OPCUA north plugin

This plugin is based on the S2OPC open source library.

OPC UA server protocol stack configuration

Connection configuration

This section provides the connection-level configuration details of an OPC UA server.

Attributes definition

Attribute	Description	Expected values	Mandatory
url	The server URL endpoint	“opc.tcp://<IP>:<port>[/sub/path]” e.g. “opc.tcp://10.5.0.1:4841”	YES
appUri	The application URI	e.g. “urn:S2OPC:localhost”	YES
productUri	The product URI	e.g. “urn:S2OPC:localhost”	YES
appDescription	Application description	Any non-empty string.	YES
localeId	The default language Id.	e.g. "en-US", "fr-FR", ...	YES
namespaces	List of namespaces URI, starting at namespace 1. Note that in current implementation, only namespace 1 is used. Thus, the array should contain only one name. (Array is kept for portability)	e.g. for 2 users namespaces (ns=1, ns=2): ["urn:S2OPC:localhost", "urn:S2OPC:localhost_2"]	YES
policies	Array of accepted policies If no security is required: - a single element containing both "None" for Mode and Policy should be used.		YES
policies.securityMode	The security mode	A string among “None”, “Sign” and “SignAndEncrypt” (case insensitive)	YES
policies.securityPolicy	The security policy	A string among “None”, “Basic256”, “Basic256Sha256”, “Aes128Sha256RsaOaep” and “Aes256Sha256RsaPss”	YES
policies.userPolicies	The user policy If no security is required, "Anonymous" should be used	A string among “Anonymous”, “username”, “username_None”, “username_Basic256”	YES
users	A map of 'user':'password'. If no user-authentication is required, it can be an empty object	e.g: {"user" : "password", "user2" : "xGt4sdE3Z+" } e.g: {}	YES
certificates	Note: all certificate files are expected to be provided in subfolders under the $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv folder		YES
certificates.serverCertPath	The Server certificate filename (DER format). The complete path for this file is $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv/server/	e.g. "server_2k_cert.der"	YES
certificates.serverKeyPath	The Server key filename (PEM format) The complete path for this file is $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv/server/	e.g. "server_2k_key.pem"	YES
certificates.trusted_root	The list of trusted root certificates (DER). Can be empty. The complete path for this file is $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv/trusted/	e.g. ["cacert.der"]	NO
certificates.trusted_intermediate	The list of trusted intermediate certificates (DER). Can be empty. The complete path for this file is $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv/trusted/	Same as “trusted_root”	NO
certificates.revoked	The list of revoked certificates (DER). Can be empty. The complete path for this file is $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv/revoked/	Same as “trusted_root”	NO
certificates.untrusted_root	The list of untrusted root certificates (DER). Can be empty. The complete path for this file is $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv/untrusted/	Same as “trusted_root”	NO
certificates.untrusted_intermediate	The list of untrusted intermediate certificates (DER). Can be empty. The complete path for this file is $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv/untrusted/	Same as “trusted_root”	NO

certificates.issued	The list of untrusted issued certificates (DER). Can be empty. The complete path for this file is $(FLEDGE_INSTALL)/data/etc/certs/s2opc_srv/issued/	Same as “trusted_root”	NO

Configuration JSON structure

```
{
   "transport_layer":{
      "url":"opc.tcp://localhost:4841/OPCUA/s2opc",
      "appUri":"urn:S2OPC:localhost",
      "productUri":"urn:S2OPC:localhost",
      "appDescription":"Application description",
      "localeId":"en-US",
      "namespaces":[ "urn:S2OPC:localhost" ],
      "policies":[
         {
            "securityMode":"None",
            "securityPolicy":"None",
            "userPolicies":[
               "anonymous"
            ]
         },
         {
            "securityMode":"SignAndEncrypt",
            "securityPolicy":"Basic256Sha256",
            "userPolicies":[
               "username_Basic256Sha256",
               "username_None"
            ]
         }
      ],
      "users":{ "user":"password", "user2":"xGt4sdE3Z+" },
      "certificates":{
         "serverCertPath":"server_2k_cert.der",
         "serverKeyPath":"server_2k_key.pem",
         "trusted_root":[ "cacert.der" ],
         "trusted_intermediate":[],
         "revoked":["cacrl.der"],
         "untrusted_root":[],
         "untrusted_intermediate":[],
         "issued":[]
      }
   }
}
```

OPC UA server endpoint interface

Connection

A client requires knowledge of parameters provided in previous section to establish a secured channel with the server:

- Endpoint URL
- Server certificate (It is the responsibility of the client to ensure that it is connecting to the expected server)
- User login/password, if applicable.

Endpoint

This section provides the user-level configuration details of an OPC UA server, once a client-server secured connection is established.

The endpoint (see "`transport_layer.url`") is an OPC UA interface and provides several means of use by a client (Browse, Read, Write, Subscribe). The following items allow any connected client to access directly all server data without prior use of browsing, provided that it has knowledge of the PIVOT object it needs and there related types.

Interface specification:

All PIVOT objects are split on OPC UA server in one variable for each field. The variables are organized as follow:

- There is a folder-type node for each PIVOT data. This folder is defined by:
  - NodeId **`ns=1;s=<PIVOT_ID>`**
  - BrowseName/DisplayName **`<PIVOT_ID>`**
  - **`IsOrganizedBy "Root.Objects" (= "i=85")`**
- There is one Variable for each exposed filed of the PIVOT data:
  - NodeId **`ns=1;s=<PIVOT_ID>/<FieldName>`**
  - BrowseName/DisplayName **`<FieldName>`**
  - **`IsOrganizedBy ns=1;s=<PIVOT_ID>`**

TeleMeasure /TeleSignal

Both TeleMeasure and TeleSignal use exactly the same OPC variables organization. They both represent data received from a south plugin and therefore only expose Read-Only Nodes.

FieldName	Type	Reading field	Default value	Details
`Cause`	UInt32 (Read-Only)	do_cot	*Mandatory*	See Cause of Transmission
`Confirmation`	Boolean (Read-Only)	do_confirmation	false	
`Source`	String (Read-Only)	do_source	"process"	"process" | "substituted"
`ComingFrom`	String (Read-Only)	do_comingfrom	*Mandatory*	Any protocol name ("iec104" ,"opcua", ...)
`TmOrg`	String (Read-Only)	do_ts_org	"genuine"	"genuine" | "substituted"
`TmValidity`	String (Read-Only)	do_ts_validity	"good"	Validity of the Timestamp of Value "good" | "invalid" | "reserved" | "questionable"

`DetailQuality`	UInt32
	(Read-Only)
		do_quality	0	OR-Mask of following values:
				
				`0x0001 = badReference`
				`0x0002 = failure`
				`0x0004 = inconsistent`
				`0x0008 = innaccurate`
				`0x0010 = oldData`
				`0x0020 = oscillatory`
				`0x0040 = outOfRange`
				`0x0080 = overflow`
				`0x1000 = test`
				`0x2000 = operator blocked`

`TimeQuality`	UInt32
	(Read-Only)
		do_ts_quality	0	OR-Mask of following values:
				
				`0x01 = clockFailure`
				`0x02 = clockNotSynch`
				`0x04 = leapSecondKnown`

`SecondSinceEpoch`	UInt64 (Read-Only)	do_ts	0	Number of seconds since Linux Epoch
`Value`	(See below) (Read-Only)	do_value do_value_quality	*Mandatory*	(See below)

Common notes

- With **`<PIVOT_ID>`** as provided in **`exchanged_data.datapoints[].pivot_id`** section configuration.
- All read-only variables have a **`OpcUa_BadWaitingForInitialData`** quality (**`0x80320000`**) initial value until a valid value is received from FledgePower.
- All non-mandatory values will be set automatically by the server if not received from a south device.
- If a mandatory value is missing, then the whole PIVOT object is **not** updated.
- As a standard OPC UA server, all functional data are organized under the **`Root.Objects`** node of namespace 0 (nodeId = **`i=85`**).
- All data are stored under the namespace 1. Its URI is configured in protocol "`namespaces`" parameter.

Value content and metadata

PIVOT timestamp

The **`ns=1;s=<PIVOT_ID>/Value`** variable contains the pivot timestamp value (t.`FractionOfSecond` + t.`SecondSinceEpoch`) is converted to OPC-UA timestamp (Unit= number of 100 nanosecond since Jan 1st,1600).

The timestamp is not optional in OPCUA. Thus, in case the timestamp were not provided by south layers, the OPC UA north plugin will set the timestamp to 0.

PIVOT value validity

The **`ns=1;s=<PIVOT_ID>/Value`** variable contains the pivot value **`<Root>.<type>.q.validity`** field. It is represented as the OPC UA Quality of the variable, using the following conversion:

PIVOT Validity	OPC UA quality
good	OPC_UA_GOOD
invalid	OPC_UA_BAD
reserved	OPC_UA_BAD (not used)
questionable	OPC_UA_UNCERTAIN

TeleControl

The TeleControls represent commands received from an OPC UA client -connected to the North plugin- that must be sent to a south service. Therefore, they expose Read/Write variables that will be written by a client, except for the feedback of operation, which is Read-Only (Variable **`Reply`**).

As a single PIVOT TC contains the information provided in several OPC variables, the following procedure has been defined to ensure atomicity of the operation. An OPC UA client must:

- first write all relevant parameters in the TC variables. This can be done in a single OPC UA 'write' operation, The plugin will **not** check that all fields have been updated. This is under the responsibility of the OPC UA client.
- then activate the command **`Trigger`**. This must be executed after the previous write command was successfully completed.

OpcUa node (**`ns=1;s=<PIVOT_ID>/`**...)	Type	Reading field	Details
	String	co_id	co_id = PIVOT ID
	String	co_type	co_type = "opcua_inc" | "opcua_apc" | "opcua_bsc" Other types not supported in current version.
`Value`	(See below)	co_value	(See below)
`Reply`	Boolean	ro_reply / ro_id	Return value from the south service Quality: ▪ reset to OpcUa_BadRefreshInProgress when the **`Trigger`** variable is written. ▪ set to OPC_UA_GOOD when the acknowledgement is received. (The "Reply" field is updated using the "opcua_reply" reading (See Filter plugin OPCUA to Pivot). Value : ▪ True: The command has been acknowledged (ro_reply=1) ▪ False : The command has not been acknowledged (ro_reply=0 or any other value) Note that there is no timeout handled in the plugin itself. If the remote equipment never acknowledges the request, **`Reply`** may never get OPC_UA_GOOD quality.
`Trigger`	UInt8	co_test co_se	Writing this node triggers the TC with current values, current timestamp and the command flags depending on the **`Trigger`** Value: • Bit 0 (0x01) : Test (bit set <=> co_test=1) • Bit 1 (0x02) : Select (bit set <=>co_se =1). Note : co_se=0 is "Execute" and co_se=1 is "Select" Examples: • Writing "0" triggers an Execute TC with Test =0 • Writing "3" triggers a Select TC with Test =1
	Timestamp	co_ts	co_ts = timestamp (Number of seconds since Linux Epoch). The value is defined by local time when **`Trigger`** is written.

PIVOT variant value

The base type of the value itself is static and depends on the configuration provide in "`exchanged_data`" section.

PIVOT Type	TypeId of `exchanged_data.datapoints.pivot_type`	BaseDataType of `ns=1;s=<PIVOT_ID>/Value`	Note
SPSTyp	opcua_sps	Boolean_Id (=1)	
DPSTyp	opcua_dps	String_Id (=12)	Enum is shown as string encoded. See CDC double point status (DPSTyp)
BSCTyp	opcua_bsc	String_Id (=12)	
MVTyp	opcua_mvi	Int32_Id (=6)	In that case, the "mag.f" field is ignored
MVTyp	opcua_mvf	Float_Id (=10)	In that case, the "mag.i" field is ignored
SPCTyp	opcua_spc	Boolean_Id (=1)	
DPCTyp	opcua_dpc	Byte_Id (=2)	
INCTyp	opcua_inc	Int32_Id (=6)	
APCTyp	opcua_apc	Float_Id (=10)	
BSCTyp	opcua_bsc	String_Id (=12)	