Interreg
Danube Transnational Programme
DAREFFORT

User manual for integrated data exchange software

WP4 Deliverable 4.2.2

October 16, 2020

SRG

The project is co-funded by the European Union

Environet Documentation

This document is the documentation of the Environet system.

System requirements

- **OS**: Linux operation system, Ubuntu 18+ or Debian 9+
- **Docker Engine**: 19+
- **Docker Compose**: 1.25+
- **RAM**: at least 2GB
- **HDD**: at least 1GB

Setup

This guide assumes you are installing Environet in a Linux environment.

Install the Docker Engine and Docker Compose

Docker Engine

The docker documentation has lots of helpful information, along with distribution specific installation instructions. If you are new to Docker, we recommend reading the overview first.
To see distribution specific installation instructions, navigate to Docker Engine > Linux > Your distribution in the documentation.

Be sure to follow the post installation steps to allow using docker as a non-root user.

You can verify that Docker is installed correctly by running:
`$ docker run hello-world`
If should output a friendly message to the terminal.

Docker Compose

Setting up Compose is a simpler process, which is described in detail on this page.
It involves downloading the docker-compose binary from github and setting executable permissions on it.

You can verify that Docker Compose is installed correctly by running:
`$ docker-compose --version`
It should output the currently installed version number.

Get the source

You will need to have Git installed to be able to download the project source, and to receive updates later. It is easiest to install Git on Linux using the preferred package manager of your Linux distribution. See the Git downloads page for details.

Checkout the Environet docker repository - Navigate to the directory where you would like to install environet
- Run
`$ git clone https://github.com/environet/environet-docker.git --recurse-submodules`

By default, the files will be downloaded to a directory named `environet-docker`, you can specify a custom name by providing a third argument to the command, e.g.:
`$ git clone https://github.com/environet/environet-docker.git my_directory --recurse-submodules`

Note: If you cloned the repository without the `--recurse-submodules` flag, you need to run `git submodule init` and `git submodule update`, to get the src files checked out.

Change to the directory you checked the code out to, and you should be ready to proceed with the setup.

If you are installing a data node, refer to the data node setup instructions

If you are installing a distribution node, refer to the distribution node setup instructions

Getting updates and maintenance

The `environet` cli script is a wrapper for some docker containers managed with docker compose. After first starting a *dist* or *data* node, these services will start automatically after a system reboot.
To stop and start them manually, you may run `./environet data up` or `./environet data down` (`./environet dist up` and `./environet dist down` in case of a distribution node).

To get the latest version, simply run `git pull` in the repository folder.

Distribution node

Overview

A Distribution node, in it's minimal form, is a service aspect that's able to maintain the central database and service API requests. In the first phase the domain of the distribution node is: https://environet.environ.hu

Central database

The distribution node has a database backend. It contains: * Data and configuration of hydro and meteo points * Measurement data tables of these monitoring points * User and access (ACL) tables (users, groups, permissions) * System runtime data

Schema of the database can be found here: Database structure

Administration area

On the administration area, administrators can maintain: * The data and configuration of monitoring points and observed properties * Operators * Users, groups, and their permissions (ACL) * Public keys of operator-users

The url of the administration area: https://distribution-node.com/admin

API endpoints

Upload

With this endpoint data nodes can upload data of some monitoring points.
The detailed documentation of this endpoint can found here: Upload API documentation

Download

With this endpoint clients can query data from the distribution node. This endpoint is available only after authentication, and authorization, so only permitted users can run queries. The response can be filtered by date, monitoring point type, and some other properties.

The detailed documentation of this endpoint can found here: Download API documentation

Setup

1. Install the environet project. Refer to Setup.
2. Create a distribution node configuration `./environet dist install`
3. Initialize database, and create admin user `./environet dist database init`

After going through these steps, the distribution node should be up and running. You can access the admin panel at YOUR_IP/admin.

Updates

After updating your deployment, you need to run `./environet dist database migrate`, to run any database migrations that might be included in the update.

Database structure

Database engine

- Required database engine is PostgreSQL
- Version compatibility: 12+

Schema sql structure

Link to sql file

Upload API documentation

URL & METHOD

- **Method**: `POST`
- **URL**: `https://domain.com/upload`

Headers

Header name	Content
Authorization	Signature (See below)
Content-Type	application/xml

Signature header:

The pattern of the header:
`Signature keyId="[username]",algorithm="rsa-sha256",signature="[signature]"`

The `keyId` is the username of the uploader user. The `signature` part is the base64 encoded openssl signature which was created with the user's private key from the from the body of the response, which is the XML data.

Body

The body of the response is the XML data.
The xml must be valid agains the environet's upload api schema: environet.xsd
The public url of this schema is: `https://distribution-node.com/schemas/environet.xsd`

Sample input XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<environet:UploadData xmlns:environet="environet">
    <environet:MonitoringPointId>EUCD</environet:MonitoringPointId>
    <environet:Property>
        <environet:PropertyId>water_level</environet:PropertyId>
        <environet:TimeSeries>
            <environet:Point>
                <environet:PointTime>2020-02-25T00:00:00+01:00</environet:PointTime>
                <environet:PointValue>22</environet:PointValue>
            </environet:Point>
            <environet:Point>
                <environet:PointTime>2020-02-26T00:01:00+01:00</environet:PointTime>
                <environet:PointValue>23</environet:PointValue>
            </environet:Point>
            <environet:Point>
                <environet:PointTime>2020-02-27T00:02:00+01:00</environet:PointTime>
                <environet:PointValue>24</environet:PointValue>
            </environet:Point>
        </environet:TimeSeries>
    </environet:Property>
</environet:UploadData>
```

Repsonses

Success

- **Status code**: 200
- **Content-type**: application/xml
- **Body**: `empty`
- **Description**: Upload was successful, the data has been successfully processed.

Invalid request

- **Status code**: 400
- **Content-type**: application/xml
- **Body**: XML: `environet:ErrorResponse`

- **Description**: Input or processing error during the upload process. The reponse in an error xml which is valid agains environet' upload api schema: environet.xsd
- **Body example**:

```
<?xml version="1.0" encoding="UTF-8"?>
<environet:ErrorResponse xmlns:environet="environet">
    <environet:Error>
        <environet:ErrorCode>101</environet:ErrorCode>
        <environet:ErrorMessage>Error</environet:ErrorMessage>
    </environet:Error>
</environet:ErrorResponse>
```

- **Error codes**
  - `101` : Unknown error
  - `102` : Server error
  - `201` : Authorization header is missing
  - `202` : Username is empty
  - `203` : User not found with username
  - `204` : Invalid Authorization header
  - `205` : Action not permitted
  - `206` : Public key for user not found
  - `301` : Signature is invalid
  - `302` : Xml syntax is invalid
  - `303` : Xml is invalid against schema
  - `401` : Error during processing data
  - `402` : Monitoring point not found with the given identifier
  - `403` : Property for the selected monitoring point not found, or not allowed
  - `404` : Could not initialize time series for monitoring point and property

Server error

- **Status code**: 500
- **Content-type**: application/xml
- **Body**: XML: `environet:ErrorResponse`
- **Description**: Unidentified error during request. The response is an xml which is valid agains environet's upload api schema: environet.xsd
- **Body example**:

```
<?xml version="1.0" encoding="UTF-8"?>
<environet:ErrorResponse xmlns:environet="environet">
    <environet:Error>
        <environet:ErrorCode>500</environet:ErrorCode>
        <environet:ErrorMessage>Error</environet:ErrorMessage>
    </environet:Error>
</environet:ErrorResponse>
```

Download API documentation

URL & METHOD

- **Method**: `GET`
- **URL**: `https://domain.com/download`

Query parameters

- **token**: `required` A random string from which the signature is generated. This is required for authentication, the server verifies the signature in the `Authorization` header based on this token.
- **type**: The type of the monitoring points. The value must be one of the following: `hydro` | `meteo`
- **start**: Minimum date of time series. Date format: ISO 8601
- **end**: Maximum date of time series. Date format: ISO 8601
- **country[]**: Query time series only for monitoring points from the given countries. Country code format: ISO 3166-1 - alpha-2
- **symbol[]**: Query time series only of the given observed properties.

Headers

Header name	Content
Authorization	Signature (See below)

Signature header:

The pattern of the header:
`Signature keyId="[username]",algorithm="rsa-sha256",signature="[signature]"`

The `keyId` is the username of the user who wants to query data. The `signature` part is the base64 encoded openssl signature which was created with the user's private key from the token in the query parameters.

Repsonses

Success

- **Status code**: 200
- **Content-type**: application/xml
- **Body**: XML: `wml2:Collection`
- **Description**: Measurement data in WaterML2.0 comaptible XML format: http://www.opengis.net/waterml/2.0

Invalid request

- **Status code**: 400
- **Content-type**: application/xml
- **Body**: XML: `environet:ErrorResponse`
- **Description**: The query request is invalid. The response is an xml which is valid agains environet's api schema: environet.xsd
- **Body example**:

```
<?xml version="1.0" encoding="UTF-8"?>
<environet:ErrorResponse xmlns:environet="environet">
    <environet:Error>
        <environet:ErrorCode>101</environet:ErrorCode>
        <environet:ErrorMessage>Error</environet:ErrorMessage>
    </environet:Error>
</environet:ErrorResponse>
```

- **Error codes**
  - `101` : Unknown error
  - `102` : Server error
  - `201` : Authorization header is missing
  - `202` : Username is empty
  - `203` : User not found with username
  - `204` : Invalid Authorization header
  - `205` : Action not permitted
  - `206` : Public key for user not found
  - `207` : Request token not found
  - `301` : Signature is invalid
  - `302` : Observation point type is missing
  - `303` : Observation point type is invalid
  - `304` : Start time filter value is invalid
  - `305` : End time filter value is invalid

Server error

- **Status code**: 500
- **Content-type**: application/xml
- **Body**: XML: `environet:ErrorResponse`
- **Description**: Unidentified error during request. The response is an xml which is valid agains environet's api schema: environet.xsd
- **Body example**:

```
<?xml version="1.0" encoding="UTF-8"?>
<environet:ErrorResponse xmlns:environet="environet">
    <environet:Error>
        <environet:ErrorCode>500</environet:ErrorCode>
        <environet:ErrorMessage>Error</environet:ErrorMessage>
    </environet:Error>
</environet:ErrorResponse>
```

Monitoring points API documentation

URL & METHOD

- **Method**: `GET`
- **URL**: `https://domain.com/api/monitoring-points`

Query parameters

- **token**: `required` A random string from which the signature is generated. This is required for authentication, the server verifies the signature in the `Authorization` header based on this token.

Headers

Header name	Content
Authorization	Signature (See below)

Signature header:

The pattern of the header:
`Signature keyId="[username]",algorithm="rsa-sha256",signature="[signature]"`

The `keyId` is the username of the user who wants to query data. The `signature` part is the base64 encoded openssl signature which was created with the user's private key from the token in the query parameters.

Repsonses

Success

- **Status code**: 200
- **Content-type**: application/json
- **Body**: `JSON`
- **Description**: Request is OK, list the hydro and meteo points of user
- **Body example**:

```
{
    "hydro": [
        {
            "id": 1,
            "station_classificationid": 1,
            "operatorid": 1,
            "bankid": 1,
            "waterbodyeuropean_river_code": "waterbody",
            "eucd_wgst": "ABC123",
            "ncd_wgst": "ABC123",
            "vertical_reference": "Vertical reference",
            "long": "1.0000000000",
            "lat": "1.0000000000",
            "z": "1.0000000000",
            "maplong": "1.0000000000",
            "maplat": "1.0000000000",
            "country": "CountryCode",
            "name": "MPoint",
            "location": "Location",
            "river_kilometer": "1.0000000000",
            "catchment_area": "1.0000000000",
            "gauge_zero": "1.0000000000",
            "start_time": "2020-01-01 00:00:00",
            "end_time": "2020-05-01 00:00:00",
            "utc_offset": 0,
            "river_basin": "Basin",
            "observed_properties": [
                "hydro_property"
            ]
        }
    ],
    "meteo": [
        {
            "id": 1,
            "meteostation_classificationid": 1,
            "operatorid": 1,
            "eucd_pst": "DEF456",
            "ncd_pst": "DEF456",
            "vertical_reference": "Vertical reference",
            "long": "1.0000000000",
            "lat": "1.0000000000",
            "z": "1.0000000000",
            "maplong": "1.0000000000",
            "maplat": "1.0000000000",
            "country": "CountryCode",
            "name": "Name",
            "location": "Location",
            "altitude": "1.0000000000",
            "start_time": "2020-01-01 00:00:00",
            "end_time": "2020-05-01 00:00:00",
            "utc_offset": 0,
            "river_basin": "Basin",
            "observed_properties": [
                "meteo_property"
            ]
        }
    ]
}
```

Invalid request

- **Status code**: 400
- **Content-type**: application/json
- **Body**: `JSON`
- **Description**: The query request is invalid. The response is a json object with the error message, under error key.
- **Body example**:

```
{
 "error": "Error message"
}
```

Server error

- **Status code**: 500
- **Content-type**: application/json
- **Body**: `JSON`
- **Description**: Unidentified error during request. The response is a json object with the error message, under error key.
- **Body example**:

```
{
 "error": "Error message"
}
```

Admin user manual

This document's goals to represents the different parts of the administration area to help to understand how it works. Help you to see through how works the specified relationships and how you can handle them.

General information

Good to know, that on each list page, there are on the top right a search field. In the following each sections you can see a "searchable" part, where you can find in what kind of fields can the system search on the specified page.

Auth pages

Login page

Path: `/admin/login`

Here you have to type your credentials data to login into the administration area. If you have no login credentials yet, you have to gain access from your webmaster.

Logout

You can logout if you click the exit icon on the top right of the page.

Path: `/admin/logout`

Dashboard

Path: `/admin`

Users

Here you can handle the users who has already added to the system.

Path: `/admin/users`

Searchable: name, username, email

There are three types of user: - Distribution node - Client node - Users who can manage the system in the admin area

All of them are listed in the users grid.

Each user, except the user type users have to have public key attached to themselves, because of they are communicate on API channel and it necessary to their authentication.

New user

You can add new user if you click the "Add user" button on the top left of the users list page.

Path: `/admin/users/add`

On the user's creating page, you have to fill the following mandatory fields: - name - email - it must be unique - username - it must be unique - permission or group

The public key field is necessary only if we want to create a client or distrubition node user. In other case if you would like to create a "system manager" user, you have to fill to password field instead of public key.

If you wouldn't like to assign a specified permission to the user, you must assign the user to a group, they will inherits the group's permissions.

Updating user

You can select a user to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/users/edit?id=[user identifier]`

You can update all user datas that you gave on the creating page except the username field.

User deleting

You can delete a user if you click the "Trash" icon at the end of the specific row. If you clicked, it shows a confirm window, where you have to approve the deleting. In fact the deleted user will never physically deleted, we have to keep its datas by security reasons.

Path: `/admin/users/delete?id=[user identifier]`

Groups

Here you can handle the groups what has already added to the system.

Path: `/admin/groups`

Searchable: name

New group

You can add new group if you click the "Add group" button on the top left of the group list page.

Path: `/admin/groups/add`

On the group's creating page you have to fill the name of the group and you have to assign permission to the specified group. On the creating page, you can assign only one permission, but later you have possibility to add more of it.

Updating group

You can select a group to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/groups/edit?id=[group identifier]`

Here you can change the name of the group and you can assign more permission to it.

Group deleting

You can delete a group if you click the "Trash" icon at the end of the specific row. If you clicked, it shows a confirm window, where you have to approve the deleting. **If any user or operator have already assigned to the specified group, the delete operation cannot be performed.** First time you have to detach these relations and after that you can delete the group.

Path: `/admin/groups/delete?id=[group identifier]`

Data providers

Here you can handle the data providers what has already added to the system.

Path: `/admin/data-providers`

Searchable: name, address, email

New data provider

You can add new data provider if you click the "Add data provider" button on the top left of the data provider's list page.

Path: `/admin/data-providers/add`

On the data providers creating page, you have to fill the following mandatory fields:

Operator data - name - phone - e-mail - url

User data - name - username - email

The user data is necessary because of each operator has to be assigned to at least one user.

Updating data provider

You can select a data provider to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/data-providers/edit?id=[data-provider identifier]`

Here you can change all of operator's data and you can assign more users or groups to the specific dataprovider.

Data provider show page

You can select a data provider to show if you click the "Eye" icon at the end of the specific row.

Path: `/admin/data-providers/show?id=[data-provider identifier]`

Here you can see the stored data of the data providers and its relations to direction of users and groups.

Data provider deleting

You cannot delete any data provider.

Hydro

Hydro monitoring point

Here you can handle the monitoring points what has already added to the system.

Path: `/admin/hydro/monitoring-points`

Searchable: european river code, country, name, location

New monitoring point

You can add new monitoring point if you click the "Add monitoring point" button on the top left of the hydro monitoring point list page.

Path: `/admin/hydro/monitoring-points/add`

On the monitoring point creating page, you have to fill the following mandatory fields: - name - classification - operator - riverbank - water body - observed properties

Updating monitoring point

You can select a monitoring point to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/hydro/monitoring-points/edit?id=[monitoring point identifier]`

Here you can change all of monitoring point's data and you can assign more observed property to the specific monitoring point.

Monitoring point show page

You can select a monitoring point to show if you click the "Eye" icon at the end of the specific row.

Path: `/admin/hydro/monitoring-points/show?id=[monitoring point identifier]`

Here you can see the stored data of the monitoring point and its relations to direction of station classification, operator, water body and observed property.

Monitoring point deleting

You cannot delete any monitoring point.

Hydro observed properties

Here you can handle the observed properties what has already added to the system.

An observed property describes, what kind of property can be measured by a monitoring point.

Path: `/admin/hydro/observed-properties`

Searchable: symbol, description

New observed property

You can add new observed property if you click the "Add observed property" button on the top left of the hydro observed property list page.

Path: `/admin/hydro/observed-properties/add`

On the observed property creating page, you have to fill to following mandatory fields: - symbol - description - type - Processed, or real time data - default is real time

Updating observed property

You can select an observed property to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/hydro/observed-properties/edit?id=[observed property identifier]`

Here you can change all of observed property's data.

Observed property show page

You can select an observed property to show if you click the "Eye" icon at the end of the specific row.

Path: `/admin/hydro/observed-properties/show?id=[observed property identifier]`

Here you can see the stored data of an observed property.

Observed property deleting

You cannot delete any observed property.

Water body

Here you can handle the water bodies what has already added to the system.

Path: `/admin/hydro/waterbodies`

Searchable: european river code

New water body

You can add new water body if you click the "Add water body" button on the top left of the water body list page.

Path: `/admin/hydro/waterbodies/add`

On the water body's creating page, you have to fill to following mandatory fields: - cname - european river code

Updating water body

You can select a water body to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/hydro/waterbodies/edit?id=[waterbody identifier]`

Here you can change the cname of the selected water body.

Water body show page

You can select a water body to show if you click the "Eye" icon at the end of the specific row.

Path: `/admin/hydro/waterbodies/show?id=[waterbody identifier]`

Here you can see the stored data of a water body.

Water body deleting

You cannot delete any water body.

Station classifications

Here you can handle the classifications of a specified station what has already added to the system.

Path: `/admin/hydro/station-classifications`

Searchable: value

New classification

You can add new station classification if you click the "Add station classification" button on the top left of the classification's list page.

Path: `/admin/hydro/station-classifications/add`

On the classification's creating page, you have to fill to following mandatory fields: - value

Updating classification

You can select a station classification to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/hydro/station-classifications/edit?id=[classification identifier]`

Here you can change the value of the selected station classification.

Classification show page

You can select a station classification to show if you click the "Eye" icon at the end of the specific row.

Path: `/admin/hydro/station-classifications/show?id=[classification identifier]`

Here you can see the stored data of a station classification.

Classification deleting

You cannot delete any classification.

Hydro results

Here you can see the results of the different monitoring point, what has arrived via API.

Path: `/admin/hydro/results`

Searchable: name, symbol

Meteo

Meteo monitoring point

Here you can handle the monitoring points what has already added to the system.

Path: `/admin/meteo/monitoring-points`

Searchable: country, name, location

New monitoring point

You can add new monitoring point if you click the "Add monitoring point" button on the top left of the meteo monitoring point list page.

Path: `/admin/meteo/monitoring-points/add`

On the monitoring point creating page, you have to fill the following mandatory fields:

- name
- classification
- operator
- observed properties

Updating monitoring point

You can select a monitoring point to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/meteo/monitoring-points/edit?id=[monitoring point identifier]`

Here you can change all of monitoring point's data and you can assign more observed property to the specific monitoring point.

Monitoring point show page

You can select a monitoring point to show if you click the "Eye" icon at the end of the specific row.

Path: `/admin/meteo/monitoring-points/show?id=[monitoring point identifier]`

Here you can see the stored data of the monitoring point and its relations to direction of station classification, operator and observed property.

Monitoring point deleting

You cannot delete any monitoring point.

Observed properties

Here you can handle the observed properties what has already added to the system.

An observed property describes, what kind of property can be measured by a monitoring point.

Path: `/admin/meteo/observed-properties`

Searchable: symbol, description

New observed property

You can add new observed property if you click the "Add observed property" button on the top left of the meteo observed property list page.

Path: `/admin/meteo/observed-properties/add`

On the observed property creating page, you have to fill to following mandatory fields:

- symbol
- description
- type - Processed, or real time data - default is real time

Updating observed property

You can select an observed property to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/meteo/observed-properties/edit?id=[observed property identifier]`

Here you can change all of observed property's data.

Observed property show page

You can select an observed property to show if you click the "Eye" icon at the end of the specific row.

Path: `/admin/meteo/observed-properties/show?id=[observed property identifier]`

Here you can see the stored data of an observed property.

Observed property deleting

You cannot delete any observed property.

Station classifications

Here you can handle the classifications of a specified station what has already added to the system.

Path: `/admin/meteo/station-classifications`

Searchable: value

New classification

You can add new station classification if you click the "Add station classification" button on the top left of the classification's list page.

Path: `/admin/meteo/station-classifications/add`

On the classification's creating page, you have to fill to following mandatory fields: - value

Updating classification

You can select a station classification to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/meteo/station-classifications/edit?id=[classification identifier]`

Here you can change the value of the selected station classification.

Classification show page

You can select a station classification to show if you click the "Eye" icon at the end of the specific row.

Path: `/admin/meteo/station-classifications/show?id=[classification identifier]`

Here you can see the stored data of a station classification.

Classification deleting

You cannot delete any classification.

Meteo results

Here you can see the results of the different monitoring point, what has arrived via API.

Path: `/admin/meteo/results`

Searchable: name, symbol

Measurement access rules

Concept of the rules

The access rules controls the time interval in which the data will be available for a user under the download API request.

The rules have three required dimensions: * **Monitoring point** - the rule is for all data of the given monitoring point * **Observed property** - the rule is for all data of the observed property of the monitoring point * **User groups** - the rule will be applied if the requesting user is under these groups

The rules has one additional optional dimension: * **Operator** - If an operator creates the rule, these property will be set automatically. An administrator can set is optionally. This property controls which rule will be visible under operator's acces rule list, and important in case of wildcard (*) rules. If the operator parameter is set, the wildcard (*) will be applied only for points/properties under the given operator.

If the rule matches for a download API request (so the **user** getting data for the **monitoring point** and the **observed property**), the user can retrieve data only in the given time period. This period's end is the current date, and the start is controlled by the `years`, `months` and `days` paramter of the access rule.

The monitoring point and the observed property can be a *, which will match all monitoring points (globally, or of an operator), or all properties.

The rules will be "merged" before checking user's access level.

Rule list

On the list page all rules are visible, with the required and optional dimensions.

Path: `/admin/measurement-access-rules`

New access rule

You can add new access rule if you click the "Add rule" button on the top left of the rules list page.

Path: `/admin/measurements-access-rules/add`

On the rules's creating page, you have to fill the following mandatory fields: - Monitoring point selector - The ID of the monitoring point, or * for all - Observed property selector - The ID (symbol) of the monitoring point, or * for all - Groups - Multiple user groups can be selected - At least one of the time interval fields: years, months, days

Optionally you can select an operator, in this case wildcard (*) will be applied for items only under the give operator.

Updating access rule

You can select a rule to update if you click the "Pencil" icon at the end of the specific row.

Path: `/admin/measurements-access-rules/edit?id=[rule identifier]`

You can update all rule datas that you gave on the creating page.

Rule deleting

You can delete a rule if you click the "Trash" icon at the end of the specific row. If you clicked, it shows a confirm window, where you have to approve the deleting.

Path: `/admin/measurements-access-rules/delete?id=[rule identifier]`

Data node

Overview

A *data node* is designed to be run at a data source with the purpose of gathering metering point measurements stored in some third party format, such as a plain text file, spreadsheet or web resource.

It transforms these data to a format compatible with the Environet *distribution node* API, and uploads the results to a distribution node.

The gathering and uploading of data is accomplished by way of uploader plugin configurations, which can then be run by calling a script, typically by a cron job.

Before a data node can start uploading measurements, the metering points and observable properties (for which the plugin will upload measurements) have to be configured beforehand on the Environet distribution node that the plugin will be uploading to.

An API user with will also need to be configured, to authenticate upload requests.

Prepare distribution node to receive data

You can set the following up if you have access to the distribution node admin panel.

API user

Configure a new or existing user with **public ssl key** for . Click here if you need help creating SSL keys. Take note of the **username**, you are going to need it later - along with the **private key** - to configure your data node.

Observed properties

Check that the distribution node has an *observed property* corresponding to each type of data to be uploaded. Take note of the **symbol** value of these for later.

Monitoring points

Finally, define the monitoring points for which data will be uploaded.
You will have to link observed properties to each monitoring point as well.

Setup steps

Before configuring the data node, you need to install the Environet project. The steps to install dependencies and download the Environet source itself, are outlined in the setup section of this document.

Configure data directory

If your data node will be reading data from a file or directory on the system where the node is running, you will have to configure the LOCAL*DATA*DIR environment variable with the path to the directory where the data will be found.
If the data node is going to access the measurements over a network connection, you can skip this step.

- Create an `.env` file by copying `.env.example`
- Uncomment the line containing LOCAL*DATA*DIR (by deleting the # character from the beginning of the line)
- Enter the path to the data directory. For example: On a system where the measurements are stored in csv files in the `/var/measurements` directory, the line would read: `LOCAL_DATA_DIR=/var/measurements`

Creating configurations

Run `./environet data plugin create` to start an interactive script, that will guide you through creating an uploader plugin configuration.

Generated configurations will be saved to the `/conf/plugins/configurations` folder, with the filename provided at the end of the process.

Running a configuration

Run `./environet plugin run [configuration name]` to run an uploader plugin configuration. (If you want to run regularly, you should set up a cron job to execute this command at regular intervals.)

SSL key pair generation tool

To generate an ssl key pair, you can run the command `./environet data tool keygen`.
Private keys should be placed in the `conf/plugins/credentials` directory, which is where the keygen tool will place them, by default.

SSL key pair generation guide

Windows

It is recommended to use the following:

https://itefix.net/dl/free-software/openssl*tool*win*x64*1.0.0.zip

Download and extract it to an optional place. In the package, you can find the executable file at the: *bin/openssl.exe*

If you open the exe, it will prompt a command window, wherein you have to type the following lines:

1. Private key generation:
`genrsa -out private.pem 2048`

2. Public key generation from private key:
`rsa -in private.pem -out public.pem -outform PEM -pubout`

Linux

You have to download one from the following link:

https://www.openssl.org/source/

After that you have to extract and install it.

Here you can find a detailed description about the installation:

https://www.tecmint.com/install-openssl-from-source-in-centos-ubuntu/

After the installation you have to run these commands:

1. Private key generation:
`openssl genrsa -out private.pem 2048`

2. Public key generation from private key:
`openssl rsa -in private.pem -out public.pem -outform PEM -pubout`

Other Environet tools

Useful CLI tools which are available through the './environet' wrapper script.

Generating keypair

Command: `./environet dist|data tool keygen`

Description: With this command you can generate a openssl keypair with an interactive process. The command will ask for: * Destination folder of the key files * Prefix for the key file names

Based on these data a keypair will be generated, and stored in the destination folder.

Sign content

Command: `./environet dist|data tool sign`

Description: With this command you can sign a content (a string, or a file) with a private key. The command will ask for: * Relative path of private key file * How do you want to enter the content (file, or pasted string) * The file path (in case of 'file' mode) or the raw string (in case of 'pasted string' mode) * Generate md5 hash before sigining, or not

Based on the input data a base64 encoded signature will be generated, which can be used in the Authorization header of any api request.