Industrial Automation

Tech Note 51 Crimson® Cloud Connectors: Azure

Abstract:

This document explains how to get Crimson 3.1 to talk to Microsoft Azure using the MQTT Connector. It assumes a basic knowledge of Crimson and its operation, and the ability to read and manipulate JSON. For more details on the Crimson Cloud Connectors, please consult the Crimson User Manual.

Products:

CR3000 HMIs / Graphite® HMI / Graphite Controllers

Use Case: Azure Connector

Transferring tag data to Azure.

Required Software:

Crimson 3.1

Required Firmware:

Build 3106.000 or higher

Step 1 – Creating an Account

If you do not have an Azure account, visit http://azure.microsoft.com/Azure to create a trial account. The trial account provides sufficient capacity for testing but is unlikely to be suitable for a real deployment. When compared to Crimson's supported update rates, Azure's free tier IoT Hub is very limited in terms of messages supported per day. You should therefore take care not to leave test devices connected too long if they are configured to update many times per minute.

Step 2 – Creating an IoT Hub

Logon to Azure and ensure the left-hand menu strip is expanded. Select the *Create a Resource* option.

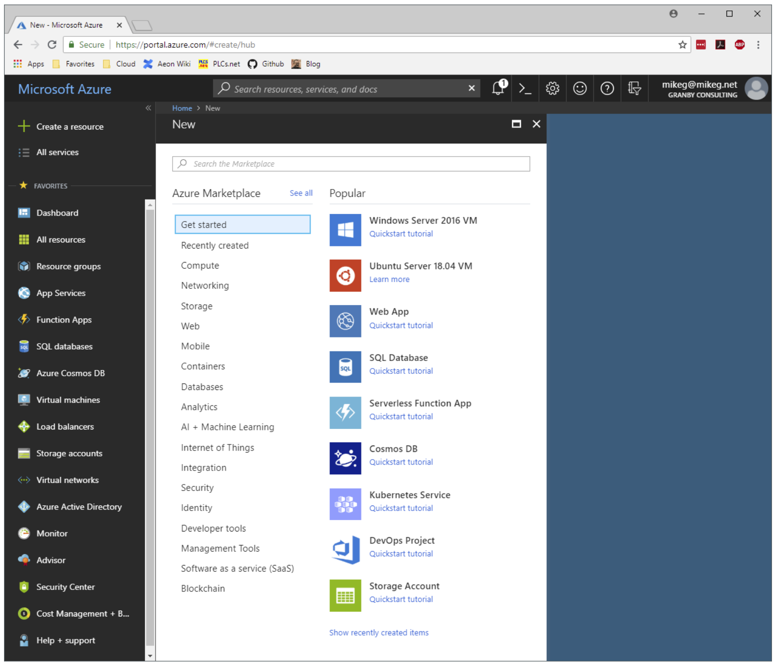
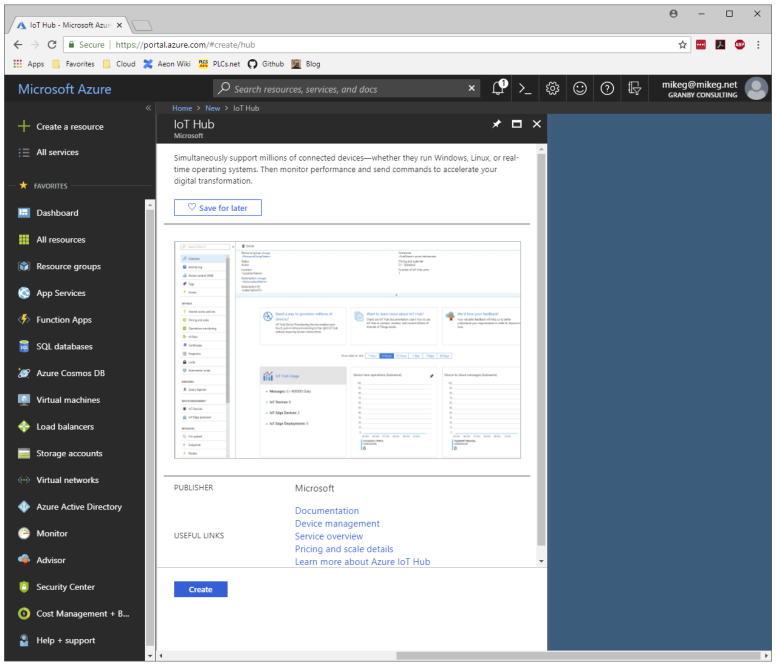


Figure 1.

Referring to Figure 1, in the search box, type a few letters of the phrase ***IoT Hub*** until you can select it from the dropdown.

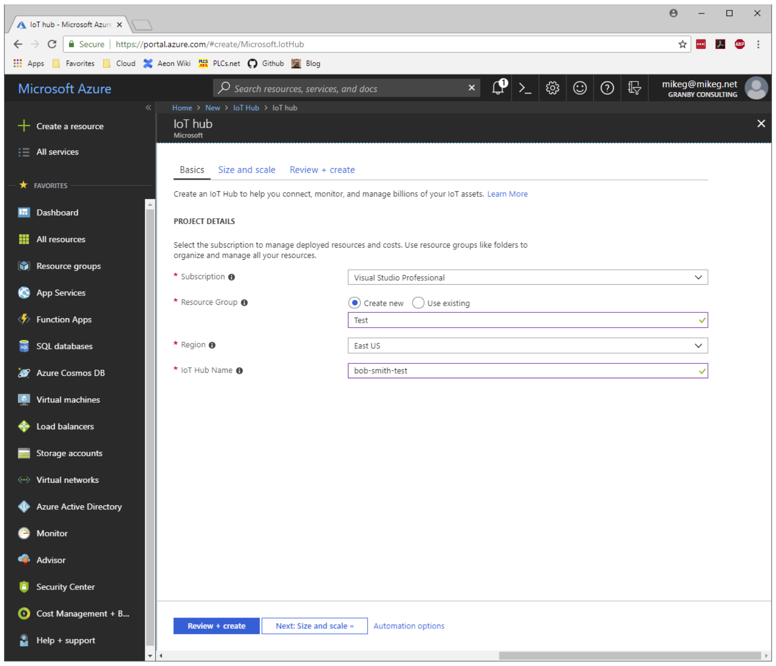


Simultaneously support millions of connected devices—whether they run Windows, Linux, or real-time operating systems. Then monitor performance and send commands to accelerate your digital transformation.

PUBLISHER	Microsoft
USEFUL LINKS	Documentation Device management Service overview Pricing and scale details Learn more about Azure IoT Hub

Figure 2.

Referring to Figure 2, click on the *Create* button to create a new IoT Hub.



IoT hub
Microsoft

Create an IoT Hub to help you connect, monitor, and manage billions of your IoT assets. Learn More

PROJECT DETAILS

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

* Subscription	Visual Studio Professional
* Resource Group	[✓ Create new] Use existing
	Test
* Region	East US
* IoT Hub Name	bob-smith-test

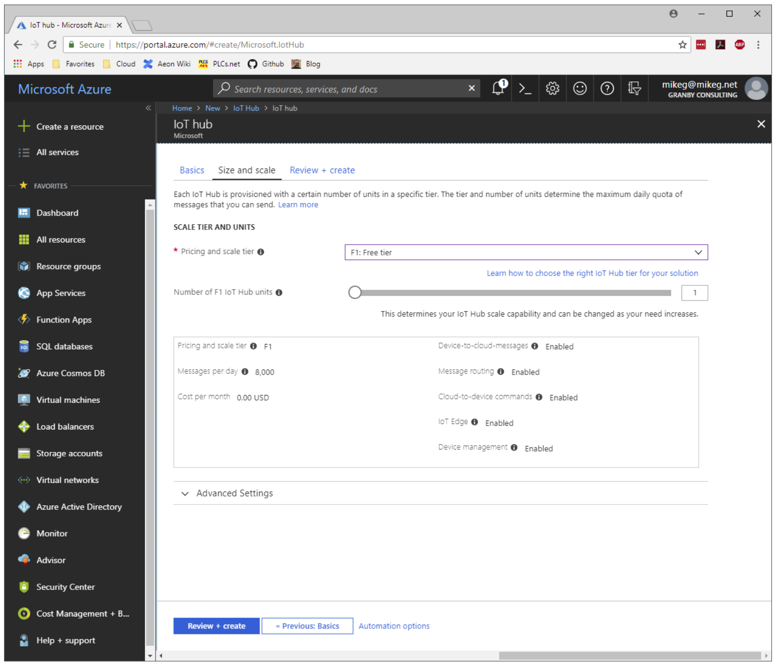
Figure 3.

Referring to Figure 3, perform the following actions.

1. In the Subscription drop-down, select the subscription you are using.
2. In the Resource Group section, select *Create New* and enter a name of Test
3. In the Region drop-down, select an appropriate geographic region.
4. In the IoT Hub Name, enter a suitable name as discussed below.

The IoT Hub name must be globally unique. For this sample, we shall use the name bob-smith-test, but you should obviously use another name that is related to yourself or your organization. If the name has already been used, Azure will not allow you to create the resource.

Next press the button labelled *Next: Size and Scale*.


IoT hub
Microsoft

Basics | Size and scale | Review + create

Each IoT Hub is provisioned with a certain number of units in a specific tier. The tier and number of units determine the maximum daily quota of messages that you can send. Learn more

SCALE TIER AND UNITS

* Pricing and scale tier: F1: Free tier

Learn how to choose the right IoT Hub tier for your solution

Number of F1 IoT Hub units: 1

This determines your IoT Hub scale capability and can be changed as your need increases.

Pricing and scale tier: F1	Device-to-cloud-messages: Enabled
Messages per day: 8,000	Message routing: Enabled
Cost per month: 0.00 USD	Cloud-to-device commands: Enabled
	IoT Edge: Enabled
	Device management: Enabled

Advanced Settings

Figure 4.

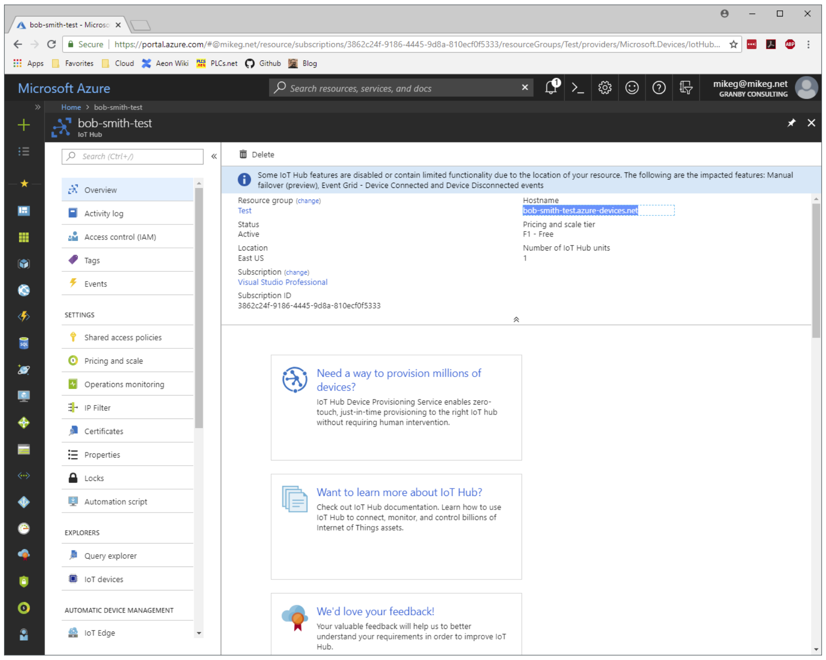
Referring to Figure 4, set the Pricing and Scale Tier to *F1: Free Tier*.

Leave the other settings unchanged and press *Review + Create*, followed by Create.

Azure will churn for a while, and after a delay of what might be several minutes the bell-shaped notification icon in the Azure web console will show a notification that your IoT Hub has been created. Once the creation process has completed, you can refresh your console window to show the resource.

Step 3 – Finding the Host Name

Select the IoT Hub from your Azure dashboard.



Some IoT Hub features are disabled or contain limited functionality due to the location of your resource. The following are the impacted features: Manual failover (preview), Event Grid - Device Connected and Device Disconnected events

Resource group (change)
Test

Status
Active

Location
East US

Subscription (change)
Visual Studio Professional

Subscription ID
3862c24f-9186-4445-9d8a-810ecf0f5333

Hostname
bob-smith-test.azure-devices.net

Pricing and scale tier
F1 - Free

Number of IoT Hub units
1

Need a way to provision millions of devices?
IoT Hub Device Provisioning Service enables zero-touch, just-in-time provisioning to the right IoT hub without requiring human intervention.

Want to learn more about IoT Hub?
Check out IoT Hub documentation. Learn how to use IoT Hub to connect, monitor, and control billions of Internet of Things assets.

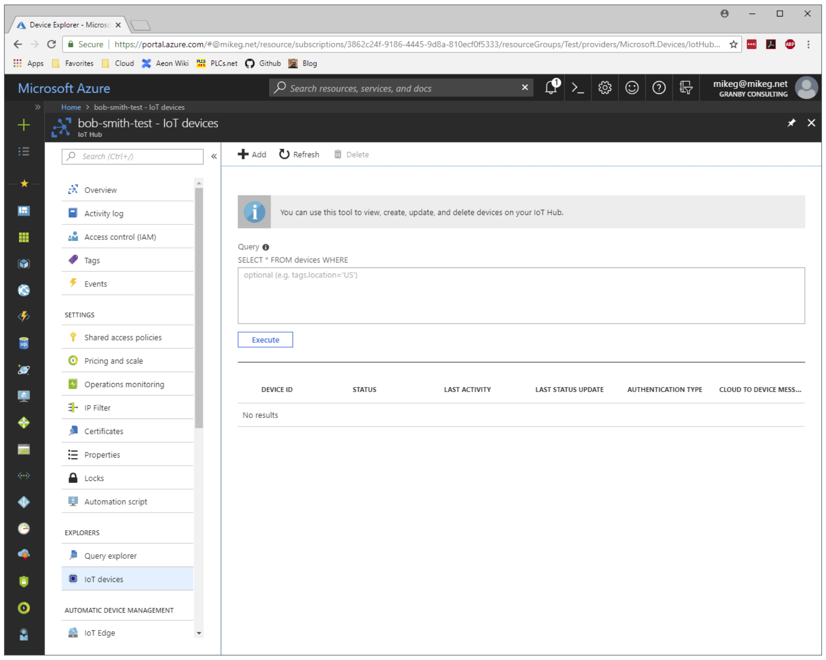
We'd love your feedback!
Your valuable feedback will help us to better understand your requirements in order to improve IoT Hub.

Figure 5.

Take note of the hostname highlighted in Figure 5. We shall be using it later when we configure the connector in Crimson. It is typically the IoT Hub Name followed by azure-devices.net.

Step 4 – Creating a Device

We can now create a device within the IoT Hub to correspond to the Crimson device from which we shall be pushing data. To do this, select the *IoT Hub* in your dashboard and scroll-down in the left-hand menu strip until you find the *IoT Devices* option in the Explorer section. Select this option to show the following.



You can use this tool to view, create, update, and delete devices on your IoT Hub.

Query

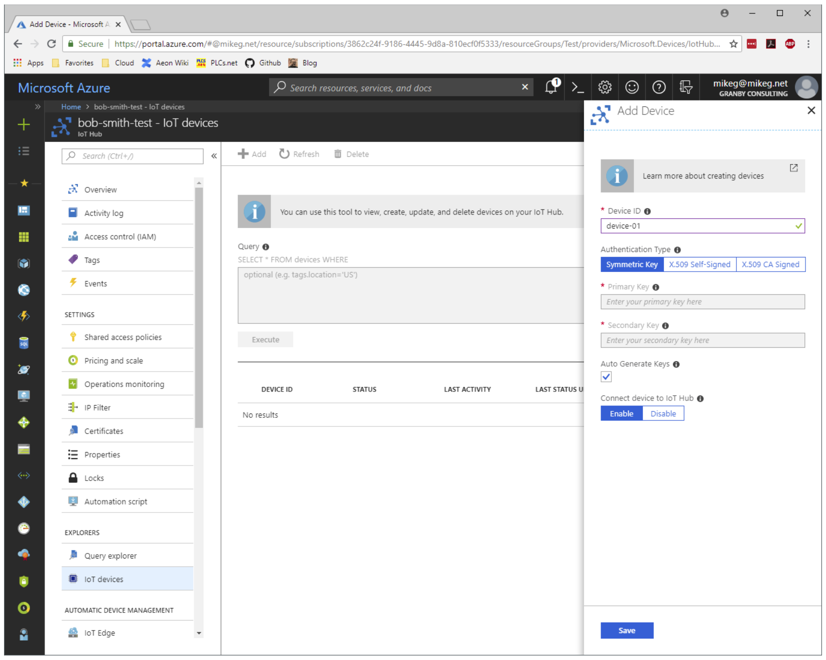
SELECT * FROM devices WHERE

optional (e.g. tags.location='US')

DEVICE ID	STATUS	LAST ACTIVITY	LAST STATUS UPDATE	AUTHENTICATION TYPE	CLOUD TO DEVICE MESS...
No results					

Figure 6.

Referring to Figure 6, press the Add button to create a new device.



You can use this tool to view, create, update, and delete devices on your IoT Hub.

Query: SELECT * FROM devices WHERE — optional (e.g. tags.location='US')

Device ID: device-01

Authentication Type: [✓ Symmetric Key] X.509 Self-Signed X.509 CA Signed

Primary Key: Enter your primary key here

Secondary Key: Enter your secondary key here

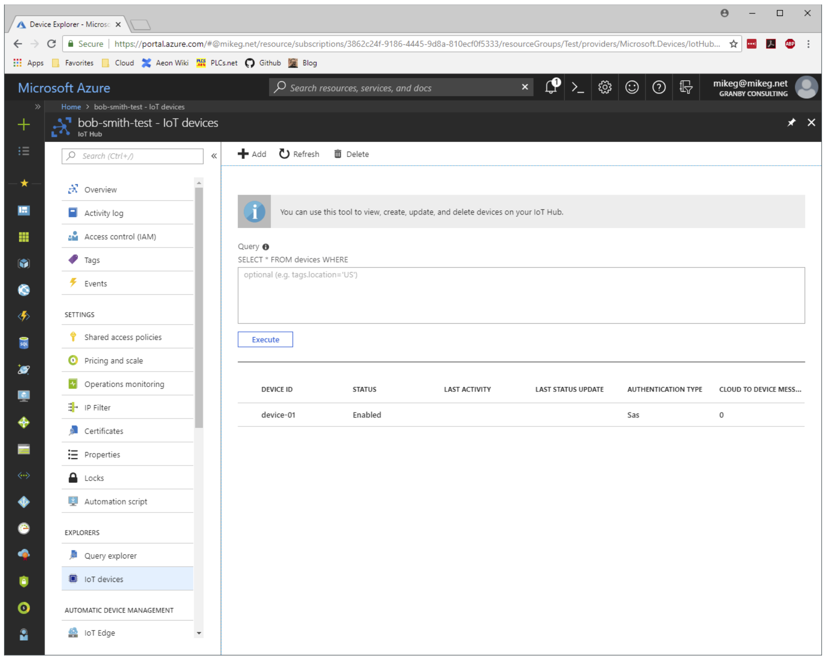
Auto Generate Keys: ✓

Connect device to IoT Hub: [✓ Enable] Disable

Figure 7.

Referring to Figure 7, in the strip that appears on the right-hand side of the window, enter ***device-01*** as the Device ID. Leave the rest of the settings at their defaults (i.e. Symmetric Key, Auto Generate Key and Enable Hub Connection) and press the *Save* button to create the device.

Your device view will update to show the device.



You can use this tool to view, create, update, and delete devices on your IoT Hub.

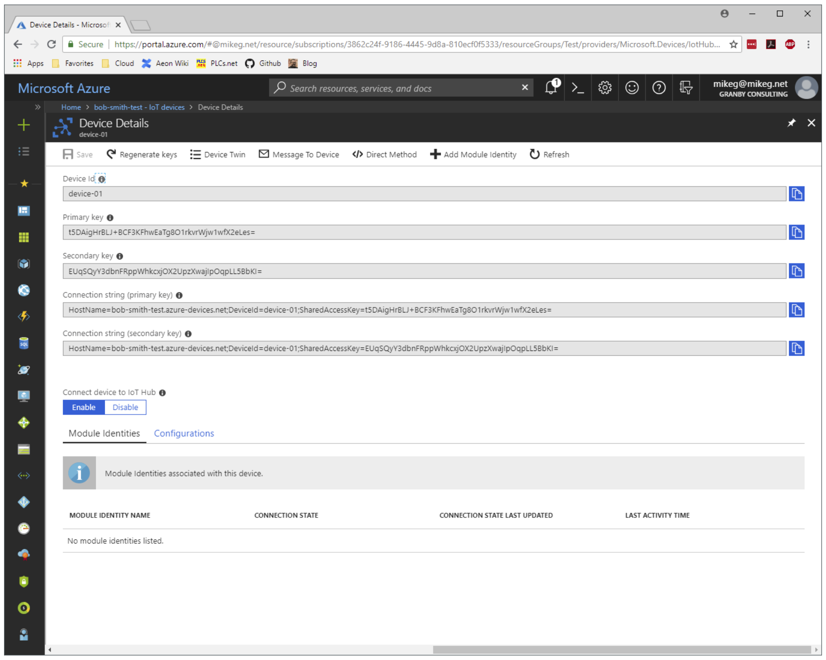
Query

SELECT * FROM devices WHERE

DEVICE ID	STATUS	LAST ACTIVITY	LAST STATUS UPDATE	AUTHENTICATION TYPE	CLOUD TO DEVICE MESS...
device-01	Enabled			Sas	0

Figure 8.

Referring to Figure 8, click on device-01 to view its properties.



Device Details

device-01

Device Id: device-01

Primary key: t5DAigHrBLJ+BCF3KFhwEaTg8O1rkvrWjw1wfX2eLes=

Secondary key: EUqSQyY3dbnFRppWhkcxjOX2UpzXwajIpOqpLL5BbKI=

Connection string (primary key): HostName=bob-smith-test.azure-devices.net;DeviceId=device-01;SharedAccessKey=t5DAigHrBLJ+BCF3KFhwEaTg8O1rkvrWjw1wfX2eLes=

Connection string (secondary key): HostName=bob-smith-test.azure-devices.net;DeviceId=device-01;SharedAccessKey=EUqSQyY3dbnFRppWhkcxjOX2UpzXwajIpOqpLL5BbKI=

Connect device to IoT Hub: Enable

Module Identities associated with this device.

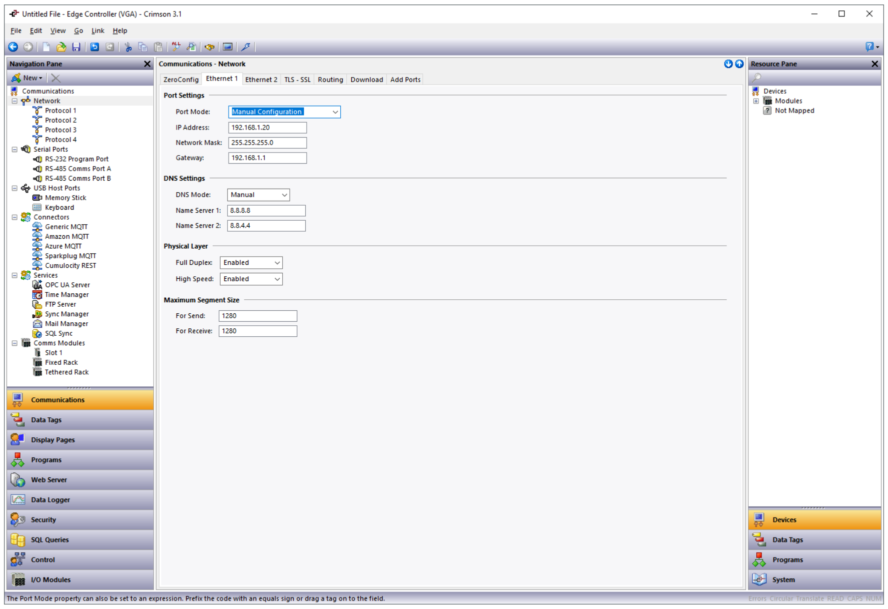
MODULE IDENTITY NAME	CONNECTION STATE	CONNECTION STATE LAST UPDATED	LAST ACTIVITY TIME
No module identities listed.			

Figure 9.

Referring to Figure 9, take note of the *Primary Key*. We shall be using this later when configuring Crimson.

Step 5 – Creating an Outline Database

You should now open Crimson 3.1 and create a new database for your device. Before we configure the Azure Connector, we shall need to perform a few steps to create the infrastructure used for testing. Red Lion will soon be making these so-called outline databases available via its website and including them in the Crimson installation. In the meantime, to manually create an outline database, start by opening the Communications category and selecting the *Ethernet 1* tab in the Network settings.


Communications - Network

Port Settings

Port Mode: Manual Configuration
IP Address: 192.168.1.20
Network Mask: 255.255.255.0
Gateway: 192.168.1.1

DNS Settings

DNS Mode: Manual
Name Server 1: 8.8.8.8
Name Server 2: 8.8.4.4

Physical Layer

Full Duplex: Enabled
High Speed: Enabled

Maximum Segment Size

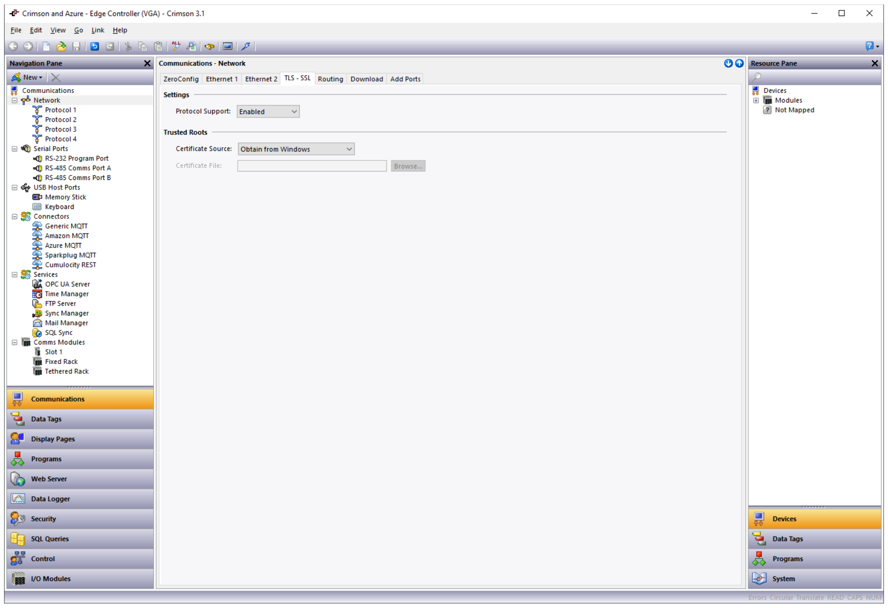
For Send: 1280
For Receive: 1280

The Port Mode property can also be set to an expression. Prefix the code with an equals sign or drag a tag on to the field.

Figure 10.

Referring to Figure 10, if the device is connected to a network that automatically allocates IP addresses via a DHCP server, you may leave these settings unchanged. **If you are manually specifying an IP address, you must also define a default gateway and manually provide DNS servers.** The gateway is necessary for your device to push data to the Internet and the DNS servers are used to convert the cloud server name into an IP address. The default server addresses of 8.8.8.8 and 8.8.4.4 will be suitable for most applications, but the DNS Mode must be explicitly set to Manual if the Port Mode is set to Manual Configuration.

Next switch to the TLS – SSL tab in the Ethernet settings.



Communications - Network

ZeroConfig | Ethernet 1 | Ethernet 2 | TLS - SSL | Routing | Download | Add Ports

Settings

Protocol Support: Enabled

Trusted Roots

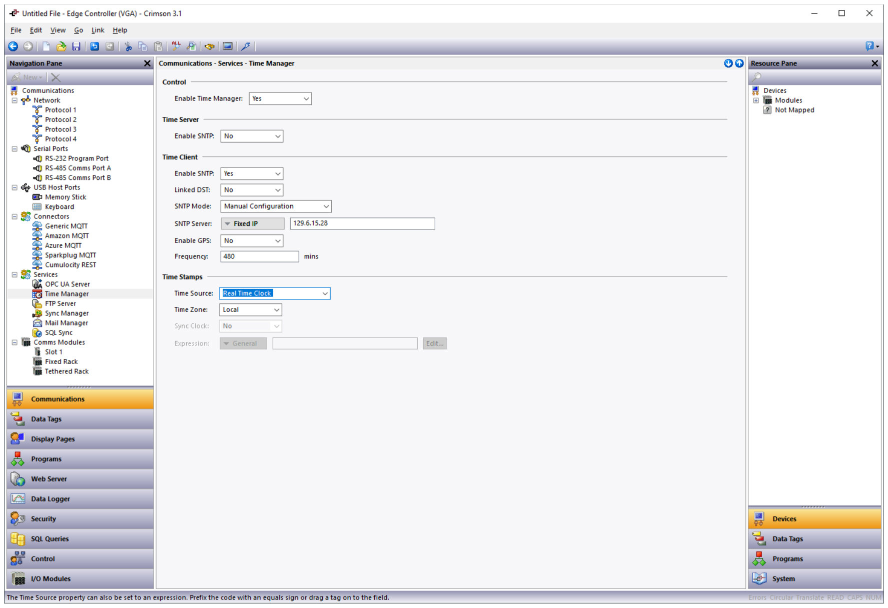
Certificate Source: Obtain from Windows

Certificate File:

Figure 11.

Referring to Figure 11, change the Certificate Source property to Obtain from Windows.

Next select the Time Manager settings within the Services group.



Communications - Services - Time Manager

Control

Enable Time Manager:	Yes

Time Server

Enable SNTP:	No

Time Client

Enable SNTP:	Yes
Linked DST:	No
SNTP Mode:	Manual Configuration
SNTP Server:	Fixed IP 129.6.15.28
Enable GPS:	No
Frequency:	480 mins

Time Stamps

Time Source:	Real Time Clock
Time Zone:	Local
Sync Clock:	No
Expression:	General

The Time Source property can also be set to an expression. Prefix the code with an equals sign or drag a tag on to the field.

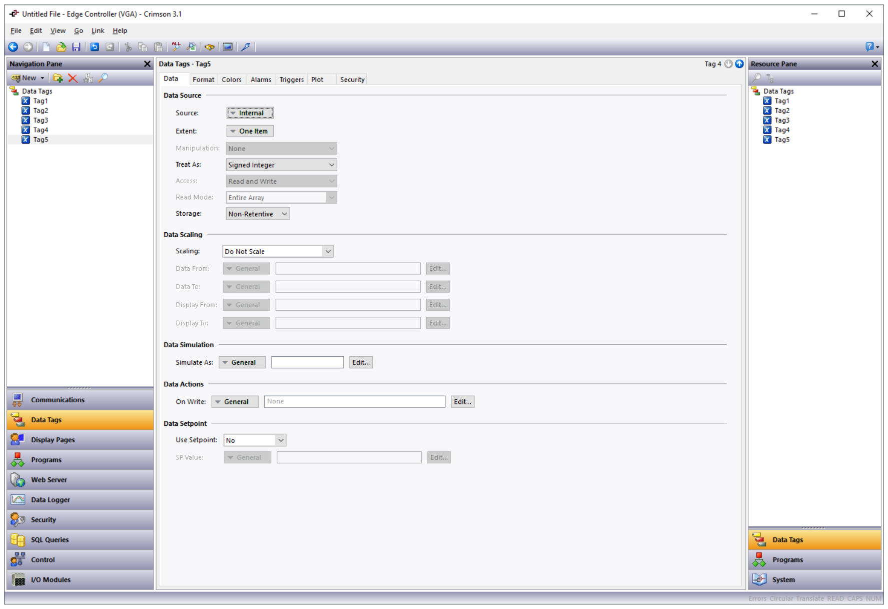
Figure 12.

Referring to Figure 12, perform the following actions:

1. In the Control section, set the Enable Time Manager property to *Yes*.
2. In the Time Client section, set the Enable SNTP setting to *Yes*.

The other settings may be left at their default values. These settings enable time synchronization, allowing your device to set its real-time clock from the Internet. Having an accurate real-time clock is important not only for time-stamping data, but also for ensuring the correct operation of certain security algorithms. **Once you have downloaded your outline database for the first time, select the Send Time option from the Link menu within Crimson to set the device's time zone to that of your PC.** If you need to set it to a different time zone, refer to the Crimson manuals for information on the time zone system variables.

Next navigate to the Data Tags category and press the *New* button five times.



Data Tags - Tag5 · Tag 4

Data | Format | Colors | Alarms | Triggers | Plot | Security

Data Source
Source: Internal
Extent: One Item
Manipulation: None
Treat As: Signed Integer
Access: Read and Write
Read Mode: Entire Array
Storage: Non-Retentive

Data Scaling
Scaling: Do Not Scale
Data From: General
Data To: General
Display From: General
Display To: General

Data Simulation
Simulate As: General

Data Actions
On Write: General None

Data Setpoint
Use Setpoint: No
SP Value: General

Figure 13.

Referring to Figure 13, select Tag5 in the Navigation pane and then rename it to Status.

Next navigate to the Display Pages category…

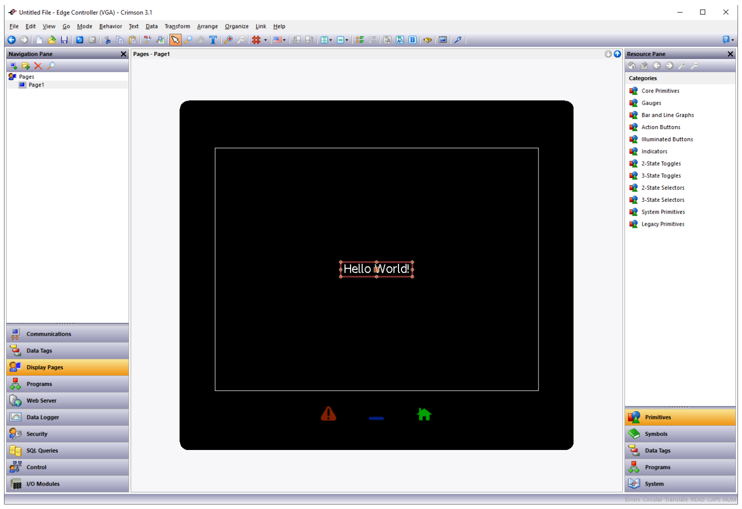


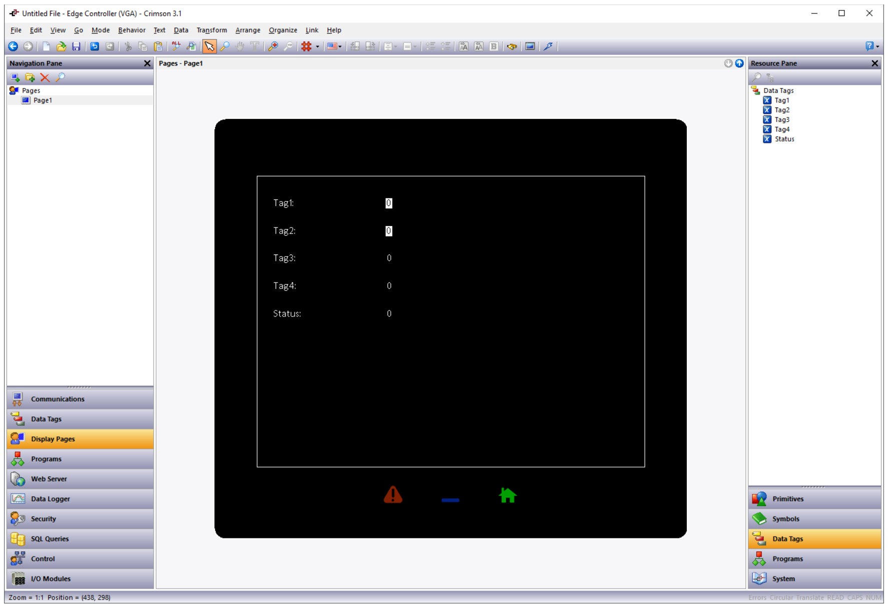
Figure 14.

Referring to Figure 14, perform the following actions:

1. Select the Hello World text and press *Delete* to remove it from the screen.
2. In the Resource Pane, select the *Data Tags* category.
3. In the Resource Pane, click on *Tag1* and, while holding down *Shift* + *Ctrl*, click on Status.
4. All five data tags should be selected.
5. Drag the resulting selection on to your display page.
6. Grab the bottom-right resizing handle and expand the data fields to a suitable size.
7. Click away from the data fields to deselect them.
8. Click on Tag1 and check the **Data Entry** box in the floating toolbar.
- Alternatively, *right-click* and select *Data* followed by *Data Entry*.

9. Repeat this operation for Tag2.

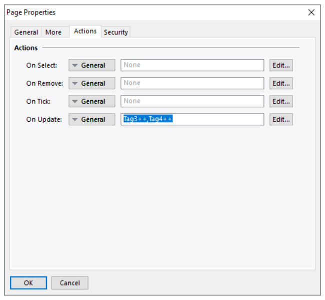
Your display page should now look like Figure 15.



Tag1: 0
Tag2: 0
Tag3: 0
Tag4: 0
Status: 0

Figure 15.

Next *right-click* on the page and select *Properties*.



Actions

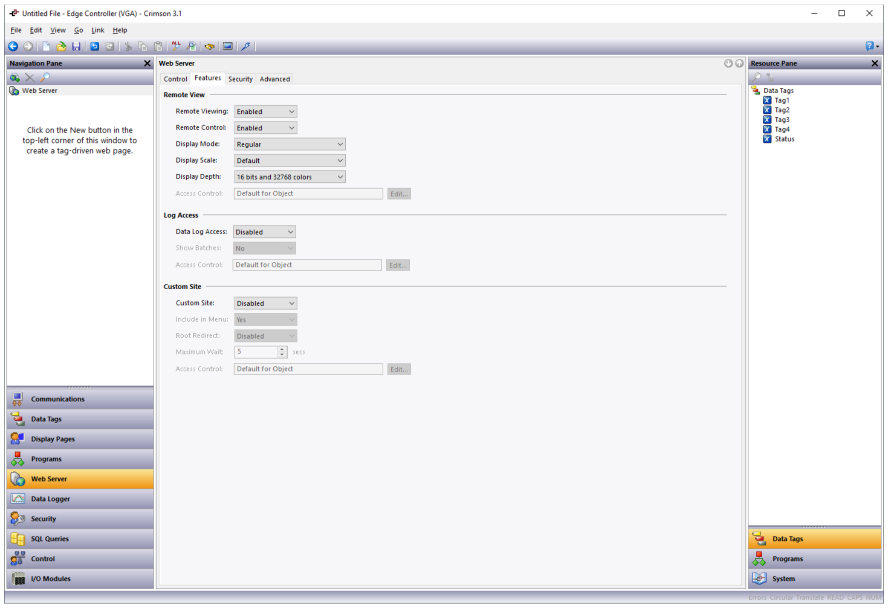
On Select:	General	None	Edit...
On Remove:	General	None	Edit...
On Tick:	General	None	Edit...
On Update:	General	Tag3++,Tag4++	Edit...

Figure 16.

Referring to Figure 16, perform the following actions…

1. Select the *Actions* tab.
2. Enter ***Tag3++,Tag4++*** in the On Update box.
3. Press *OK* to save the changes.

Next navigate to the Web Server category.



Click on the New button in the top-left corner of this window to create a tag-driven web page.

Remote View

Remote Viewing: Enabled
Remote Control: Enabled
Display Mode: Regular
Display Scale: Default
Display Depth: 16 bits and 32768 colors
Access Control: Default for Object

Log Access

Data Log Access: Disabled
Show Batches: No
Access Control: Default for Object

Custom Site

Custom Site: Disabled
Include in Menu: Yes
Root Redirect: Disabled
Maximum Wait: 5 secs
Access Control: Default for Object

Figure 17.

Referring to Figure 17, perform the following actions:

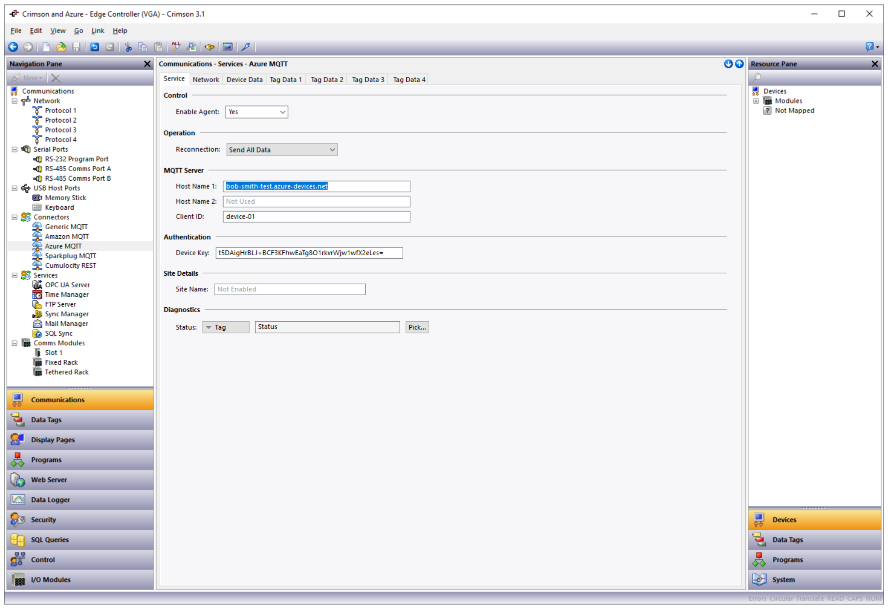
1. Select the *Features* tab.
2. In the Remove View section, set the Remote Control property to *enabled*.

You have now created an outline database. Press *F9* to download this to your device, and if this is the first time you are doing so, select the Send Time option from the Link menu to set the device's time and time zone. If the device has a display, you should be able to see the five tags. You will be able to edit the first two tags using the touchscreen, and the next two will be incrementing steadily as the display updates.

If your device does not have a display, open your web browser. In the address bar, enter the http:// followed by the IP address that you have allocated to your device to access the Crimson web server. Within the Crimson web server, select *Remote View* to view the device's virtual display and again note the behavior of the tags.

Step 6 – Configuring the Azure Connector

We are now in a position to configure and test the Azure Connector. We shall be configuring it to talk to the device that we created in earlier sections and pushing four of the data tags that we have created. The fifth tag will be used to display the connection status. Start by navigating to the Communications category and select the Azure MQTT settings in the Connectors section.



Communications - Services - Azure MQTT

Service | Network | Device Data | Tag Data 1 | Tag Data 2 | Tag Data 3 | Tag Data 4

Control
Enable Agent: Yes

Operation
Reconnection: Send All Data

MQTT Server
Host Name 1: bob-smith-test.azure-devices.net
Host Name 2: Not Used
Client ID: device-01

Authentication
Device Key: t5DAigHrBLJ+BCF3KFhwEaTg8O1rkvrWjw1wfX2eLes=

Site Details
Site Name: Not Enabled

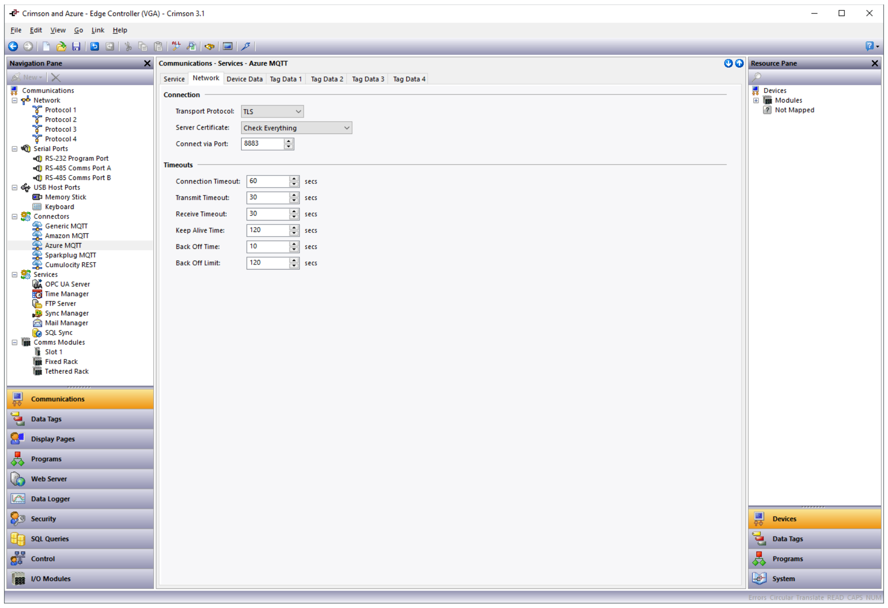
Diagnostics
Status: Tag Status Pick...

Figure 18.

Referring to Figure 18, perform the following actions:

1. In the Control section, set the Enable Agent property to *Yes*.
2. In the MQTT Server section, set the Host Name 1 property to the host name from **Step 3**.
3. In the MQTT Server section, set the Client ID property to ***thing-01***
4. In the Authentication section, set the Device Key property to the primary key from **Step 4**.
5. In the Diagnostics section, set the Status property to ***Status***

Next select the Network tab.


Communications - Services - Azure MQTT

Service | Network | Device Data | Tag Data 1 | Tag Data 2 | Tag Data 3 | Tag Data 4

Connection

Transport Protocol: TLS
Server Certificate: Check Everything
Connect via Port: 8883

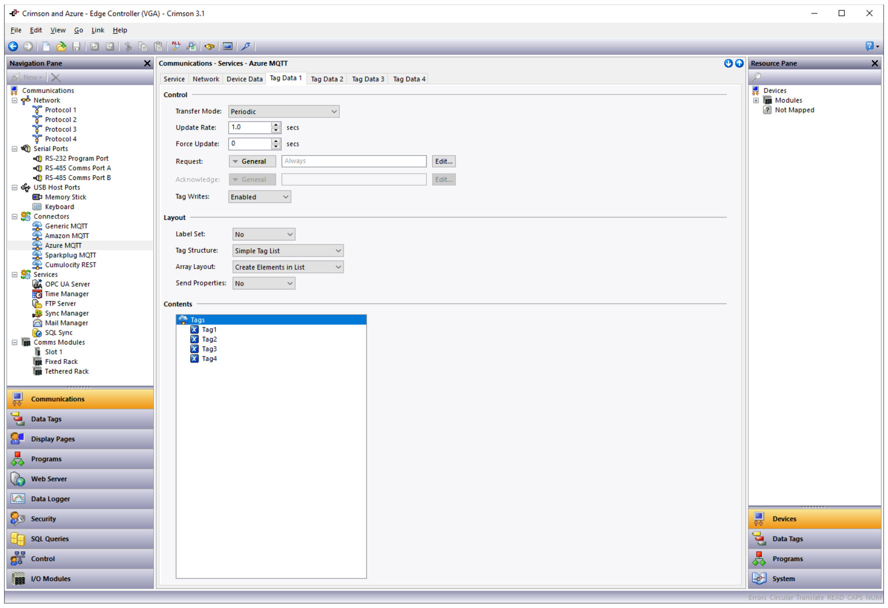
Timeouts

Connection Timeout: 60 secs
Transmit Timeout: 30 secs
Receive Timeout: 30 secs
Keep Alive Time: 120 secs
Back Off Time: 10 secs
Back Off Limit: 120 secs

Figure 19.

Referring to Figure 19, set the Server Certificate property to Check Everything. We are able to do this because our database contains an appropriate set of root certificates obtained from Windows. If we do not have root certificates available, we should leave this setting at Ignore.

Next select the Tag Data 1 tab.



Communications - Services - Azure MQTT

Service | Network | Device Data | Tag Data 1 | Tag Data 2 | Tag Data 3 | Tag Data 4

Control

Transfer Mode: Periodic
Update Rate: 1.0 secs
Force Update: 0 secs
Request: General Always Edit...
Acknowledge: General Edit...
Tag Writes: Enabled

Layout

Label Set: No
Tag Structure: Simple Tag List
Array Layout: Create Elements in List
Send Properties: No

Contents

Tags
Tag1
Tag2
Tag3
Tag4

Figure 20.

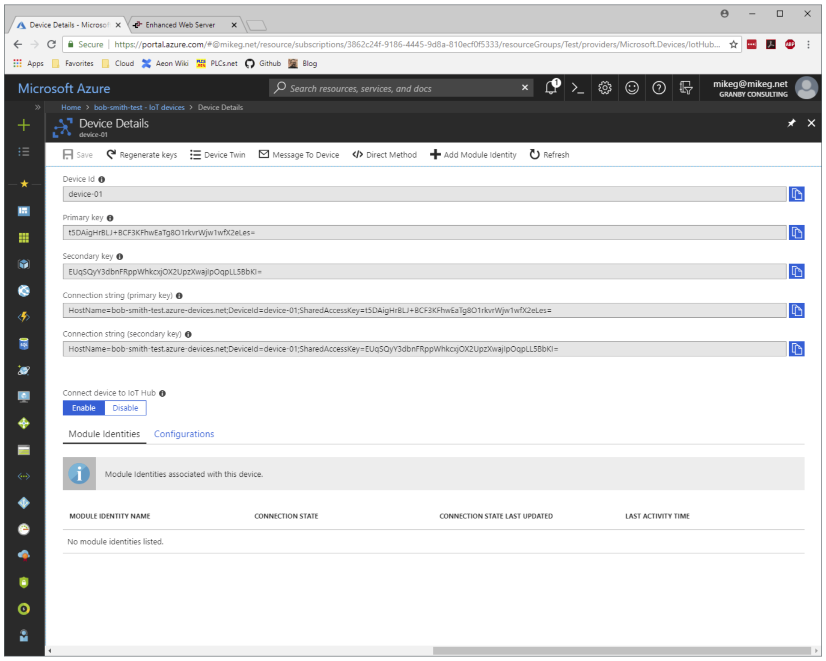
Referring to Figure 20, perform the following actions:

1. In the Control section, set the Tag Writes property to Enabled.
2. Select the Data Tags category of the Resource Pane.
3. Drag Tag1 through Tag4 into the Contents field in the editing pane.

You have now configured Crimson to push Tag1 through Tag4 to the cloud once per second. Press *F9* to download the database to your device and check the Status tag on your display or via the web browser. A value of 4 should be displayed, indicating that the cloud connection has been established and that data is being pushed. A value of 0 typically indicates an issue with network connectivity or with DNS, while a value of 1 indicates that the server name was resolved but that the connection could not be established. A value of 3 indicates that the connection has been made, but that data has not been transferred. If you do not see a value of 4, check each item in this note carefully and ensure your Crimson configuration matches your Azure settings.

Step 7 – Interacting with the Device

Now that we have configured Crimson to pass data to the cloud, we can view this data from the Azure web console and optionally write data back to the device. Return to the IoT Hub within the console, and once again select *IoT Devices* from the Explorers. Select *device-01* from the presented list of devices.



Device Details
device-01

Device Id: device-01

Primary key: t5DAigHrBLJ+BCF3KFhwEaTg8O1rkvrWjw1wfX2eLes=

Secondary key: EUqSQyY3dbnFRppWhkcxjOX2UpzXwajIpOqpLL5BbKI=

Connection string (primary key): HostName=bob-smith-test.azure-devices.net;DeviceId=device-01;SharedAccessKey=t5DAigHrBLJ+BCF3KFhwEaTg8O1rkvrWjw1wfX2eLes=

Connection string (secondary key): HostName=bob-smith-test.azure-devices.net;DeviceId=device-01;SharedAccessKey=EUqSQyY3dbnFRppWhkcxjOX2UpzXwajIpOqpLL5BbKI=

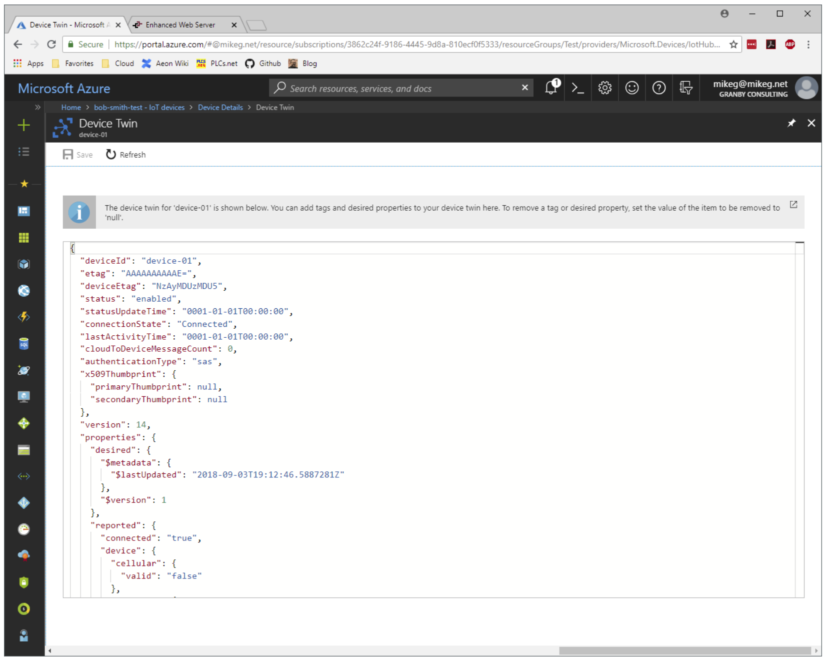
Connect device to IoT Hub: [✓ Enable] Disable

Module Identities associated with this device.

MODULE IDENTITY NAME	CONNECTION STATE	CONNECTION STATE LAST UPDATED	LAST ACTIVITY TIME
No module identities listed.			

Figure 21.

Referring to Figure 21, select the Device Twin option. Please note that Azure will not allow a device twin to grow over 8K (the sum of all the tag names and longest possible tag values, plus all the JSON formatting) in size.

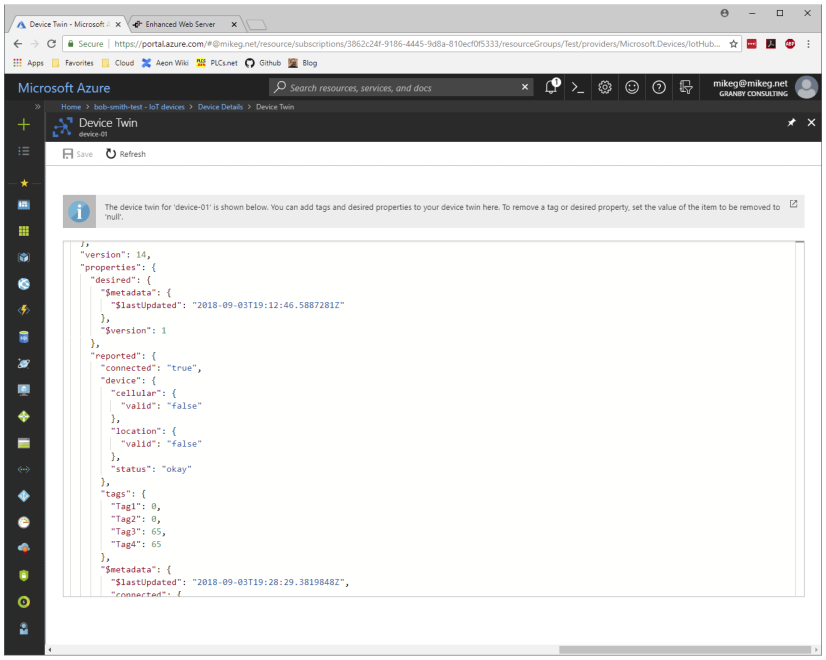

Device Twin - Microsoft A
Enhanced Web Server
Secure
https://portal.azure.com/#@mikeg.net/resource/subscriptions/3862c24f-9186-4445-9d8a-810ecf0f5333/resourceGroups/Test/providers/Microsoft.Devices/IotHub...
Apps
Favorites
Cloud
Aeon Wiki
PLCs.net
Github
Blog
Microsoft Azure
Search resources, services, and docs
mikeg@mikeg.net
GRANBY CONSULTING
Home > bob-smith-test - IoT devices > Device Details > Device Twin
Device Twin
device-01
Save
Refresh

The device twin for 'device-01' is shown below. You can add tags and desired properties to your device twin here. To remove a tag or desired property, set the value of the item to be removed to 'null'.

```
{
  "deviceId": "device-01",
  "etag": "AAAAAAAAAAE=",
  "deviceEtag": "NzAyMDUzMDU5",
  "status": "enabled",
  "statusUpdateTime": "0001-01-01T00:00:00",
  "connectionState": "Connected",
  "lastActivityTime": "0001-01-01T00:00:00",
  "cloudToDeviceMessageCount": 0,
  "authenticationType": "sas",
  "x509Thumbprint": {
    "primaryThumbprint": null,
    "secondaryThumbprint": null
  },
  "version": 14,
  "properties": {
    "desired": {
      "$metadata": {
        "$lastUpdated": "2018-09-03T19:12:46.5887281Z"
      },
      "$version": 1
    },
    "reported": {
      "connected": "true",
      "device": {
        "cellular": {
          "valid": "false"
        },
```

Figure 22.

The device twin is a persistent object that stores the values sent by the device, whether or not those values were included in the latest message. The twin contains data items created by Azure plus the data items reported by your device. If you scroll down a little, you will see the **reported** object within the **properties** object.


Device Twin - Microsoft A Enhanced Web Server
Secure | https://portal.azure.com/#@mikeg.net/resource/subscriptions/3862c24f-9186-4445-9d8a-810ecf0f5333/resourceGroups/Test/providers/Microsoft.Devices/IotHub...

The device twin for 'device-01' is shown below. You can add tags and desired properties to your device twin here. To remove a tag or desired property, set the value of the item to be removed to 'null'.

```
},
"version": 14,
"properties": {
  "desired": {
    "$metadata": {
      "$lastUpdated": "2018-09-03T19:12:46.5887281Z"
    },
    "$version": 1
  },
  "reported": {
    "connected": "true",
    "device": {
      "cellular": {
        "valid": "false"
      },
      "location": {
        "valid": "false"
      },
      "status": "okay"
    },
    "tags": {
      "Tag1": 0,
      "Tag2": 0,
      "Tag3": 65,
      "Tag4": 65
    },
    "$metadata": {
      "$lastUpdated": "2018-09-03T19:28:29.3819848Z",
      "connected": {
```

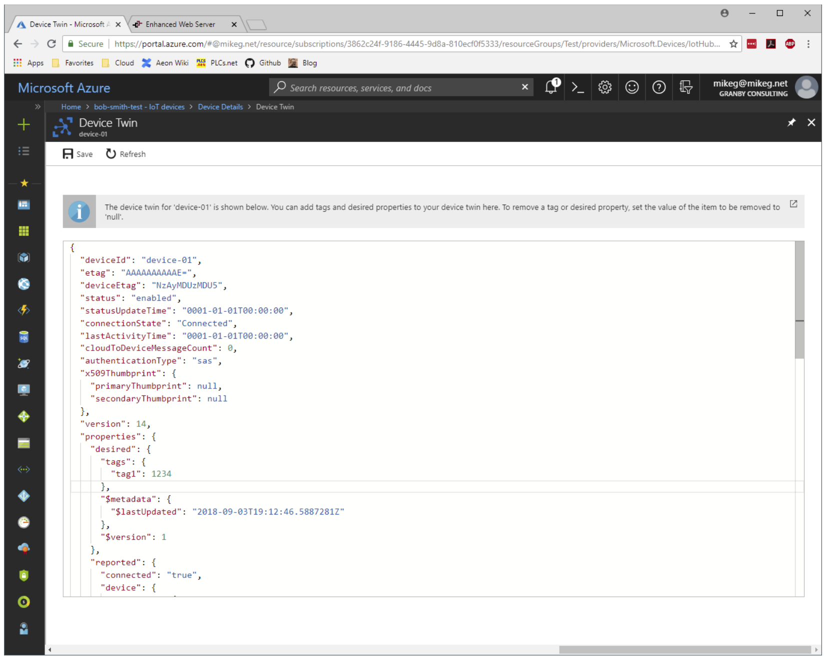
Figure 23.

The text below shows the part of the twin more clearly with certain items omitted.

```
"properties": {
  "desired": {
    [contents ommitted for brevity]
  },
  "reported": {
    "connected": "true",
    "device": {
      "cellular": {
        "valid": "false"
      },
      "location": {
        "valid": "false"
      },
      "status": "okay"
    },
    "tags": {
      "Tag1": 0,
      "Tag2": 0,
      "Tag3": 65,
      "Tag4": 65
    }
  },
  [balance ommitted for brevity]
```

As you can see, the twin is a JSON fragment that contains an object called **properties**, which in turn contains an object called **reported**. Within this latter object, a further object called **tags** contains the data tags being pushed by Crimson, and an object called **device** acts as a placeholder for yet-to-be-supported device status information. Refer to the Crimson User Manual for details on how the format of this JSON fragment can be adjusted to suit your application.

To write data to the device, we must edit the JSON object called **desired** within the **properties** object to contain its own **tags** object that in turn contains the values that we want to write. To modify the twin, click on the twin text within the Azure console and make the following changes.


Device Twin - Microsoft A
Enhanced Web Server
Secure
https://portal.azure.com/#@mikeg.net/resource/subscriptions/3862c24f-9186-4445-9d8a-810ecf0f5333/resourceGroups/Test/providers/Microsoft.Devices/IotHub...
Apps
Favorites
Cloud
Aeon Wiki
PLCs.net
Github
Blog
Microsoft Azure
Search resources, services, and docs
mikeg@mikeg.net
GRANBY CONSULTING
Home > bob-smith-test - IoT devices > Device Details > Device Twin
Device Twin
device-01
Save
Refresh

The device twin for 'device-01' is shown below. You can add tags and desired properties to your device twin here. To remove a tag or desired property, set the value of the item to be removed to 'null'.

```
{
  "deviceId": "device-01",
  "etag": "AAAAAAAAAAE=",
  "deviceEtag": "NzAyMDUzMDU5",
  "status": "enabled",
  "statusUpdateTime": "0001-01-01T00:00:00",
  "connectionState": "Connected",
  "lastActivityTime": "0001-01-01T00:00:00",
  "cloudToDeviceMessageCount": 0,
  "authenticationType": "sas",
  "x509Thumbprint": {
    "primaryThumbprint": null,
    "secondaryThumbprint": null
  },
  "version": 14,
  "properties": {
    "desired": {
      "tags": {
        "tag1": 1234
      },
      "$metadata": {
        "$lastUpdated": "2018-09-03T19:12:46.5887281Z"
      },
      "$version": 1
    },
    "reported": {
      "connected": "true",
      "device": {
```

Figure 24.

It is very important that you get the formatting right when editing the twin. JSON is particularly finicky about commas and other punctuation. The portion that you are creating is shown in red below…

```
"properties": {
  "desired": {
    "tags": {
      "tag1": 1234
    },
    "$metadata": {
      "$lastUpdated": "2018-09-03T19:12:46.5887281Z"
    },
    "$version": 1
  },
  "reported": {
    "connected": "true",
    [balance ommitted for brevity]
```

Note the comma at the end of line immediately after the **tags** object and note the absence of commas at the end of the other lines. JSON requires commas only when another element follows within the current object. If you get this wrong, a red squiggle should appear warning you of your error.

Once you have edited the JSON, press click *Save* to commit the change. The value of Tag1 in your device should be updated both on the device display and in **reported** object in the shadow, although you might have to click Refresh to see this latter change. If this does not happen, check your formatting and check that you have writes enabled in the corresponding tag set.

For more information: http://www.redlion.net/support/policies-statements/warranty-statement