PI Web API 2017

User Guide

OSIsoft, LLC
1600 Alvarado Street
San Leandro, CA 94577 USA
Tel: (01) 510-297-5800
Fax: (01) 510-357-8136
Web: http://www.osisoft.com

PI Web API 2017 User Guide

Version: 1.9

Published: 05 May 2017

Contents

PI Web API overview

PI Web API is a RESTful PI System access layer that provides a cross-platform programmatic interface to the PI System. RESTful interfaces enable cross-platform development of web, desktop, and mobile applications across many different programming languages. PI Web API enables you to retrieve and manipulate time series data from the PI Data Archive, and asset and event frame data from the PI AF server.

PI Web API belongs to the OSIsoft PI Developer Technologies family of products, designed to support both the implementation of custom PI System applications and the integration of PI System data with other applications and business systems, such as Microsoft Office or SQL Server, Enterprise Resource Planning systems (ERPs), reporting and analytics platforms, web portals, geospatial and maintenance systems, and so on. The PI Developer Technologies suite covers a wide range of use cases in various environments, programming languages, operating systems and infrastructures.

PI Indexed Search

PI Web API provides PI Indexed Search to enable you to develop a powerful search experience using this RESTful API. PI Indexed Search finds assets by using multiple fields, such as description, template, category, point source, unit of measure, and other metadata. For more introductory details, see Indexed Search. For complete information, access PI Web API Indexed Search: Getting Started (https://techsupport.osisoft.com/Documentation/PI-Web-API/help/topics/pi-web-api-indexed-search_-getting-started.html) on OSIsoft Tech Support, or in the PI Web API Reference after you have completed PI Web API installation.

Bearer authentication

Beginning with the PI Web API 2017 release, claims-based authentication is supported, with the OpenID Connect protocol using access tokens in the authorization header. The access token either directly contains the claims as a JSON Web Token (JWT) or from calling the identity provider's UserInfo endpoint.

OpenID Connect is a protocol that is implemented by various identity providers and allows clients to obtain claims about the logged in user. Information about OpenID Connect can be found at OpenID Foundation (http://openid.net/ (http://openid.net/)) with detailed information at OpenID Specifications and Developer Information (http://openid.net/developers/specs/ (http://openid.net/developers/specs/)).

PI Web API requires that a User Principle Name UPN claim be returned from the identity provider for to get a Windows identity using the Claims to Windows Token Service (C2WTS). The Windows identity is then used as with other authentication schemes for connecting to the AF and Data Archive servers. The PI Web API can be configured to use another claim type for the UPN like the email address claim type if the identity provider cannot easily be configured to return different claims.

The following JavaScript sample shows how the request is made from a browser by adding the access token to the Authorization header:

```
var url = "https://localhost/piwebapi/assetservers";
var xhr = new XMLHttpRequest();
xhr.open("GET", url); xhr.onload = function () {
    log(xhr.status, JSON.parse(xhr.responseText));
}
```

```
xhr.setRequestHeader("Authorization", "Bearer " + user.access_token);
xhr.send();
```

Support for FIPS Community Technology Preview (CTP)

Beginning with the PI Web API 2016 R2 release, both PI Web API service and PI Web API Crawler service support Windows environments that enforce the use of U.S. Federal Information Processing Standards (FIPS) cryptographic algorithms.

Shared Index CTP

Beginning with the PI Web API 2016 release, the Shared Index CTP feature is included for use with PI Indexed Search. It enables multiple installations of PI Web API, each with its own PI Indexed Search, to share access to a single common index, and thereby avoid having multiple Crawlers waste resources by performing duplicate work against the same PI Data Archive and PI AF servers. With the Shared Index CTP enabled, PI Indexed Search builds and maintains a primary searchable index, and other PI Web API instances have read-only access to that primary index.

For more information on other PI Developer Technologies products and about licensing these products in your environment, please contact your account manager.

PI Web API installation

The PI Web API installer has a graphical user interface, which enables you to perform an installation or upgrade of PI Web API. At the end of an installation or upgrade, the PI Web API Admin Utility is automatically launched. You must follow all the steps in the PI Web API Admin Utility to complete the installation or upgrade. For more information, see Configuration with PI Web API Admin Utility.

Silent installation

A `silent.ini` file is included in the PI Web API setup kit wrapper if you prefer to perform a silent installation. For more information, see PI Web API silent installation.

Topics in this section

- System requirements
- Configure Claims to Windows Token Service
- Install PI Web API
- PI Web API silent installation
- Upgrade PI Web API
- Change, repair, or remove PI Web API installation

System requirements

Refer to the PI Web API 2017 Release Notes for details on operating systems, server platforms, and other system requirements.

Note:
OSIsoft recommends you create the PI Web API application data folder on a non-system drive, if one is available. The size of the application folder can grow to be very large, which can cause Windows performance degradation if it resides on the system drive.

Configure Claims to Windows Token Service

You enable bearer authentication with OpenID Connect to obtain a user principal name (UPN) claim for the user. The Claims to Windows Token Service is used to get the Windows identity.

Before you start

You only need to configure Claims to Windows Token Service (C2WTS) if you intend to use Bearer authentication with PI Web API.

Procedure

1. Enable the Claims to Windows Token Service feature.

a. Open Microsoft Server Manager, and on the dashboard click **Add roles and features**.

b. In the Add Roles and Features Wizard, go to the Features page and select **Windows Identity Foundation 3.5**.

c. Click **Next** and then **Install.**

d. Follow the instructions on the Microsoft support page How to configure Claim to Windows Token Services in SharePoint 2010 with Kerberos Authentication (https://support.microsoft.com/en-us/help/2722087).

Tip:
As a best practice, OSISoft recommends you run C2WTS using a dedicated service account and not as Network Service.

2. Edit the Claims to Windows Token Service configuration file.

   a. In a text editor, open `C:\Program Files\Windows Identity Foundation\v3.5` named `c2wtshost.exe.config`.

   b. In the `allowedCallers` section, add this entry to the list:

```
<add value="NT SERVICE\piwebapi" />
```

Note:
If the PI Web API service identity is different, change to use that identity.

   c. Save and close the configuration file.

3. Change the Claims to Windows Token Service startup.

   a. As administrator, open a command window and run this command:

```
sc config "c2wts" depend= CryptSvc
```

   b. Open the Services Control Panel and select **Claims to Windows Token Service.**

   c. Change the startup type to **Automatic (Delayed Start)**.

   d. Save changes and start the Claims to Windows Token Service.

Install PI Web API

Before you start

Verify that:

- The system you plan to use is running a supported operating system. See the PI Web API 2017 Release Notes.
- You can run the installer as an Administrator.

Procedure

1. Go to the directory where you downloaded the PI Web API install kit.

   a. Double-click the installation kit.

You may be prompted by a User Account Control message to allow the installation to run. Click **Yes** to allow the installation to continue.

b. In the Self-Extracting Executable window, click 📁, select the directory where you want to extract the files, and click **OK**.

The files are extracted, and the Welcome window opens. A list displays all the modules that will be installed.

2. Review the list of modules and scroll over to the comments to ensure there are no warnings displayed.

   Click **OK** to continue.

3. When the PI Web API Setup wizard opens, click **Next** to continue to the Custom Setup window.

4. Select the components to be installed. The **PI Web API** component has a sub-component:

   - **PI Indexed Search**: Resources to provide an indexed search experience in the PI System.

   The **PI Indexed Search Crawler** indexes PI Data Archive and PI AF databases for quicker data searches.

   a. Optional. To change the PI Web API installation folder from the default location (`C:\Program Files\PIPC\WebAPI\`), select a primary component, click **Browse** and select a different location in the Change destination folder window.

   b. Click **Next** when finished.

5. Select a directory where application data, such as PI Search indexes, will be stored. The default directory is `C:\ProgramData\OSIsoft\WebAPI`.

   a. Optional. To store in a different directory, click **Change** and select a different location in the Change destination folder window.

   b. Click **Next** to continue.

6. In the PI Web API Admins Group window, create a dedicated administrator group to administer PI Web API or select an existing group.

   - Recommended. Select **Create and use "PI Web API Admins" group**.
   - Select **Use an existing local group** and click a group on the existing group list.

7. Click **Install** to begin the installation, or **Back** to review or change your installation setup settings.

8. When the installation is complete, click **Finish** to close the PI Web API Setup wizard.

   The Change PI Web API Installation Configurations window opens so you can configure additional settings required by PI Web API. For more information, see Configure installation settings.

9. In the Installation Complete window, you can review the status of the installed modules. Click **Close** to end the installation.

PI Web API silent installation

A `silent.ini` file is included in the PI Web API setup kit wrapper. To invoke a silent installation, you enter:

```
setup.exe -f silent.ini
```

Note:
The OSIsoft pre-requisite kit should be installed on a target system before you silently install PI Web API.

If you make no changes in the `silent.ini` file, default installation setting values are used to install PI Web API. You can, however, customize the `silent.ini` file to specify the installation folder, the application data folder, and other configuration settings with the following parameters:

ADMINS_GROUP_TYPE

You specify the type of the administrator group.

- Use `Default` to indicate that you wish to create and use a `PI Web API Admins` group as the administrator group for PI Web API.
- Use `Existing` to indicate that you wish to use an existing Windows group as the administrator group for PI Web API.

Example:

```
ADMINS_GROUP_TYPE=Existing
```

ADMINS_GROUP_NAME

You specify an existing administrator group when **ADMINS_GROUP_TYPE** is set to `Existing`.

Example:

```
ADMINS_GROUP_NAME=Administrators
```

INSTALLATION_DIR

You enter the full path to the directory where you want PI Web API to be installed, in the following format:

```
INSTALLATION_DIR=”c:\path\to\installation\folder”
```

If there are no spaces in the path, you can omit the double quotes around the path.

DATA_DIR

You enter the full path to the directory where you want PI Web API application data to be installed, in the following format:

```
DATA_DIR=”d:\path\to\data\folder”
```

You can also specify a directory using a UNC path, for example:

```
DATA_DIR="\\MyFileServer\appdata"
```

If there are no spaces in the path, you can omit the double quotes around the path. OSIsoft recommends you create the PI Web API application data folder on a non-system drive, if one is available. The size of the application folder could grow to be very large, and it might cause Windows performance degradation if the application folder resides on the system drive.

CONFIG_FILE

You can change other installation settings, such as the configuration asset server name, the configuration instance name, the listen port number, and so on, from the default values. The configuration file normally created by the installation is called `InstallationConfig.json`, which can be found in the PI Web API installation directory.

```
CONFIG_FILE=”c:\path\to\config\file”
```

Refer to Options for CONFIG_FILE parameter for details on the content of the configuration file.

Sample silent.ini file

```
; PI Web API
11 = /qn REBOOT=Suppress
ADMINS_GROUP_TYPE=Existing
ADMINS_GROUP_NAME=Administrators
INSTALLATION_DIR=”c:\path\to\installation\folder”
DATA_DIR=”d:\path\to\data\folder”
CONFIG_FILE=”c:\path\to\config\file”
```

Options for CONFIG_FILE parameter

You define the options for the **CONFIG_FILE** parameter in a Json file.

Refer to the following table for a description of required options.

Option	Description
PI Web API	
ConfigAssetServer	Specifies the name of the PI AF server where configuration data is stored.
ConfigInstance	Specifies the configuration instance. The default value is the same name as the hostname of the server on which PI Web API is installed.
IsFirewallExceptionEnabled	Specifies whether remote clients can access the PI Web API services. The default value is `true`.
IsTelemetryEnabled	Specifies whether you want to participate in the PI System Customer Experience Improvement Program. The default value is `true`.
ListenPort	Specifies the communication port for PI Web API. Port numbers can be between 1 and 65535. The default value is 443.
SslCertificateThumbprint	Represents in hexadecimal notation the SSL certificate that is bound to the communication port. It can be self-signed or purchased from a certification authority.
ApiServiceAccountType	Specifies the Windows NT Service account type for the PI Web API service. The default value is `"Default"`. To change the account type, type `"Custom"`.
ApiServiceAccountUsername	Specifies the Windows NT Service account name for the PI Web API service. The default value is `"NT Service\\piwebapi"`. You can enter a custom account name in `"Domain\\User"` format.
ApiServiceAccountPassword	If you enter a custom account name for the PI Web API service, specifies the password for that account in plain text. **Note:** After configuration is completed, this password does not persist and therefore does not appear in the `InstallationConfig.json` file.
PI Indexed Search Crawler	
CrawlerServiceAccountType	Specifies the Windows NT Service account type for the PI Crawler Windows service. The default value is `"Default"`. To change the account type, type `"Custom"`.
CrawlerServiceAccountUsername	Specifies the Windows NT Service account name for the PI Crawler Windows service. The default value is `"NT Service\\picrawler"`. You can enter a custom account name in `"Domain\\User"` format.

Option	Description
`CrawlerServiceAccountPassword`	If you enter a custom account name for the PI Crawler Windows service, specifies the password for that account in plain text. **Note:** After configuration is completed, this password does not persist and therefore does not appear in the `InstallationConfig.json` file.
`CrawlBuiltInGroups`	*OSIsoft recommends you do not modify this option.* Specifies that built-in Windows SIDs for identity mappings are included. The default value is `true`.
`CrawlerSubmitTimeout`	*OSIsoft recommends you do not modify this option.* Indicates the amount of time in milliseconds before a crawler submission to store results times out. The default value is `1200.0`.
`CrawlerSubmitUrl`	*OSIsoft recommends you do not modify this option.* Specifies the URL that the crawler uses to submit results for storage in the index. The default value is the piwebapi endpoint, for example: `https://MyServer.com/piwebapi/`
`MaxConcurrentCrawlers`	*OSIsoft recommends you do not modify this option.* Specifies the number of Indexed Search crawlers that can be run in parallel. The default value is 2.
`AutoCrawlRebuild`	*OSIsoft recommends you do not modify this option.* Determines whether the index should be rebuilt when PI AF security mappings change. The default value is `true`.
`CrawlerMaxPathCount`	*OSIsoft recommends you do not modify this option.* Specifies the maximum number of paths that the crawler will crawl. This is used to prevent the crawler from never finishing on very large systems. The default value is `1000000`.

Example

```
{
  "ConfigAssetServer": "JSH-C35-WS2012",
  "ConfigInstance": "JSH-C35-WS2012",
  "IsFirewallExceptionEnabled": true,
  "IsTelemetryEnabled": false,
  "ListenPort": 443,
  "SslCertificateThumbprint": "1BE03E3728A77BB11D07B17EAD8DF9A0FEB9CD13",
  "ApiServiceAccountType": "Default",
  "ApiServiceAccountUsername": "NT Service\\piwebapi",
  "CrawlerServiceAccountType": "Default",
  "CrawlerServiceAccountUsername": "NT Service\\picrawler",
  "CrawlBuiltInGroups": true,
  "CrawlerSubmitTimeout": 1200.0,
  "CrawlerSubmitUrl": "https://jsh-c35-ws2012.dev.osisoft.int/piwebapi/",
  "MaxConcurrentCrawlers": 2,
  "AutoCrawlRebuild": true,
  "CrawlerMaxPathCount": 1000000
}
```

Upgrade PI Web API

If an earlier version PI Web API is already installed, the installer will uninstall the earlier version before installing the new one. The installation procedure remains the same.

Note:
The application data folder and the administrator group selection cannot be changed during an upgrade.

Change, repair, or remove PI Web API installation

You can change, repair, or remove a PI Web API installation in two different ways.

Procedure

- If the version of PI Web API that you are installing is the same as one that is already installed, run the `PIWebApi.msi` file of the installed version directly.
- Select the PI Web API program from the list of installed programs in **Control Panel** > **Programs and Features.** You can select **Change**, **Repair**, or **Uninstall**.
    - **Change**

      Select this option to change one or more installation settings. The change options are the same as the installation options. You cannot change the installation or destination folder. To change the installation or destination folder, you should uninstall and then reinstall PI Web API.
    - **Repair**

      Select this option to repair problems with the PI Web API installation, such as restoring missing files or registry values. You cannot change installation settings using this option.
    - **Uninstall**

      Select this option to remove the PI Web API. Uninstalling the product does not remove:
        - Any SSL certificates that were created during the installation process.
        - The binding of the selected SSL certificate to the port used by PI Web API in the Windows HTTP service's configuration.
        - The URL reservation for PI Web API in the Windows Kernel routing table.

PI Web API configuration

PI Web API has two types of configuration.

- As soon as the installation is completed, you use the PI Web API Admin Utility to ensure that the product operates successfully.
- You also use either the RESTful interface or PI System Explorer to control aspects of PI Web API behavior when it is running, such as authentication methods and cross-origin resource sharing (CORS). Configuration information for these properties are stored under a Configuration Instance Name in the configuration database of a PI AF server.

Configuration with PI Web API Admin Utility

When installation is successfully completed, the PI Web API automatically presents a series of configuration options in the Change PI Web API Installation Configurations window.

You need to provide configuration settings for the following components:

Component	Description
`PI Web API`	
Telemetry	Enables OSIsoft to collect anonymous usage data to improve the PI System and to prioritize new features. The collected data does not include business data or logic, but can include information such as IP addresses, host names, and names of PI System objects.
Configuration Store	Specifies the PI AF server and the name of the space in the Configuration database where PI Web API can store operating configurations and persist other data.
Listen Port	Specifies the communication port that PI Web API uses to listen for HTTPS traffic. Also enables a Windows Firewall exception to allow remote clients to access the service when Windows Firewall is enabled.
Certificate	Enables an SSL certificate to be selected or changed to encrypt traffic between the server and clients on the selected listen port. If no SSL certificate is selected or set on the port, PI Web API creates and uses a self-signed certificate.
API Service	Specifies the account under which the API Windows service will run. For integrated Windows security that uses the Kerberos authentication protocol, this account must be trusted for delegation in Active Directory, ideally the default NT Service account.
`PI Indexed Search Crawler`	
Crawler Service	Specifies the account under which the Crawler Windows service operates, with privileges to read all PI System objects for indexing.
Submit Url	Specifies the root URL that the Indexed Search Crawler uses to connect to PI Web API.

PI Web API Admin Utility location

If you later need to change any of the configuration settings you established initially for the above components, you can open the PI Web API Admin Utility manually. This utility is located under **PI System** on the **Start** menu.

Configure installation settings

As soon as the PI Web API installation is completed, configure settings that PI Web API requires to operate correctly.

> **Note:**
> Installation configuration settings are dependent on which features are installed. If the PI Web API Indexed Search crawler is not installed, **Crawler Service** and **Submit Url** settings are not displayed.

Procedure

1. In the **PI System Customer Experience Improvement Program** pane, decide if you wish to participate in the program.
   - To participate, select **Yes, I would like to participate.**

   Click **Next** to continue.
2. In the **Select Operating Configuration Store** pane, select the PI AF server where you wish to store configuration data. The default value is the currently connected PI AF server. To select a configuration space on a different server:
   a. Click ....
   b. In the PI AF Servers window, select a server from the list and click **OK.**
   c. If you are not currently connected to the server, click **Connect.**
3. The default configuration instance is the same name as the hostname of the server on which you installed PI Web API. You can change the instance name, as needed.

   Click **Next** to continue.
4. In the **Select Communication Port** pane, enter a communication port number for PI Web API. You can enter a port number between 1 and 65535 if you do not wish to use the default value of 443.
5. To create a Windows Firewall exception to enable remote clients to access the PI Web API service, leave **Yes, please create a Firewall Exception for PI Web API** selected. You can clear the check box if you wish to set up firewall exceptions manually or wish to prevent remote client access.

   Click **Next** to continue.
6. In the **Select an SSL Certificate for Encrypting Traffic** pane, select or change an SSL certificate for the communications port you just selected.

To ...	Do this ...
Set an SSL certificate	a. Click **Select**. b. In the Windows Security window, select the certificate. To view certificate properties, click the link. c. Click **OK**. d. Click **Next** to continue.

Set a self-signed certificate created by PI Web API	Click **Next**. **Note:** A self-signed certificate is not issued if the port has been bound to another certificate or a certificate has been selected.
View or change a previously set certificate	a. Click the certificate thumbprint to view detailed information about the certificate. b. To change a certificate, click **Change**. c. Click **Yes** in response to the warning that changing the binding certificate can disrupt an existing application. d. In the Windows Security window, select the certificate. To view certificate properties, click the link. e. Click **OK**. f. Click **Next** to continue.

7. In the **Configure PI Web API Windows Service** pane, choose the account under which the installed API Windows Service will operate.

To use a ...	Do this ...
Default NT Service account	a. Leave the **Account Type** field value at `Default`. b. Leave the **Account name** field value at `NT Service\piwebapi`. c. Click **Next** to continue.
Custom account	a. In the **Account Type** field, select `Custom`. b. In the **Account name** field, enter an account name in `Domain\User` format. c. In the **Account password** field, enter a valid password for the account name. d. Click **Test**. ▪ If the test is successful, **Next** is enabled. ▪ If the test failed, click **OK** and enter the correct password. e. Click **Next** to continue.

8. In the **Configure PI Crawler Windows Service** pane, choose the account under which the installed Crawler Windows Service will operate. Choose the default values (`NT Service \picrawler`) or specify a custom account, as described above.

   Click **Next** to continue.

9. In the **Configure the PI Indexed Search Crawler Submit Url** pane, enter the root URL of the PI Web API instance. You need to enter the fully qualified domain name (FQDN), comprised of the domain name and the hostname.

   ◦ The default URL is `https://`*`FQDN_of_local_host`*`/piwebapi`.

   ◦ If the target PI Web API server is not on the machine of the crawler, change the URL to `https://`*`FQDN_of_PI_Web_API_host_machine`*`/piwebapi`.

   Click **Next** to continue.

10. In the **Ready to Apply Changes** pane, review the settings you have just configured.
    - If you are satisfied that the settings are correct, select **Accept all the configurations and click "Next" to apply** and click **Next**.
    - If you need to modify a setting, click **Back** until the setting you wish to change is displayed.

    In the **Applying Changes** pane as the configuration is in progress, you can click **Cancel** if necessary. When all jobs are completed, the **Confirmation** pane is displayed.
11. Click **Finish**.

Configuration at runtime

PI Web API stores configuration data in the PI System. This enables the system to scale horizontally by having multiple PI Web API instances behind a network load balancer share a common configuration space, as described in Cluster management.

You can configure several properties, either in the RESTful interface or in PI System Explorer.

Configuration with the RESTful interface

PI Web API contains resources under `/piwebapi/system/configuration` to view, add, modify, and remove configuration items. Consult PI Web API online help for information about how to interact with these resources.

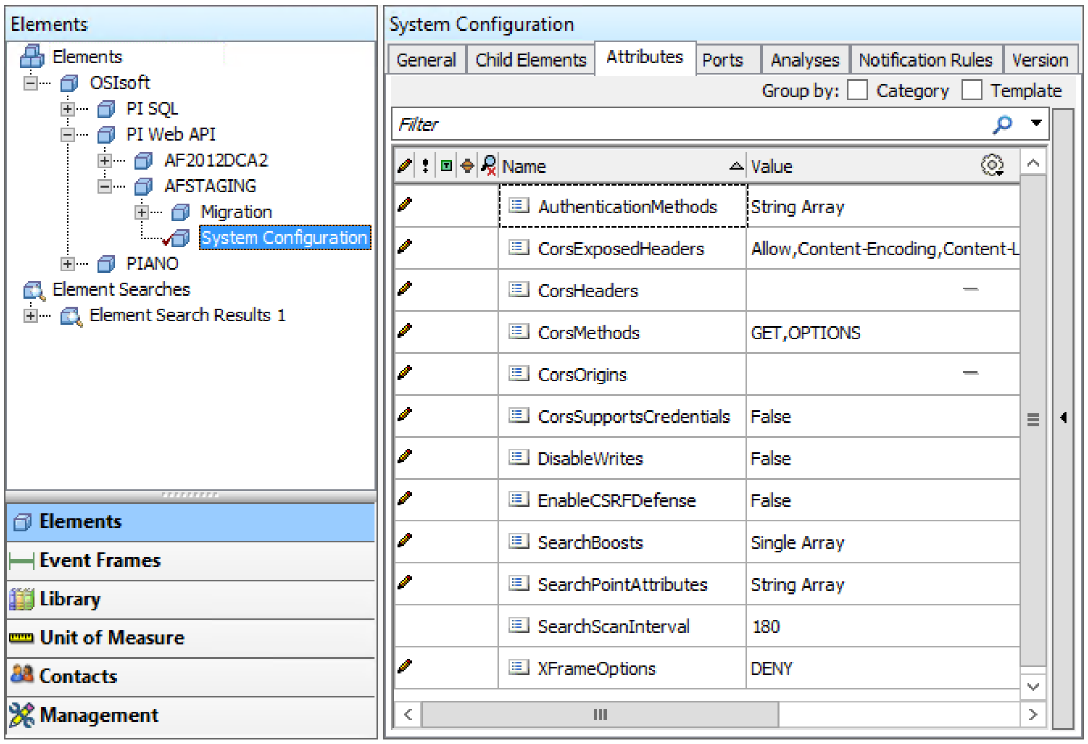
Configuration with PI System Explorer

You can also modify configuration values directly in PI System Explorer. In the **Elements** pane, you navigate to the **Configuration** database on the PI AF server that you selected when installing PI Web API.

In the element hierarchy, the PI Web API configuration is stored under **OSIsoft** > **PI Web API** > ***Configuration Instance Name***. By default, your configuration instance name is the hostname of the server on which you have installed PI Web API.

You add or modify configuration values in the **Attributes** tab on the **System Configuration** element. When completed, use the **Check In** feature to register configuration changes in the PI AF database.

Note:
Not all attributes described in this section are available by default.



Elements

- Elements
  - OSIsoft
    - PI SQL
    - PI Web API
      - AF2012DCA2
      - AFSTAGING
        - Migration
        - System Configuration
    - PIANO
- Element Searches
  - Element Search Results 1

Elements | Event Frames | Library | Unit of Measure | Contacts | Management

System Configuration

General | Child Elements | Attributes | Ports | Analyses | Notification Rules | Version

Group by: ☐ Category ☐ Template

Filter

Name	Value
AuthenticationMethods	String Array
CorsExposedHeaders	Allow,Content-Encoding,Content-L
CorsHeaders	–
CorsMethods	GET,OPTIONS
CorsOrigins	–
CorsSupportsCredentials	False
DisableWrites	False
EnableCSRFDefense	False
SearchBoosts	Single Array
SearchPointAttributes	String Array
SearchScanInterval	180
XFrameOptions	DENY

Sample System Configuration attributes

Authentication methods

PI Web API supports the following authentication methods:

- Kerberos
- Basic
- Bearer
- Anonymous

The PI Web API configuration item **`AuthenticationMethods`** takes an array of authentication methods, and multiple authentication methods can be specified. If multiple methods are enabled, and multiple authentication types are supplied in a single request, many browsers select Basic authentication. However, if Anonymous authentication is one of the many enabled methods, it overrides other authentication types.

Note:
Google Chrome has limitations that prevent browsing the PI Web API reliably. Chrome is *not* supported when both Kerberos and Basic are selected as authentication methods.

- **Kerberos**

  Kerberos is the only authentication mechanism that is enabled by default in PI Web API. Kerberos provides per-user security that is native to Windows and Active Directory, and that is well supported in Microsoft clients. Kerberos does not rely on credentials being transmitted across the wire, which makes it ideal for use with untrusted connections.

Use of Kerberos authentication requires the correct configuration of Active Directory delegation for the account hosting the PI Web API service in Active Directory. Correctly configured delegation requires that an Active Directory Domain Administrator grant delegation privileges to the account hosting PI Web API (or in the case of the default virtual service account, to the computer account of the computer hosting PI Web API). Correctly configured delegation also requires that Service Principal Names be correctly set on the account hosting PI Web API. Refer to Commonly encountered problems for detailed steps for resolving issues related to Kerberos delegation.

- **Basic**

  Basic authentication is defined in the Request for Comments document RFC 2617 HTTP Authentication (https://www.ietf.org/rfc/rfc2617.txt) and is widely supported across vendors, platforms, and HTTP clients. Basic authentication as implemented in PI Web API is simple to use, provides granular, per-user security based on Windows identity, and can help avoid configuration problems like those related to Kerberos delegation. When combined with SSL, as in all PI Web API requests, Basic authentication is reasonably secure.

  Basic authentication is nevertheless less secure than Kerberos, since user credentials are transmitted across the wire. In addition, Basic authentication requires that PI Web API keeps the decrypted username and password in memory for the duration of the request. Even after the request is completed, the credentials can continue to reside in memory until new data takes its place in memory. You should not use Basic authentication unless you are confident of the security of the server on which you are running PI Web API.

- **Bearer**

  Bearer authentication uses the OpenID Connect protocol to obtain information about the user.

  Clients use an access token obtained from the identity provider in the header to identify the user. The access token is used to obtain a UPN claim, which is then mapped to a Windows identity using the Claims to Windows Token Service configured on the PI Web API machine.

- **Anonymous**

  Anonymous indicates no authentication at all on requests to the PI System from the PI Web API. All requests against PI Web API that use Anonymous authentication are served using the Windows account that hosts PI Web API (by default, the virtual service account `NT Service\piwebapi`).

  OSIsoft discourages use of the Anonymous authentication setting. When Anonymous authentication is enabled, OSIsoft strongly recommends the use of Read-only mode for PI Web API.

Cross-Origin Resource Sharing

Cross-Origin Resource Sharing, or CORS, is a W3C-recommended mechanism that enables restricted resources (such as stylesheets, images and fonts) on a web page to be requested securely from another domain outside the domain from which the resource originated.

All modern web browsers enforce a Same-Origin Policy, which limits scripts running in one domain from accessing resources that are hosted in another. For example, a script that is running at `http://some-questionable-site.com` can only make AJAX (asynchronous

JavaScript and XML) calls to the `some-questionable-site.com` domain; it cannot make requests to `http://your-bank.com`. The Same-Origin Policy is a cornerstone of internet security and helps protect against a wide array of attacks that would otherwise be possible.

Unfortunately, this means that modern, trustworthy applications that are running in one domain cannot make AJAX requests to resources that are hosted in another domain. This presents a problem if, for example, you are developing an application that will run at `http://my-internal-site.internal/mygreatapp` and you are trying to make AJAX calls to the PI Web API, which is running at `https://my-pisystem.internal/piwebapi`.

The solution is to enable CORS in the PI Web API configuration, and indicate the origins from which PI Web API accepts cross-domain requests. All modern browsers support CORS as a workaround for the same-origin policy.

More information about CORS is available at W3C Recommendation on Cross-Origin Resource Sharing (http://www.w3.org/TR/cors/). Information about its use in ASP.NET Web API is available in an MSDN magazine article titled ASP.NET Web API - CORS Support in ASP.NET Web API 2 (http://msdn.microsoft.com/en-us/magazine/dn532203.aspx).

The following CORS configuration items are available in PI Web API:

CorsHeaders

A comma-separated list of HTTP header keys that are allowed for cross-domain requests. The client browser enforces this restriction.

The default value of `null`, or an empty string, indicates that no HTTP headers can be sent with the request. The client browser suppresses any HTTP headers that you attempt to set manually.

The asterisk (`*`) character is a wildcard, and means that the client browser is free to forward any HTTP headers with the request.

If you are making requests that include content, for example creating or updating a resource in PI Web API, you can add the `Content-Type` header as an allowed header in the **`CorsHeaders`** property.

CorsMethods

A comma-separated list of HTTP methods (verbs) that are allowed for cross-domain requests. PI Web API inspects the provided HTTP method against the list of allowed values to make an authorization decision.

The default value, `GET,OPTIONS`, means that only HTTP calls that use one of those two HTTP methods will be accepted. This limits cross-domain requests to read access only, for enhanced security.

A value of `null`, or an empty string, indicates that cross-domain requests cannot be accepted with any HTTP method. This means that CORS is effectively disabled.

The asterisk (*) character is a wildcard, and indicates that cross-domain requests can be accepted using any HTTP method.

CorsOrigins

A comma-separated list of domains from which CORS requests may originate. The originating browser should automatically send the domain in the HTTP `Origin` header of the cross-domain request. PI Web API inspects the `Origin` header of the request against the list of allowed origins to make an authorization decision.

The default value, `null` or an empty string, indicates that cross-domain requests cannot be accepted from any origin. This means that CORS is effectively disabled.

The asterisk (*) character is a wildcard, and indicates that cross-domain requests can be accepted from any origin. This configuration is not recommended.

`CorsExposedHeaders`

A comma-separated list of HTTP response headers that browsers are allowed to access, in addition to the simple response headers that are exposed by the browser by default. The following simple response headers are available by default:

- Cache-Control
- Content-Language
- Content-Type
- Expires
- Last-Modified
- Pragma

To make other headers available to an application, include them in a list after the **`CorsExposedHeaders`** attribute. For example, if you are creating an element using **`CreateElement (link)`** and would like to obtain the URL resource for the newly created element from the Location header, specify the Location header explicitly in **`CorsExposedHeaders`**.

`CorsSupportsCredentials`

A Boolean property that indicates whether or not the browser can send and receive authentication information along with a cross-origin request and response.

The default value, `false`, means that authentication information is not sent with the request and will not be accepted in the response. If you are running PI Web API with an authentication type other than `Anonymous`, PI Web API returns a `401 Unauthorized` response to cross-origin requests if the browser includes user credentials in the request.

The value `true` means that the browser can send authentication information with the request, and receive it in the response. Per the CORS specification, **`CorsSupportsCredentials`** cannot be used in conjunction with a wildcard (*) value for **`CorsOrigins`**. The list of allowed origins must be explicitly defined when **`CorsSupportsCredentials`** is enabled.

Bearer authentication settings

OpenID Connect can be used with claims embedded in the access token as a JWT, or from claims obtained from the identity provider's UserInfo endpoint as configured by an administrator. When configured for bearer authentication, PI Web API supports access tokens in the Authorization header for requests.

When using an identity provider that cannot be configured with CORS to return metadata for browser based calls, the PI Web API can act as a relay to get the configuration and JSON Web Key Set (`jwks`) information from the identity provider. The browser can obtain these by configuring the **`BearerIssuer`** setting.

The following bearer settings are available in the PI Web API, although they are not present by default.

`BearerIssuer`

Identifies the URL to the authentication provider, for example, `https://sts.windows.net/{Guid}/`. Also allows the PI Web API to relay calls for metadata information from the identity provider. The URL and the client ID that is needed for client side authentication is provided by the administrator of the identity provider.

The default value is null. This setting is required for all OpenID Connect configurations.

`BearerEnableJwt`

This Boolean setting indicates whether the access token is a JWT containing the embedded UPN claim for the Windows identity. The identity provider administrator determines if a JWT is returned when the client application is created. If the access token does not have embedded claims, the OpenID Connect standard UserInfo endpoint of the identity provider is used to obtain the claims for the user.

The default value is false. This setting is true if the access token is a JWT.

`BearerValidAudiences`

This setting is used with a JWT to verify that it is valid for the PI Web API server. If the setting is missing, no check is made. It is usually a URL of the PI Web API machine.

The default value is null. If the setting is present, it must match the audience in the JWT.

`BearerValidIssuer`

This setting is used with a JWT to verify that the issuer (identity provider) is valid for the PI Web API server. If the setting is missing, no check is made. The URL may be different from the **`BearerIssuer`** setting.

The default value is null. If the setting is present, it must match the issuer in the JWT.

`BearerUpnClaimType`

This setting is used to change the claim type for the UPN that is mapped to a Windows identity. If the setting is missing, the default is `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn`.

Read-only mode

PI Web API supports a read-only mode, whereby the ability to modify your PI System is completely disabled. This is especially desirable if the **`AuthenticationMethods`** parameter is configured for `Anonymous` authentication.

The following configuration item is available in PI Web API:

`DisableWrites`

Set the configuration item in PI System Explorer with a **Value Type** of `Boolean`. The default setting is `false`. When set to `true`, read-only access is enabled in PI Web API.

Caution:

- Read-only mode prevents the Indexed Search crawler from updating search indexes, which prevents Indexed Search from working properly.
- Read-only mode prevents the Batch controller from working since requests use the POST HTTP method.

Restrictions on request data rate

You can limit the rate at which clients request data by setting rate limit parameters in the Configuration database.

The following configuration items are available in PI Web API:

`RateLimitMaxRequests`

The maximum number of requests per client (identified by IP address) over a specified duration of time. Add or set the configuration item in PI System Explorer:

Field	Description or value
Value Type	`Int32`
Value	Enter a numeric that represents the maximum number of requests per client. Minimum value is `1`. Maximum value is `2,147,483,647`. The default value is `1,000`.

`RateLimitDuration`

The period of time, in seconds, that the client is limited to a specific number of requests. Add or set the configuration item in PI System Explorer:

Field	Description or value
Value Type	`Int32`
Value	Enter a numeric that represents the number of seconds per client. Minimum value is `1`. Maximum value is `2,147,483,647`. The default value is `1`.

The rate limit per client IP address is calculated with the formula:

```
Rate limit per second per IP = RateLimitMaxRequests / RateLimitDuration
```

With default parameter values, this limit resets every single second, meaning that a maximum of 1000 requests can be made per second.

If the rate limit is reached or exceeded, a response is returned with HTTP status `429 (Too Many Requests)`. The response body contains the following error message:

```
Rate limit was reached
```

Restrictions on number of items per call

You can set up a configuration item to restrict the number of items returned per call. This prevents unnecessarily large returns. The type of items returned depends on the nature of the call. For example, your maximum items could consist of elements, attributes, or time series data values depending on how your call is defined.

The following configuration item is available:

`MaxReturnedItemsPerCall`

Restricts the number of items returned per call. Add or set the configuration item in PI System Explorer:

Field	Description or value
Value Type	`Int32`
Value	Enter a numeric that represents the upper bound for the items returned per call. Minimum value is `1`. Maximum value is `2,147,483,647`. The default value is `150,000`.

Responses are paged so long as each page request returns a number of items that are less than the maximum number set by this parameter.

If the maximum number of items per call is reached or exceeded, a response is returned with HTTP status `400 Parameter (name) is greater than the maximum allowed (maximum defined number)`.

Restrictions on request size

You can set up a configuration item to control the size of a request to PI Web API. The following configuration item is available:

`MaxRequestContentLength`

Controls the size of a request to PI Web API. Add or set the configuration item in PI System Explorer:

Field	Description or value
Value Type	`Int64`
Value	Enter a numeric that represents the upper bound for the size of the request in bytes. Minimum value is `0`. Maximum value is `9,223,372,036,854,775,807`. The default value is `4194304` (4 MB).

If a request content size is reached or exceeded, a response is returned with HTTP status `413 (Request Entity Too Large)`.

Search boost settings

PI Web API establishes a default array of weighting values for specific fields, which it uses to determine the priority of fields in search results. You can modify those weighting values as needed.

The following configuration item is available:

`SearchBoosts`

The **`SearchBoosts`** configuration item is set in PI System Explorer with a **Value Type** of `Single Array`. It contains an array of values that are assigned to specific fields, as shown in the following table. Each value is used to weight the priority of the associated field when calculating a score. A higher number implies a higher boost to ranking score than a lower number.

Field	Sequence	Value
Name	0	1
Description	1	0.8
AFCategories	2	0.5
AFElementTemplate	3	0.5
AttributeName	4	0.5
AttributeDescription	5	0.5
AttributeCategories	6	0.5

Note that for fields that are not in the **`SearchBoosts`** array, boosts can be assigned explicitly in a search query. In the following example, the 1.5 value assigns a higher weight to hits on the particular `field:term` pair.

```
?q=(instrumenttag:sql)^1.5 OR (instrumenttag:R)
```

Other security settings

Starting in the 2017 version, you can establish the following additional security settings to prevent Cross Site Request Forgery (CSRF) attacks and improve the protection of web applications against clickjacking.

`EnableCSRFDefense`

Set the configuration item in PI System Explorer with a **Value Type** of `Boolean`. The default setting is `true`. When set to `true`, the Cross-Site Request Forgery defense is enabled in PI Web API, and PI Web API checks whether a custom HTTP request header **`X-Requested-With`** is present with a request, whose method is POST, PUT, PATCH or DELETE. This defense relies on the Same-Origin Policy restriction and CORS settings. Ensure that the value for the **`CorsHeaders`** setting is either the asterisk (*) character or contains **`X-Requested-With`**.

`XFrameOptions`

The X-Frame-Options HTTP response header can be used to indicate whether or not a browser should be allowed to render a page in a `<frame>`, `<iframe>` or `<object>`. Set the configuration item in PI System Explorer with one of the following values:

Field	Description or value
Value Type	`String`

Field	Description or value
Value	• `DENY` The page cannot be displayed in a frame, regardless of the site attempting to do so. • `SAMEORIGIN` The page can only be displayed in a frame on the same origin as the page itself. • `ALLOW-FROM` *`URI`* The page can only be displayed in a frame on the specified origin. • Empty string Protection is turned off.

PI Web API administration

As a PI Web API administrator, you need to ensure that users who access protected PI Web API resources be members of either a specially designated local Windows group or the `PI Web API Admins` group that the install program can create instead. You also may need to manage log collection so that it includes analytic and debug events.

PI Web API administrators

When the setup program is run, PI Web API allows you to designate a local Windows group, or creates a Windows group called `PI Web API Admins`, whose members are the users authorized to call protected resources in the PI Web API service.

Because of the nature of Microsoft Windows group membership, and how such groups are added to the Windows token, the change might not appear to take effect immediately, especially if a user is logged on interactively. In such cases, logging out and back in will produce the desired change in a user's Windows token. You do not need to restart the PI Web API service.

Protected resources

The following resources that are hosted in PI Web API are sensitive, and require that the calling user be a member of the local Windows group designated for the PI Web API service during initial setup, or the `PI Web API Admins` group on the server that hosts PI Web API.

```
GET /piwebapi/system/cacheinstances
PUT /piwebapi/system/configuration/{key}
DELETE /piwebapi/system/configuration/{key}
POST PUT DELETE /piwebapi/search/sources
POST /piwebapi/search/sources/crawl
POST /piwebapi/search/settings
```

Log collection

The PI Web API service is an Event Tracing for Windows (ETW) event source, so any application capable of recording ETW events (such as PerfView) can capture PI Web API logs. By default, the installer registers the PI Web API event source with the Windows Event Log, under **Applications and Services Logs** > **PIWebAPI**. Admin events, which include unhandled service exceptions and other critical errors and messages, are logged by default.

PI Web API also generates verbose analytic and debug events.

Configure log collection of analytic and debug events

When you configure PI Web API to write debug information to analytic and debug logs, you access them with the Microsoft Event Viewer.

Note:
To capture PI Web API logs, you can also use other event logs that are integrated with Event Tracing for Windows (ETW).

Procedure

1. Start the Microsoft Event Viewer.
   a. Click **Start.**
   b. Click **Control Panel.**
   c. Click **System and Security.**
   d. Click **Administrative Tools.**
   e. Select **Event Viewer.**
2. Select **View** > **Show Analytic and Debug Logs.**
3. In the navigation pane:
   a. Expand **Applications and Services Logs.**
   b. Expand **PIWebAPI.**
   c. Click **Analytic.**
4. In the **Actions** pane, click **Enable Log.**

   In response to the warning that analytic and debug logs may lose events when enabled, click **OK.**
5. In the navigation pane, click **Debug.**
6. In the **Actions** pane, click **Enable Log.**

   In response to the warning that analytic and debug logs may lose events when enabled, click **OK.**

PI Web API features

The PI Web API includes several features to customize your user experience.

Cache management

PI Web API is built on top of AF SDK, which is designed to be run within the context of a single user. To support multiple simultaneous users, PI Web API hosts multiple instances of AF SDK. These instances of AF SDK are provisioned automatically by PI Web API. Advanced users might need to understand the mechanisms by which caching occurs.

Cache mechanism

PI Web API allocates a separate cache to each user. Caches are marked for refresh at five-minute intervals. After the five-minute interval has elapsed, or if an appropriate Cache-Control header is sent with the request, the cache is refreshed during the first request that is served by the cache.

Each cache can serve multiple concurrent read requests. In addition, requests to write time-series data to resources under the `/piwebapi/streams` hierarchy execute concurrently with read requests, because no locking is required to fulfill such writes.

With the exception of writes to resources under the `/piwebapi/streams` hierarchy, simultaneous requests to write data to the PI System require exclusive access to the cache, which means that all concurrent read and write requests are queued.

Caches expire if they have not been used for longer than 15 minutes after their most recent refresh. This frees the memory associated with the cache for use by other caches within PI Web API, or for other processes on the system.

Refreshing the cache requires exclusive access to the cache instance. While the cache is refreshing, all requests to read or write from the cache are queued.

Note:
Refreshes of AF cache are resource-intensive and should be used with care.

Cache-Control headers

To specify the operation of the cache, any request can assign a Cache-Control header.

For example, to request the latest copy from the PI System, you can add the following HTTP header to the request:

```
Cache-Control: no-cache
```

Alternatively, you can request a maximum age of the data served by the cache with a `max-age` directive in your Cache-Control header:

`Cache-Control: max-age=0` (equivalent to no-cache)

`Cache-Control: max-age=120` (the response must be no more than 120 seconds old)

If the cache instance has not been refreshed within the time period specified in the *Cache-Control* header, the cache instance will be refreshed before the request is served.

Note:
Due to performance considerations, you cannot use the Cache-Control header on bulk data operations by default. However, you can change this setting by adding/setting a Runtime configuration **`IgnoreCacheControlForBulkRequests`** item and setting **Value Type** to `Boolean` and **Value** to `true`. See Configuration at runtime.

View cache instances

You can use a PI Web API resource to list cache instances currently in use and their last refresh, scheduled refresh, and expiration times:

```
GET /piwebapi/system/cacheinstances
```

WebId

When you retrieve information, PI Web API encodes both the ID and the path to the requested object to form a `WebId`. The `WebId` forms a new, redundant, unique identifier for use in further queries. Most PI Web API queries require that one or more `WebIds` be included in the call.

When you make a query with an input `WebId`, PI Web API decodes the `WebId` and finds the object's unique ID along with the object's path. PI Web API then attempts to perform the query using the object's unique ID. If the query fails, PI Web API performs the query using the path instead. This gives the PI Web API a redundancy-protected, efficient way to reference objects in the PI System. As such, it is safe to persist URLs that contain `WebIds` over long periods of time.

Indexed Search

PI Web API provides Indexed Search to enable you to develop a powerful search experience using this RESTful API. Indexed Search finds assets by using multiple fields, such as description, template, category, point source, unit of measure, and other metadata. For example, PI Vision, PI Coresight 2014, and later versions use indexed search to find points, elements, attributes, and event frames.

Indexed Search is a plug-in within PI Web API. Metadata from the PI System is gathered by each instance of the PI Indexed Search Crawler service, a Windows service application that usually resides on the same computer as PI Web API, but can be located on another server if circumstances require. The PI Indexed Search Crawler service periodically runs the crawler to gather metadata from which to build and maintain indexes to improve the performance of commonly used queries.

Note:
Kerberos is the only supported security method for Indexed Search.

The values that are always indexed are names of PI points, PI AF elements, attributes, templates and categories, descriptions, the unit of measure, and data type. You can configure other point attributes to be indexed, for example, you might want to filter your data by **`pointsource`**, **`exdesc`**, or **`location1`**, and so on.

The following queries benefit from indexed search:

- Finding PI points, PI AF elements and attributes by name or description.

You can constrain a query by item type, data type, PI point attribute, PI AF template, or PI AF category. You can restrict the query to a list of PI Data Archive servers and PI AF databases, or to specific paths within a PI AF hierarchy.

- Finding children of PI AF objects.

Note:
The crawler service can index only PI Data Archive servers that are configured with identity mappings, which are supported by PI Server version 3.4.380, or later. The crawler service cannot be used with earlier versions of PI Server.

Security configuration for Indexed Search Crawler

PI Web API Indexed Search Crawler requires read access to the PIUSER, PIDBSEC, PIMAPPING and PIPOINT tables. Access to these tables can be set in the Database Security plugin in PI SMT. Note that by default, the PIWorld identity has read access to PIUSER, PIPOINT and PIDBSEC. Therefore, explicit access is only required for PIMAPPING, unless you have disabled PIWorld. If PIWorld is disabled, you must grant explicit access to PI Web API Indexed Search Crawler on each of the four tables mentioned above.

Further information on configuring the crawler service is available in the Crawler Configuration (PI Web API Indexed Search) (https://techsupport.osisoft.com/Documentation/PI-Web-API/help/topics/crawler-configuration.html) help topic.

Configuration of Indexed Search

You can make changes to your Indexed Search configuration with the PI Web API Admin Utility and on the settings page for Search Service Administration.

PI Web API Admin Utility configuration

You can change the following settings in the PI Web API Admin Utility, which is located under **PI System** on the **Start** menu.

Parameter	Description
`Crawler Service`	Specifies the account under which the Crawler Windows Service operates. For more information on configuring this account, refer to step 8 in Configure installation settings.
`Submit Url`	Specifies the root URL that the Indexed Search crawler uses to connect to PI Web API. For more information on configuring this URL, refer to step 9 in Configure installation settings.

Search Service Administration configuration

You can change the following settings on the settings page of Search Service Administration, which is located at `https://`*`web_API_server`*`/piwebapi/admin/search/settings.html`.

Parameter	Description
`SearchPointAttributes`	Determines which PI point attributes are indexed. The default settings are **exdesc**, **instrumenttag**, **location1**, and **pointsource** but you can click **Add Attribute** and modify those values or select from many other attributes, as needed. Click **Save and Close** to set which attributes are indexed.

Parameter	Description
`SearchScanInterval`	Establishes the time interval between successive scans. The default value is 3 minutes (high data refresh rate). You can also set 1 minute (high system load), 30 minutes, 2 hours, 12 hours (low data refresh rate), and 1 day (low system load). Click **Apply** to set the scan interval.

Cluster management

PI Web API is designed to be clustered behind web load balancers. You can use the load balancer of your choice. You should consider the following points when configuring a cluster:

Cluster configuration

- **Configuration Store**

  To prevent configuration mismatches across the cluster nodes, OSIsoft recommends that all nodes share a common configuration store in PI AF. This configuration is performed during installation by choosing the same PI AF server and choosing identical Instance Names across all installations. By sharing the configuration store, all nodes in the PI Web API cluster are guaranteed to have identical configurations.

- **SSL certificates**

  You must configure appropriate SSL Certificates.

  Simple load balancers (such as DNS, Windows Server Network Load Balancing software load balancing, and Azure Load Balancer) require that each node in the cluster be configured with an identical SSL certificate. The SSL certificate must match the DNS name of the load balancer's public endpoint address, which might mean that local interactions with PI Web API appear to have a certificate name mismatch.

  Advanced load balancers such as the F5 LTM, which do full request proxying, have the ability to set SSL settings at the load balancer level, so that SSL configuration of the individual cluster nodes is less important.

  With all load balancers, OSIsoft recommends that node affinity be retained if possible. It is more efficient when sequential requests from the same user get serviced by the same node.

- **Search crawlers**

  The indexes for the PI Indexed Search functionality are stored in the filesystem on the PI Web API server. In a clustered scenario, this means that each node creates and builds its own copy of the index. State will eventually be consistent across the cluster, but between indexing operations, different nodes in the cluster might produce different results. A search crawler should be configured for each PI Web API node, and should be configured to write directly to the individual node rather than to the load balanced public endpoint. Only one search crawler should be servicing or writing the index.

  Since PI Web API 2016, the Shared Index CTP has supported only a single primary instance in a cluster, as described in Indexed Search Shared Index CTP White Paper (https://techsupport.osisoft.com/Downloads/File/53b2552a-2d62-4ae0-aa75-82b02c5a0ac6).

- **AF SDK caching**

  It is technically possible for an application to create an element but then be unable to read it. This can happen if the request to create was handled by one node but then the request to read is handled by another that has not updated its cache yet. One way to avoid this is documented above: set up the load balancer so that users are re-rerouted to the same node for repeat requests. Another way is to use Cache-Control to disable caching, as described in Cache-Control headers. Reading data without writing is generally safe. Writing time series data is usually safe (unless buffering is enabled on multiple nodes, with one node buffering while others are flowing to the PI Data Archive).

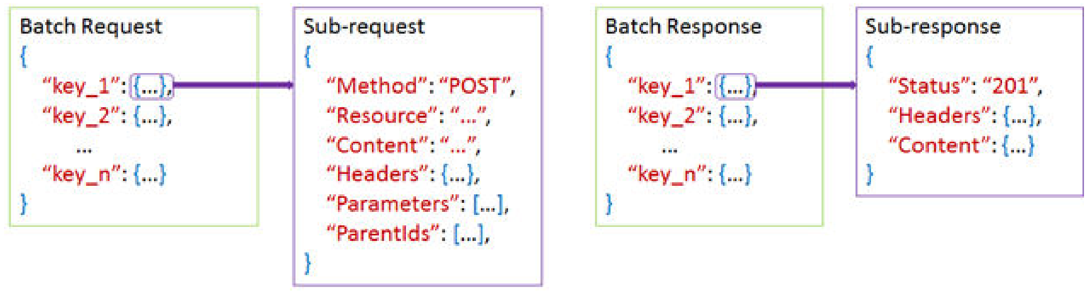
Batch support

Batch enables you to submit multiple related or unrelated requests into a single HTTPS POST and thereby reduce the number of HTTP operations and improve network latency. Individual sub-requests can be independent, or dependent on each other, so you can build relationships between sub-requests, as desired. Because all processing logic is built into a single request, PI Web API can optimize execution. PI Web API supports batching multiple logical REST requests into a single POST HTTP request.

The input is a dictionary with IDs as keys and request objects as values. Each request object specifies the HTTP method, the resource and, optionally, the content and a list of Parent IDs. The list of Parent IDs specifies which other requests must complete before the given request will execute.

The resource can be an absolute URL or a JsonPath that references the response to the parent request. JsonPath is a tool that you use to analyze, transform, and selectively extract data out of Json document. For more information, refer to JSONPath - XPath for JSON (http://goessner.net/articles/JsonPath/).

The following illustration and sample code shows how you can construct the batch request and the batch response you expect. The batch request is a dictionary of HTTP requests that contain an infinite set of keys. The content of each batch request is an HTTP response in which you can specify any method, such as POST, GET, PUT or PATCH. The Resource is a URL stream of the sub-request. If it is a POST request, you can include a string representation of the payload of the POST request (and if it is a GET request you can omit this property). Optionally, you can include a number of headers for the request. The Parameters and the Parent IDs identify relationships between batch requests. The batch response corresponds to the set of keys in the batch request. Each batch response contains a sub-response that contains a status, headers, and content (if any).



Batch request and response structure

Sample batch request

```
{
  "1": {
    "Method": "POST",
    "Resource": "https://localhost/piwebapi/assetdatabases/D0NxzXSxtlKkGzAaZhKOB-
KABJ2buwfWrkye3YhdL2FOUAUEhMQUZTMDRcQgYUUEVSRk9STUFOQ0UgVEVTVElORw/elements",
    "Content": "{\"Name\":\"New Element\"}",
    "Headers": {
      "Cache-Control": "no-cache"
    }
  },
  "2": {
    "Method": "GET",
    "Resource": "$.1.Headers.Location",
    "ParentIds": [
      "1"
    ]
  },
  "3": {
    "Method": "POST",
    "Resource": "$.2.Content.Links.Attributes",
    "Content": "{\"Name\":\"New Attribute\"}",
    "ParentIds": [
      "2"
    ]
  },
  "4": {
    "Method": "GET",
    "Resource": "https://localhost/piwebapi/points?path=%5C%5CMyPIServer
%5Csinusoid",
    "Headers": {
      "Cache-Control": "no-cache"
    }
  },
  "5": {
    "Method": "GET",
    "Resource": "https://localhost/piwebapi/points?path=\\\\MyPIServer\\cdm158",
    "Headers": {
      "Cache-Control": "no-cache"
    }
  },
  "6": {
    "Method": "GET",
    "Resource": "https://localhost/piwebapi/streamsets/value?
webid={0}&webid={1}",
    "Parameters": [
      "$.4.Content.WebId",
      "$.5.Content.WebId"
    ],
    "ParentIds": [
      "4",
      "5"
    ]
  },
  "7": {
    "Method": "GET",
    "Resource": "https://localhost/piwebapi/assetdatabases/D0NxzXSxtlKkGzAaZhKOB-
KABJ2buwfWrkye3YhdL2FOUAUEhMQUZTMDRcQgYUUEVSRk9STUFOQ0UgVEVTVElORw/elements"
  },
  "8": {
    "Method": "GET",
    "RequestTemplate": {
      "Resource": "$.7.Content.Items[*].Links.Attributes"
    },
    "ParentIds": [
      "7"
    ]
  }
}
```

The input is a dictionary with IDs as keys and request objects as values. Each request object specifies the HTTP method and the resource and, optionally, the content and a list of parent IDs. The list of parent IDs specifies which other requests must complete before the given request will be executed. The example first creates an element, then gets the element by the response's Location header, then creates an attribute for the element. The batch's response is a dictionary that uses keys corresponding to those provided in the request, with response objects containing a status code, response headers, and the response body. A request can alternatively specify a request template in place of a resource. In this case, a single JsonPath may select multiple tokens, and a separate sub-request will be made from the template for each token. The responses of these sub-requests will be returned as the content of a single outer response.

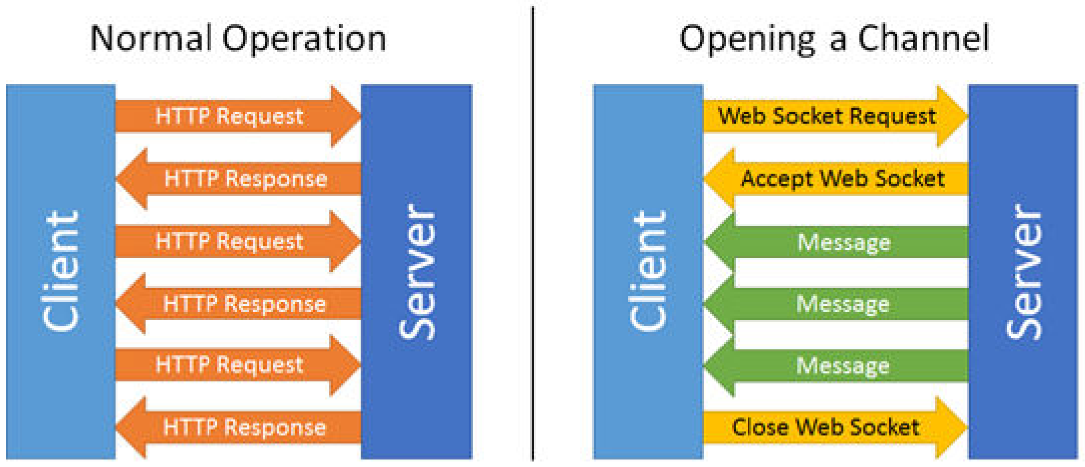
Channels

A channel is a way to receive continuous updates about a stream or stream set, and therefore of particular relevance to updates of PI AF attribute data reference and PI point values.

Channels use the WebSocket protocol (defined in the Request for Comments document RFC 6455 The WebSocket Protocol (https://tools.ietf.org/html/rfc6455)) to enable the PI Web API server to send messages containing new stream time series data value changes. Once a channel is opened, the client receives continuous updates from the server without having to poll. This increases efficiency by reducing unnecessary requests, as well as reducing the overhead that comes with HTTP headers.

Explanation of channels

Normally, when clients apply the PI Web API, they send an http request to the server and the server sends back a response. Although this is satisfactory if clients need information only once or twice, it becomes rather inefficient if they require information constantly. In such situations, the preferred solution would be to open a channel in which clients make web socket requests to the server, and the server accepts those web socket connections in return. Thereafter, the server can send messages to clients whenever a value is changed or added. Clients no longer have to constantly poll the server for information. The server simply messages clients whenever new information is available. When clients no longer want messages, they can close the connection. If a client crashes, the server reacts and closes the connection itself.



Normal vs. Channel communication

For more information on channels and how to implement them, refer to the PI Web API online help.

WebSocket protocol

Channels are accessed using the WebSocket protocol. Numerous WebSocket client libraries are available in several languages for use with channels. Because the WebSocket protocol is a relatively new technology, be advised that there are still authentication issues in some browsers, especially with Basic and Kerberos authentication. Clients may need to authenticate using `https` before they can open any channels. This can be done by issuing any PI Web API request.

Stream value changes

Once connected, the server may send a message with current values for all streams if the optional URL parameter **`includeInitialValues`** is set to `true` (default is `false`). Thereafter, the server sends messages containing all stream value changes since the last message. The payload of these messages is the same as the **`GetEnd`** method for streams and stream sets. Any streams whose values have not changed since the last message will not be included in the message. Value changes are sent in the order they were received by the PI System. There is no guarantee that value changes are sent in chronological order.

Support for FIPS CTP

As noted in PI Web API overview, both PI Web API service and PI Indexed Search support Windows environments that enforce the use of FIPS cryptographic algorithms.

No additional configuration is required once the local security policy **System cryptography: Use FIPS compliant algorithms for encryption, hashing, and signing** has been enabled.

Note:
If you previously disabled FIPS compliance at the application level (rather than at the machine level) by adding `<enforceFIPSPolicy enabled="false"/>` in the `<runtime>` section of `C:\Program Files\PIPC\WebAPI\OSIsoft.REST.HOST.exe.config`, remove that setting, and restart PI Web API service and PI Web API Crawler service.

PI Web API troubleshooting

As a PI Web API administrator, you can resolve common problems that you might encounter during installation and administration of PI Web API.

What you need to start

You need the PI Web API setup kit, a server machine, and a web browser. PI Web API requires access to a PI AF server. You also need to be a Windows administrator on the server that hosts PI Web API.

Commonly encountered problems

Problems on installation

- **SSL Certificate Requirements**

  Problem: An existing, installed certificate cannot be selected when configuring installation properties for the first time, or when running the PI Web API Admin Utility.

  Solution: The certificate must meet the following requirements in order to be selected:

  - The certificate must be installed in the Personal Store of the Local Machine.
  - The certificate must not be expired.
  - The certificate must have a private key.
  - The certificate must be an SHA-2 certificate.

- **Installation Fails Unexpectedly**

  Problem: Especially during silent installation, invalid inputs can cause setup to fail.

  Solution: Re-run the PI Web API setup from the command line, and specify verbose log output to discover the root cause of the problem:

  ```
  msiexec /l*v c:\piwebapisetup.log /i
  c:\path\to\piwebapi.msi
  ```

  Problem: During installation on a computer that is not connected to a domain, installation can fail unexpectedly when a remote PI AF server is selected for the configuration store. If the user manually enters credentials to connect to the PI AF server when selecting a server to store configuration, those credentials cannot be retrieved after installation begins. This causes the installation to fail.

  Solution: Any of the following approaches will mitigate the problem:

  - Ensure that the username used for installation on the PI Web API host also exists on the PI AF server, and that the passwords are the same.
  - Add credentials to the remote PI AF server in the installing user's Windows Credentials Store.
  - Install PI Web API on the same host as the PI AF server.

Problems after the service is installed

- **Kerberos Delegation**

  Problem: Requests to the service fail with the response `It may be that the impersonated client user account cannot be delegated to the remote server.`

  Solution: Configure Kerberos delegation to allow credentials to take a "double hop":

  - If running PI Web API using the default service account (NT SERVICE\piwebapi):
    - Ensure that the computer is trusted for delegation in Active Directory.
  - If running PI Web API as a custom account:
    - Ensure that the account is trusted for delegation in Active Directory.
    - Ensure that the account has Service Principal Names registered for:
      - HTTP/*hostname*
      - HTTP/*fully.qualified.hostname*
    - Ensure that the account is not marked “sensitive and cannot be delegated” in Active Directory.

- **Same-Origin Policy**

  Problem: Requests made within browser scripts (for example, AJAX requests) fail with a message about same-origin or cross-origin request policy.

  Solution: By default, browsers prevent scripts served from one site from making requests to other sites. To enable this behavior, enable the Cross-Origin Resource Sharing settings in the PI Web API Configuration database. For further details on setting these values, see Cross-Origin Resource Sharing.

- **Changing the Service Account**

  Problem: Changing the PI Web API service account after installation causes problems with the service: the service may fail to start or fail to serve requests as expected.

  Solution: Run the PI Web API Admin Utility and set the PI Web API service account.

- **Firefox Kerberos Authentication**

  Problem: When Kerberos is configured as the PI Web API authentication method, requests made in Firefox fail with the response `Authorization has been denied for this request.`

  Solution: Refer to the OSIsoft Tech Support article KB01223 - Kerberos and Internet Browsers (https://techsupport.osisoft.com/Troubleshooting/KB/KB01223).

- **IE Compatibility Mode**

  Problem: Visiting PI Web API resources in Internet Explorer returns a prompt to download raw JSON, instead of the expected HTML response with help page links. In another scenario, responses are returned but the response is not formatted (that is, returns everything in a single line) and links are not displayed as hyperlinks.

  Solution: In Internet Explorer, in **Tools** > **Compatibility View settings** clear **Display intranet sites in Compatibility View**.

- **Administrative Operations Return Unauthorized**

  Problem: Administrative operations (for example, updating service configuration) fail with the response `Authorization has been denied for this request.`

  Solution: Administrative operations require the user to be a member of the `PI Web API Admins` group on the server computer. Users added to this group must log off and on before administrator privileges take effect.

Problems on uninstallation

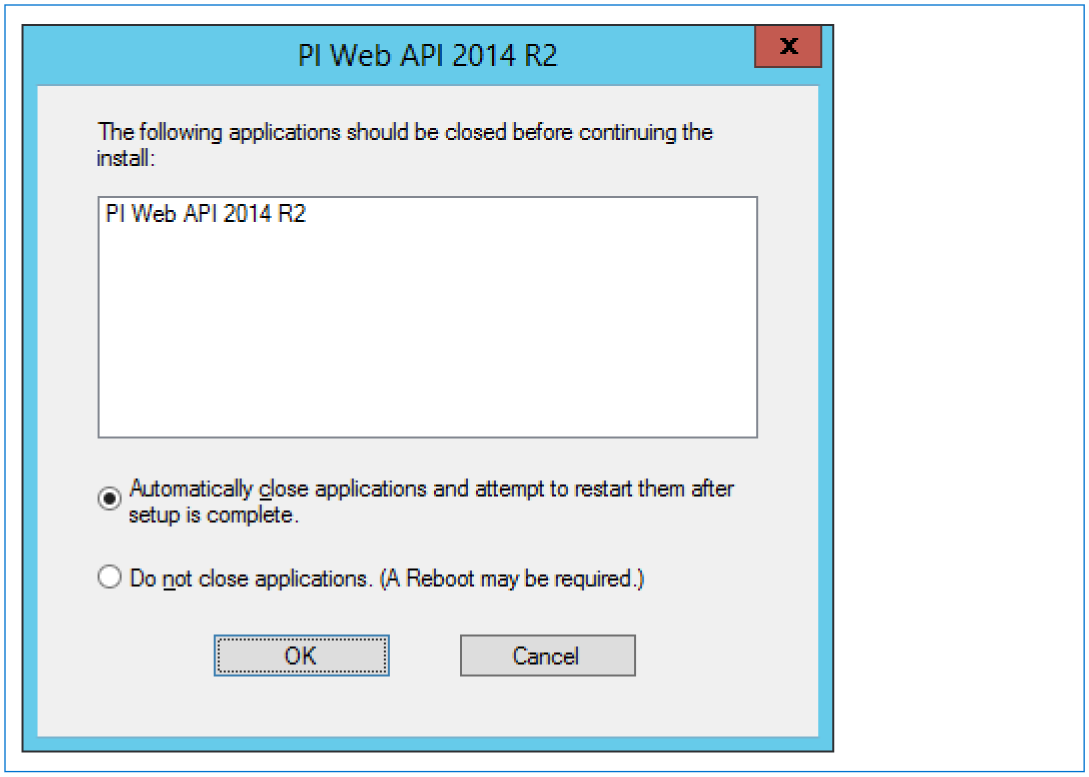
- **Windows Service disabled but not uninstalled**

  Problem: On uninstallation, the Windows Service is not removed. It is, however, disabled.

  Solution: This is the expected behavior if a file on which the Windows Service depends is locked or in use. The dependent file and the service are removed the next time the computer is rebooted.

- **Requesting to close many applications**

  Problem: During a normal uninstallation, if the PI Web API Windows Service is not yet stopped, the following window is displayed:



PI Web API 2014 R2

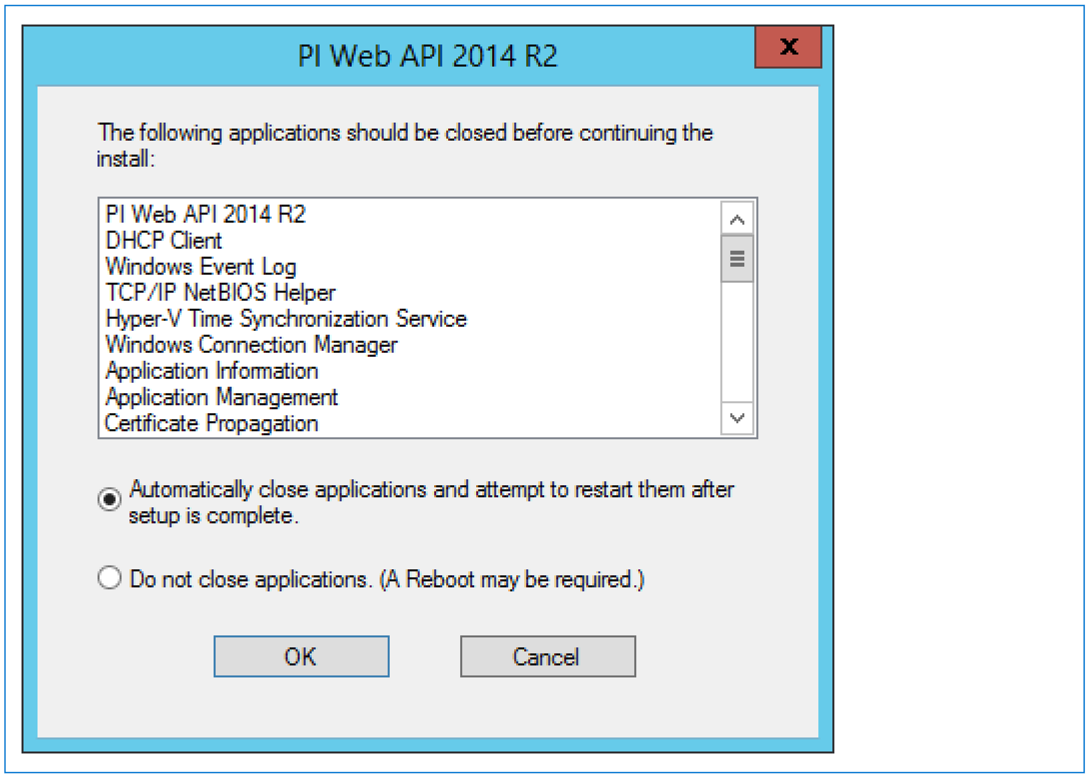
The following applications should be closed before continuing the install:

PI Web API 2014 R2

● Automatically close applications and attempt to restart them after setup is complete.

○ Do not close applications. (A Reboot may be required.)

Usually, the window displays only one application, which is the PI Web API application. This is normal behavior. Select **Automatically close applications and attempt to restart them after setup is complete** and click **OK**. However, if the Windows Event Viewer is open when the uninstallation starts, the window might show many other applications:


PI Web API 2014 R2

The following applications should be closed before continuing the install:

PI Web API 2014 R2
DHCP Client
Windows Event Log
TCP/IP NetBIOS Helper
Hyper-V Time Synchronization Service
Windows Connection Manager
Application Information
Application Management
Certificate Propagation

[✓ Automatically close applications and attempt to restart them after setup is complete.]

Do not close applications. (A Reboot may be required.)

Solution: Click **Cancel** to terminate the uninstallation process, close the Windows Event Viewer, and then rerun the uninstallation.

Problems with Bearer authentication

- **C2WTS (Claims to Windows Token Service) not started or configured incorrectly**

  Solution: Check the PI Web API Admin event log and review events that describe the error.

- **UPN (Universal Principal Name) claim is missing or incorrect**

  Solution: Check the PI Web API Debug event log and review events that describe the error.

- **Issues validating the JWT (JSON Web Token)**

  Problem: Errors in validating the JWT show up in the Debug event log.

  Solution: Inspect the JWT on the client side for issuer and audience and make sure they match the **`BearerValidIssuer`** and **`BearerValidAudiences`** configuration settings.

General troubleshooting

The first step in troubleshooting is to determine whether the PI Web API service or the client application is responsible for the problem. One effective way to do this is to emulate the failing request in a basic HTTP/REST client, such as the Postman REST client for Chrome. It can also be helpful to view the raw request and response data using the browser's developer console or a traffic inspector like Fiddler. These tools remove much of the browser- and client application-specific configuration that can affect the service's behavior.

Once the source of problem is determined to be the service, additional information may be available in the service application's logs. For more information, see Log collection.

Technical support and other resources

For technical assistance, contact OSIsoft Technical Support at +1 510-297-5828 or through the OSIsoft Tech Support Contact Us page (https://techsupport.osisoft.com/Contact-Us/). The website offers additional contact options for customers outside of the United States.

When you contact OSIsoft Technical Support, be prepared to provide this information:

- Product name, version, and build numbers
- Details about your computer platform (CPU type, operating system, and version number)
- Time that the difficulty started
- Log files at that time
- Details of any environment changes prior to the start of the issue
- Summary of the issue, including any relevant log files during the time the issue occurred

To ask questions of others who use OSIsoft software, join the OSIsoft user community, PI Square (https://pisquare.osisoft.com). Members of the community can request advice and share ideas about the PI System. The PI Developers Club space within PI Square offers resources to help you with the programming and integration of OSIsoft products.