



OpenID Connect with Azure Blob Storage

Manual

Copyright

Copyright © 2020 CAXperts GmbH. All Rights Reserved.

Including software, file formats, and audio-visual displays; may be used pursuant to applicable software licence agreement; contains confidential and proprietary information of CAXperts and/or third parties which is protected by copyright law, trade secret law, and international treaty, and may not be provided or otherwise made available without proper authorisation.

Restricted Rights Legend

Rights reserved under the copyright laws of the Federal Republic of Germany.

Warranties and Liabilities

All warranties given by CAXperts about equipment or software are set forth in your purchase contract, and nothing stated in, or implied by, this document or its contents shall be considered or deemed a modification or amendment of such warranties. CAXperts believes the information in this publication is accurate as of its publication date.

The information and the software discussed in this document are subject to change without notice and are subject to applicable technical product descriptions. CAXperts is not responsible for any error that may appear in this document.

The software discussed in this document is furnished under a licence and may be used or copied only in accordance with the terms of this licence. THE USER OF THE SOFTWARE IS EXPECTED TO MAKE THE FINAL EVALUATION AS TO THE USEFULNESS OF THE SOFTWARE IN HIS OWN ENVIRONMENT.

Trademarks

CAXperts is a registered trademark of CAXperts GmbH. Intergraph, the Intergraph logo, SmartSketch, FrameWorks, SmartPlant, INtools, MARIAN, PDS, IGDS, RIS and IntelliShip are registered trademarks of Intergraph Corporation. IGDS file formats ©1987-1994 Intergraph Corporation. Microsoft and Windows are registered trademarks of Microsoft Corporation. Bentley, the Bentley logo "B," and MicroStation are registered trademarks of Bentley Systems, Inc. ISOGEN is a registered trademark of Alias Limited. Other brands and product names are trademarks of their respective owners.

Table of Contents

I.	Introduction	3
II.	High-level schema for Azure AD	3
III.	Configuration in UniversalPlantViewer.....	4
	authenticationConfig.json	
IV.	Configuration in Azure	7
	Azure Blob storage	
	App registration in Azure Active Directory	
V.	Contact	10
	Helpdesk	

Introduction

OpenID Connect is an authentication protocol built on OAuth 2.0 that allows an application to request access on behalf of an end user.

As this document is only meant to provide a very basic overview, use the official documentation for further information, e.g.: <https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-v2-protocols>

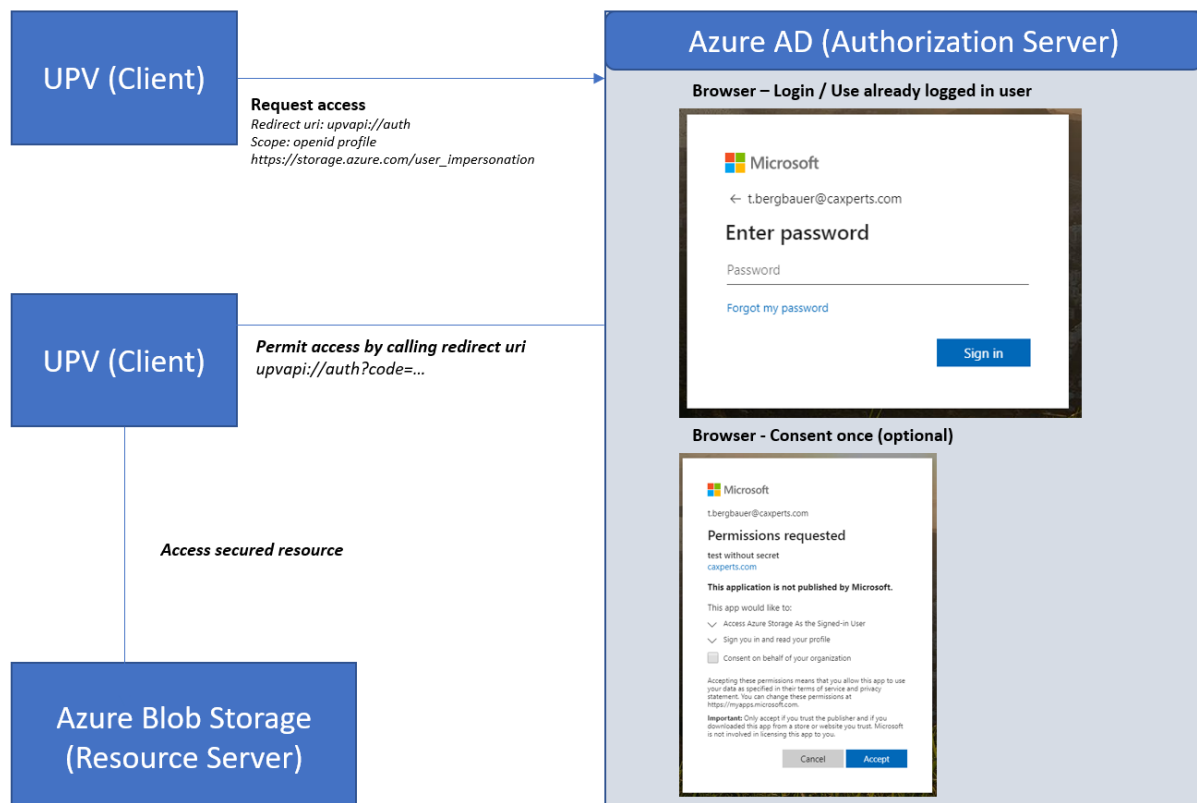
In short – OpenID Connect is a protocol for providing SSO (single sign-on) using the native browser via a common identity provider (the **Authorization Server**). It is built on top of OAuth 2.0 and allows an application to request access to a system on behalf of an end user. The **Client** is the application requesting access to the secured **Resource Server** on behalf of the end user (**Resource Owner**). Of course, every **Authorization Server** supporting OpenID Connect can be used in combination with UniversalPlantViewer. The **Resource Server** needs to support plain file access in a folder hierarchy (opposed to an API like for example in Google Drive).

Supporting specific Resource Server APIs will require additional development effort on CAXperts side and will have to be negotiated separately.

In this example we will specifically address Azure AD in combination with Azure Blob Storage.

The OpenID Connect protocol is supported in UniversalPlantViewer **version 05.02** and upwards.

High-level schema for Azure AD



Conceptually the UniversalPlantViewer does not handle any user secret. The **Authorization Server** handles the login independently and will issue a token which can be used by UniversalPlantViewer to access the gated resource.

Therefore, it is not possible for the **Client** (UniversalPlantViewer) to leak a user password. Only the token could be a target for attacks and when stolen be used to access the **Resource Server** for a limited time.

Refer to the **Authorization Server** documentation for further security measures for example Multi-Factor Authentication or specific authorization rules.

Configuration in UniversalPlantViewer

The UniversalPlantViewer uses the IdentityModel.OidcClient library internally (currently version 2.9.0.0). Please refer to the library's options.

The following extensions to the library's options are available:

- **FrontChannelExtraParameters**: set additional request query parameters for the auth request (i.e. the resource parameter)
- **BackChannelExtraParameters**
- **AuthorizationHeaders**: set static header values which should be sent when authenticated
- **EnableLog**: when set will include the OidcClient2-Log into the application

The configuration file example for Azure – Missing configuration options will be using the libraries default value.

You should set the **EnableLog** property while setting up the configuration. Authentication problems will then be logged to the general UniversalPlantViewer log.

When finished it is advisable to disable this setting so security relevant information is not persisted in the log file.

authenticationConfig.json

```
{
  "Oidc": {
    "FrontChannelExtraParameters": {
      "resource": https://upv.blob.core.windows.net
    },
    "AuthorizationHeaders": {
      "x-ms-version": "2017-11-09"
    },
    "EnableLog": true,
    "Authority": "https://sts.windows.net/e1765baf-2fe4-4b9e-87ee-d130dde50afa",
    "ClientId": "5c2f88a0-766f-464d-bdad-d40ec1d4fb33",
    "Scope": "openid profile offline_access https://storage.azure.com/user_impersonation",
  }
}
```

```

    "RedirectUri": "upvapi://auth",
    "RefreshDiscoveryDocumentForLogin": true,
    "RefreshDiscoveryOnSignatureFailure": false,
    "ResponseMode": 1,
    "LoadProfile": false,
    "Flow": 0,
    "TokenClientAuthenticationStyle": 1,
    "Policy": {
      "Discovery": {
        "ValidateEndpoints": false
      },
      "RequireAccessTokenHash": false,
      "RequireIdentityTokenOnRefreshTokenResponse": false,
    },
  },
}

```

The marked options will require to be adjusted to your case:

- **Resource:** The name of the blob storage
- **Authority:** The Microsoft authorization endpoint for your company using e.g. the Directory (tenant) ID below – please refer to Microsoft documentation for further options
- **ClientId:** The identifier for the application registration – Application (client) ID below

Azure-specific settings:

- **ValidateEndpoints:** set to false when the endpoint is on a different host than the authority
- **AuthorizationHeader:** x-ms-version – determines the Azure API version
- **RequireAccessTokenHash:** false
- **LoadProfile:** false (not required by UniversalPlantViewer – needs additional permission)

You can find the necessary configuration IDs in the app registrations overview:

The screenshot shows the Azure Portal interface for an application registration named "test without secret". The left-hand navigation pane includes links for Overview, Quickstart, Manage, Branding, Authentication, Certificates & secrets, and Token configuration (preview). The main content area displays the following details:

Display name	test without secret	Supported account types	My organization only
Application (client) ID	5c2f88a0-766f-464d-bdad-d40ec1d4fb33	Redirect URIs	0 web, 1 public client
Directory (tenant) ID	e1765baf-2fe4-4b9e-87ee-d130dde50afa	Application ID URI	Add an Application ID URI
Object ID	2c0b9e1a-cf81-462e-952b-8524d76ec7b9	Managed application in local directory	test without secret

The configuration file will be recognized automatically in the Data folder of the UniversalPlantViewer model at **Data\authenticationConfig.json**. It needs to be accessible without authentication (anonymous access).

It is not possible to set single files to anonymous access in Azure Blob Storage.

As a workaround the file can be stored at any other location for example in a separate blob storage container with global anonymous access. The name of the file does not matter in this context.

The user will have to open the model using an upvapi:// link like below once. The configuration will be persisted on the user device and automatically recognized for the next attempts in the specified domain.

```
upvapi://https://upv.blob.core.windows.net/$web/demoplantModel/?CMD!SetAuthConfig=
https%3A%2F%2Fupv.blob.core.windows.net%2F%24web%2FdemoplantModel%2F!https%3A%2F%2
Fupv.blob.core.windows.net%2FauthenticationConfig.json
```

- 1) [https://upv.blob.core.windows.net/\\$web/demoplantModel/](https://upv.blob.core.windows.net/$web/demoplantModel/)
→ Model location for opening directly over the link
- 2) SetAuthConfig (Parameter 1)
`https%3A%2F%2Fupv.blob.core.windows.net%2F%24web%2FdemoplantModel%2F`
→ Domain where the configuration is relevant (subroutes will use the provided configuration when no other is specified)
- 3) SetAuthConfig (Parameter 2)
`https%3A%2F%2Fupv.blob.core.windows.net%2FauthenticationConfig.json`
→ Path to the configuration file (needs to be accessible without authentication) – Supports file/web or URI paths

The parameters of SetAuthConfig need to be UrlEncoded.

Configuration in Azure

Azure Blob storage

For a general overview see <https://docs.microsoft.com/en-us/azure/storage/blobs/>

Set up a storage account

<https://docs.microsoft.com/en-us/azure/storage/common/storage-account-create?tabs=azure-portal>

Set up the containers

<https://docs.microsoft.com/en-us/azure/storage/blobs/storage-quickstart-blobs-portal>

You will need a container for the configuration file – Set the access level to anonymous.

Upload the UniversalPlantViewer configuration file here.

You will need another container for the secured UniversalPlantViewer model data – Set the access level to private.

Uploading multiple files/folders can be accomplished using the Azure Storage Explorer.

<https://docs.microsoft.com/en-us/azure/vs-azure-tools-storage-explorer-blobs>

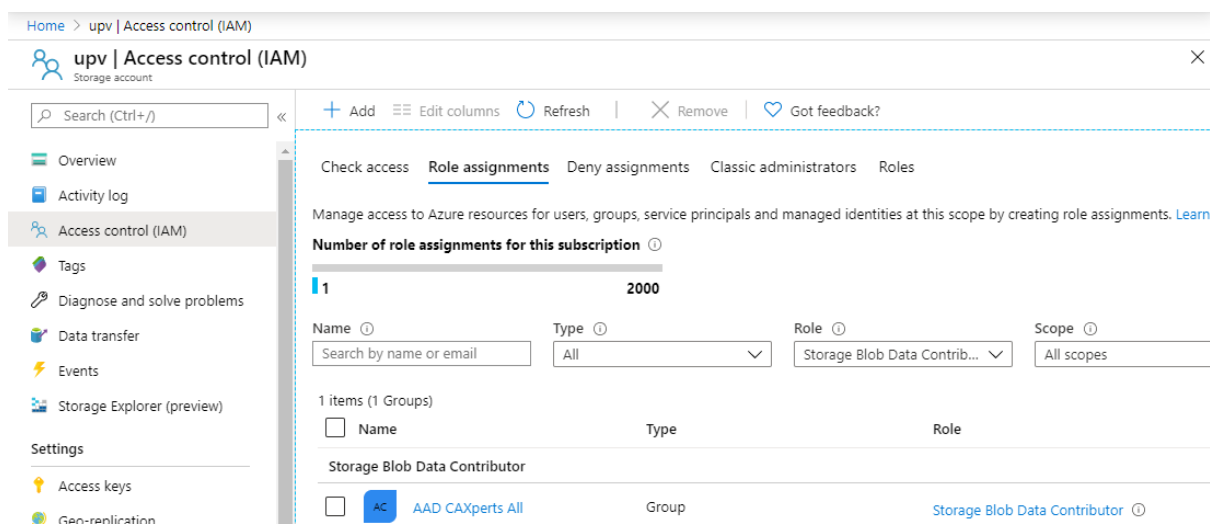
Add user permissions

<https://docs.microsoft.com/en-us/azure/storage/common/storage-auth-aad-rbac-portal>

For the container you will have to set up user and group permissions.

For allowing delegated access to the storage account you will have to set the **Storage Blob Data Reader** role for read-only access.

Depending on your needs you can of course use other roles in the **Storage Blob Data** group.



The screenshot shows the Azure Access control (IAM) interface for a storage account named 'upv | Access control (IAM)'. The left sidebar contains navigation links: Overview, Activity log, Access control (IAM) (selected), Tags, Diagnose and solve problems, Data transfer, Events, Storage Explorer (preview), Settings, Access keys, and Geo-replication. The main pane displays the 'Role assignments' tab. It includes a search bar, a table of role assignments, and a 'Number of role assignments for this subscription' bar showing 1 out of 2000. The table has columns for Name, Type, Role, and Scope. One role assignment is listed: 'Storage Blob Data Contributor' assigned to the 'AAD CAXperts All' group with the scope 'All scopes'.

Name	Type	Role	Scope
Storage Blob Data Contributor	Group	Storage Blob Data Contributor	All scopes

App registration in Azure Active Directory

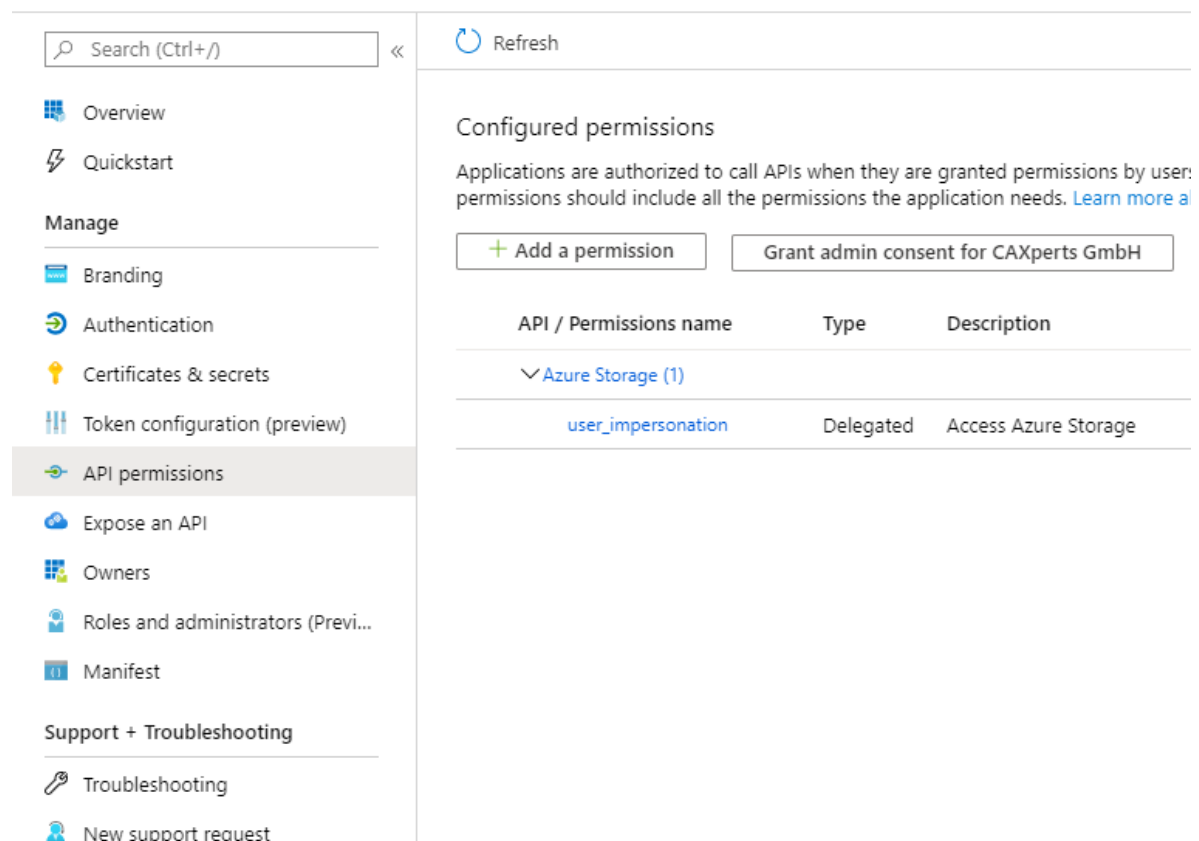
<https://docs.microsoft.com/en-us/azure/active-directory/develop/quickstart-register-app>

Set Redirect URI to upvapi://auth

Redirect URI must match the one provided in the configuration file.

A secret is not necessary as UniversalPlantViewer is a public client.

Add required permission



Search (Ctrl+ /) << Refresh

Overview
Quickstart

Manage

Branding
Authentication
Certificates & secrets
Token configuration (preview)
API permissions
Expose an API
Owners
Roles and administrators (Previ...
Manifest

Support + Troubleshooting

Troubleshooting
New support request

Configured permissions

Applications are authorized to call APIs when they are granted permissions by user: permissions should include all the permissions the application needs. [Learn more about permissions](#)

+ Add a permission Grant admin consent for CAXperts GmbH

API / Permissions name	Type	Description
▼ Azure Storage (1)		
user_impersonation	Delegated	Access Azure Storage

The UniversalPlantViewer needs to access the storage on behalf of the user, therefore add following permission to the registration: **Azure Storage – user_impersonation**

Optionally you can grant admin consent so the end user will no longer have to consent independently.

Add relay page (optional)

In Chrome we observed some calls to upvapi:// being silently ignored when the browser does not have focus which can for example happen when the user is already logged in.

Chrome regards this as a security feature.

As an additional problem there is no indicator in the browser when the login is finished and it is possible to close the browser window.

As a workaround to this problem, it is possible to use a relay webpage as an additional Redirect URI (see attached example site).

Type	Redirect URI
Public client/native (mobile & desktop)	https://upv.blob.core.windows.net/public/loginRedirect.html
Public client/native (mobile & desktop)	upvapi://auth
Web	e.g. https://myapp.com/auth

The redirect URI in the UniversalPlantViewer configuration file will have to be adjusted accordingly.

loginRedirect.html (example code)

```
<html>

  <head>
    <title>UniversalPlantViewer Login</title>
  </head>

  <body>
    <p>
      UniversalPlantViewer login should be succeeded <br/>
      You can close this window now
    </p>

    <p> If there was a problem, you can reissue the login request with
      below link </p>
    <a id="#authLink">Login</a>
  </body>

  <script>
    var url = 'upvapi://auth' + window.location.search;

    function login() {
      window.location.assign(url);
    }

    window.onload = function() {
      var link = document.getElementById('#authLink');

      link.href = url;

      login();
    }
  </script>

</html>
```

Case sensitivity of urls

Urls in Azure Blob Storage are – unlike other systems - case sensitive. Therefore your UPV model needs to have unchanged file casings.

Regarding additional configuration files following casing needs to be considered:

- defaultConfig.upv
- authenticationConfig.json

- volumes.xls/xlsx
- attributes.xls/xlsx
- intelliPidAttributes.xls/xlsx
- attributeData.xlsx
- intelliPidAttributeData.xlsx
- links.xls/xlsx/txt
- intelliPidLinks.xls/xlsx/txt
- defaultPackages.json/xlsx
- upvcolorindex.txt
- ReportDefinition.xls/xlsx
- intelliPidReportDefinition.xls/xlsx
- upvobjectsindex.txt
- upvsketchitemindex.txt
- upvintellipidsketchitemindex.txt
- Measurements.xlsx
- Endpreps.xlsx
- projectMessage.png/jpg/jpeg

Contact

Contact CAXperts' support by email, online, or phone:

CAXperts GmbH
Carl-Zeiss-Ring 4
85737 Ismaning
Germany

<https://www.caxperts.com/contact/>
Phone: +49 (89) 969772-0
Email: info@caxperts.com

Helpdesk

Available Monday to Friday 08.00 a.m. – 5.00 pm (UTC +1)
Phone: +49 (89) 969772-250
support@caxperts.com