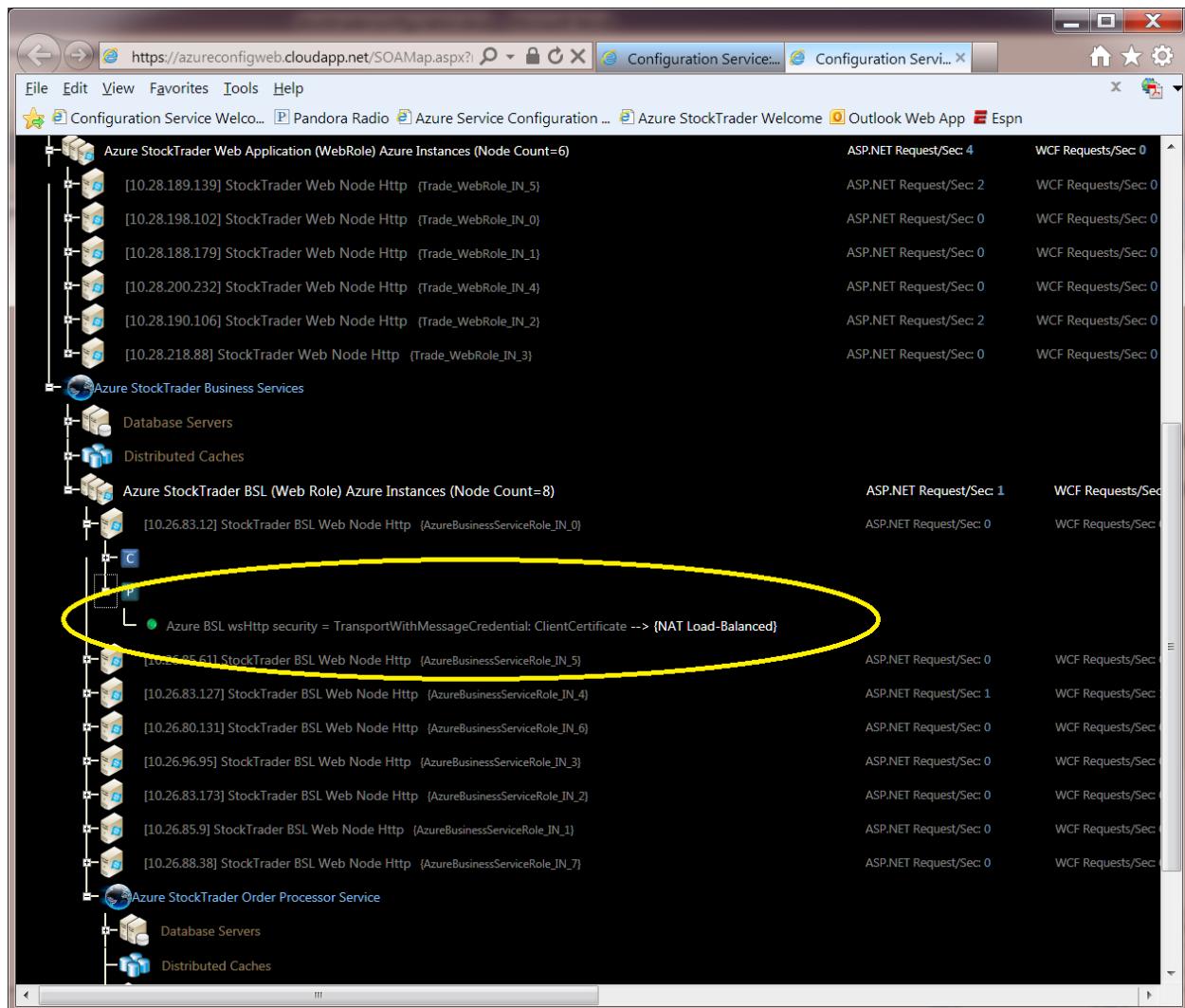
```
/// <summary>
/// The Business Services uses this custom X.509 certificate validator, that uses the base class
/// provided with Configuration Service. This class is referenced in the config file, with the
/// BSL_M_Security_Behavior behavior configuration for the host exe.
/// </summary>
public class CustomCertValidator : CustomCertificateValidator
{
    /// <summary>
    /// Override to provide our list of valid cert thumbprints for the service.
    /// </summary>
    /// <returns></returns>
    protected override string[] getAllowedThumbprints()
    {
        List<string> thumbprints = new List<string>();

        //This is the thumbprint for the Test Certificate CN=StockTraderBSLClient.Com
        // (StockTraderBSLClient.pfx).
        //This Cert is used by the Business Service Layer to validate the StockTrader Client. When
        using this binding in
        //conjunction with the service behavior 'BSL_M_Security_ClientCert_Behavior' (which also
        assigns the service certificate),
        //only clients presenting the EXACT client certificate will be allowed in. The thumbprint is
        unique, meaning,
        //no one can create another cert with this thumbprint. Hence, only an app with this exact
        client-side
        //cert can ever access the BSL when using this WCF message security behavior.
        thumbprints.Add("EB0C8C302C4F5E22E4492006F1D16D01008E7826");

        return thumbprints.ToArray();
    }
}
```

The thumbprint is for the client certificate for the BSL that we already uploaded to the **StockTrader Web application** when installing Azure StockTrader. It's the **StockTraderBSLClient.Com** certificate, contained in the \certs\ StockTraderBSLClient.pfx certificate file.

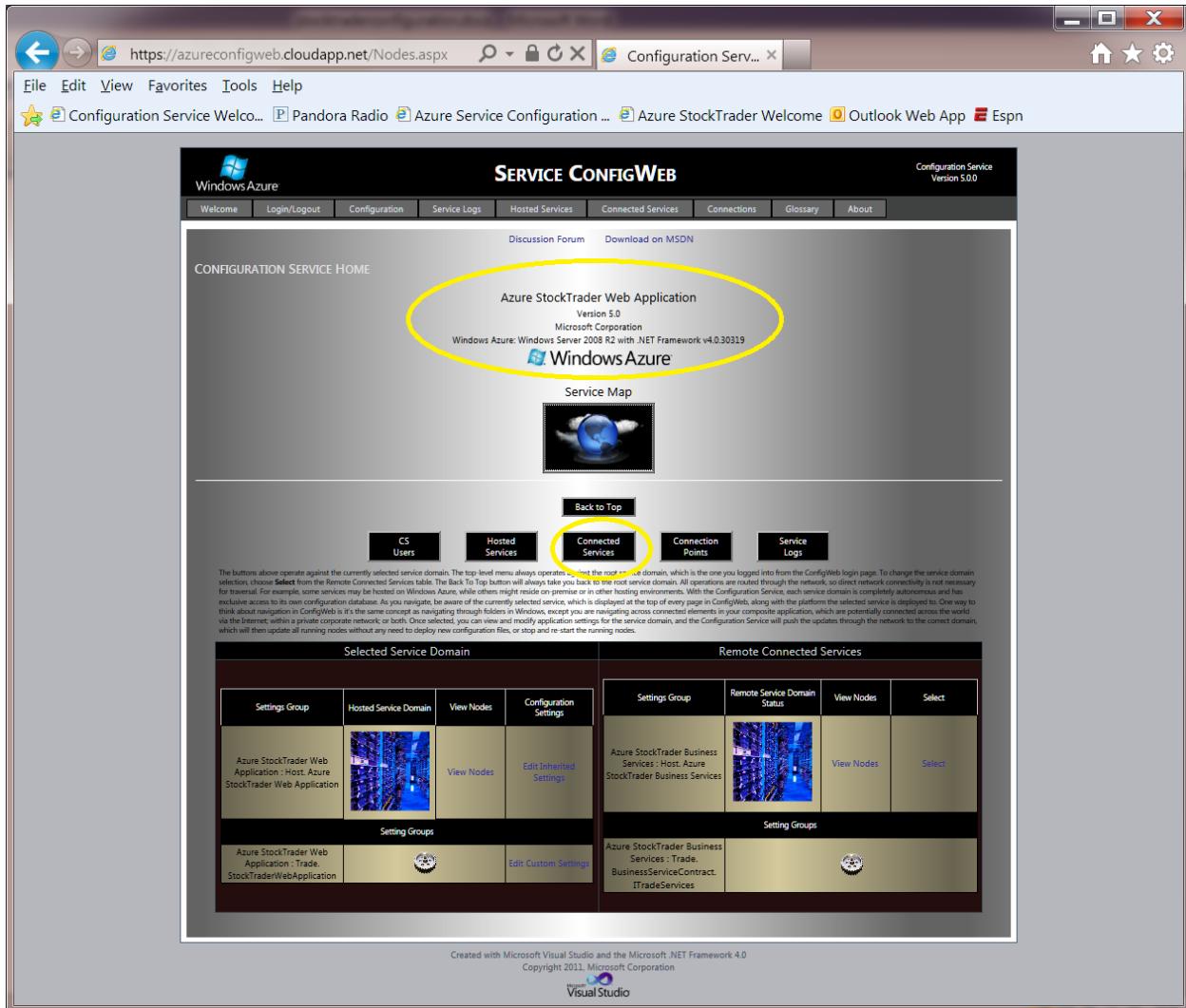
If you go back to the home page of ConfigWeb, and click the Service Map button, you will see the new endpoint is now active and online (green), and the old one using User Name is not active anymore.



Note that the Hosted Service endpoint for the BSL is already configured with an internal client configuration that allows the BSL nodes to check their own endpoints, in this case using the correct client certificate which we uploaded to the Business Service domain via the Azure Management portal when we first created the hosted service during StockTrader setup. The Service Map page works by requesting remote services to validate their own endpoints for each node (in this case Azure role instance) running. BUT: the StockTrader Web application is not yet configured to use a WCF endpoint requiring a client certificate. So we next need to create a new Connected Service Definition for the **StockTrader Web** application that will use this service endpoint, instead of the User Name authenticated endpoint as installed by default.

Changing the StockTrader Web Application to Supply a Client Certificate to the StockTrader Business Services.

- Click on the Configuration menu top-level menu item to return to the root StockTrader Web application home page for ConfigWeb.



- Click on the **Connected Services** button.

The screenshot shows a web browser window with the URL <https://azureconfigweb.cloudapp.net/CServices.asp>. The page title is "SERVICE CONFIGWEB". The top navigation bar includes links for Configuration Service Version 5.0.0, Home, Favorites, Tools, Help, Configuration Service Welcome, Pandora Radio, Azure Service Configuration, Azure StockTrader Welcome, Outlook Web App, and ESPN.

The main content area displays the "Azure StockTrader Web Application" configuration. It shows the service version as "Version 5.0" and the provider as "Microsoft Corporation". Below this, it states "Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319". A "Windows Azure" logo is present.

A "Connected Services" section lists two entries:

Virtual Host ID	Service Name	Service Type	Service Contract	Client Configuration	Edit/ Delete
1	Azure BSL wsHttp security = TransportWithMessageCredential: UserName	Primary Connected Service	Trade.BusinessServiceContract. ITradeServices	Client_Ws2007HttpBinding_T_Security_MCredential_UserName_BSL Binding Type: ws2007HttpBinding SecurityMode: TransportWithMessageCredential	Edit
3	On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate	Primary Connected Service	Trade.BusinessServiceContract. ITradeServices	Client_Ws2007HttpBinding_T_Security_MCredential_ClientCert_BSL Binding Type: ws2007HttpBinding SecurityMode: TransportWithMessageCredential	Edit

Below the table is a "Return to Home Page" link. At the bottom of the page, there is a note about creation: "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0. Copyright 2011, Microsoft Corporation" and a "Visual Studio" logo.

- Click on the **Add Service** button.

Instructions

This page enables you to establish connections to remote services your application will utilize. A primary service implements the Configuration Service; a generic service does not. For example, a generic service might be one written in Java, PHP or .NET. When establishing connections to **primary services**, you will simply connect to the configuration service of the remote host, and a list of available services (contracts) will be returned. You can also establish connection to **generic services** simply by providing the endpoint to the service, and filling out the requested details. A service implementing the Configuration Service can always be treated as a generic service, since the Configuration Service imposes no requirement on the language clients are developed in, or whether they also implement the Configuration Service.

Connected Service Type

Primary Connected Service
 Generic Connected Service

Remote Configuration Service Details - Get Connected!

Address to Configuration Service:	Once connected, the remote host will return the available services your client application can subscribe to. You can connect over http, https, or tcp to the remote Configuration Service to retrieve this list. Make sure to enter the address to the configuration service endpoint. If the host does not implement the configuration service, choose the Generic Service type above. https://[azure stocktrader bsl DNS address].cloudapp.net/config.svc
Client Configuration Name:	Please make sure to select a valid Configuration Service Client. Service configurations displayed below are always prefixed with "client_configs" in the config file 'client' section to appear in this list. You can use prcutil.exe to generate new client definitions to remote configuration services if these services expose their metadata. Client_ConfigSvc_BasicHttpBinding_T_Security
User Id:	azurebslcsuser (see information below left)
Password:	*****

The credentials above are used by the Configuration Service to send/receive notifications between hosts and subscribed clients. Hosts can be configured to automatically return these credentials for any client attempting to subscribe, in which case nothing needs to be entered before connecting, the credentials will be returned. If the remote host is not configured to automatically provide these credentials, you will need to enter valid credentials to the remote host above, with the userid supplied having at least 'Connected Service Rights' as defined by the remote host in its users database.

Connected!

Please Select the Primary Service

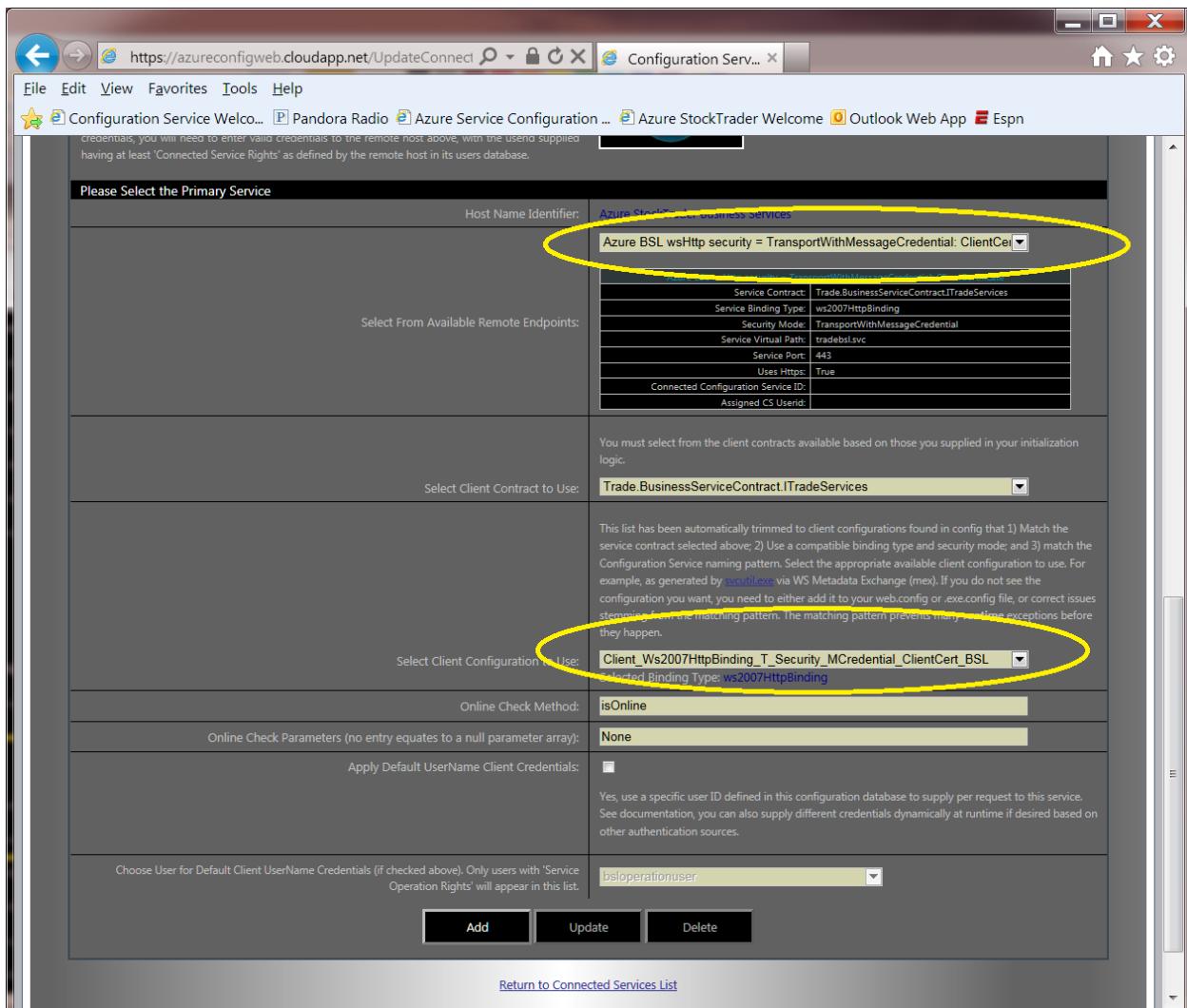
Host Name Identifier:	Azure StockTrader Business Services
-----------------------	-------------------------------------

These are the available service endpoints by friendly names as assigned by the Service Hoster. Each has

Get Remote Services!

We will now connect up to the BSL's Configuration service to get the new endpoint information and create our new Connected Service definition.

- Type in the address to your Azure hosted service domain for the Business Services project you deployed.
- Choose the **Client_ConfigSvc_BasicHttpBinding_T_Security** client configuration, and click on the **Get Remote Services!** button.



- The BSL has only one active endpoint, but you need to make sure to select the correct client configuration: **Client_Ws2007Binding_T_Security_MCredential_ClientCert_BSL**.
- Click the **Add** button.
- Next, click on the **Connections** top-level menu item.

The screenshot shows the Configuration Service web interface at <https://azureconfigweb.cloudapp.net/ConnectionPoint>. The main page displays the 'Azure StockTrader Web Application' version 5.0 from Microsoft Corporation, running on Windows Azure. Below the title, there's a 'Back to Top' button and two links: 'View Services' and 'View Clients'. A large section titled 'Add Connection' contains a detailed description of WCF connections between client and service domains. Below this, a table lists 'Connections To Host: Azure StockTrader Business Services'. The table has columns for 'Remote Address', 'Client Details', 'Service Status', and 'Remove'. A row for 'Azure BSL wsHttp security' is selected, showing its configuration details: Client Configuration (Client_Ws2007HttpBinding_T_Security_MCredential_UserName_BSL), Service Contract (Trade.BusinessServiceContractITradeServices), Binding Configuration (Client_Ws2007HttpBinding_T_Security_MCredential_UserName), Binding Type (ws2007HttpBinding), Security Mode (TransportWithMessageCredential), and Endpoint Behavior (USERNAME_CLIENT_CREDENTIAL_BEHAVIOR_BSL). The 'Delete' link in the 'Remove' column for this row is circled in yellow.

Remote Address	Client Details	Service Status	Remove
https://azur stocktraderbsl.cloudapp.net:443/tradebsl.svc	Azure BSL wsHttp security = TransportWithMessageCredential: UserName Client Configuration: Client_Ws2007HttpBinding_T_Security_MCredential_UserName_BSL Service Contract: Trade.BusinessServiceContractITradeServices Binding Configuration: Client_Ws2007HttpBinding_T_Security_MCredential_UserName Binding Type: ws2007HttpBinding Security Mode: TransportWithMessageCredential Endpoint Behavior: USERNAME_CLIENT_CREDENTIAL_BEHAVIOR_BSL		Delete

You will notice we still have a Connection Point defined to the old, inactive User Name endpoint. Since the BSL is not hosting this anymore (its deactivated), it shows as red from the client connections page.

We can delete this non-active connection (it does not hurt anything to leave it be, since we will not be using it, but to keep things clean, we will get rid of it).

- Click on the Delete link, and go through the next page to delete the old connection point.

The screenshot shows a web browser window with the URL <https://azureconfigweb.cloudapp.net/UpdateConnect>. The page title is "Configuration Serv...". The main content area is titled "SERVICE CONFIGWEB" and features a "Windows Azure" logo. A navigation menu at the top includes "Welcome", "Login/Logout", "Configuration", "Service Logs", "Hosted Services", "Connected Services", "Connections", "Glossary", and "About". The "Connections" tab is active. Below the menu, there are links to "Discussion Forum" and "Download on MSDN". The main content area is titled "DELETE CONNECTION POINT" and displays information about the "Azure StockTrader Web Application" (Version 5.0, Microsoft Corporation). It also mentions "Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319" and the "Windows Azure" logo. A "Back to Top" button is visible. A table lists a single connection point:

Remote Service Host	Remote Address	Action
Azure StockTrader Business Services	https://azur stocktraderbsl.cloudapp.net:443/tradebsl.svc	Delete

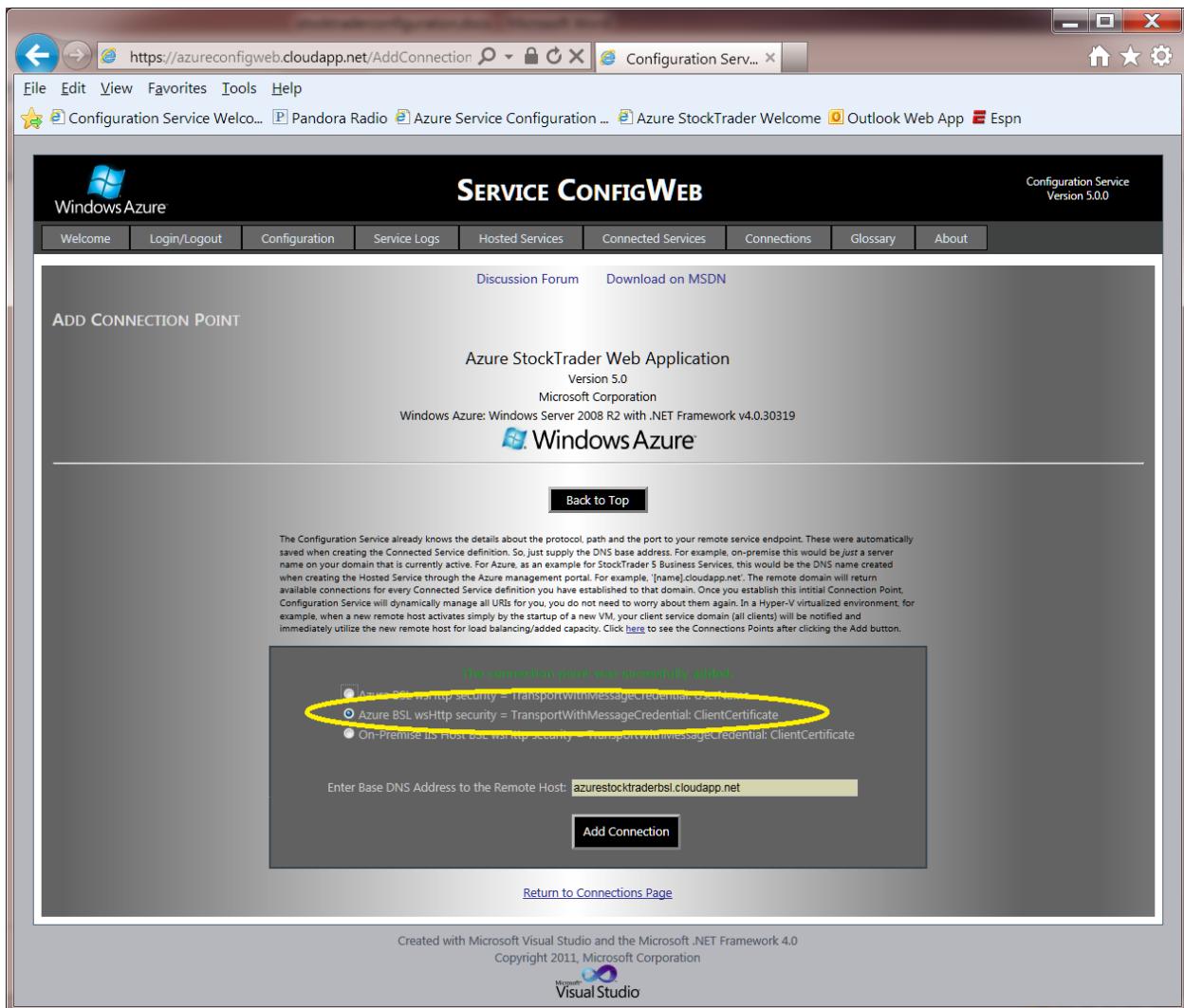
A "Delete" button is located below the table. At the bottom of the page, there is a link "Return to Connection Page".

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation
Microsoft Visual Studio

- Click on the Delete button.
- Click on the Return to Connection Page link at the bottom of the page.

The screenshot shows a web browser window with the URL <https://azureconfigweb.cloudapp.net/ConnectionPoint>. The title bar reads "Configuration Serv...". The menu bar includes File, Edit, View, Favorites, Tools, Help, and several links like Configuration Service Welcome, Pandora Radio, Azure Service Configuration, Azure StockTrader Welcome, Outlook Web App, and ESPN. The main content area has a header "Windows Azure" and "SERVICE CONFIGWEB". A sub-header "Configuration Service Version 5.0.0" is on the right. Below the header is a navigation menu with tabs: Welcome, Login/Logout, Configuration, Service Logs, Hosted Services, Connected Services, Connections, Glossary, and About. Under "Welcome", there are links to "Discussion Forum" and "Download on MSDN". The main content area is titled "SERVICE CONNECTION POINTS" and displays information about the "Azure StockTrader Web Application". It shows the application version as "Version 5.0", the provider as "Microsoft Corporation", and the runtime environment as "Windows Azure: Windows Server 2008 R2 with .NET Framework v4.0.30319". A "Windows Azure" logo is present. At the bottom of this section are three buttons: "Back to Top", "View Services", and "View Clients". A prominent "Add Connection" button is located below this section. A note explains the nature of connection points behind a NAT load balancer. A message states that the service currently has no defined connection points to other services. At the bottom, it says "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation" with the Visual Studio logo.

- Click on the **Add Connection** button.



- Select the newly created Connected Service Definition: **Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate**
- Enter the base DNS address to **YOUR** Azure StockTrader Business Services Azure Hosted Service.
- Click the **Add Connection** button.
- Click the **Return to Connections** link.

The screenshot shows the Service ConfigWeb interface for the Azure StockTrader Business Services. The main page title is "SERVICE CONFIGWEB" and it indicates "Configuration Service Version 5.0". The "Service Connection Points" section shows a single entry for the "Azure StockTrader Web Application" with "Version 5.0" and "Microsoft Corporation". The "Windows Azure" logo is present. Below this, there are buttons for "Back to Top", "View Services", and "View Clients", along with an "Add Connection" button. A note at the bottom explains the visibility of connection points behind a NAT. The "Connections To Host: Azure StockTrader Business Services" section lists a single connection point:

Remote Address	Client Details	Service Status	Remove
https://azur stocktraderbsl.cloudapp.net:443/tradebsl.svc	Azure BSL wsHttp security = TransportWithMessageCredential; ClientCertificate Client Configuration: Client_Ws2007HttpBinding_T_Security_1 Credential_ClientCert_BSL Service Contract: TradeBusinessServiceContractTradeServices Binding Configuration: Client_Ws2007HttpBinding_T_Security_1 Credential_ClientCert Binding Type: ws2007HttpBinding Security Mode: TransportWithMessageCredential Endpoint Behavior: CERTIFICATE_CLIENT_CREDENTIAL_BEHAVIOR_BSL		Delete

At the bottom, there are links for "View/Configure This Service Host's Connection Points" and "Purge All Connections To This Service Host". The footer notes "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation".

- We can see from this page that the StockTrader Web application is seeing the endpoint as online, and hence the client certificate is working correctly for authentication and being passed by the client to the service. The reference to the client certificate StockTraderBSLService.Com has been already configured in the StockTrader Web's client configuration we selected previously, and this client configuration is contained in the StockTrader web application's Web.config (so it knows where to find the client certificate to pass to the WCF BSL service).
- However, the Web application is still configured to use the other hosted service, via its AccessMode setting. Notice the title text for the connection is not green above (its black text), meaning the client connection exists, but it's not configured to be used by the Web app via the AccessMode setting. So the last thing we need to do is change the AccessMode setting so the Web application uses this endpoint/connection point(s).

- Click on the Back to Top button to get back to the home page for ConfigWeb, with the Web application selected.

The screenshot shows the Service ConfigWeb interface at <https://azureconfigweb.cloudapp.net/Nodes.aspx>. The top navigation bar includes File, Edit, View, Favorites, Tools, Help, and a search bar. Below the navigation is a toolbar with links to Configuration Service Welcome, Pandora Radio, Azure Service Configuration, Azure StockTrader Welcome, Outlook Web App, and ESPN. The main content area is titled "SERVICE CONFIGWEB" and "Configuration Service Version 5.0". It displays the "Azure StockTrader Web Application" settings, version 5.0, Microsoft Corporation, Windows Azure, and a Service Map. A "Back to Top" button is visible above a row of navigation buttons: CS Users, Hosted Services, Connected Services, Connection Points, and Service Logs. A note explains the function of these buttons against the currently selected service domain. The "Selected Service Domain" section shows a "Setting Groups" table with rows for "Azure StockTrader Web Application : Host: Azure StockTrader Web Application" and "Azure StockTrader Web Application : Trade: StockTraderWebApplication". The "Edit Custom Settings" link in the second row is highlighted with a yellow circle. The "Remote Connected Services" section shows a similar table for "Azure StockTrader Business Services : Host: Azure StockTrader Business Services" and "Azure StockTrader Business Services : Trade: BusinessServiceContract, ITradeServices". The bottom of the page includes a note about creation with Microsoft Visual Studio and the Microsoft .NET Framework 4.0, Copyright 2011 Microsoft Corporation, and a Visual Studio logo.

- Click on the Edit Custom Setting link for the Web application.

The screenshot shows the Service ConfigWeb interface for the Azure StockTrader Web Application. The page title is "SERVICE CONFIGWEB" and the sub-page title is "Azure StockTrader Web Application". The "AccessMode" setting is highlighted with a yellow circle around its "Change Value" link.

Setting Name	Current Value	Description	Change Value
AccessMode	Azure BSL viaHttp security = TransportWithMessageCredential; Username	This setting determines how the Trade Web application makes calls to the Business Service Layer (BSL) over WCF. As a sample application, there are a wide variety of settings. First, there are three distinct SSL layers hosting service endpoints: 1) Windows Azure running in a Web Role; 2) An on-premise BSL-based service; 3) An on-premise EXE application that ships with the sample. For each endpoint, there are two types of transport: 1) A secure connection that the Web application will only use one service endpoint at a time so there is a switch that can for example be changed to have the Web application port communicating to our on-premise BSL vs. the Azure-hosted BSL with no update from ConfigWeb.	Change Value
Always Check Order Alerts	false	Whether to run the closed orders check on every visit to an authenticated page to display the Order Alert message, or to use the check frequency setting value.	Change Value
Check Order Alert Frequency	15	Time in seconds to check for closed orders: only used if CheckOrderAlertsOnlyOverRequest is false	Change Value
DisplayOrdersDefault	10	This setting determines the default number of orders to display in the Account.aspx page.	Change Value
DisplayOrdersMax	100	This setting determines the maximum number of orders to display in the Account.aspx page.	Change Value

- Click on **Change Value** for the AccessMode setting.

You can change a setting's value here, and all running nodes will automatically receive the new setting and be updated. The setting update will be pushed through the network to the correct service domain automatically. Once updated, new nodes for that domain will always start up with the updated value since it has been persisted into the configuration repository, which nodes read on startup. Running nodes are updated live via the node service and reflection. Base-level validation is provided on integral types, but developers can easily supply their own server-side validation logic for custom settings, and perform custom actions across all nodes when specific settings change, if they desire. For StockTrader 5, a key custom setting for the Web application is AccessMode, which determines which Business Service domain to utilize, and over which network/encoding/security standard. The OrderMode setting for the StockTrader Business Service domain serves a similar purpose, specifying which remote Order Processor Service domain to use, and via what type of communication channel.

Setting Name	Current Value	Setting Description	Valid Values	Last Updated
AccessMode	Azure BSL wsHttp security = TransportWithMessageCredential: UserName	This setting determines how the Trade Web application makes calls to the Business Service Layer (BSL) over WCF. As a sample application, there are a wide variety of settings. First, there are three distinct BSL layers hosting service endpoints: 1) Windows Azure running in a Web Role; 2) An on-premise IIS-based service; 3) An on-premise .NET application that ships with the sample. For each, there are a variety of transports and security modes illustrated. The Web application will only use one service endpoint at a time; so this setting is a switch that can for example be changed to have the Web application start communicating to our on-premise BSL vs. the Azure-hosted BSL with a simple update from ConfigWeb.	In-Process Activation; Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate; Azure BSL wsHttp security = TransportWithMessageCredential: UserName; On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: UserName; On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate	5/15/2011 3:08:10 PM

Enter/Select New Value:

In-Process Activation
 Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate
 Azure BSL wsHttp security = TransportWithMessageCredential: UserName
 On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: UserName
 On-Premise IIS Host BSL wsHttp security = TransportWithMessageCredential: ClientCertificate

Update

[Return to Configuration Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0

- Select the '**Azure BSL wsHttp security = TransportWithMessageCredential: ClientCertificate**' option.
- Click the **Update** button.

OK! At this point, we have completely changed both the service and client, so now the StockTrader Web Application is making all service requests to the Business Services via an endpoint requiring a client-side X.509 certificate. User Name credentials are no longer being used. You can now login as an end-user to the Azure StockTrader web application. End users will not notice a difference, but you can verify the application is working in its new configuration.

Changing the Samples to Use Different Certificates vs. Default Install Certificates

Now that we have seen how the X.509 certificates are used, the next logical question is how to change them from the ones provided in the default install? After all, everyone using the default install will have the same certificates, so anyone could access our services when running in Client Certificate mode as set above.

To change certificates, you need:

- To create or obtain your own certificate. There is a batch file for creating .pfx self-signed certificates installed in the \certs folder with the StockTrader certificates for this purpose.
- You need to upload these certificates to the correct Azure Hosted Service domain(s).
- You need to modify the correct service and client behaviors in configuration to use these certificates as opposed to the default certificates. Simply do a global search for 'StockTraderBSLService.Com', for example, to find its uses in Config; but remember that 'StockTraderBSLClient.Com' will be found in both the Business Services .config files (app.configs and web.config); and the StockTrader Web app's web.config.
- You need to look at the method 'getAllowedThumbprints' (see page 96) on the custom certificate validator and change this method to use your certificate(s) thumbprints.

These rules apply to both the StockTrader Business Services, and the Order Processor, since both can be set to use client certificates vs. Username client credentials.

Configuring the MSMQ Durable Message Queue Endpoint for Order Processing On-Premise StockTrader

The following sections will step through an additional StockTrader configuration change using the Hosted Services pages in ConfigWeb. This topic relates only to on-premise .NET StockTrader, which uses MSMQ. For Windows Azure, the new Azure AppFabric Service Bus is bringing a cloud-optimized durable queuing mechanism to the party, but it is only in CTP right now (as of June 9, 2011, it will be released commercially later but you can sign up for the CTP if you want).

So we will use on-premise StockTrader in this walkthrough, not Azure StockTrader.

Activating and Using the Order Processor Transacted MSMQ Endpoint

By default, the OPS MSMQ endpoint is marked as de-activated in the OPS Repository. We can activate this MSMQ endpoint, which operates against a durable, transacted MSMQ in conjunction with a WCF service endpoint. The following section assumes you have StockTrader configured (or Azure StockTrader) for remote modes with OrderMode set to the net.tcp or http endpoint, and valid connections established to the OPS as covered in previous steps.

- Launch the Order Processor Windows Service Host from the StockTrader Start menu, if not already running.
- Login to the StockTrader Web application's Configuration Service via ConfigWeb.

The screenshot shows the Configuration Service Home page in a Windows Internet Explorer browser. The title bar reads "Configuration Service: Home - Windows Internet Explorer" and the address bar shows "http://localhost/ConfigWeb/Nodes.aspx". The main content area is titled "SERVICE CONFIGWEB" and includes a "Discussion Forum" and "Download on MSDN" link. Below this is the "CONFIGURATION SERVICE HOME" section, which displays the ".NET StockTrader Web Application" version 5.0, Microsoft Corporation, and Windows Server 2008 R2 with .NET Framework v4.0.30319. A "Service Map" section features a globe icon. At the bottom of this section are buttons for "CS Users", "Hosted Services", "Connected Services", "Connection Points", and "Service Logs". A note explains the functionality of these buttons. The main content area also contains a table titled "Selected Service Domain" and "Remote Connected Services". The "Selected Service Domain" table has four columns: "Settings Group", "Hosted Service Domain", "View Nodes", and "Configuration Settings". It lists ".NET StockTrader Web Application : Host .NET StockTrader Web Application" and ".NET StockTrader Web Application : Trade .StockTraderWebApplication". The "Remote Connected Services" table has four columns: "Settings Group", "Remote Service Domain Status", "View Nodes", and "Select". It lists ".NET StockTrader Business Services IIS Host : Host .NET StockTrader Business Services IIS Host" and ".NET StockTrader Business Services IIS Host : Trade .BusinessServiceContract .ITradeServices". The "Select" button in the second row of the "Remote Connected Services" table is highlighted with a yellow circle.

- Choose **Select** to navigate to the Business Service IIS Host configuration as the selected service domain. (note: you may be connected to the BSL self-host; in which case you should add connections for the Web app to the IIS Host service host, and set the StockTrader Web AccessMode setting to either the http or net.tcp BSL IIS Host endpoint setting).

.NET StockTrader Business Services IIS Host
Version 5.0
Microsoft Corporation
Windows Server 2008 R2 with .NET Framework v4.0.30319
Windows Server 2008 R2

Service Map

Selected Service Domain

Setting Group	Hosted Service Domain	View Nodes	Configuration Settings
.NET StockTrader Business Services IIS Host : Host .NET StockTrader Business Services IIS Host		View Nodes	Edit Inherited Settings
Setting Groups			
.NET StockTrader Business Services IIS Host : Trade. BusinessServiceContract. ITradeServices		Edit Custom Settings	

Remote Connected Services

Setting Group	Remote Service Domain Status	View Nodes	Select
.NET StockTrader Order Processor Service : Host .NET StockTrader Order Processor Service		View Nodes	Select
Setting Groups			
.NET StockTrader Order Processor Service : Trade. OrderProcessorContract. IOrderProcessor			

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

- Choose **Select** again to select the Order Processor Service as the selected service domain.

The screenshot shows the Configuration Service Home page in Microsoft Internet Explorer. At the top, the title bar reads "Configuration Service: Home - Windows Internet Explorer" and the address bar shows "http://localhost/ConfigWeb/Nodes.aspx?name=.NET StockTrader Order Pro". The main content area is titled "SERVICE CONFIGWEB" and "CONFIGURATION SERVICE HOME". It displays information about the ".NET StockTrader Order Processor Service" (Version 5.0, Microsoft Corporation, Windows Server 2008 R2 with .NET Framework v4.0.30319) and a "Service Map" icon. Below this, there is a navigation bar with buttons: "CS Users", "Hosted Services" (which is circled in yellow), "Connected Services", "Connection Points", and "Service Logs". A tooltip for the "Hosted Services" button explains its function: "The button above operates against the currently selected service domain. The context menu slugs operate against the root service domain, which is the one you logged into from the ConfigWeb login page. To change the service domain for traversal, choose Select from the Remote Connected Services table. The Back To Top button takes you back to the root service domain. All operations are routed through the network, so direct network connectivity is not necessary for traversal. For example, some services may be hosted on Windows Azure, while others might reside on-premise or in other hosting environments. With the Configuration Service, each service domain is completely autonomous and has exclusive access to its own configuration database. As you navigate, be aware of the currently selected service, which is displayed at the top of every page in ConfigWeb, along with the platform the selected service is deployed to. One way to think about navigation in ConfigWeb is it's the same concept as navigating through folders in Windows, except you are navigating across connected elements in your composite application, which are potentially connected across the world via the Internet, within a private corporate network, or both. Once selected, you can view and modify application settings for the service domain, and the Configuration Service will push the updates through the network to the correct domain, which will then update all running nodes without any need to deploy new configuration files, or stop and re-start the running nodes." Below the navigation bar is a "Selected Service Domain" table and a "Remote Connected Services" table.

- Click on the Hosted Services button.

.NET StockTrader Order Processor Service
Version 5.0
Microsoft Corporation
Windows Server 2008 R2 with .NET Framework v4.0.30319

Service Host ID	Service Host Type	Service Host Name / Implementation Class	Is a Workflow Host	Number of Service Endpoints	Modify Services for this Virtual Host	Edit This Virtual Host
2	Configuration Service	Trade.OrderProcessorHostConfigurationImplementation.ConfigurationService	No	4	Select	Edit
1	Node Service	ConfigService.ServiceNodeCommunicationImplementation.NodeCommunication	No	1	Select	Edit
3	Primary Service	Trade.OrderProcessorImplementation.OrderProcessor	No	5	Select	Edit

[Return to Home Page](#)

Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0
Copyright 2011, Microsoft Corporation

You will see a list of Virtual Service Hosts.

- Select the `Trade.OrderProcessorImplementation.OrderProcessor` Primary Service Host as shown.

.NET StockTrader Order Processor Service
 Version 5.0
 Microsoft Corporation
 Windows Server 2008 R2 with .NET Framework v4.0.30319
 Windows Server 2008 R2

[Back to Top](#)

Defined Endpoints for:
 Trade.OrderProcessorImplementation.OrderProcessor

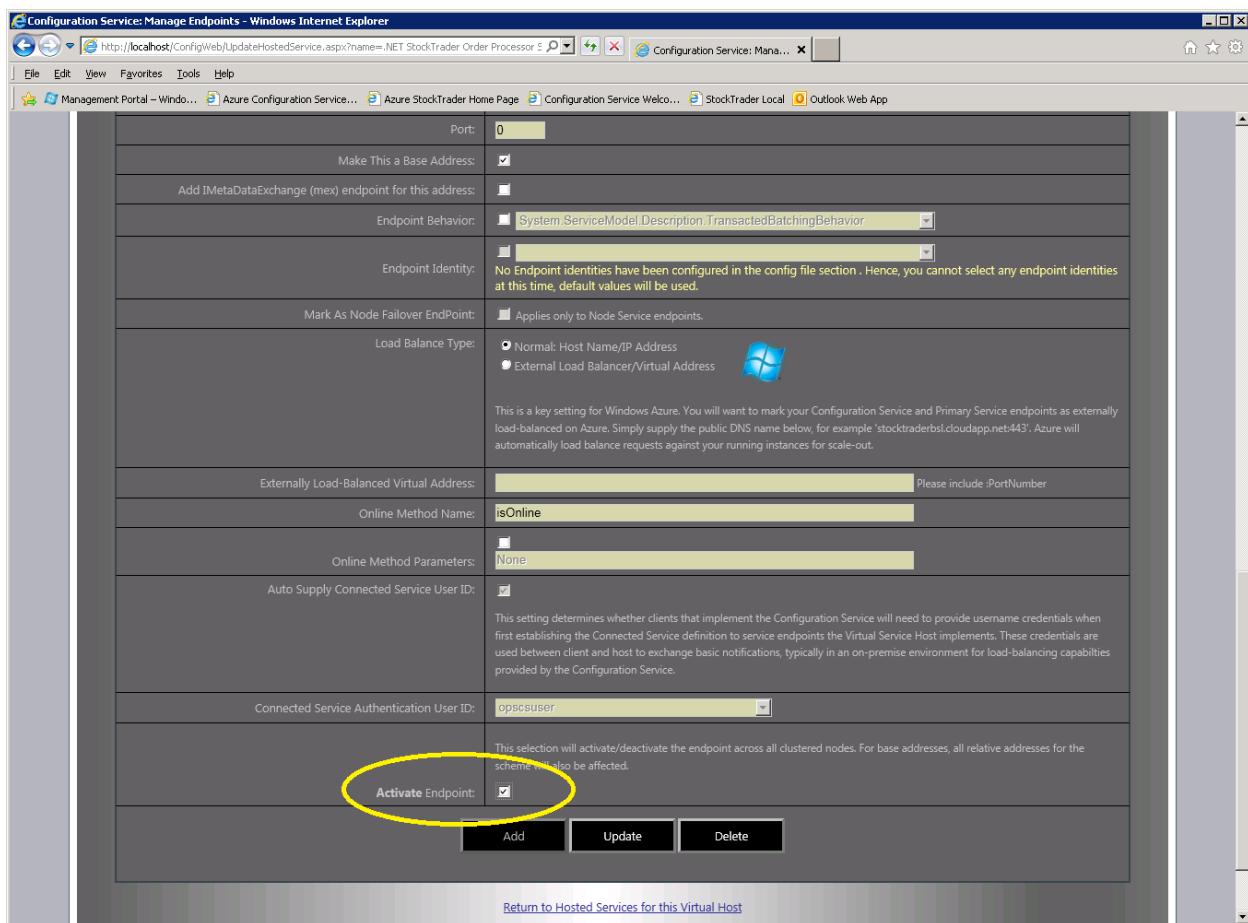
[Add Service Endpoint](#)

This page shows endpoints defined for a service host. A WCF ServiceHost can host multiple endpoints at once, potentially using different network schemes (http, https, net.tcp, etc.), different security configurations, and the like. Configuration Service lets you add, remove, edit, activate endpoints dynamically, without having to write new code. The options shown below are basically point and click options chosen in the hosted service definition page. Choose Edit to view/modify existing endpoints as defined. Keep in mind that if changing an endpoint, WCF requires that the service host be closed and then re-created with the new description. Configuration Service does this automatically across all active nodes.

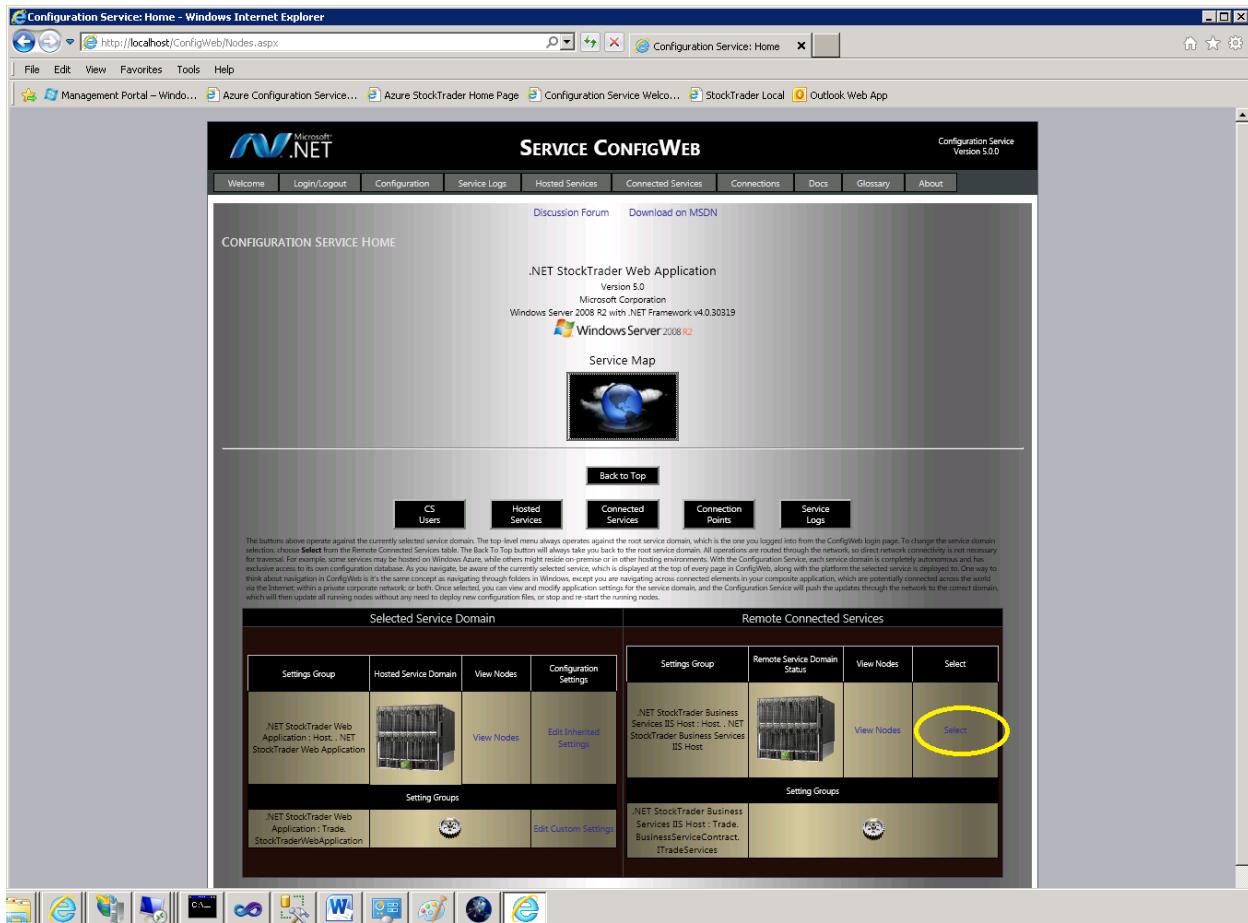
Hosted Service ID	Service Assigned Name For the Endpoint	Service Contract	Virtual Path and Port	Base Address	Binding Configuration	Activated	Edit
41	On-Premise Self-Host OPS http	Trade.OrderProcessorContract. IOrderProcessor	Port: 8000 Path: orders NAT Address: N/A	yes	Host_BasicHttpBinding Binding Type: basicHttpBinding UseHttps: False	True	Edit
32	On-Premise Self-Host OPS netTcp	Trade.OrderProcessorContract. IOrderProcessor	Port: 8001 Path: orders NAT Address: N/A	yes	Host_CustomTcpBinding Binding Type: customTcpBinding UseHttps: False	True	Edit
22	On-Premise Self-Host OPS MSMQ Transacted Queue	Trade.OrderProcessorContract. IOrderProcessor	Port: 0 Path: tradeorders NAT Address: N/A	yes	Host_MsmqDurable Binding Type: netMsmqBinding UseHttps: False	False	Edit
20	On-Premise Self-Host OPS netTcp security = TransportWithMessageCredential: ClientCertificate	Trade.OrderProcessorContract. IOrderProcessor	Port: 8001 Path: orders NAT Address: N/A	yes	Host_TcpBinding_T_Security_MCredential_ClientCert Binding Type: netTcpBinding UseHttps: False	False	Edit
36	On-Premise Self-Host OPS netTcp security = TransportWithMessageCredential: ClientUserName	Trade.OrderProcessorContract. IOrderProcessor	Port: 8001 Path: orders NAT Address: N/A	yes	Host_TcpBinding_T_Security_MCredential_UserName Binding Type: netTcpBinding UseHttps: False	False	Edit

[Return to Virtual Host List](#)

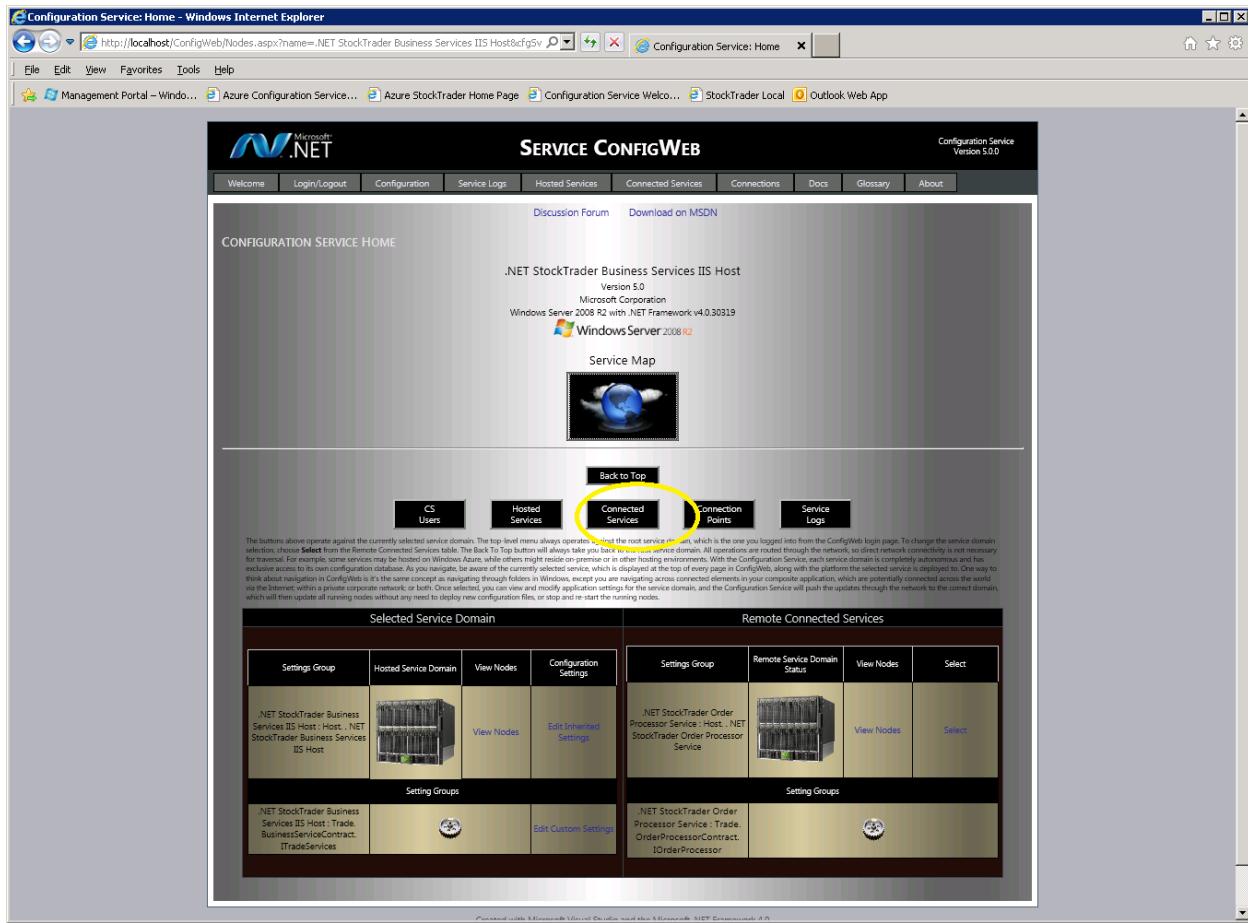
- This is now a list of the **service endpoints** defined for this service host. Not all endpoints are active, and not all can be active at the same time.
- Click the **Edit** link for the On-Premise Self-Host OPS MSMQ Transacted Queue endpoint.



- Scroll down to the bottom of the page. Here you will select **Activate Endpoint**.
- Click the **Update** button.
- The service host will be restarted across all running nodes, this time with the MSMQ endpoint activated. You can look at the Main window of the Order Processor Windows host and see the service has been restarted with the new definition.
- Now we must create a new Connected Service definition for the IIS host, so it can use this new endpoint.
- Go back to the home page by clicking on the Back to Top button on the page.



- Select the IIS Host service domain as shown above.



- Now click on the **Connected Services** button.

The screenshot shows a Microsoft Internet Explorer window titled "Configuration Service: Connected Services - Windows Internet Explorer". The address bar displays the URL "http://localhost/ConfigWeb/CServices.aspx?name=.NET StockTrader Business Services IIS Host&c=1". The page header includes the ".NET" logo and the title "SERVICE CONFIGWEB". A navigation menu at the top includes links for "Welcome", "Login/Logout", "Configuration", "Service Logs", "Hosted Services", "Connected Services", "Connections", "Docs", "Glossary", and "About". Below the menu, there are links for "Discussion Forum" and "Download on MSDN". The main content area is titled "CONNECTED SERVICES" and displays a table of service definitions. One row in the table is highlighted with a yellow circle around the "Add Service" button. The table columns are: Virtual Host ID, Service Name, Service Type, Service Contract, Client Configuration, and Edit/Delete. The highlighted row shows the following data:

Virtual Host ID	Service Name	Service Type	Service Contract	Client Configuration	Edit/Delete
102	On-Premise Self-Host OPS http	Primary Connected Service	Trade.OrderProcessorContract, IOrderProcessor	Client, BasicImpBinding Binding Type: basicImpBinding Security Mode: None	Edit
102	On-Premise Self-Host OPS netTcp	Primary Connected Service	Trade.OrderProcessorContract, IOrderProcessor	Client, TcpBinding Binding Type: netTcpBinding Security Mode: None	Edit

At the bottom of the page, there is a link "Return to Home Page". The footer contains copyright information: "Created with Microsoft Visual Studio and the Microsoft .NET Framework 4.0" and "Copyright 2011, Microsoft Corporation". There is also a "Visual Studio" logo.

- Click on the **Add Service** button.

Configuration Service: Connected Service Definitions - Windows Internet Explorer

File Edit View Favorites Tools Help

Management Portal – Windo... Azure Configuration Service... Azure StockTrader Home Page Configuration Service Welco... StockTrader Local Outlook Web App

Back to Top

Instructions

This page enables you to establish connections to remote services your application will utilize. A primary service implements the Configuration Service; a generic service does not. For example, a generic service might be one written in Java, PHP or .NET. When establishing connections to **primary services**, you will simply connect to the configuration service of the remote host, and a list of available services (contracts) will be returned. You can also establish connection to **generic services** simply by providing the endpoint to the service, and filling out the requested details. A service implementing the Configuration Service can always be treated as a generic service, since the Configuration Service imposes no requirement on the language clients are developed in, or whether they also implement the Configuration Service.

Connected Service Type

Primary Connected Service
 Generic Connected Service

Remote Configuration Service Details - Get Connected!

Address to Configuration Service:	Once connected, the remote host will return the available services your client application can subscribe to. You can connect over http, https, or top to the remote Configuration Service to retrieve this list. Make sure to enter the address to the configuration service endpoint. If the host does not implement the configuration service, choose the Generic Service type above. <input type="text" value="http://localhost:8002/orders/config"/>
Client Configuration Name:	Please make sure to select a valid Configuration Service Client. Service configurations displayed below are always prefixed with "client_configsvc" in the config file 'client' section to appear in this list. You can use wsutil.exe to generate new client definitions to remote configuration services if these services expose their metadata. <input type="text" value="Client_ConfigSvc_BasicHttpBinding"/> Selected Binding Type: basicHttpBinding
User Id:	<input type="text"/> <small>(see information below left)</small>
Password:	<input type="password"/>

The credentials above are used by the Configuration Service to send/receive notifications between hosts and subscribed clients. Hosts can be configured to automatically return these credentials for any client attempting to subscribe, in which case nothing needs to be entered before connecting, the credentials will be returned. If the remote host is not configured to automatically provide these credentials, you will need to enter valid credentials to the remote host above, with the user supplied having at least 'Connected Service Rights' as defined by the remote host in its users database.

Get Remote Services!

[Return to Connected Services List](#)

- Enter the address of the remote Order Processor configuration service. This is <http://localhost:8002/orders/config>.
- Click **Get Remote Services!**

Please Select the Primary Service

Host Name Identifier:	.NET StockTrader Order Processor Service																
<p>These are the available service endpoints by friendly names as assigned by the Service Host. Each has its own binding requirements (network scheme, security, encoding, etc.). Please select the desired service from this list. Client configurations and bindings can be generated by svutil.exe, named to Configuration Service standards, and inserted into this application's configuration file to appear in this page as selectable. You will also need to supply the service contract your client uses in your startup call per the StockTrader example and the documentation.</p> <p>On-Premise Self-Host OPS MSMQ Transacted Queue</p> <table border="1"> <tr> <td>Service Contract:</td> <td>Trade.OrderProcessorContract.IOrderProcessor</td> </tr> <tr> <td>Service Binding Type:</td> <td>netMsmqBinding</td> </tr> <tr> <td>Security Mode:</td> <td>None</td> </tr> <tr> <td>Service Virtual Path:</td> <td>tradeorders</td> </tr> <tr> <td>Service Port:</td> <td>0</td> </tr> <tr> <td>Uses Https:</td> <td>False</td> </tr> <tr> <td>Connected Configuration Service ID:</td> <td></td> </tr> <tr> <td>Assigned CSUserId:</td> <td></td> </tr> </table>		Service Contract:	Trade.OrderProcessorContract.IOrderProcessor	Service Binding Type:	netMsmqBinding	Security Mode:	None	Service Virtual Path:	tradeorders	Service Port:	0	Uses Https:	False	Connected Configuration Service ID:		Assigned CSUserId:	
Service Contract:	Trade.OrderProcessorContract.IOrderProcessor																
Service Binding Type:	netMsmqBinding																
Security Mode:	None																
Service Virtual Path:	tradeorders																
Service Port:	0																
Uses Https:	False																
Connected Configuration Service ID:																	
Assigned CSUserId:																	
Select From Available Remote Endpoints:	<p>You must select from the client contracts available based on those you supplied in your initialization logic.</p> <p>Select Client Contract to Use:</p> <p>Trade.OrderProcessorContract.IOrderProcessor</p>																
Select Client Configuration to Use:	<p>This list has been automatically trimmed to client configurations found in config that 1) Match the service contract selected above; 2) Use a compatible binding type and security mode; and 3) match the Configuration Service naming pattern. Select the appropriate available client configuration to use. For example, as generated by <code>svutil.exe</code> via WS Metadata Exchange (mex). If you do not see the configuration you want, you need to either add it to your web.config or .exe.config file, or correct issues stemming from the matching pattern. The matching pattern prevents many runtime exceptions before they happen.</p> <p>Client_MsmqDurable</p> <p>Selected Binding Type: netMsmqBinding</p>																
Online Check Method:	<p>isOnline</p> <p>None</p>																
Apply Default UserName Client Credentials:	<p><input checked="" type="checkbox"/></p> <p>Yes, use a specific user ID defined in this configuration database to supply per request to this service. See documentation, you can also supply different credentials dynamically at runtime if desired based on other authentication sources.</p>																
Choose User for Default Client UserName Credentials (if checked above). Only users with 'Service Operation Rights' will appear in this list.	<p>bsloperationuser</p>																
<p>Add Update Delete</p> <p><i>(There are services you already have established definitions for, and these have been removed from the available list!)</i></p>																	

- Make sure the correct contract and client configuration are selected, they should be correct by default.
- Click the **Add** button.
- Click on the Back to Top Page to go back to the root home page.

- Once again select the IIS Business Service host as shown, since we want to configure it.

Configuration Service: Home - Windows Internet Explorer

File Edit View Favorites Tools Help

Management Portal – Wind... Azure Configuration Service... Azure StockTrader Home Page Configuration Service Welco... StockTrader Local Outlook Web App

CONFIGURATION SERVICE HOME

.NET StockTrader Business Services IIS Host
Version 5.0
Microsoft Corporation
Windows Server 2008 R2 with .NET Framework v4.0.30319

Service Map

Back to Top

CS Users **Hosted Services** **Connected Services** **Connection Points** (highlighted with a yellow circle) **Service Logs**

The buttons above operate against the currently selected service domain. The top-level menu always operates against the root service domain. When you log in, you logged into from the ConfigWeb login page. To change the service domain selection, choose **Select** from the Remote Connected Services table. The Back To Top button will always take you back to the root service domain. All operations are routed through the network, so direct network connectivity is not necessary for traversal. For example, some services may be hosted on Windows Azure, while others might reside on-premises or in other hosting environments. With the Configuration Service, each service domain is completely autonomous and has exclusive access to its own configuration database. As you navigate, be aware of the currently selected service, which is displayed at the top of every page in ConfigWeb, along with the platform the selected service is deployed to. One way to think about navigation is that you are always operating on your local service, except you are navigating across connected elements in your composite application, which are potentially connected across the world via the internet, within a private corporate network, or both. Once selected, you can view and modify any configuration settings for the service domain, and the Configuration Service will push the updates through the network to the correct domain, which will then update all running nodes without any need to deploy new configuration files, or stop and re-start the running nodes.

Selected Service Domain				Remote Connected Services			
Settings Group	Hosted Service Domain	View Nodes	Configuration Settings	Settings Group	Remote Service Domain Status	View Nodes	Select
.NET StockTrader Business Services IIS Host : Host .NET StockTrader Business Services IIS Host		View Nodes	Edit Inherited Settings	.NET StockTrader Order Processor Service : Host .NET StockTrader Order Processor Service		View Nodes	Select
Setting Groups				Setting Groups			
.NET StockTrader Business Services IIS Host : Trade. BusinessServiceContract. ITradeServices		Edit Custom Settings		.NET StockTrader Order Processor Service : Trade. OrderProcessorContract. IOrderProcessor			

- Click on the **Connection Points** button (which will operate, like the other buttons, against the currently selected service domain).

SERVICE CONNECTION POINTS

.NET StockTrader Business Services IIS Host
Version 5.0
Microsoft Corporation
Windows Server 2008 R2 with .NET Framework v4.0.30319
Windows Server 2008 R2

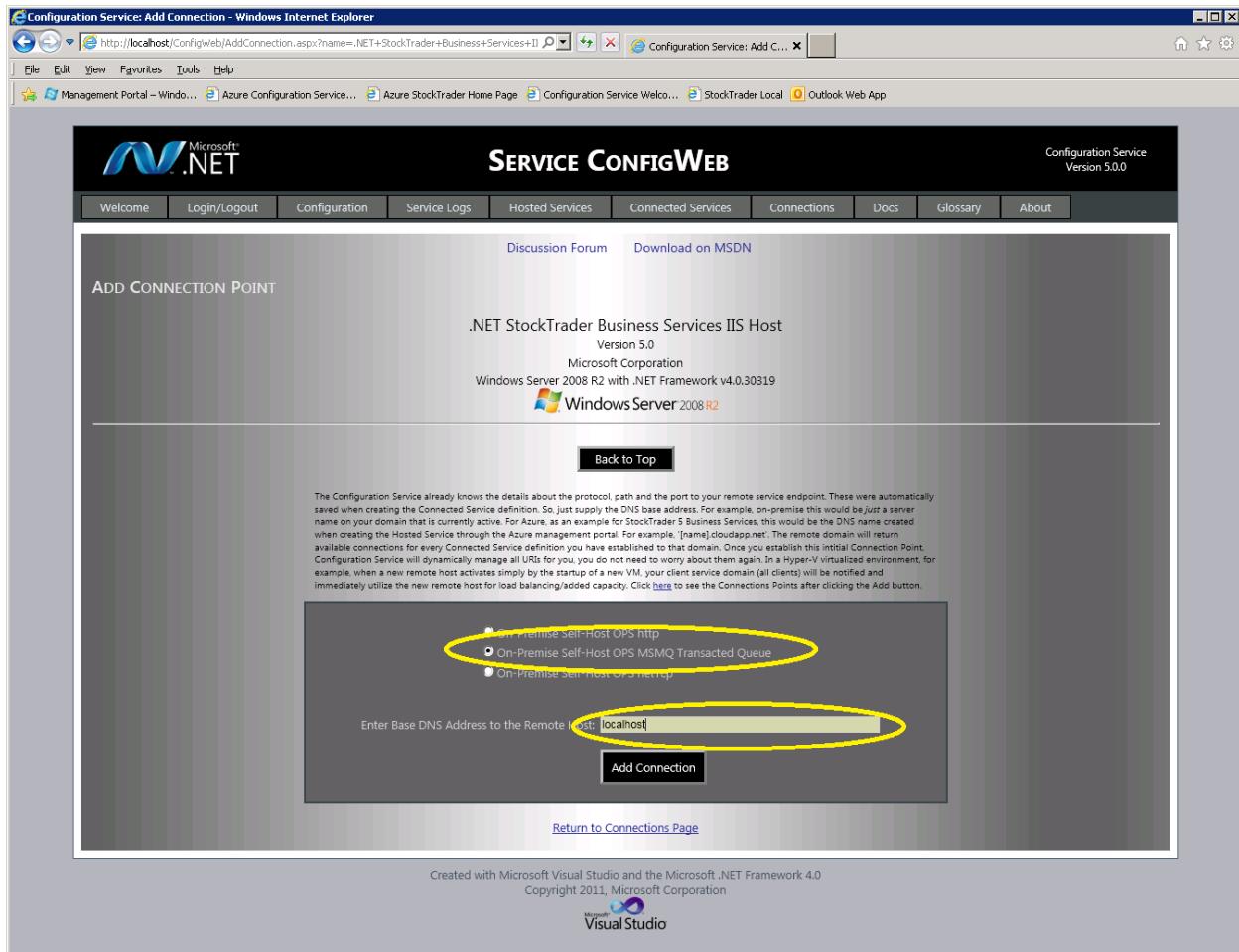
Add Connection

This page shows the actual WCF connections between the client service domain and the remote service domains for any business service endpoint maintained by the Configuration Service. If the remote service domain is running behind any type of NAT/internal load balancer, as is the case with Windows Admin service domains, then you will see one connection point per domain with a corresponding port, just as the client service domain sees - even though in reality there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly to each load balanced node, with load balancing and request-level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all nodes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the WCF client being utilized, including the endpoint address, the name of the client definition as defined in configuration, the binding type, binding configuration name, and security mode the client is utilizing.

If a connection point below shows red, hover over it to see the actual exception the client is receiving. If a connection below shows as a grey-colored server, a timeout was exceeded (also indicating an issue).

Remote Address	Client Details	Service Status	Remove
http://igwin2008-lab.corp.dotnettrade.com:8000/orders	On-Premise Self-Host OPS http Client Configuration: Client_BasicHttpBinding Service Contract: Trade.BusinessServiceContract.ITradeServices Binding Configuration: Client_BasicHttpBinding Binding Type: basicHttpBinding Security Mode: None Endpoint Behavior: None Assigned		Delete
net.tcp://igwin2008-lab.corp.dotnettrade.com:8001/orders	On-Premise Self-Host OPS netTcp Client Configuration: Client_TcpBinding Service Contract: Trade.BusinessServiceContract.ITradeServices Binding Configuration: Client_TcpBinding Binding Type: netTcpBinding Security Mode: None Endpoint Behavior: None Assigned		Delete

- Click on the **Add Connection** Button.



- Select the MSMQ endpoint.
- Type in localhost as the host DNS name.
- Click Add Connection.
- Select the Return to Connections Page link at the bottom of the page when the connection has been added.

Configuration Service: Connection Points - Windows Internet Explorer

File Edit View Favorites Tools Help

Management Portal – Wind... Azure Configuration Service... Azure StockTrader Home Page Configuration Service Welcome StockTrader Local Outlook Web App

reality, there may be many nodes servicing this endpoint behind the NAT load balancer. On private cloud deployments, unless you are also using a NAT to translate addresses, you will see every node as a distinct connection point, as clients communicate directly to each load-balanced node, with load balancing and request-level failover provided by the Configuration Service itself with no NAT required. While the Node Map and Service Map pages display all nodes even if behind a NAT, this page is important in that it shows the view the client service domain is actually operating against. As such, you can see the detail on the WCF client being utilized, including the endpoint address, the name of the client definition as defined in configuration, the binding type, binding configuration name, and security mode the client is utilizing.

If a connection point below shows red, hover over it to see the actual exception the client is receiving. If a connection below shows as a grey-colored server, a timeout was exceeded (also indicating an issue).

Connections To Host: .NET StockTrader Order Processor Service

[View/Configure This Service Host's Connection Points](#)

[Purge All Connections to This Service Host](#)

Remote Address	Client Details	Service Status	Remove
http://lgwin2008-lab.corp.dotnettrade.com:8000/orders	On-Premise Self-Host OPS http Client Configuration: Client_BasicHttpBinding Service Contract: Trade.BusinessServiceContract.ITradeServices Binding Configuration: Client_BasicHttpBinding Binding Type: basicHttpBinding Security Mode: None Endpoint Behavior: None Assigned		Delete
net.msmq://lgwin2008-lab.corp.dotnettrade.com/private/tradeorders	On-Premise Self-Host OPS MSMQ Transacted Queue Client Configuration: Client_MsmqDurable Service Contract: Trade.OrderProcessorContract.IOrderProcessor Binding Configuration: Client_MsmqDurable Binding Type: netMsmqBinding Security Mode: None Endpoint Behavior: None Assigned		Delete
net.tcp://lgwin2008-lab.corp.dotnettrade.com:8001/orders	On-Premise Self-Host OPS netTcp Client Configuration: Client_TcpBinding Service Contract: Trade.BusinessServiceContract.ITradeServices Binding Configuration: Client_TcpBinding Binding Type: netTcpBinding Security Mode: None Endpoint Behavior: None Assigned		Delete

Note that MSMQ nodes will show up as online as long as the MSMQ messaging service is running on that node, even if the service host at that node is not active. This is by design, since these 'loosely-coupled' nodes will actively receive messages from clients as long as the MSMQ message service is running (and process these messages when the service host is started again).

If the message service is stopped on a node, it will show up as offline, and StockTrader will automatically failover to the other nodes. You will only be able to view connection points for a remote service, however, if the host process itself is running, regardless of the status of the MSMQ messaging service.

[Return to Home Page](#)

- You can see the MSMQ endpoint above. Note that in this screen shot, we already changed the **OrderMode** setting to On-Premise Self-Host OPS MSMQ Transacted Queue for the Business Service IIS host domain. Hence it shows as green, meaning currently being used (only one endpoint is used at a time for StockTrader—but not necessarily for all applications).
- You should make sure the BSL IIS Host is configured with **OrderMode** per above, or else the Trade application will not actually be using the MSMQ endpoint/durable message queue.

Note

If the MSMQ endpoint above shows up as red, you have an issue. Hover over the red icon to see the exception message (the complete exception will also be in the Service Logs page). It is likely you did not follow the post-install steps (see Readme.html in the start menu) to configure the MSMQ to allow permissions for the anonymous IIS account to write to the queue. If so, follow the readme post-install instructions related to MSMQ and then just refresh the page above.

New Hosts and Configuration Service Dynamic URI Management

You have used ConfigWeb to configure a variety of endpoints to a variety of hosts. You only have to do these steps once, even if new host nodes are started, however. When a new host now starts up, clients with definitions will all be notified, and Configuration Service will manage the URIs and load balancing. You can start the Order Processor program, for example, on a new server, and then in ConfigWeb the Service Map page and the Connections page will show it; and the BSL layer will already be using the new node for load balancing/failover. Note that both the Business Services and Order Processor Service can optionally be installed as **Windows NT services**, and set to Auto-start when a machine starts up or is restarted.

Re-Configuring StockTrader to Work with Oracle 11G

While the configuration repositories must be SQL Server/SQL Azure in this release, a complete Oracle DAL is provided for the core application and it can run against Oracle 11G databases. You will want to install the latest Oracle 11G 32-bit or 64-bit client, downloadable from Oracle Corporation's Web site. You will also want to make sure you install Oracle's latest ADO.NET ODP, which installs their Oracle.DataAccess assembly. The client and the ODP need to be installed on each computer running any component/service for StockTrader.

You will use TNS Names to create a name for the database, and simply update the configuration via ConfigWeb to use the Oracle DAL instead of the SQL Server DAL. To first create the Oracle database, you will find the DDL script located in:

[stocktrader setup dir]\stocktrader\DatabaseLoaders\TradeOracleLoader\Create_Oracle_Trade_Database_DDLScript

Simply use **SQLPlus** to create the database schema by running this file. To create the Oracle schema, please see the .NET StockTrader Oracle Database Loader program install directory for the DDL which can be run from SQLPlus. It is ideal to first create a new tablespace, then create a new user 'trade'; password 'trade' assigned to this tablespace, and then login to **SQL Plus** as 'trade' to create the Trade database schema from the provided DDL.

Next, create a TNS Names entry for your SQL Server instance, with the login trade, trade. This will be your reference to the database from the various StockTrader service domains, and this entry and the Oracle client hence must be installed on every node running .NET StockTrader.

The .NET Oracle Database loader program can then be used to load the Oracle database with data. Simply run this from the Start menu, and for the server name use your TNS name entry.

- Once loaded with data, login to ConfigWeb against the StockTrader Web application service domain, while the application is still configured to run against SQL Server.
- Next, select the Business Service domain, to configure its custom settings, and select **Edit Custom Settings** for the Business Services.
- Select the Detailed Settings (button at top of page).
- Click on Change Value for the setting named **DataAccessLayer**, and change the value from Trade.DALSQLServer to Trade.DALOracle.
- Click Update to apply the change.
- Next change the setting for **Stocks Database Name** to be your TNS name entry for your Oracle server. It is also under “Detailed” settings for the Business Services.
- Finally, change the **Stocks DatabaseUserID** and **Stocks Database Password** to be the values you set when creating the trade userid (which should have its own tablespace where the database lives) and the trade password.
- These application settings are also part of the Order Processor tier. If the OPS is running when making these changes to the BSL above, they are pushed down to the OPS. But you should check these corresponding settings for the Order Processor Service via ConfigWeb to make sure they are set the same as above.
- You can now try to login to the Trade Web application as an end-user, to check everything is working against Oracle!

Separation of Implementation from Schema and Contract

In the service-oriented world, a properly designed system cleanly separates implementation of services into autonomous black boxes, to avoid ever having to share classes/code between layers. When running with AccessMode set to any setting other than ‘InProcess’; this is the way StockTrader is designed. However, StockTrader supports the ‘InProcess’ collapsed mode to run as a standalone Web application as well, which is also a perfectly valid configuration. Hence, the Visual Studio StockTrader Web Application solution does share projects and classes from Business Services to support this mode. You should keep in mind, as noted in the source code comments, that this is a special case; and that normally the only sharing between the StockTrader Web application projects and the Business Service projects would be the **BusinessServiceContract** project, and the **BusinessServiceDataContract** project. The same is true for the Business Service solution, which references Order Processor implementation projects to support the in-process order mode (OrderMode=‘InProcess Activation’).

The information for the service and data contract can also optionally be generated via the **SVCUTIL.exe** tool that is part of the Windows 7.x SDK; for example if someone else is hosting the service outside of your organization; or it is written in a different platform like J2EE. This tool generates both the Data Contracts and the Service Contracts for building clients; it also generates the binding information to be used. This can then be used as-is; or easily modified into a custom client as we did with .NET StockTrader. We do not use a purely generated proxy because:

- a) We already have a DataContract and ServiceContract designed in separate re-usable projects that can simply be imported into other solutions.
- b) We use the WCF **ChannelFactory** class with cached instances of channels to Business Services (and the Order Processing Service) for performance reasons.

Why A Configuration Service and ConfigWeb?

As a composite application, .NET StockTrader can have many services running across multiple servers. In addition, as a performance sample application, .NET StockTrader has a wealth of different configuration settings to allow developers to test the performance of various different technologies, protocols and deployment topologies for an application. Without a way to centrally manage configuration data, it would be extremely difficult to ensure the composite application, as whole, was configured as desired and to keep configuration files in sync across various servers—especially if services are deployed in geographically disperse locations. The solution implemented in StockTrader is the configuration management service, which is a WCF Web service hosted in each component (StockTrader Web Application, StockTrader Business Services, and StockTrader Order Processor Service). The configuration system relies on a SQL-Server (or SQL Azure) based repository schema.

This service utilizes a central SQL repository/schema, and each element of StockTrader has its own separate repository. All of the data is not stored in a single shared repository because, to a large degree, this would degrade many of the intended benefits of service-orientation. For example, if

all elements of the application shared one central database repository, then all would need to be deployed within one application domain, and typically on the same subnet; or a subnet with direct connectivity to the single central database.

The configuration service takes a different approach such that each service can be deployed anywhere. The only requirement of course is that services that directly connect to each other must have network connectivity. But a composite application might be made up of services that connect several layers deep, in different geographic regions, and all elements might not require (or even be able to achieve) direct connections to all other elements. A composite application should not be required to be deployed on the same internal network just to surface an integrated management and configuration experience. Hence, the configuration management service maintains the principle that ***services should be autonomous***—and each gets its own configuration repository, although the schemas for the repository are common.

Starting Point for Implementing the Configuration Service

If you are interested in implementing the configuration service, you should start with the new Visual Studio 2010 template and tutorial. With 5.0, the process is very straightforward since everything is provided in base classes. Once done, you can use all the features of the Configuration Service and ConfigWeb for your own applications and services.