Sending OPC Tag Data with Transmission

Prerequisites:

- Installing the Java Runtime Environment
- Installing Ignition
- Installing the following MQTT Modules
 - MQTT Distributor
 - v2.1.X if using Ignition 7.7.x or 7.8.x
 - v3.1.0 or greater if using Ignition 7.9.X
 - MQTT Engine
 - v2.1.X if using Ignition 7.7.x or 7.8.x
 - v3.1.0 or greater if using Ignition 7.9.X
 - MQTT Transmission
 - v2.1.X if using Ignition 7.7.x or 7.8.x
 - v3.1.0 or greater if using Ignition 7.9.X
- A device that supports Modbus over TCP

Overview:

Transmission is an MQTT module for Ignition that can convert Ignition tag data and tag change events into MQTT messages to be consumed by MQTT Engine or other MQTT clients. This tutorial will show how to configure MQTT Transmission to send OPC tag data in Ignition as MQTT messages via MQTT Distributor to MQTT Engine where they will be displayed.

The topology of this example shows MQTT Distributor, MQTT Engine, and MQTT Transmission all running in the same Ignition instance. This is done for simplicity of the tutorial. But, this isn't required or even intended to be a real use case. In a more realistic scenario MQTT Transmission and MQTT Engine would be located on separate machines.

Variations:

This tutorial shows how to use OPC tags and MQTT Transmission to generate MQTT messages based on tag change events. However, the tag source does not have to be an OPC tag. Instead, as long as the tag structure for MQTT Transmission is followed, any Ignition tag can be used to generate MQTT messages and/or be controlled via MQTT messages.

Sending OPC Tag Data with Transmission:

The first step is to configure the tag provider in Ignition in a way that MQTT Transmission understands. Start by configuring your OPC server, client, and tags. This can be done using the Inductive Automation documentation here. Once this is done, the Tag Provider needs to be set up in Ignition via the Ignition Designer. Using a Web Browser, browse to the Ignition Gateway on your Ignition Gateway. If it is running on your development machine, that is: ht tp://localhost:8088. You should see this:

) (i) localhost:8088/main/web	v E3 C Q Search	USER MANUAL SUPP
gnition.	A HOME de STATUS O CONFIGURE	Launch Designer
rial Version 🗉	:00:53 We're glad you're test driving our software. Have fun.	Activate Ignition
	Welcome to the Ignition Gateway	-
	hide this panel Congratulations, Ignition is installed and running! Whether you need an OPC-UA server, a SQL datalogger, an HMI, or a full-blown SCADA or MES solution, Ignition can handle it. Here are a few common steps to help get you started: Legin to the configuration section. The default username and password are: admin / password Change the password or configure your own authentication profile in the Security > Authentication section Congratulations. Congratulations and the security is a section. Congratulation section. Congratulations and password or configure your own authentication profile in the Security > Authentication section Congratulations. Congratulations and the security is a section. Congratulation section. Congratulation section. Congratulations and the security is a section section. Congratulation section. Congratul	
	Connectivity is what gliniton is all about: Connect to a FLC bit your network or using the internal of Conserver, or a situ party OFC server. Don't have a PLC handy? No problem, you can skip this step or use one of the simulator drivers. Connect to a database. Database connectivity is at the heart of Ignition's most powerful features, like Transaction Groups and SQLTags Historian. If you have a SQL database, you can greatly increase Ignition's capabilities by adding a connection to it. No database? You can come back to this step later or skip it entirely.	
	Launch the Ignition Designer C ⁶ . This is where the magic happens. Create a project and add windows and transaction groups. Besides the usual status and control functionality, take advantage of advanced charting, tables and reporting capabilities. Launch a client. Or two. Or twenty.	
	web-launched clients can be launched anywhere on your network from the panel below. With lgnition, you don't have licensing restrictions to limit you.	

Near the upper right corner, click 'Launch Designer'. This will open the following window after downloading the JNLP file and executing it. Note the default username/password is admin/password. Type those into the appropriate fields and click 'Login'.

Use of this application is subject	desigr	ner	
to the acceptance of the terms and conditions set forth in the	Username Password		——X
accompanying <u>incense agreemen</u>		Login	Cancel
Copyright © 2003–2016 Inductive Auto	mation All rights reserv	ved.	inductive automation

This will bring you to a new Window where you can select an Ignition Project or create a new one. Create a new project by giving it a name and clicking 'Create New Project'.

	Open/Create Project
Ignition by inductive automation	
Now Project	Project Name MyProject
	Project Title
Open Recent	Authentication Profile default
	Default Database
	Default Tags Provider default 💽
	Project Template Blank 🔍
	Description
	Create New Project

In this example we're going to use the "default" tag provider within Ignition Enterprise. Expand 'All Providers' in the Tag Browser and select 'default'. Note that if Ignition Edge is being used, the tag provider will be named 'edge'.



With 'default' (or 'edge') selected, click the 'OPC' icon in the Tag Browser icon list:



This will open a new window as shown below. If the OPC server and client were set up and configured properly, you should see something similar to the following:



Note there is a device with an attached PLC and two sets of registers. Yours will look different based on the device you are using and how it is configured. At this point, we can select the device (CLTEST002DIRECTOR in this case) and drag it into the Tag Browser under the 'default' Tag Provider. This is shown below:

Tag Browser		급 무 ×
🗚 🛞 🔢 🔹 I 📎 📼 🕷 📰 💩 I 🥔 d	ð.	
Tag	Value	Data Type 🔄
🕀 🦳 Tags		
🕂 🛄 System		
🕂 闻 Client		
占 🏐 All Providers		
🗗 🔄 default		
🕂 🔯 Data Types		
🗄 🔄 Edge Nodes		
🗄 🔄 Group 1		
Ė- 🔄 PLC1		
⊨- 🔄 1_V1-1_V10		
	234	Int2
	888	Int2
	333	Int2
	444	Int2
	5,555	Int2
	-678	Int2
	777	Int2
	-128	Int2
	999	Int2
<u></u>	1,010	Int2
Ė- 🔄 1_Y1−1_Y5		
		Boolean
		Boolean
🕂 💿 1_Y3		Boolean
		Boolean
│		Boolean

The first level of folders is very important in terms of layout and how the tags will be understood and represented by MQTT Transmission. These are the rules:

- For terminology reference, please review the Sparkplug specification if you have not done so.
- Under the Tag Provider, the first folder must be exactly 'Edge Nodes'
- Under the 'Edge Nodes' folder a 'Group ID' must be next. This can be anything you want but realize it represents a group of edge nodes. You can have as many group IDs as you want.
- Under 'Group ID' you can one or more Edge Nodes.
- Under each Edge Node, you can have one or more devices.
- Under each device, you can have any depth of folder/tag structure to represent the tags.

In our example, we have a single Edge Node of 'CLTEST002DIRECTOR', under it a single Device called PLC1, and under it, 15 registers in two different folders. We could rearrange this by renaming and moving folders and tags to a different representation as desired. The layout below is also valid as was done simply by moving/renaming the tags shown above.

Tag Browser		⊡ ₽ ×
🙈 🚯 🔢 📼 I 🖏 🖛 😻 📰 💩 I 🌛 🌛	i	
		A
E- 🕥 All Providers		
E- 🔍 default		
🕀 🔯 Data Types		
E- 🔄 Edge Nodes		
E- 🔄 Kansas City		
🔄 🔄 Cirrus Link Lab		
È- 🔄 PLC1		
🗗 🔄 My Analogs 1		
│ │	234	Int2
│ │	888	Int2
│ │ │ │ │ │ │ │ ↓-	333	Int2
	444	Int2
	5,555	Int2
🖨 🔄 My Analogs 2		
	-678	Int2
📘 🔤 🕂 🔂 🛓 🗍	777	Int2
	-128	Int2
	999	Int2
↓	1,010	Int2
- Sin My Booleans 1	,	
- 🕤 Sub Booleans		
		Boolean
F⊢ Soolean 2		Boolean
+- \ssac 1 Y1		Boolean
\downarrow		Boolean
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓		Boolean
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓		Boolean
H Soolean 5		Boolean
		Boolean

Note while the folders and tags were moved and renamed, the required basic structure stayed the same with:

default

lt ◦ Edge Nodes ■ Group ID (1 to n) • Edge Node ID (1 to n) ◦ Device ID (1 to n) ■ Free form folders and tags

With our tags set up as we want them, we now must configure MQTT Transmission. Do so by browsing to the the Configure section of the Ignition Gateway web UI in the left-hand navigation pane. Find and select 'Configure MQTT Transmission Settings':

In this example we're using the default MQTT Server of MQTT Distributor:

Ignition	Ignition Gateway × +	
← → ♂ ☆	0 localhost:8088/main/web/config/mqtttransmission.settings?48	⊻ II\ 🖸 📽 🛈 Ξ
Ignition	A HOME A STATUS CONFIGURE	USER MANUAL SUPPORT ≟admin Sign Out Launch Designer 🖌
q Search	Trial Version 0:41:01 We're glad you're test driving our software. Have fun.	Activate Ignition
SYSTEM Overview Backup/Restore Licensing Modules Projects Redundancy Gateway Settings NETWORKING Gateway Network	MQTT Transmission Settings General Servers Sets Transmitters Records Name URL Server Type Server Set Username Certificate Files Connected Chariot SCADA tcp://localhost:1883 MQTT_Distributor Default admin 0 of 0	delete edit
Email Settings SECURITY Auditing Users, Roles Service Security Security Zones	→ Create new MQTT Server Note: For additional details on configuring MQTT Transmission, see the documentation here	
DATABASES Connections Drivers		

Under the Transmitters tab, we're also using the defaults:

←)→ ୯ û	I i localhost:8	088/main/web/config/mqtttransmission.settings?49 😶 🔂 👱 🔢 🗊 🕲
SYSTEM Overview Backup/Restore Licensing	MQTT Transmi	ission Settings
Modules Projects Redundancy Gateway Settings	General Serve	rs Sets Transmitters Records
NETWORKING Gateway Network Email Settings	Default Cus	
SECUDITY	Configuration	
Auditing Users, Roles Service Security	Name	Default Transmitter Name for the Default Transmitter
Security Zones	Enabled	Z Enable Default Transmitter
DATABASES Connections Drivers	Tag Provider Name	default The Name of the tag provider
Store and Forward ALARMING General	Tag Pacing Period	1000 The waiting period in milliseconds after an initial tag change event before publishing all changed tags
Journal Notification On-Call Rosters Schedules	Set	Default The MQTT Server Set to use with the Default Transmitter
TAGS History Realtime	Discovery Delay	0 The Discovery Delay in milliseconds. This is useful when using MQTT Engine as the tag provider (default: 0)
OPC-UA SERVER Certificates Devices Settings	History Store	- none - The MQTT Transmission History Store to use with the Default Transmitter
DPC CONNECTIONS	In-Order History	Flush history in-order (synchronously) before live data resumes (default: false)
Quick Client	Aliased Tags	□ Use aliases for tag names to optimize payload sizes when publishing data
MOBILE Settings	Compression	NONE The algorithm to use for compressing payloads before publishing
ENTERPRISE ADMINISTRATION	Block Commands	Black incoming commande (united to Edge Mode and Deuice Tare

Note the Tag Provider Name is the same Tag Provider we build out our Edge Nodes from in the Tag Browser earlier. Now we can go back to Designer and force MQTT Transmission to update. MQTT Transmission does not dynamically look for changes in the tag structure and update them. If it did, you could end up with a lot of improperly structured data while the changes to the tag tree are being made. So, the update must be forced via 'Refresh'. This is a tag under the MQTT Transmission tag provider as shown below in the Tag Browser.



In order to refresh, Designer must be in read/write and preview mode. Do so by selecting these two buttons in the top menu of Designer:



Once this is done, click 'Refresh' in the Tag Browser. This will force MQTT Transmission to read the default Tag Browser tree, find 'Edge Nodes', and begin sending MQTT messages based on tag change events. You can see this by clicking the refresh icon in the Tag Browser menu:



At this point, you should be able to expand the MQTT Engine tag provider and see all of the tags in MQTT Engine:

Tag Browser

.........

다 다 X

🙈 📵 🔢 🔹 I 😓 👻 📅 💩 I 🌲 🎝		
Tag	Value	Data Type 🔺
🕀 🦲 Tags		
🔃 🛄 System		
🔃 🛄 Client		
🖻 🔄 All Providers		
🕂 🔯 default		
🕂 🛄 MQTT Transmission		
E- 🔄 MQTT Engine		
🗗 🔄 Edge Nodes		
E- 🔄 Kansas City		
🖻 🔄 Cirrus Link Lab		
🕂 🛄 Node Control		
🕂 🎑 Node Metrics		
E- 🔄 PLC1		
🕂 🎑 Device Metrics		
🖻 🔄 My Analogs 1		
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	234	Int4
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	888	Int4
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	333	Int4
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	444	Int4
±- % 1_V5	5,555	Int4
🗗 🔄 My Analogs 2		
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	-678	Int4
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	777	Int4
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	-128	Int4
⊕- <u></u> 1_V9	999	Int4
<u>⊡</u> _ _ _ _ _ _ _ _ _ _ _ _ _	1,010	Int4
🖻 🔄 My Booleans 1		
E- Sub Booleans	_	
E Boolean 1		Boolean
E- Solean 2		Boolean
- · · · · · · · · · · · · · · · · · · ·		Boolean
□ □ □ <u>1 Y2</u>		Boolean
⊕- № 1_Y3 ↓		Boolean
₽- <u>></u> 1_Y4		Boolean
\square		Boolean
Engine Metrics		
Hessage Diagnostics		

In addition to the tags being displayed in Engine, they are also writable if this enabled in MQTT Engine. By default, MQTT Engine blocks command messages from being sent to devices. To enable this feature, in the Ignition web console browse to the MQTT Engine Module Settings.



Make sure the "Block Node Commands" and "Block Device Commands" settings are disabled, as shown below.

ost:8088/main/web/config/r	nqttengine.settings?61		C Q Search	
estore	MOTT Engine	- Settings		
	General Se	rvers Namespaces		
cy				
Settings	Conoral Co	ttinge		
G	General se	lungs		
letwork				
ings	Main			
	Enabled	C Enable the MQTT Engine		
		(66.66.6.7.66.)		
urity				
nes	Primary Host ID			
		The Primary Host ID to allow connecting clients to ensure they remain connected to this application (optio	nal)	
د				
	Group ID Filters			
orward		A comma separated list of Group IDs to listen for (optional)		
	Chariot Access			
	charlot Access			
1	Chariot Cloud			
sters	Access Key	The optional Chariot Cloud Access Key used for Cirrus Link hosted Chariot MOTT Servers (optional)		
	Chariot Cloud			
	Secret Key	The optional Charlot Cloud Secret Key used for Circus Link bested Charlot MOTT Servers (optional)		
		The optional charlot cloud Secret Rey used for cirrus Link hosted charlot mg in Servers (optional)		
FD				
	Miscellaneous			
	Block Node			
	Commands	Enable blocking of outbound edge node tag writes		
TIONS				
	Block Device Commands	Enable blocking of outbound device tag writes		
t				
		Ignore		
	File Policy	ignore t		
		The policy for handling incoming files		
ION				
ion	File Location			
		The directory to store files in when using the "Store" file policy (optional)		
UNCTION				
		Save Changes		

With this enabled and Designer in read/write and preview mode, you can write to the outputs of the modbus device from the Tag Browser:



Note there is some delay in the response. This is due in part to MQTT Transmissions 'Tag Pacing Period'. This is the delay for MQTT messages to wait before being sent to allow multiple change events to buffer before putting them into a single MQTT message. This can be changed in the MQTT Transmission module configuration in the Ignition web console.

Additional Resources

- Inductive Automation's Ignition download with free trial https://inductiveautomation.com/downloads/
- Azure Injector download with free trial
- https://inductiveautomation.com/downloads/third-party-modules • Questions about this tutorial?
 - ° Check out the Cirrus Link Forum: https://forum.cirrus-link.com/ Contact support: support@cirrus-link.com
- Sales questions
 - Email: sales@cirrus-link.com
 Phone: +1 (844) 924-7787
- About Cirrus Link
 - https://www.cirrus-link.com/about-us/