Deltek Authorization Server Pro

ProPricer Authorization Server Pro Azure Deployment Guide

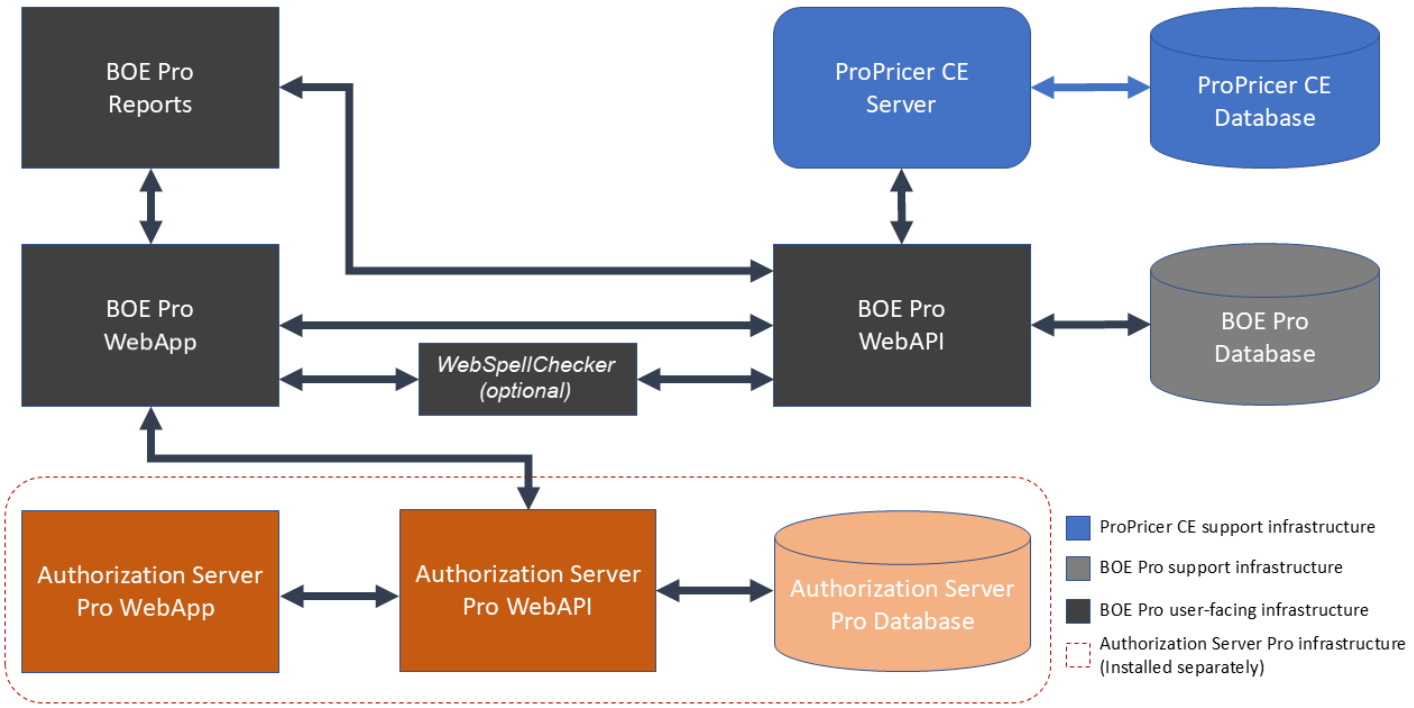
Contents

Introduction

Authorization Server Pro, an n-tier web application developed with ASP.NET Core, supports various deployment options. Authorization Server Pro provides centralized user management and licensing for ProPricer companion products including BOE Pro and Cost Volume Pro.

The focus of this guide is deployment on Microsoft Azure using App Service.

Architecture



Authorization Server Pro

Minimum requirements

- Azure account with required permissions to create App Services.
- Windows App Service Plan with the Premium or Isolated plan depending on your workload.
    - The Isolated plan is strongly recommended. It includes improved performance and security features and is ideal for providing access to company-approved users only.
- Web applications required:

App Name	.NET version (Runtime stack)
WebAPI	.NET 8
WebApp	.NET 8

- SQL Server (Azure SQL Database or SQL Server 2016 or newer).
- Optional: SSL certificate or certificates for custom domains.
- Optional: PFX certificate for encryption of authentication tokens.
- Optional: Azure Application Registration to enable Azure Active Directory login and client certificate.

Known limitations

- Authorization Server Pro fully supports Azure Active Directory logins, but Windows authentication is not supported when using Azure. However, Authorization Server Pro supports Windows authentication when it is running on Windows Server.
- Bookmarking the URL of the Log In page can cause login issues for users. To quickly open Authorization Server Pro in a browser, users should bookmark the web application start URL instead.

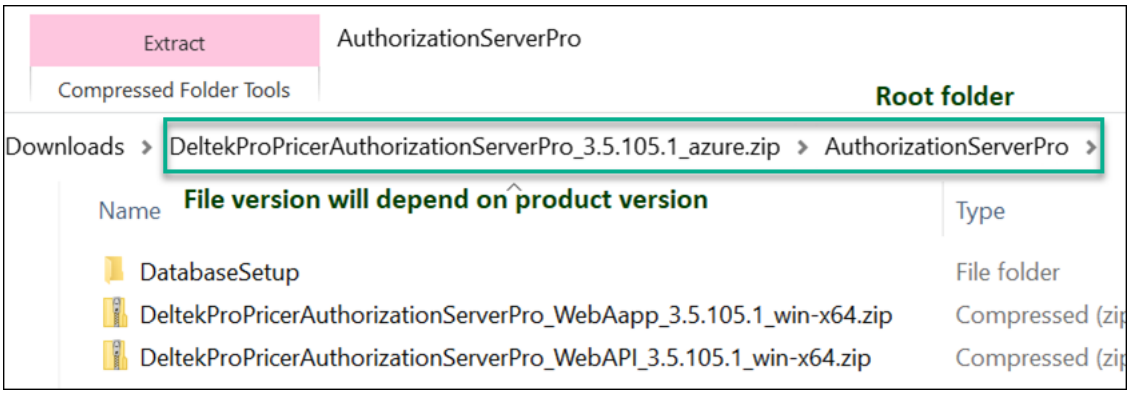
Authorization Server Pro deployment package

The platform-specific and framework-dependent package includes only the application and its dependencies. The .NET runtime is provided by the ASP.NET Hosting Bundle. The deployment package includes a Windows x64 platform-specific executable.

Typically, the package is deployed on one and the same Windows server, but you can deploy each site on different servers for scalability reasons.

The ZIP package is available in the Deltek Software Manager (DSM).

The file name of the ZIP package is **DeltekProPricerAuthorizationServerPro_[version]_azure.zip**. Download and extract it to a temporary folder.



Authorization Server Pro Web API

Back-end Authorization Server Pro web API that receives requests from the Authorization Server Pro web application, provides the WebSockets implementation for live collaboration updates, and communicates with the database to persist the information. This component requires .NET 8 ASP.NET Core runtime.

Authorization Server Pro WebApp

Front-end Authorization Server Pro web application. This component serves only static files such as JS, HTML, CSS, etc. (the files are included in the zip file).

Authorization Server Pro Database Setup

Console application that creates and upgrades Authorization Server Pro databases. This component uses .NET 8 single .exe deployment (.NET 8 is included in the .exe).

Create an Authorization Server Pro database

The first thing you must do is create your Authorization Server Pro database. Authorization Server Pro Database Setup is a command line tool that allows database administrators to create and upgrade Authorization Server Pro databases.

1. Open a command prompt window.
2. Go to the folder where AuthorizationServerProDatabaseSetup.exe was extracted.
3. Run the tool with the create command depending on the authentication method.
    - To use SQL authentication over Azure SQL, run the tool with the create command and follow the prompts (database must already exist):

      ```
      AuthorizationServerProDatabaseSetup create -e
      ```

    - To use SQL authentication, run the tool with the create command and follow the prompts:

      ```
      AuthorizationServerProDatabaseSetup create
      ```

    - To use Windows authentication, run the tool with the create command, add the -w parameter, and follow the prompts:

      ```
      AuthorizationServerProDatabaseSetup create -w
      ```

See the Authorization Server Pro Database Setup reference section for further guidance.

Install Authorization Server Pro WebAPI

The Authorization Server Pro web API connects to the Authorization Server Pro database. Once connected, it provides all required data to the Authorization Server Pro web application and all other ProPricer companion products.

Prerequisites

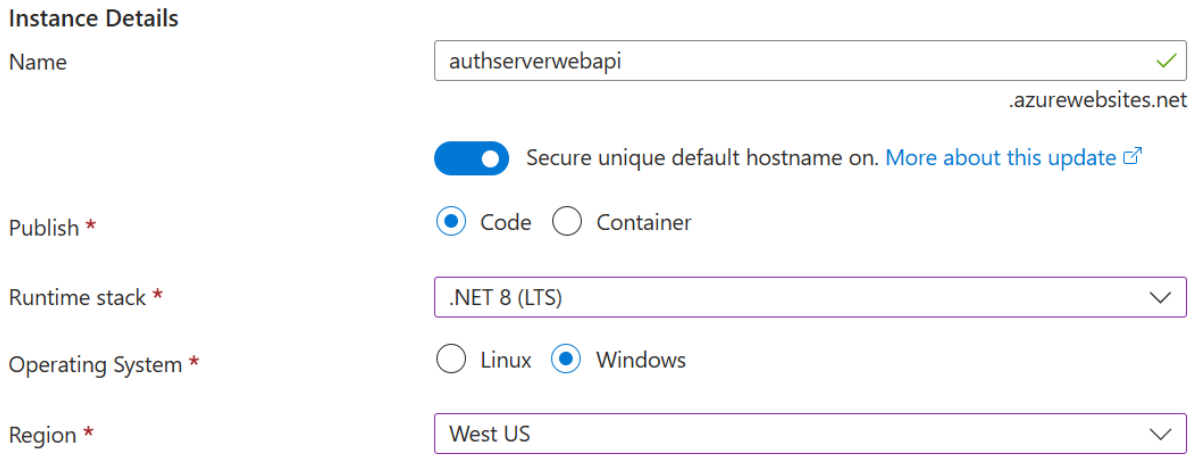
Complete all prerequisites before continuing.

AuthorizationServerProWebAPIManageConfiguration

- Authorization Server Pro WebApp URL.
- Authorization Server Pro database connection information.
- Optional: Application registration’s Application (client) ID, Directory (tenant) ID to enable Azure AD logins, and certificate private key (.pfx) for client certificate.
- BOE Pro WebApp URL (optional)
- Cost Volume Pro WebApp URL (optional)

App Service

1. In the Azure Portal, add a Web App (App Service).
2. Enter the Web App name.
3. Select the following settings:
   - **Publish**: Code
   - **Runtime stack**: .NET 8 (LTS)
   - **Operating System**: Windows
   - **Region**: Select the desired region



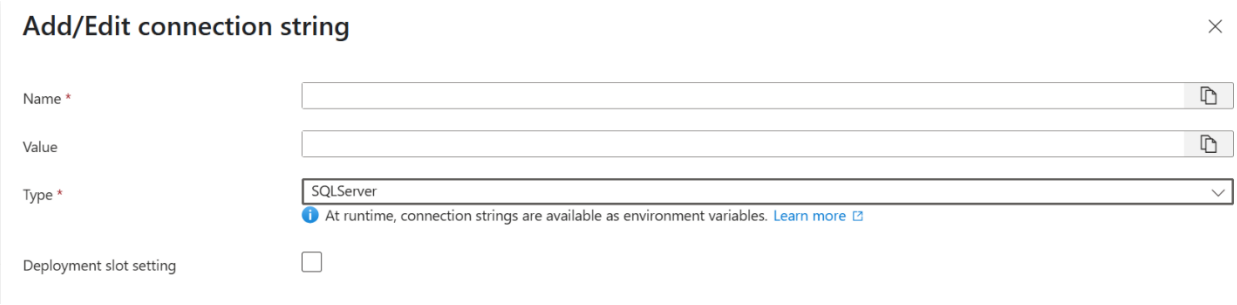
***Recommendation**: Select the same subscription, resource group, and region so you can select the same App Service Plan for all Authorization Server Pro App Services.*

4. On the **Deployment** tab, set **Continuous deployment** to **Disable**.
5. On the **Networking** tab, set **Enable public access** to **On**, and **Enable network injection** to **Off**.
6. On the **Tags** tab, create the desired tags.
7. On the **Review + create** tab, click **Create**.

Configuration

To finish the configuration of Authorization Sever Pro WebAPI:

1. In the App Service in the Azure Portal, go to the **Settings** section and select the **Environment variables** option.
2. On the **Connection strings** tab, click **Add** to create the following connection string:



Name	Value	Example
Name	DBConnection	
Value	<your-database-connection-string>	Server=tcp:my.database.windows.us,1433;Database=authserverprodb;Persist Security Info=False;User ID=authprodbo;Password={your-password};MultipleActiveResultSets=True;Encrypt=True; TrustServerCertificate=True;Connection Timeout=30;
Type	SQLServer	
Deployment slot setting	Unselected	

*If you are using Azure SQL Database with the Azure Portal, go to the database > **Overview** section > **Show database connection strings**, then select the connection string on the **ADO.NET** tab. Replace **Initial Catalog** with **Database**.*

3. Ensure SQL Database Server and App Service are connected to a Virtual Network (VNet) that allows SQL server traffic. Typically, the easiest method is to add a subnet to the SQL server VNet and link the App Service to the new subnet.
4. Create the following Environment Variables:

Name	Value	Example
AzureAD:ClientCertificates:0:CertificateStorePath	<your certificate store path>	CurrentUser/My
AzureAD:ClientCertificates:0:CertificateThumbprint	<your certificate thumbprint>	F2B8C6E2RR2B7B4BC2971RR7ED786A0C9CW56D75
AzureAD:ClientCertificates:0:SourceType	StoreWithThumbprint	ProPricer_Prod_SQL
AzureAd:ClientId	<your Azure client id GUID value>	f7b8f8b1-bafb-4aa6-9d3f-891f37b6e677
AzureAd:Instance	https://login.microsoftonline.us/	
AzureAd:TenantId	<your Azure tenand id GUID value>	0f695235-1127-4a74-99fc-6409615100fe
Certificate:StoreLocation		CurrentUser
Certificate:StoreName		My
Certificate:ThumbPrint	<your certificatethumbprint>	F2B8C6E2RR2B7B4BC2971RR7ED786A0C9CW56D75
Clients:0:AllowedCorsOrigins:0	<your authorizationserver pro webappurl>	https://aspwebapp.azurewebsites.us

Name	Value	Example
Clients:0:AllowedPostLogoutRedirectUrls:0	<your Authorization Server Pro webapp url>	https://aspwebapp.azurewebsites.us
Clients:0:AllowedPostLogoutRedirectUrls:1	<your Authorization Server Pro webapp url>/logout	https://aspwebapp.azurewebsites.us/logout
Clients:0:AllowedRedirectUrls:0	<your Authorization Server Pro webapp url>/callback	https://aspwebapp.azurewebsites.us/callback
Clients:0:AllowedRedirectUrls:1	<your Authorization Server Pro webapp url>/silent	https://aspwebapp.azurewebsites.us/silent
Clients:1:AllowedCorsOrigins:0	<your Cost Volume Pro webapp url>	https://cspwebapp.azurewebsites.us
Clients:1:AllowedPostLogoutRedirectUrls:0	<your Cost Volume Pro webapp url>	https://cspwebapp.azurewebsites.us
Clients:1:AllowedPostLogoutRedirectUrls:1	<your Cost Volume Pro webapp url>/logout	https://cspwebapp.azurewebsites.us/logout
Clients:1:AllowedRedirectUrls:0	<your Cost Volume Pro webapp url>/callback	https://cspwebapp.azurewebsites.us/callback
Clients:1:AllowedRedirectUrls:1	<your Cost Volume Pro webapp url>/silent	https://cspwebapp.azurewebsites.us/silent
Clients:2:AllowedCorsOrigins:0	<your BOE Pro webapp url>	https://boeprowebapp.azurewebsites.us

Name	Value	Example
Clients:2:AllowedPostLogoutRedirectUrls:0	<your BOE Pro webapp url>	https://boeprowebapp.azurewebsites.us
Clients:2:AllowedPostLogoutRedirectUrls:1	<your BOE Pro webapp url>/logout	https://boeprowebapp.azurewebsites.us/logout
Clients:2:AllowedRedirectUrls:0	<your BOE Pro webapp url>/callback	https://boeprowebapp.azurewebsites.us/callback
Clients:2:AllowedRedirectUrls:1	<your BOE Pro webapp url>/silent	https://boeprowebapp.azurewebsites.us/silent
Host:Cors:AllowOrigin:0	<your Authorization Server Pro webapp url>	https://aspwebapp.azurewebsites.us
Host:Cors:AllowOrigin:1	<your Cost Volume Pro webapp url>	https://cvpwebapp.azurewebsites.us
Host:Cors:AllowOrigin:2	<your BOE Pro webapp url>	https://boeprowebapp.azurewebsites.us
Options:HostDomain	<your Authorization Server Pro webapi url> without protocol	asp-api.azurewebsites.us
Options:HostScheme	https	
Security:Authority	<your Authorization Server Pro webapi url>	https://aspapi.azurewebsites.us

Name	Value	Example
WEBSITE_LOAD_CERTIFICATES	<your certificate thumbprint>	F2B8C6E2RR2B7B4BC2971RR7ED786A0C9CW56D75

5. On the **General Settings** tab, verify or adjust the following settings:
    - **Stack**: .NET
    - **.NET Version**: .NET 8 (LTS)
    - **Platform**: 64 Bit
    - **Basic Auth Publishing Credentials**: Off
    - **Manage pipeline version**: Integrated
    - **FTP state**: Disabled
    - **HTTP version**: 1.1
    - **Web sockets**: On
    - **Always on**: On
    - **ARR affinity**: Off
    - **HTTPS Only**: On
    - **Minimum TLS Version**: 1.2
    - **Remote debugging**: Off
    - **Client certificate mode**: Ignore
6. Click **Save**.

Application Logging

Enable application logging to collect diagnostic information from this web application. Logging is optional but highly recommended.

To enable application logging in Azure, create the following application settings in the **Configuration** option:

Name	Value
Serilog:WriteTo:1:Name	AzureApp
Serilog:WriteTo:1:Args:outputTemplate	{Message}{NewLine}|[{Url}] {UserAgent}{NewLine}{Exception}

To view logs in a log stream:

1. In the Azure Portal, go to **Monitoring** > **App Service logs**.
2. Enable **Application logging (Filesystem)**.
3. Set **Level** to **Information**.
4. Go to **Log stream** to see the log trace.

To preserve logs to a storage account:

1. In the Azure Portal, go to **Monitoring** > **App Service logs**.
2. Enable **Application logging (Blob)**.
3. Set **Level** to **Information**.
4. Select the **Storage Container** to store the logs.
5. Set the **Retention Period (Days)**.

Web Logging

Enable web server logging to collect diagnostic information from the web server. Logging is optional but highly recommended.

To view logs in a log stream:

1. In the Azure Portal, go to **Monitoring** > **App Service logs**.
2. Set **Web server logging** to **File System**.
3. Enter the **Quote (MB)**.
4. Set the **Retention Period (Days)**.

To preserve web logs:

1. In the Azure Portal, go to **Monitoring** > **App Service logs**.
2. Set **Web server logging** to **Storage**.
3. Select the **Storage** to send the logs to.
4. Set the **Retention Period (Days)**.

Recommended TLS/SSL settings

Azure App Service is created with an SSL certificate by default to provide https and a subdomain, like propricer9webapi.azurewebsites.us.

In the App Service in the Azure Portal, you should verify that the following TLS/SSL settings were selected during configuration:

- **HTTPS Only**: On
- **Minimum TLS Version**: 1.2

Optionally, you can configure your own domain in this section, like **mysite.mycompany.com**.

Deploy ZIP file using ZipDeployUI

This ZIP file deployment uses the same Kudu service that powers continuous integration-based deployments.

1. In the App Service options pane, select **Advanced Tools**, click **Go**, expand the **Tools** menu, then select **Zip Push Deploy**. Alternatively, in your browser, go to **https://<app_name>.scm.azurewebsites.us/ZipDeployUI**.
2. Upload the Authorization Server Pro WebAPI ZIP file from your temporary folder where Authorization Server Pro ZIP file is extracted by dragging it to the file explorer on the web page.

When deployment is in progress, an icon in the top-right corner shows the progress percentage. The page also shows verbose messages for the operation below the explorer area. When it is finished, the last deployment message should say **Deployment successful**.

Alternatively, use az webapp deployment source config-zip to deploy the ZIP file using Azure CLI, or the Publish-AzWebApp cmdlet to deploy the ZIP file using PowerShell.

Create Authorization Server Pro Web Application

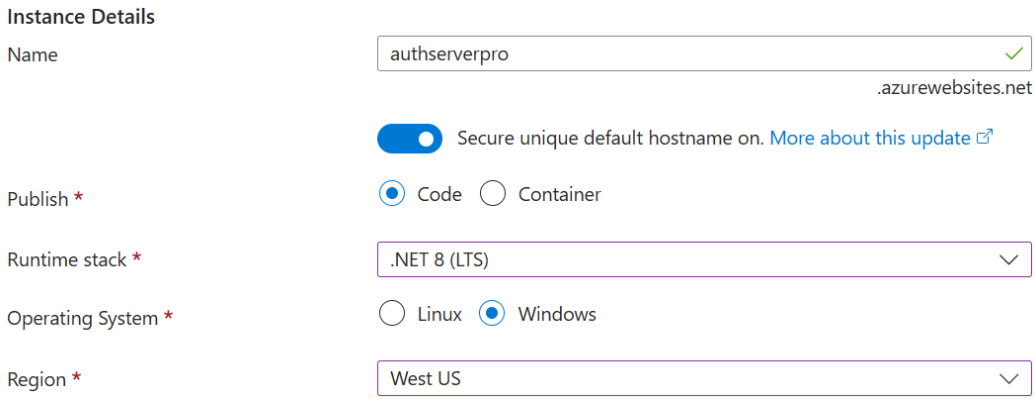
Prerequisites

Configuration

- Authorization Server Pro URL.

App Service

1. In the Azure Portal, add a Web App (App Service).
2. Enter the Web App name.
3. Select the following settings:
   - **Publish**: Code
   - **Runtime stack**: .NET 8 (LTS)
   - **Operating System**: Windows
   - **Region**: Select the desired region



***Recommendation**: Select the same Subscription, Resource group, and Region so you can select the same App Service Plan for all Authorization Server Pro App Services.*

4. On the **Deployment** tab, set **Continuous deployment** to **Disable**.
5. On the **Networking** tab, set **Enable public access** to **On**, and **Enable network injection** to **Off**.
6. On the **Monitoring** tab, set **Enable Application Insights** to **No**.
7. On the **Tags** tab, create the desired tags.
8. On the **Review + create** tab, click **Create**.

Configuration

You can configure Authorization Server Pro WebApp with the Manager tool, which is recommended, or you can edit the file **config.js** in the target folder.

1. Open a command prompt window.
2. Go to the folder where Authorization Server Pro was extracted and extract **DeltekProPricerAuthServerPro_WebApp_[version]_win-x64.zip** to a temporary folder.
3. At the command prompt in your temporary folder, enter:
   `AuthorizationServerProWebAppManager.exe config`

   The Application settings in the Configuration option in Azure are not supported by BOE Pro WebApp.

4. In Windows Explorer, select all content in your temporary folder and zip it into a new file.

To finish the configuration of Authorization Server Pro WebApp:

1. In the App Service in the Azure Portal, go to the **Settings** section and select the **Configuration** option.
2. On the **General Settings** tab, verify or adjust the following settings:
    - **Stack**: .NET
    - **.NET Version**: .NET 8 (LTS)
    - **Platform**: 64 Bit
    - **Basic Auth Publishing Credentials**: Off
    - **Manage pipeline version**: Integrated
    - **FTP state**: Disabled
    - **HTTP version**: 1.1
    - **Web sockets**: Off
    - **Always on**: On
    - **ARR affinity**: Off
    - **HTTPS Only**: On
    - **Minimum TLS Version**: 1.2
    - **Remote debugging**: Off
    - **Client certificate mode**: Ignore
3. Click **Save**.

Web Logging

Enable web server logging to collect diagnostic information from the web server. Logging is optional but highly recommended.

Application logging does not apply to Authorization Server Pro WebApp.

To view logs in a log stream:

1. In the Azure Portal, go to **Monitoring** > **App Service logs**.
2. Set **Web server logging** to **File System**.
3. Enter the **Quote (MB)**.
4. Set the **Retention Period (Days)**.

To preserve web logs:

1. In the Azure Portal, go to **Monitoring** > **App Service logs**.
2. Set **Web server logging** to **Storage**.
3. Select the **Storage** to send the logs to.
4. Set the **Retention Period (Days)**.

Recommended TLS/SSL settings

Azure App Service is created with an SSL certificate by default to provide https and a subdomain, like propricer9webapi.azurewebsites.us.

In the App Service in the Azure Portal, you should verify that the following TLS/SSL settings were selected during configuration:

- **HTTPS Only**: On
- **Minimum TLS Version**: 1.2

Optionally, you can configure your own domain in this section, like mysite.mycompany.com.

Deploy ZIP file using ZipDeployUI

This ZIP file deployment uses the same Kudu service that powers continuous integration-based deployments.

1. In the App Service options pane, select **Advanced Tools**, click **Go**, expand the **Tools** menu, then select **Zip Push Deploy**. Alternatively, in your browser, go to **https://<app_name>.scm.azurewebsites.us/ZipDeployUI**.
2. Upload the Authorization Server Pro WebApp ZIP file from your temporary folder by dragging it to the file explorer area on the web page.

When deployment is in progress, an icon in the top-right corner shows the progress percentage. The page also shows verbose messages for the operation below the explorer area. When it is finished, the last deployment message should say **Deployment successful**.

Alternatively, use az webapp deployment source zip to deploy the ZIP file using Azure CLI, or the Publish-AzWebApp cmdlet to deploy the ZIP file using PowerShell.

Authorization Server Pro Database Setup reference

Authorization Server Pro Database Setup is a command line tool that allows database administrators to create and upgrade Authorization Server Pro databases.

Usage

```
AuthorizationServerProDatabaseSetup [options] [command]
```

Options

`-v|--version` Show version information.

`-?|-h|--help` Show help information.

Commands

`addsysadmin` Adds a system administrator account to an Authorization Server Pro database.

`create` Creates a new Authorization Server Pro database.

`upgrade` Upgrades a database for Authorization Server Pro to the current version.

create command

`create` Creates an Authorization Server Pro database.

Usage

```
AuthorizationServerProDatabaseSetup create [options]
```

Options

`-?|-h|--help` Show help information.

`-f|--scripttofile <fileName>` Output the script to a file.

`-s|--server <servername>` Hostname of the database server.

`-w|--windowsauth` Use Windows authentication.

`-d|--dbname <databasename>` Name of the database. Default value is `AuthServerProDb`

`-al|--adminlogin <login>` Authorization Server Pro admin user email. Default value is `sysadmin@propricer.com`

`-ap|--adminpass <password>` Authorization Server Pro admin user password. Default value is `sysadmin`

`-an|--adminname <name>` Authorization Server Pro admin username. Default value is `System Administrator`

`-u|--dbauser <dbalogin>` SQL Server authentication database user login.

`-p|--dbapass <dbapassword>` SQL Server authentication database user password.

`-e|--useexistingdb` Use an existing database (for example, when using Azure SQL Database).

Examples

Create an Authorization Server Pro database on an on-premises SQL server using Windows authentication:

```
AuthorizationServerProDatabaseSetup create -s sqlserver.mycompany.com -d
AuthServerProDb -al sysadmin@mycompany.com -ap MyPassword4AuthServerPro -an
"System Administrator" -w
```

Create an Authorization Server Pro database on an SQL server on a VM using Windows authentication:

```
AuthorizationServerProDatabaseSetup create -s sqlserver.eastus.cloudapp.azure.com
-d AuthServerProDb -al sysadmin@mycompany.com -ap MyPassword4AuthServerPro -an
"System Administrator" -w
```

Create an Authorization Server Pro database on an Azure SQL database using SQL authentication:

```
AuthorizationServerProDatabaseSetup create -s mycompany.database.windows.net -u
myazureuser -p MyAzurePassword -d AuthServerProDb -al sysadmin@mycompany.com -ap
MyPassword4AuthServerPro -an "System Administrator" -e
```

upgrade command

`upgrade` Upgrades an existing Authorization Server Pro database.

Usage

```
AuthorizationServerProDatabaseSetup upgrade [options]
```

Options

`-?|-h|--help` Show help information.

`-s|--server <servername>` Hostname of the database server.

`-w|--windowsauth` Use Windows authentication.

`-u|--dbauser <dbalogin>` SQL Server authentication database user login.

`-p|--dbapass <dbapassword>` SQL Server authentication database user password.

`-d|--dbname <databasename>` Name of the database. Default value is `AuthServerProDb`

`-f|--scripttofile <fileName>` Output the script to a file.

Examples

Upgrade a database named **AuthServerProDb** on an on-premises SQL server using Windows authentication:

```
AuthorizationServerProDatabaseSetup upgrade -s sqlserver.mycompany.com -d
AuthServerProDb -w
```

Upgrade a database named **AuthServerProDb** on an Azure SQL database using SQL authentication:

```
AuthorizationServerProDatabaseSetup upgrade -s mycompany.database.windows.net -u
myazureuser -p MyAzurePassword -d AuthServerProDb
```

addsysadmin command

`addsysadmin` Creates an administrator-type user in a previously created Authorization Server Pro database. The user will have the Administrator permission to access Authorization Sever Pro but cannot access any ProPricer companion products.

Usage

```
AuthorizationServerProDatabaseSetup addsysadmin [options]
```

Options

`-?|-h|--help` Show help information.

`-s|--server <servername>` Hostname of the database server.

`-w|--windowsauth` Use Windows authentication.

`-d|--dbname <databasename>` Name of the database. Default value is `AuthServerProDb`

`-al|--adminlogin <login>` Authorization Server Pro admin user email. Default value is `sysadmin@propricer.com`

`-ap|--adminpass <password>` Authorization Server Pro admin user password. Default value is `sysadmin`

`-an|--adminname <name>` Authorization Server Pro admin username. Default value is `System Administrator`

`-u|--dbauser <dbalogin>` SQL Server authentication database user login.

`-p|--dbapass <dbapassword>` SQL Server authentication database user password.

Database Setup reference

BOE Pro Database Setup is a command line tool that allows the database administrators to create and upgrade BOE Pro databases.

Usage

`BOEProDatabaseSetup [options] [command]`

Options

`-v|--version` Show version information.

`-?|-h|--help` Show help information.

Commands

`addsysadmin` Add a system administrator account to a BOE Pro database.

`create` Create a new BOE Pro database.

`upgrade` Upgrade a BOE Pro database to the current version.

create command

`create` Create a BOE Pro database.

Usage

```
BOEProDatabaseSetup create [options]
```

Options

`-?|-h|--help` Show help information.

`-f|--scripttofile <fileName>` Output the script to a file.

`-s|--server <servername>` Name of the database server.

`-w|--windowsauth` Use Windows authentication.

`-d|--dbname <databasename>` Name of the database. Default value is `BOEPro`.

`-al|--adminlogin <login>` BOE Pro admin user email. Default value is `sysadmin@propricer.com`.

`-ap|--adminpass <password>` BOE Pro admin user password. Default value is `sysadmin`.

`-an|--adminname <name>` BOE Pro admin username. Default value is `System Administrator`.

`-u|--dbauser <dbalogin>` Database user login.

`-p|--dbapass <dbapassword>` Database user password.

`-e|--useexistingdb` Use an existing database (for example, when using Azure SQL Database).

Examples

Create a BOE Pro database on a SQL server on a VM using Windows authentication:

```
BOEProDatabaseSetup create -s sqlserver.eastus.cloudapp.azure.com -d BOEPro -al
sysadmin@mycompany.com -ap MyStrongPassword4BOEPro -an "System Administrator" -w
```

Create a BOE Pro database on an Azure SQL database using SQL authentication:

```
BOEProDatabaseSetup create -s mycompany.database.windows.net -u myazureuser -p
MyAzurePassword -d BOEPro -al sysadmin@mycompany.com -ap MyStrongPassword4BOEPro
-an "System Administrator" -e
```

upgrade command

`upgrade` Upgrade an existing BOE Pro database.

Usage

```
BOEProDatabaseSetup upgrade [options]
```

Options

`-?|-h|--help` Show help information.

`-s|--server <servername>` Name of the database server.

`-w|--windowsauth` Use Windows authentication.

`-u|--dbauser <dbalogin>` Database user login.

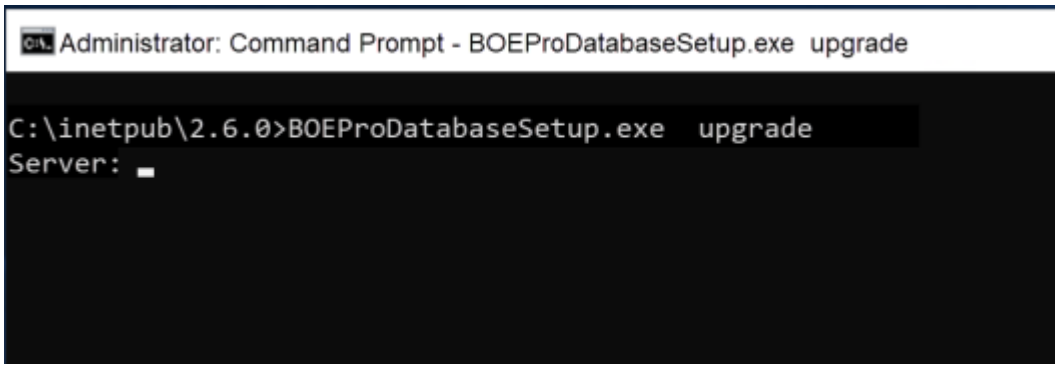
`-p|--dbapass <dbapassword>` Database user password.

`-d|--dbname <databasename>` Name of the database. Default value is `BOEPro`.

`-f|--scripttofile <fileName>` Output the script to a file.

Upgrade a BOE Pro database

1. Download the **BOEProDatabaseSetup** package.
2. Extract **BOEProDatabaseSetup.exe**.
3. Open a command prompt window.
4. Go to the folder where **BOEProDatabaseSetup.exe** is extracted.
5. At the command prompt, use the `upgrade` command to upgrade your database:
   `BOEProDatabaseSetup upgrade`



6. If you need to use Windows authentication, you can add `-w`:
   `BOEProDatabaseSetup upgrade -w`
7. Enter the information requested.

Examples

Upgrade a database named **BOEPro** on a SQL server on a VM using Windows authentication:

```
BOEProDatabaseSetup upgrade -s sqlserver.eastus.cloudapp.azure.com -d BOEPro -w
```

Upgrade a database named **BOEPro** on an Azure SQL database using SQL authentication:

```
BOEProDatabaseSetup upgrade -s mycompany.database.windows.net -u myazureuser -p
MyAzurePassword -d BOEPro
```

addsysadmin command

`addsysadmin` Create an administrator-type user in a previously created BOE Pro database. The user will have the Admin-only role. There can be only one user with the Admin-only role.

Usage

```
BOEProDatabaseSetup addsysadmin [options]
```

Options

`-?|-h|--help` Show help information.

`-s|--server <servername>` Name of the database server.

`-w|--windowsauth` Use Windows authentication.

`-d|--dbname <databasename>` Name of the database. Default value is `BOEPro`.

`-al|--adminlogin <login>` BOE Pro admin user email. Default value is `sysadmin@propricer.com`.

`-ap|--adminpass <password>` BOE Pro admin user password. Default value is `sysadmin`.

`-an|--adminname <name>` BOE Pro admin username. Default value is `System Administrator`.

`-u|--dbauser <dbalogin>` Database user login.

`-p|--dbapass <dbapassword>` Database user password.

References

- Azure App Service
  https://azure.microsoft.com/en-us/services/app-service
- Deploy your app to Azure App Service with a ZIP or WAR file
  https://docs.microsoft.com/en-us/azure/app-service/deploy-zip
- National clouds
  https://docs.microsoft.com/en-us/azure/active-directory/develop/authentication-national-cloud#azure-ad-authentication-endpoints
- Quickstart: Register an application with the Microsoft identity platform
  https://docs.microsoft.com/en-us/azure/active-directory/develop/quickstart-register-app
- Configure a custom container for Azure App Service
  https://learn.microsoft.com/en-us/azure/app-service/configure-custom-container?tabs=debian&pivots=container-linux
- Microsoft Clarity
  https://clarity.microsoft.com