Getting Started: Azure Injector Quick Start

Prerequisites

- Knowledge of Ignition and Module installation process: Cloud and MQTT Module Installation.
- An existing Microsoft Azure account with an active IoT Hub and a registered device.
 - Documentation on creating an IoT Hub and registering an new device can be found here

Summary

This tutorial will provide step-by-step instructions for the following:

- · Installing the Ignition Gateway Industrial Application Platform with the Azure Injector Module
- · Configuring the Azure Injector Module to connect to an existing Azure IoT Hub
- Publishing live Tag data and events to the connected Azure IoT Hub

Upon completion of this module you will have an Ignition Gateway connected and publishing live Tag data to an Azure IoT Hub.

Architecture



Tutorial

Step 1: Download and Install Ignition

Ignition is an Industrial Application Platform that can be used to create SCADA and HMI solutions. A fully functional Ignition system can be downloaded and run in trial mode. Using Ignition as a tool in this way, we can install the Sparkplug MQTT Modules and observe everything working.

Go to the Inductive Automation download page and download the desired version (select version from the 'Ignition Version' dropdown) of the Ignition installer for Windows, Linux or MacOS; https://inductiveautomation.com/downloads/archive

Once the Ignition installer has been downloaded, follow the instructions provided by Inductive Automation to install and startup Ignition.

Step 2: Download and Install the Cirrus Link Azure Injector Module

Go to the Inductive Automation download page again and scroll down to the Third Party modules section. Find the Cirrus Link modules section and download the Azure Injector Module.

https://inductiveautomation.com/downloads/archive. The download links should look similar to what is shown below.

C 🗅	i A https://inductiveautomation.co	① ▲ https://inductiveautomation.com/downloads/ignition					
	Third Party Modules All third party Ignition modules require the Ignitio To install third party modules: 1. Install Ignition: See Ignition installation guide 2. Once Ignition is installed, download the modu	on platform to be installed. Ile and install it in the Ignition Gateway: See module insta	allation guide				
	Cirrus Link Solutions MQTT Modules for Ign	ition					
	See the release notes and usage documenta	tion for all Cirrus Link modules here.					
	MQTT Distributor Module	MQTT-Distributor-signed.modl (12MB)	Version: 3.4.4				
	MQTT Engine Module	MQTT-Engine-signed.modl (13MB)	Version: 3.4.4				
	MQTT Transmission Module	MQTT-Transmission-signed.modl (9MB)	Version: 3.4.4				
	MQTT Remote Management - Agent	MQTT-Remote-Agent-signed.modl (18MB)	Version: 3.4.4				
	MQTT Remote Management - Manager	MQTT-Remote-Manager-signed.modl (9MB)	Version: 3.4.4				
	AWS Injector Module	AWS-Injector-signed.modl (19MB)	Version: 3.4.4				
	Azure Injector Module	Azure-Injector-signed.modl (12MB)	Version: 3.4.4				
	Opto 22 groov EPIC and SNAP PAC Driver Module	Opto-22-groov-EPIC-and-SNAP-PAC-Driver- signed.modl (9MB)	Version: 3.4.4				
	EFM Emerson ROC Driver Module	EFM-Emerson-ROC-Driver-signed.modl (10MB)	Version: 3.4.4-beta				
	MQTT Recorder	MQTT-Recorder-signed.modl (9MB)	Version: 3.4.4-beta				
	encertaine areas and a construction						

Step 3: Configure the MQTT Modules

Once you have Ignition installed and running, and the Azure Injector module downloaded, browse to the Ignition Gateway console (e.g. http://localhost: 8088). Login using the default credentials of <u>admin/password</u>. Click on Configuration tab and then click on the Modules tab on the left side of the page. Scroll to the bottom of the Modules section and click on the Download/Upgrade modules button. When prompted, select the Azure Injector module from the file browser and install it. When complete, the Ignition Gateway Web UI module section should look similar to what is shown below:

$) ightarrow$ C* $rac{1}{2}$	localhost:8088/main/web/c	config/system.modules?55	💟 🏠 🔍 Sea	rch				Ţ
QUENTIAL FUNCTION IARTS Settings	Tag Historian	2.9.5 (b2017111615)	Turns any database into a powerful historian that can store and drive data in Ignition.	Trial	Running	More 🔻	restart	
URE INJECTOR Settings	UDP and TCP Drivers	4.9.5 (b2017111615)	Drivers for receiving and parsing UDP or TCP packets.	Trial	Running	More 🔻	restart	
Ŭ	User Manual	4.9.5 (b2017111615)	Provides an offline version of http://docs.inductiveautomation.com.	Free	Running	More 🔻	restart	
	Vision	9.9.5 (b2017111615)	A module that provides web-launched HMI/SCADA clients.	Trial	Running	More 🕶	restart	
	Cirrus Link Solution	ns						
	Cirrus Link Solution View Certificate Name	15 Version	Description	License	State			
	Cirrus Link Solution View Certificate Name Azure Injector	Version 3.3.4-SNAPSHOT (b0)	Description An Ignition Tag to Azure Bridge	License Trial	State Running	More 🕶	restart	
	Cirrus Link Solution View Cerificate Name Azure Injector → Install or Upgrade a Moduli	NS Version 3.3.4-SNAPSHOT (b0)	Description An Ignition Tag to Azure Bridge	License Trial	State Running	More 🔻	restart	
	Cirrus Link Solutior View Certificate Name Azure Injector → Install or Upgrade a Modul Note: For details about a modul	IS Version 3.3.4-SNAPSHOT (b0) le le's status, see the Module Status p	Description An Ignition Tag to Azure Bridge page.	License Trial	State Running	More 🔻	restart	
	Cirrus Link Solutior View Certificate Name Azure Injector → Install or Upgrade a Modul Note: For details about a modul	NS Version 3.3.4-SNAPSHOT (b0) le le's status, see the Module Status p	Description An Ignition Tag to Azure Bridge page.	License Trial	State Running	More v	restart	

Select the "AZURE INJECTOR" "Settings" link on the lower left of the page to navigate to the Azure Injector Module's configuration page. A detailed explanation of each configuration tab can be found here. For this tutorial, we will only be adding a new Azure IoT Hub Setting.

) → C 🏠	localhost:8088/main/web/e	config/azure.settings?47	💟 🕁	Q Search	5
nition	•	-		4	USER MANUAL SUPPO
gnition.	A HOME 🔟 STATUS	¢ CONFIGURE			Launch Designer 💡
	Trial Version 1:18:57	We're glad you're test driving our software. Have fun.			Activate Ignition
TEM verview ackup/Restore icensing lodules rojects edundancy ateway Settings WORKING ateway Network mail Settings	Azure Injector Set	tings ubs Sets Tag Agents ettings Protocol Set Certificates	Store & Forward Enabled	Sta	rtus
URITY Juditing Jsers, Roles iervice Security iecurity Zones	→ Create new Azure IoT H	ıb Setting			
ABASES connections Irivers tore and Forward					
RMING eneral ournal otification					

Click on the "Create new Azure IoT Hub Setting..." link to bring up the following configuration form:

) → 健 @	i localhost:8088/mair	n/web/config/azure.settings?58 👽 🔂 🛛 🔍 Search	1
nition			USER MANUAL SUPPO
			🚨 admin 🛛 Sign 🤇
gnition.	A HOME Ju STATU		Launch Designer 2
Search	Trial Version 🕕		Activate Ignition
YSTEM			
Overview	A Turo Inicato	- Cattinga	
Backup/Restore	Azure injector	Settings	
Licensing			
Projects	General Azu	re IoT Hubs Sets Tag Agents	
Redundancy			
Gateway Settings	Now Azuro I	aT Hub Satting	
TWORKING	New Azure I	or hub setting	
Gateway Network			
mail Settings	Main		
יווסודע			
luditing	Setting Name	TestSetting	
Jsers, Roles		A friendly name for this Azure IoT Hub setting	
Service Security			
Security Zones	Connection String	HostName=SomeHostNameTest.azure	
TABASES	connection outring	The connection string used for establishing a connection with the IoT Hub.	
Connections			
Drivers		MQTT T	
Store and Forward	Protocol	The IoT Hub client protocol	
ARMING		(default: MQTT)	
General			
Journal	Set	Default 🔻	
Notification		The Set this IoT Hub is associated with	
On-Call Rosters			
schedules	Certificates	Browse No file selected.	
SS		FIIE5.	
listory			
ceattime	Store & Forward		
C-UA SERVER			
Certificates	Store & Forward Enabled	Enable Store and Forward capabilities for this stream	
Devices	Linubicu	(neunaur unva)	
settings	Store & Forward	Choose One	
C CONNECTIONS	Store & Forward Type	The Time of this Plane & Plane and a plane	
Somore		The Type of this Store & Porward mechanism	

For the Setting Name you can enter any unique identifier, we will use "TestSetting".

For the Connection String you will need to obtain an IoT Hub Device Connection String from your Azure IoT Hub. This can be found by navigating to a provisioned Device within the Device Explorer of the IoT Hub that you wish to use (see below).



The Protocol, Set, and Certificates fields can be left with the default values. Click on "Create New Azure IoT Hub Setting" to finish creating the new configuration setting.

) → C û	localhost:8088/main/	web/config/azure.settings?59)		Search	1
nition gnition by inductive automation	A HOME A STATUS	CONFIGURE				USER MANUAL SUPP Ladmin Sign (Launch Designer)
	Trial Version 111	0:28 We're glad you're test d	iving our software. Have	fun.		Activate Ignition
STEM Overview Backup/Restore Licensing Modules Projects Redundancy Gateway Settings TWORKING Gateway Network Enail Settings	Azure Injector General Azure Azure IoT Hu Successfully cre	Settings IIOT Hubs Sets T b Settings ated new Azure IOT Hub Se	ag Agents			
CURITY	Setting Name	Protocol Set	Certificates	Store & Forward Enabled	Status	
Auditing Users, Roles Service Security Security Zones	TestSetting	MQTT Defaul		false	Connected	delete edit
TABASES Connections Drivers Store and Forward	Create new Azure	or nuo setting				
ARMING						

Now the Azure Injector module is connected to the IoT Hub and ready to push Tag data.

If you click on the "Tag Agents" tab you will see that out-of-the-box the Azure Injector module will have one default Tag Agent defined. For this tutorial we will not need to make any configuration changes to the Tag Agents.

$ ightarrow$ C* $rac{1}{2}$	🛈 localhost:8088/main/web/config/azure.settings?60 🛡 🏠 🗌 🔍 Search	1
on		USER MANUAL SUPPO
		Ladmin Sign C
by inductive automation	A HOME JE STATUS CONFIGURE	Launch Designer
	Trial Version 1:09:47 We're glad you're test driving our software. Have fun.	Activate Ignition
verview ackup/Restore censing	Azure Injector Settings	
odules ojects	General Azure IoT Hubs Sets Tag Agents	
edundancy ateway Settings	Tag Provider Name Tag Path Convert UDTs Set Auto-discover Tags Group ID Edge Node ID Device ID	
NORKING ateway Network nail Settings	default Edge Nodes false Default false	delete edit
JRITY	→ Create new Tag Agent Settings	
sers, Roles ervice Security		
ecurity Zones		
ABASES		
rivers ore and Forward		
RMING		

The Default Tag Agent will monitor tags that are in the "Edge Nodes" folder of the "default" Tag Provider. In the next step we go into more detail about the tags in this folder.

Step 4: Use Ignition Designer to Examine the Initial Tag Structure

With Ignition running and the Azure Injector module loaded now we can open the Ignition Designer to create/observe the initial Tag structure. Regardless of the OS Ignition is running on, there is a "Launch Designer" button on the Ignition Gateway Console. From here you can launch your Designer on any machine. This is shown below. The default credentials for the designer are the same as the Gateway Console, admin/password. Once you have logged into the Designer enter a new project name and open the project. The project name that we used for this tutorial is simply called "Tutorial1".

Ignition - Ignition Gateway * +	
O localhost:8088/main/web/home?13	El C Search
inition	JSER MANUAL SUPP
Spinicities automation A HOME di STATUS & CONFIGURE	Launch Designer 3
Frial Version 1:04:01 We're glad you're test driving our software. Have fun.	Activate Ignition

After Designer opens, you will see the default Designer screen as shown below.



With the Azure Injector module installed in Ignition, a new folder is created under the "All Providers" folder and is called "Azure Injector". This folder will contain both information tags about the module's version and state, as well as control tags for refreshing the module and it's Tag Agents.



Next, we need to create a folder structure where we will create a virtual Edge device and some tags to be published by the Azure Injector module. When the Azure Injector module is installed in Ignition, a folder is automatically created in the Ignition tag structure with the following path:

• All Providers/default/Edge Nodes

	Tutorial1 -	Ignition - Ignition Designer	
File Edit View Project	t Component Alignment Shape Tools	Heln	
		[2622] ● ● ● 目目目目目目目目目目目目目目目目目目目目目目目目目目目目目目	▲▲▲ ╞∃ 판 止⊕ ≑ [*]
Project Browser		급 무 ×	
🕀 🌐 Global			Getting Started
Tag Browser		급 무 ×	
#8 🕵 🔢 📼 I 🗞 📼 🐚	5 📅 💩 I 🚓 🎝		
Tag	Value	Data Type	
 			2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
All Flowlers			- New Main Window
			<
::			(-,-) % 197 / 910 mb

Step 5: Use Ignition Designer to Create New Tags

For this tutorial, right click on the Edge Nodes folder and create a new folder called Tutorial Group. Then right click on the Tutorial Group folder and create another new folder called Tutorial Edge Node. Finally, right click on the Tutorial Edge Node folder and create another new folder called Tutorial Device. This folder structure creates the same hierarchy that is described in the Sparkplug B specification of Group ID, Edge ID, and Device ID.



With this folder structure in place, now we can create some memory tags of various data types to publish. Right click on the Tutorial Device folder and select 'New Tag'/Memory Tag'. In the tag editor change the Name of the tag to "Boolean001", and change the Data Type to Boolean. Follow this same procedure for new memory tags called "Integer001" of type Integer, "Float001" of type Float, and "String001" of type String. The resulting folder structure should look as follows.



Step 6: Use Ignition Designer to Publish Tag Data (Current Tag Values)

Now that we have a folder structure with some tags we can refresh the Azure Injector module. Make sure that the Ignition Designer has read/write communications turned on by selecting Project/Comm Read/Write.



To refresh the default Tag Agent with the folder structure we've created, open the folder "All Providers/Azure Injector/Azure Injector Control" and click on the Refresh Boolean. Note the Boolean tag will not change to true. This is really a one-shot and as a result, the tag will not change to true.

When this happens, the Tag Agent will scan the "Edge Nodes" folder and find the new Memory Tags that we have created, construct JSON payloads representing those tags with their current values and publish the payload to the Azure IoT Hub that we have configured.



The Azure Injector Tag Agent will publish two JSON payloads to the Azure IoT hub. The format of these messages closely follows the Sparkplug B Specification's payload structure.

The first payload represents the Edge Node and will contain the following:

- The Sparkplug elements: Namespace, Group ID, Edge Node ID. They will be grouped under "topic".
- A "timestamp" for when the payload was constructed.
- A "bdSeq" sequence number to track the "session" of the Tag Agent.
- Any Edge Node tags defined in the "Tutorial Edge Node" folder (in our example we have none).

It will look something like this:

First Payload

```
{
 "topic": {
   "namespace": "spBv1.0",
    "groupId": "Tutorial Group",
   "edgeNodeId": "Tutorial Edge Node"
 },
  "payload": {
    "timestamp": 1504739061495,
    "metrics": [
     {
       "name": "bdSeq",
       "timestamp": 1504739061495,
       "dataType": "Int64",
       "value": 0
     }
    ],
    "seq": 0
 }
}
```

The second payload represents the Device and will contain the following:

The Sparkplug elements: Namespace, Group ID, Edge Node ID, Device ID. They will be grouped under "topic".
A "timestamp" for when the payload was constructed.
Any Device tags defined in the "Tutorial Device" folder.

It will look something like this:

Second Payload

```
{
 "topic": {
   "namespace": "spBv1.0",
   "groupId": "Tutorial Group",
   "edgeNodeId": "Tutorial Edge Node",
   "deviceId": "Tutorial Device"
 },
  "payload": {
   "timestamp": 1504739061501,
    "metrics": [
     {
       "name": "Boolean001",
       "timestamp": 1504739061546,
        "dataType": "Boolean",
        "properties": {
         "Quality": {
           "type": "Int32",
          "value": 192
        }
        },
        "value": true
      },
      {
        "name": "String001",
       "timestamp": 1504739061546,
       "dataType": "String",
        "properties": {
         "Quality": {
           "type": "Int32",
           "value": 192
         }
        },
        "value": "onetwothree"
      },
      {
       "name": "Integer001",
       "timestamp": 1504739061546,
       "dataType": "Int32",
        "properties": {
         "Quality": {
           "type": "Int32",
           "value": 192
         }
        },
        "value": 123
      },
      {
       "name": "Float001",
        "timestamp": 1504739061546,
       "dataType": "Float",
       "properties": {
         "Quality": {
           "type": "Int32",
           "value": 192
         }
       },
        "value": 1.23
     }
   ],
    "seq": 1
 }
}
```

Step 7: Use Ignition Designer to Publish Tag Data (Live Tag Values Changes)

Now we can change the values of the new Memory Tags and generate payloads that contain the Tag change events.

Click on the value of the "Boolean001" Memory Tag to change it's value.



This will result in the following payload to be constructed to represent this Tag change event and pushed to the Azure IoT Hub:

Change Event Payload

```
{
 "topic": {
   "namespace": "spBv1.0",
   "groupId": "Tutorial Group",
   "edgeNodeId": "Tutorial Edge Node",
   "deviceId": "Tutorial Device"
 },
 "payload": {
   "timestamp": 1504740884529,
   "metrics": [
     {
       "name": "Boolean001",
       "timestamp": 1504740883526,
       "dataType": "Boolean",
       "value": false
    }
   ],
   "seq": 2
 }
}
```

Step 8: Azure lot Hub Applications

It is beyond the scope of this tutorial to show how to design an application in Azure to handle the payloads as they are pushed in to the Azure IoT Hub. For additional information on developing applications to consume this data see https://docs.microsoft.com/en-us/azure/.