Technical Manual

ACI400

IIoT Edge Device

METTLER TOLEDO

Table of Contents

Pictures

Tables

1 Introduction

Overview

Figure 1: ACI400 IIoT Edge

The METTLER TOLEDO ACI400 IIoT Edge device functions as a secure gateway that allows a user to connect an array of METTLER TOLEDO weighing devices to ERP/MES systems and cloud services. The ACI400 provides an OPC UA server and MQTT Clients for connection to Azure, AWS (Amazon Web Services), and IBM cloud services. A generic MQTT broker is also available for connecting to MQTT brokers not listed here.

1.1 Getting Started

Getting Started

The ACI400 hardware/connection layout is shown in the following three figures.

The D/C input is for connecting the provided external power supply to the ACI400. The ACI400 is programmed start automatically when power is connected. The power button will turn the device OFF and ON manually, however, this manual action is not recommended.

Other connections include:

- Three USB ports which can be used with USB to serial converters to connect weighing devices with RS-232 serial ports
- One RS-232 serial (COM) port for connecting a weighing device with an RS-232 port
- Two 1-Gigabit Ethernet ports (LAN 1 and LAN 2)
- The two mini-display ports (mDP) are not used under normal conditions. In challenging service situations, the mini-display ports, along with the USB ports, can be used to connect an external PC display, USB keyboard and USB mouse
- The audio connection and MicroSD card slot are not used

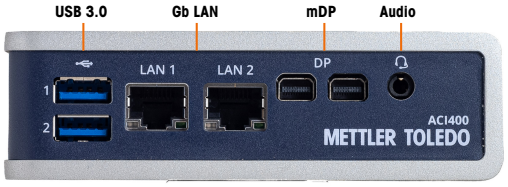

Figure 2: ACI400, Front

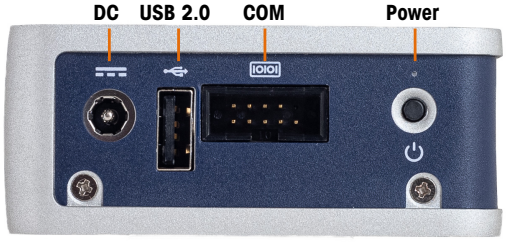

Figure 3: ACI400, Side

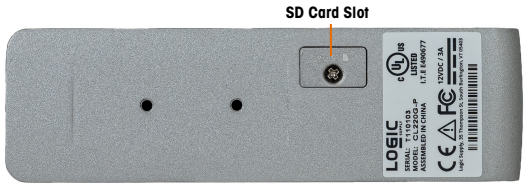

Figure 4: ACI400, Rear

1.2 Device Connections

The ACI400 uses serial, USB and Ethernet connections to communicate with weighing devices. Each ACI400 IIoT Edge can connect up to four unique weighing channels in any combination of Ethernet or serial/USB connections. This can mean four scale channels from a single terminal, four single-scale terminals, four smart weigh modules, or any combination of such devices.

System Layout Examples

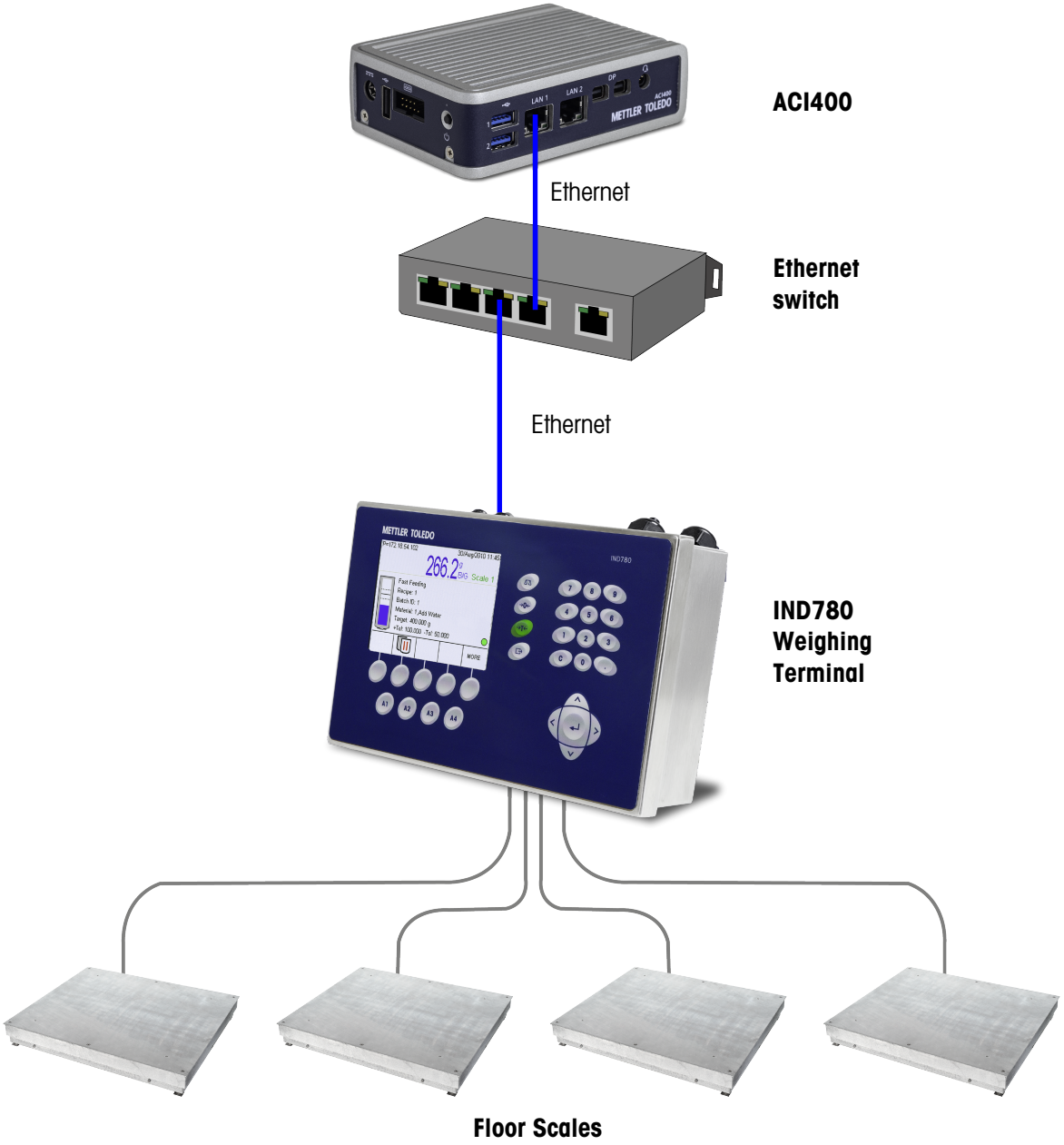

Figure 5: ACI400 Connected via Ethernet to One IND780 Terminal Supporting Four Floor Scales

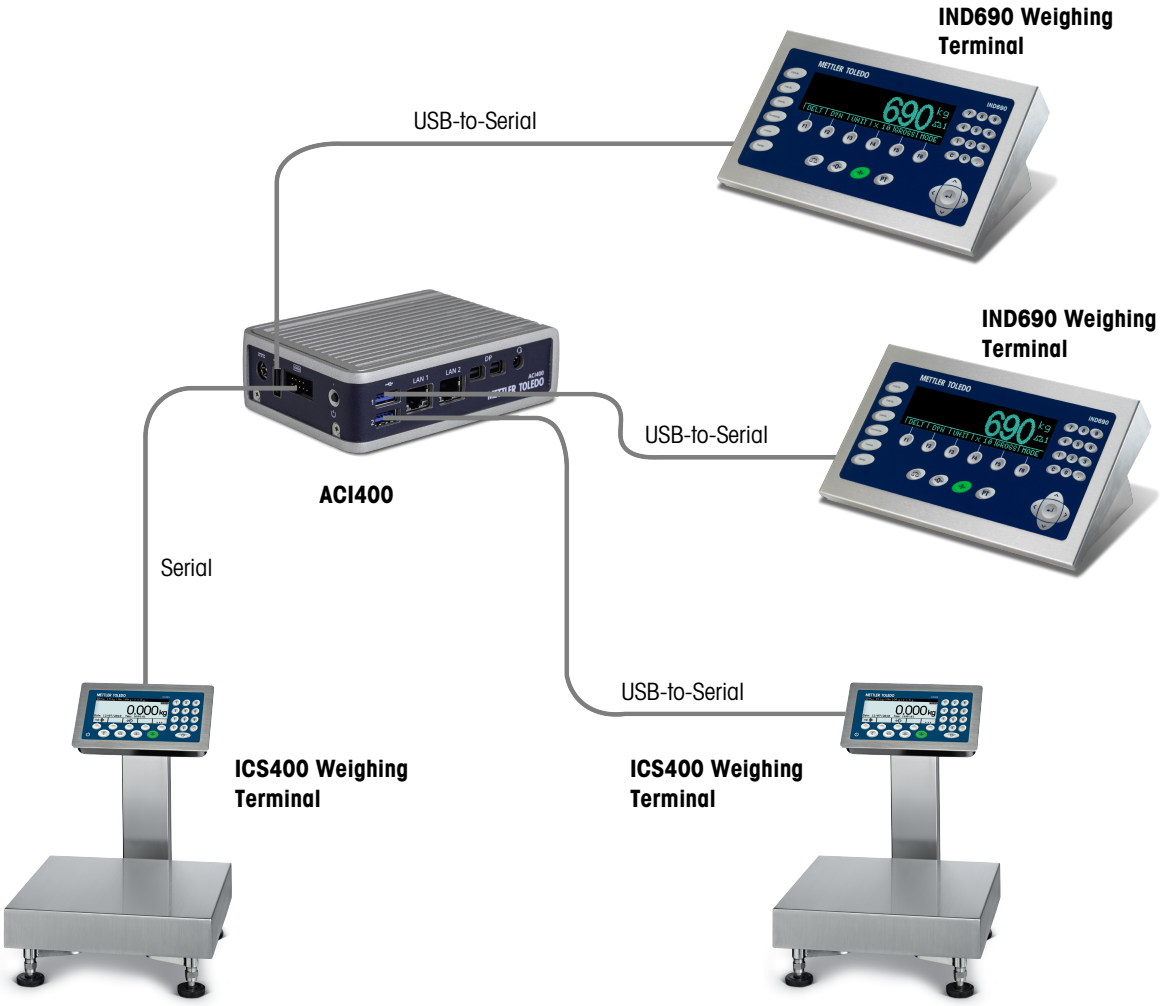

Figure 6: ACI400 Connected Serially with IND690s and ICS4-Series Terminals

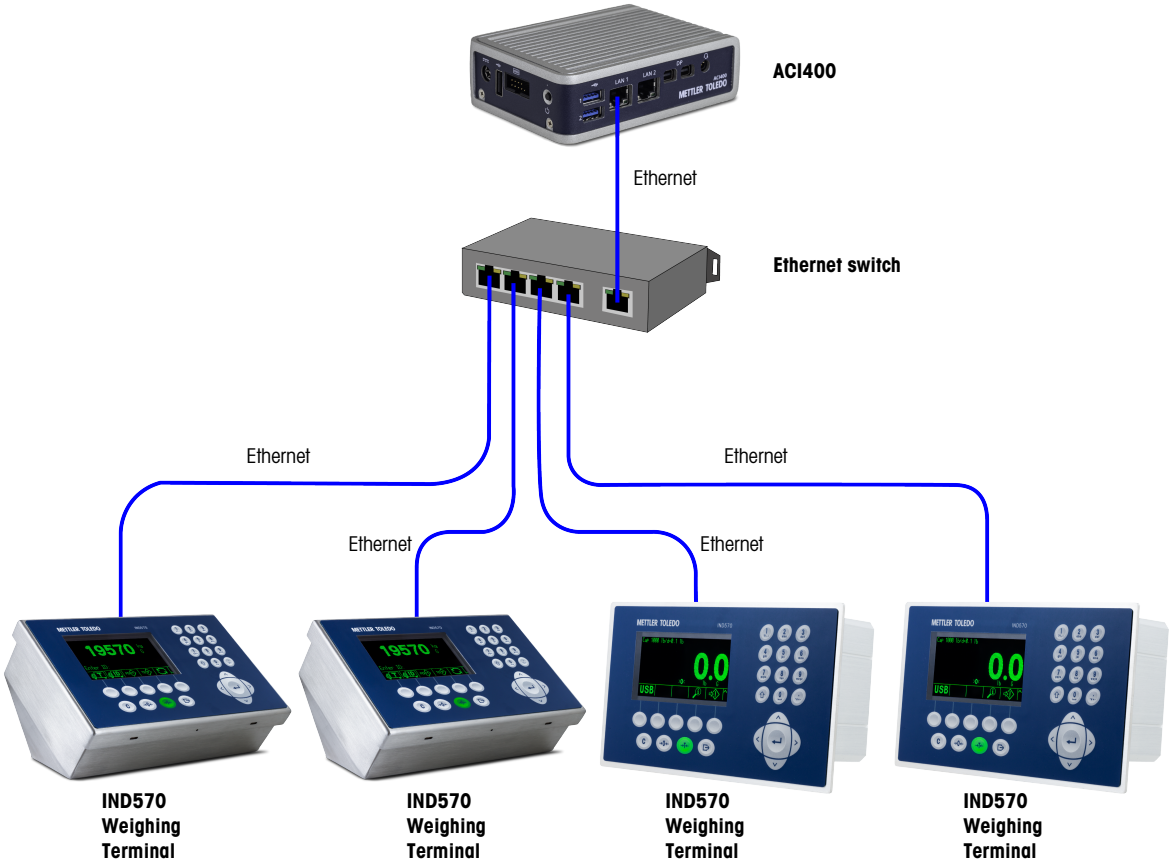

Figure 7: ACI400 Connected via Ethernet to Four Single-Scale IND570 Terminals

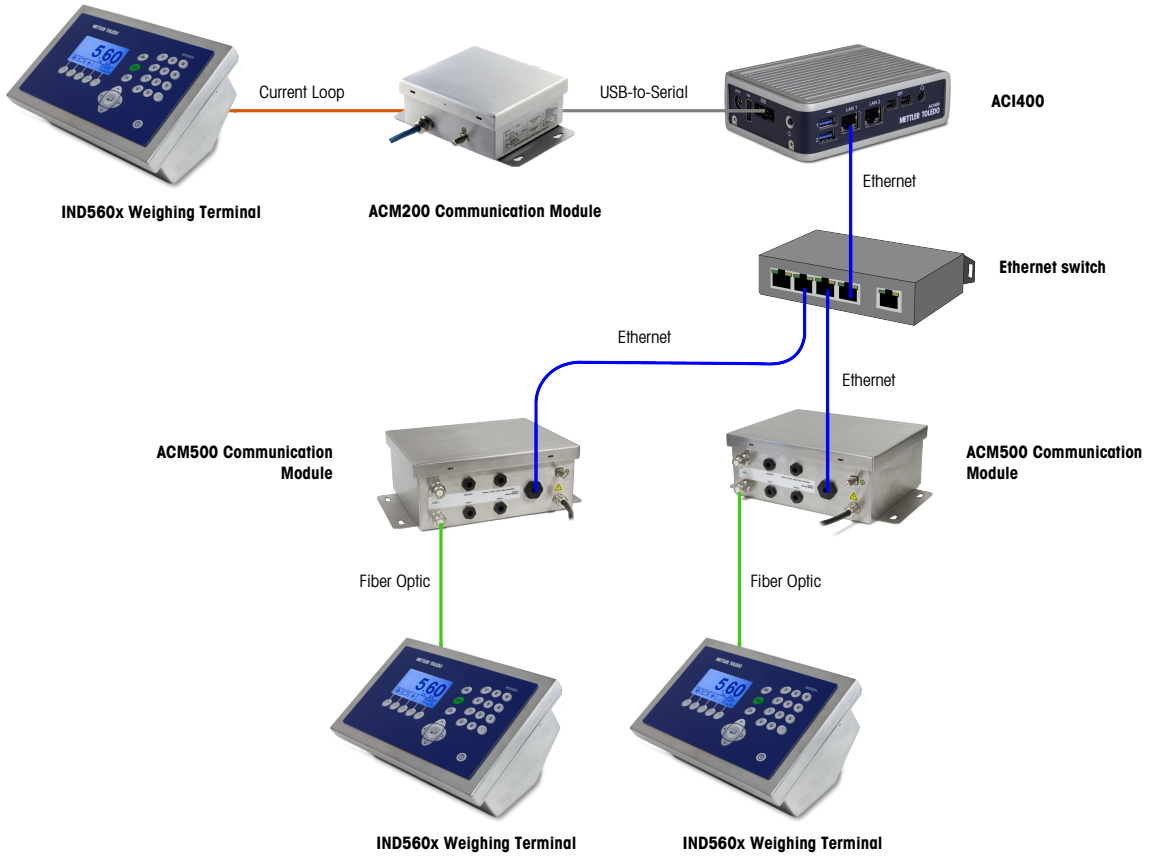

Figure 8: ACI400 Connected with IND560x Hazardous Area Terminls Using Both Serial and Ethernet

USB to Serial Converter Cable

METTLER TOLEDO recommends using the USB to Serial converter cable, part number 64088427, as it has proven performance with the ACI400

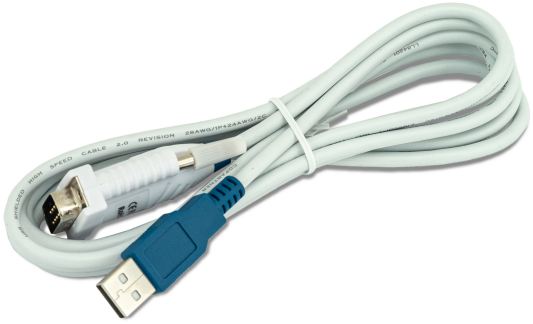

Figure 9: USB to Serial Converter Cable (Part Number 64088427)

2 ACI400 Configuration Interface

The ACI400 runs the Windows 10 IoT Enterprise LTSC operating system. An internal web service used to configure the ACI400.

2.1 Accessing the Web Interface

The ACI400 configuration webserver is accessed through an Ethernet connection between the ACI400 and a user PC on the same local network.

ACI400 has two physical LAN connections that correspond to two separate Ethernet adapters. LAN1 is configured at the factory with a static IP address to simplify initial access to the configuration webserver. Table 2 1 lists the default characteristics of each LAN port. The network settings of both Ethernet adapters are fully configurable through the configuration webserver.

ACI400 Default Network Adapter Settings

Physical Connection	Industrial Ethernet Adapter	Default IP Address
LAN 1	Ethernet 2	192.168.0.100
LAN 2	Ethernet 1	DHCP

In order to access the webserver on the ACI400, the accessing PC and the ACI400 must have correct IP configuration within the same local network. Based on the operating system of the accessing PC, the method to set the IP configuration of the PC can be different. To connect an ACI400 to a PC running Windows 10:

1 On the PC, go to **Control Panel | Network and Internet | Change Adapter Options**

2 Right-click the Ethernet network.

3 Click **Properties**.

4 Double click **Internet Protocol Version 4 (TCP/IP)**

5 Select **Use the following IP address**.

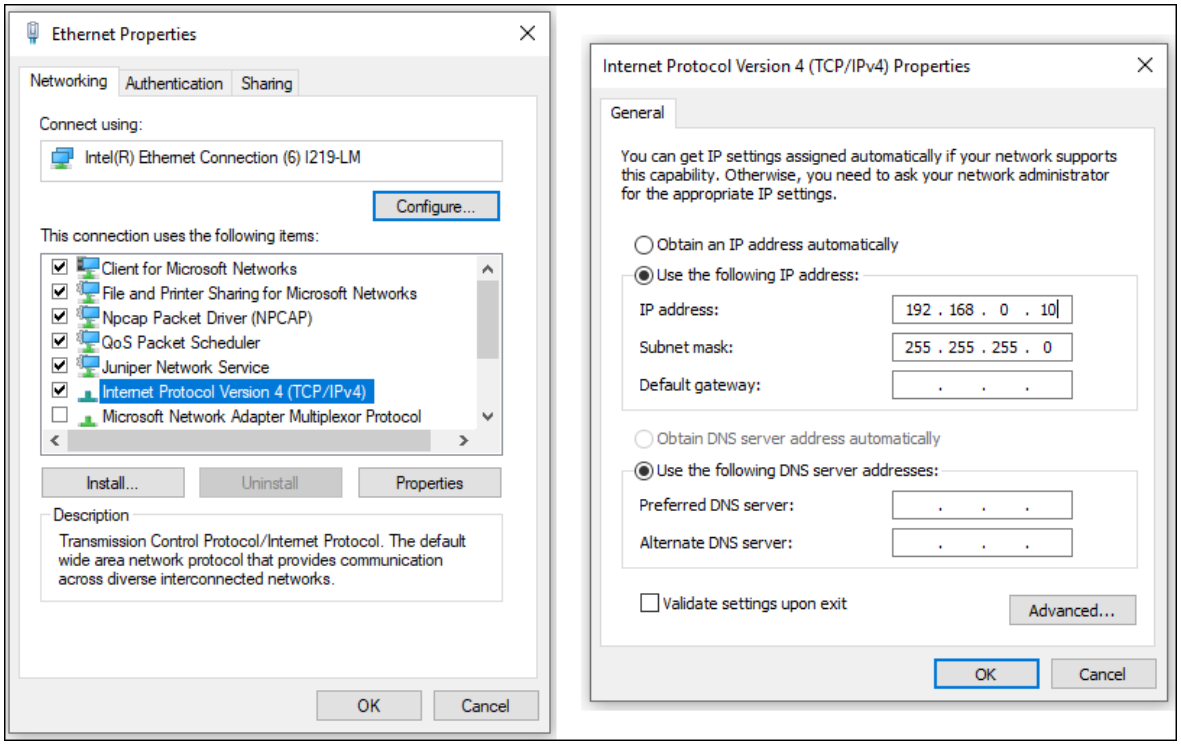

Figure 10: PC Connection Settings Example

6 Fill in the IP address settings of the PC to match the ACI400 network settings; for example --- MISSING LINK ---:

- ACI400 LAN1 default IP address: 192.168.0.100, default subnet: 255.255.255.0

- PC IP address: 192.168.0.10, subnet: 255.255.255.0

7 Using the LAN1 port on the ACI400, physically connect the ACI400 and the accessing PC to the same local network. Mettler Toledo recommends making this connection through a switch or network hub.

8 To access the ACI400 web server, use a Chrome or Microsoft Edge browser on the accessing PC to navigate to https://192.168.0.100:23491.

9 Because the ACI400 uses a self-signed certificate, a warning screen will appear. Accept the warning. Click Advanced (Chrome) or Details (Edge), and proceed to IP address 192.168.0.100.

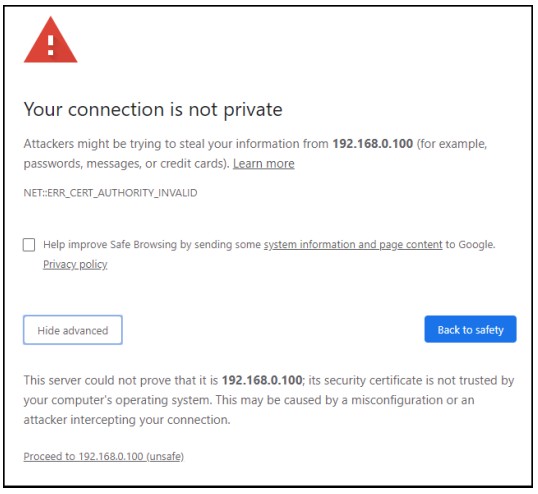

Figure 11: Certificate Warning Screen, MS Edge

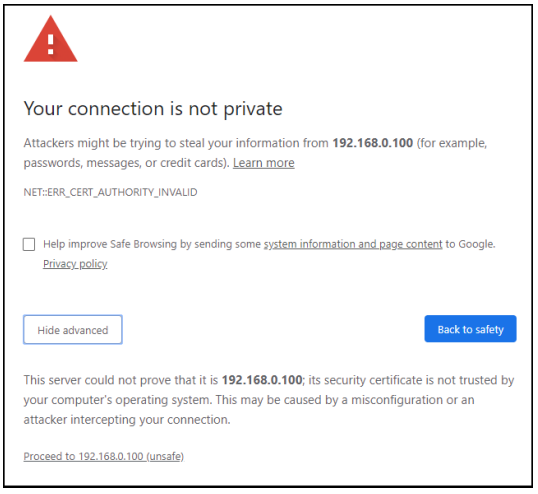

Figure 12: Certificate Warning Screen, Chrome

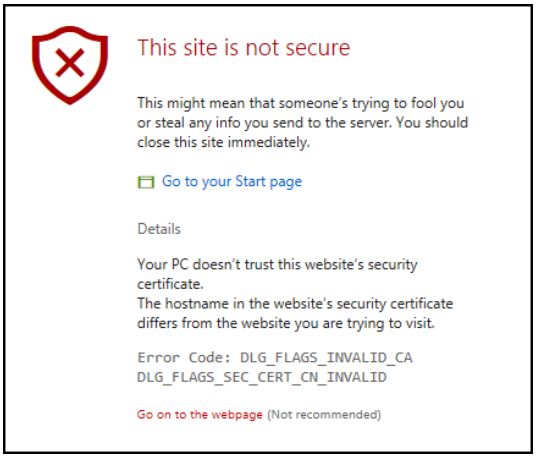

Figure 13: Cerfiticate Warning Screen, Edge

2.2 ACI400 IIoT Edge Home Page and Mode Selection

The configuration web pages consist of tabs, each of which displays configurable parameters and default settings for the ACI400. On the Home page, a configuration quick guide lists the minimum areas of configuration the user must program to set up the ACI400 for successful OPC UA or MQTT communication.

Current communication status of the ACI400 is always displayed in the upper right corner of each web page – either Running or Stopped.

NOTICE

Always close the ACI400 web browser from the Home page or the Admin>Diagnostics page

When a user accesses certain tabs of the configuration web pages, the ACI400 IIoT Edge device will stop all communication functions. Device connections, MQTT Clients, OPC UA Server, Admin and Communication functions will all restart automatically once the user returns to the Home page or the Admin>Diagnostics page.

If the ACI400 configuration web browser closes from any other page other than the Home page (or the Diagnostic page found on the Admin tab), all communication functions will remain stopped. All ACI400 communication functions will automatically restart when the web browser is accessed again and the user is directed to the Home page by default.

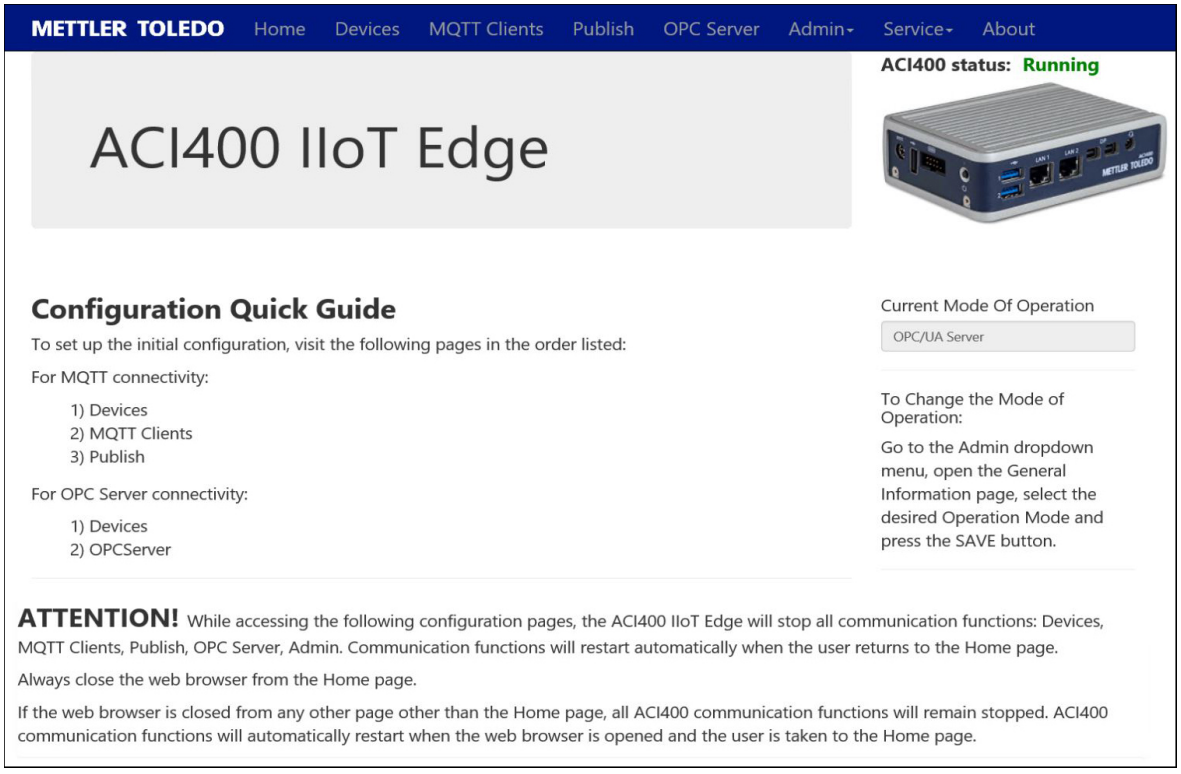

Figure 14: AC400 Configuration Browser Home Page with Status Display at Upper Right

To begin configuring the ACI400, first select **Admin** from the menu at the top of the Home Page. A dropdown list will appear.

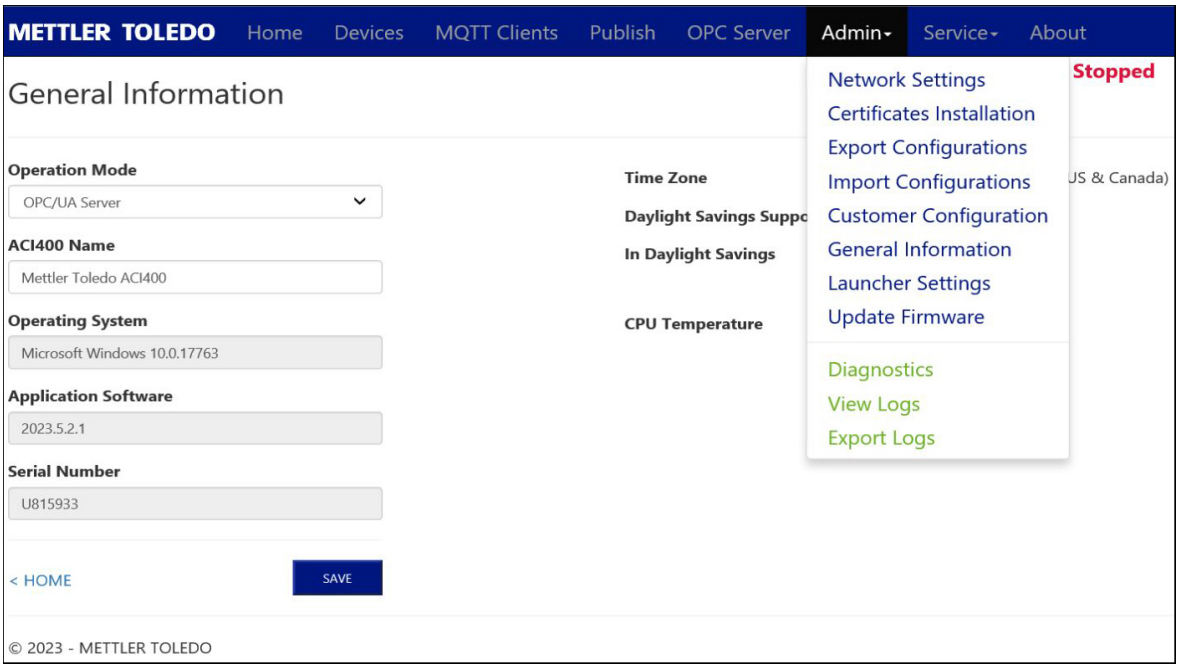

Figure 15: Admin Dropdown List

Select **General Information** from the Admin **dropdown** list, and touch the **Operation Mode** box.

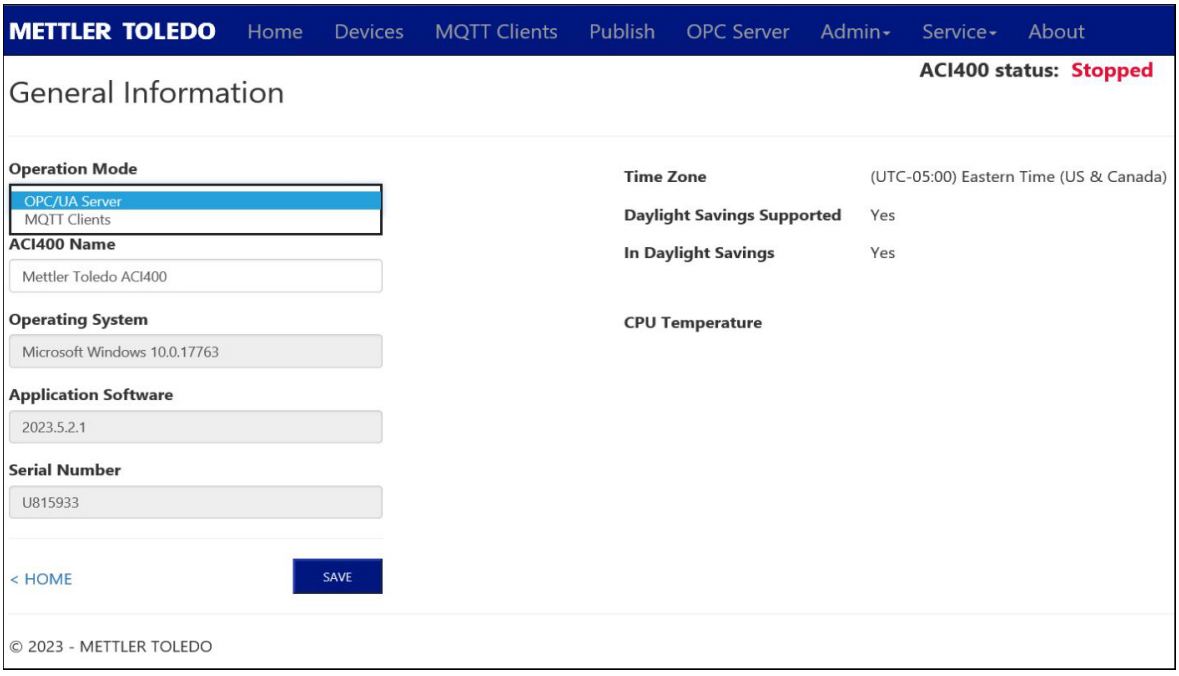

Figure 16: Selecting Mode of Operation

Depending on which **Operation Mode** has been selected, proceed to [MQTT Communication Configuration ▸ Page 18] or [OPC UA Communication Configuration ▸ Page 37], and follow the instructions to continue configuration.

2.3 Login

When faced with a login challenge, use the following login information to access most of the web pages:

- USERNAME: **mettler**

- PASSWORD: **mettlertoledo**

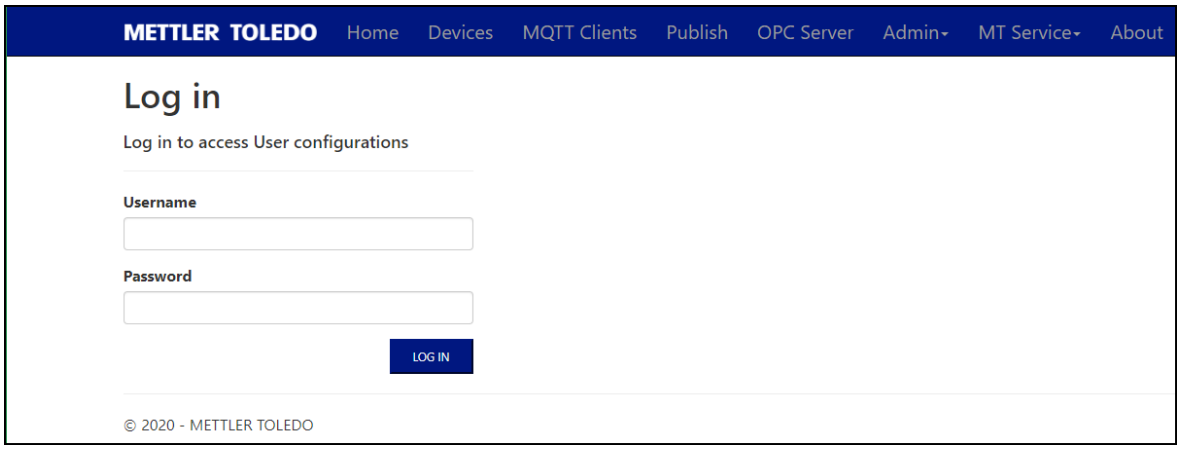

Figure 17: ACI400 Login Challenge

NOTICE

User name and password

It is recommended that customers change the default user name and password in the user management page under the service menu. Refer to [User Account Management ▸ Page 61].

3 MQTT Communication Configuration

To enable MQTT communication services in the ACI400, navigate to the following web browser tabs in the order listed:

- Devices
- MQTT Clients
- Publish

3.1 Devices

Configuration options available through the Devices tab allow a user to:

- Identify the weighing device/s with which the ACI400 will communicate
- Allow selection of communication settings such as protocol and connection types

The default DeviceInterfaceConfig.xml file (shown in [Device Interface File -- Add ▸ Page 18]) is provided as a reference for Device Interface File creation. This default file can either be edited or deleted.

To edit this device file, select the **EditName** or **EditFile** option on the far right of the **Device Interface Setup** table. To remove the default file (or any Device Interface File) select the **Delete** option to the right of the file name ([Device Interface File Configuration-Edit/Delete File Options ▸ Page 19]).

Note

If reusing a previous Device file name, ensure that any required changes are made before using the file.

Follow the steps below to create and set up a new Device Interface File:

1 Click the ADD button on the Launch Devices screen ([Device Interface File - Add ▸ Page 18]) to begin setup of a new device.

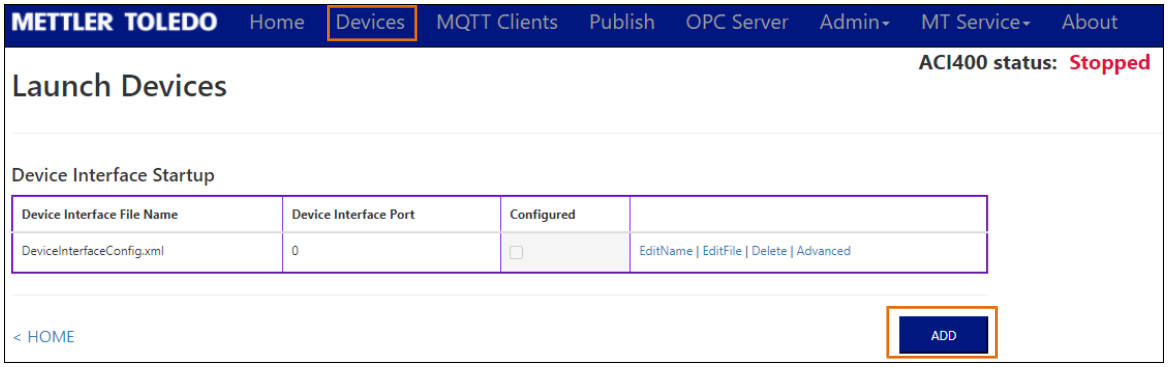

Figure 18: Device Interface File - Add

2 2. Type in a descriptive name for the **Device Interface File Name** – for example, IND570_1 – and click CREATE. The browser will return to the **Launch Devices** screen.

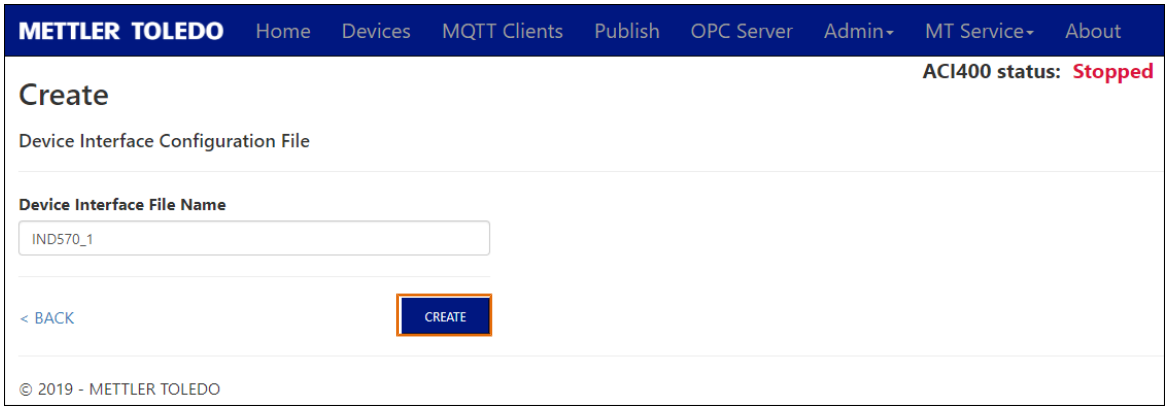

Figure 19: Device Interface File - Create

3 After creating the Device Interface File Name, select the **EditFile** option to the far right of the file name. The Device Interface Configuration parameters page will display. Descriptions of all Device Interface Configuration settings begin at [Device Interface Port ▸ Page 19].

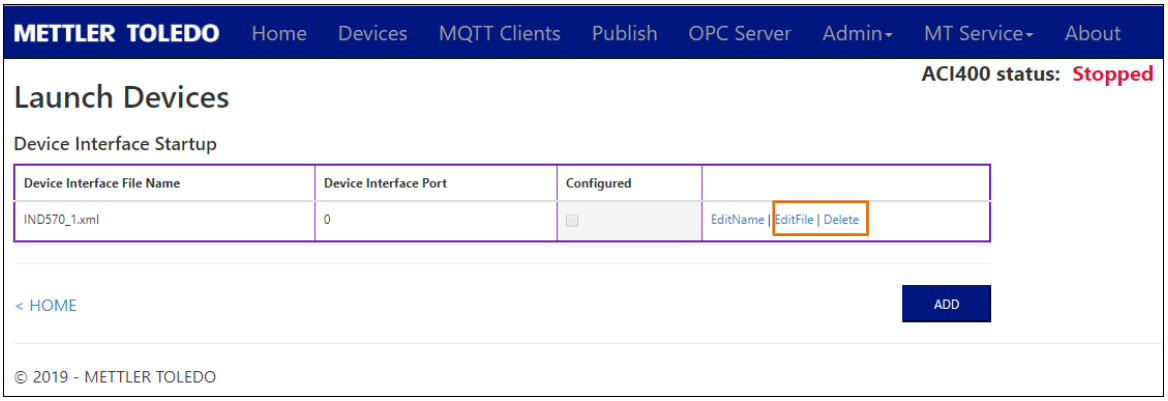

Figure 20: Device Interface File Configuration - Edit/Delete File Options

NOTICE

The Configured checkbox visible in [Devices ▸ Page 19] **is a visual aid only and is not selectable. Once a Device Interface File is created, it must be configured to be functional. The Configured checkbox will be greyed out until all required parameters are set in the Device Interface File.**

3.1.1 Device Interface Port

ACI400 uses the **Device Interface Port** to identify and route communication to the correct connected device.

The Device Interface Port number must be unique among **all** connected devices. The Device Interface Port number must also be outside the range of well-known restricted port numbers used by common networking features and software applications. Restricted port numbers generally fall between 0 and 1024. METTLER TOLEDO suggests a safe range for Device Interface Port settings between 5555 and 65535.

5555 is the default Device Interface Port number.

NOTICE

Port number 4840 should not be used as it is the default TCP/IP port for communication with the ACI400 OPC UA server.

The Device Port Number is used not only in communication between ACI400 and a connected device, but also in the [OPC UA server output ▸ Page 41] and [MQTT Payloads ▸ Page 48]. METTLER TOLEDO recommends that unique Device Interface Port numbers be assigned to **all** devices connected to **all** ACI400s in use on a network.

3.1.2 Device Connection Type

The **Device Connection Type** determines the physical connection that the ACI400 will use to communicate with the weighing device. There are two options for Device Connection Type: **Ethernet [default]** or **Serial**.

Select the connection type to use. The screen will update with a set of properties appropriate for the selected connection type.

Device Connection Type - Ethernet

When **Device Connection Type** is **Ethernet**, the following parameters are required:

- Device Protocol

- Device IP Address
- Device Port

3.1.3 Device ConnectionType - Ethernet

When **Device Connection Type** is **Ethernet**, the following parameters are required:

- Device Protocol
- Device IP Address
- Device Port

Device Protocol

The **Device Protocol** is the communication protocol the ACI400 will use to communicate with the weighing device. Options are **SDS** (Shared Data Server) [default], **SICS** (METTLER TOLEDO Standard Interface Command Set), **Print**, and **RPA455**.

Figure 21: Device Protocol Selections

Refer to Chapter 5, [Device Protocols ▸ Page 46], for additional details on the available Device Protocols.

Device IP Address

Device IP Address is the IP address of the weighing device with which the ACI400 will communicate. The default setting is **1922.168.0.1**.

Enter the IP address for the connected weighing device. The ACI400 and weighing device must have similar network interface settings (IP address, subnet) in order to communicate.

Network settings for the ACI400 can be modified under the **Admin** tab on the Network Settings page.

Device Port

The **Device Port** is the TCP/IP port the connected weighing device uses to communicate with the ACI400. Enter the appropriate TCP/IP port number for the connected device.

When **SDS** (Shared Data Server) is selected as the **Device Protocol over Ethernet**, the **Device Port** must always be 1701.

When **SICS** is selected as the **Device Protocol over Ethernet**, the port number will vary. For example, an ICS terminal uses port 4305 as the default Ethernet communication port.

Refer to the documentation of the connected device for details on the correct port setting.

Saving Changes

After entering the required Ethernet settings, click SAVE, then click <BACK to return to the Launch Devices screen.

3.1.4 Device Connection Type - Serial

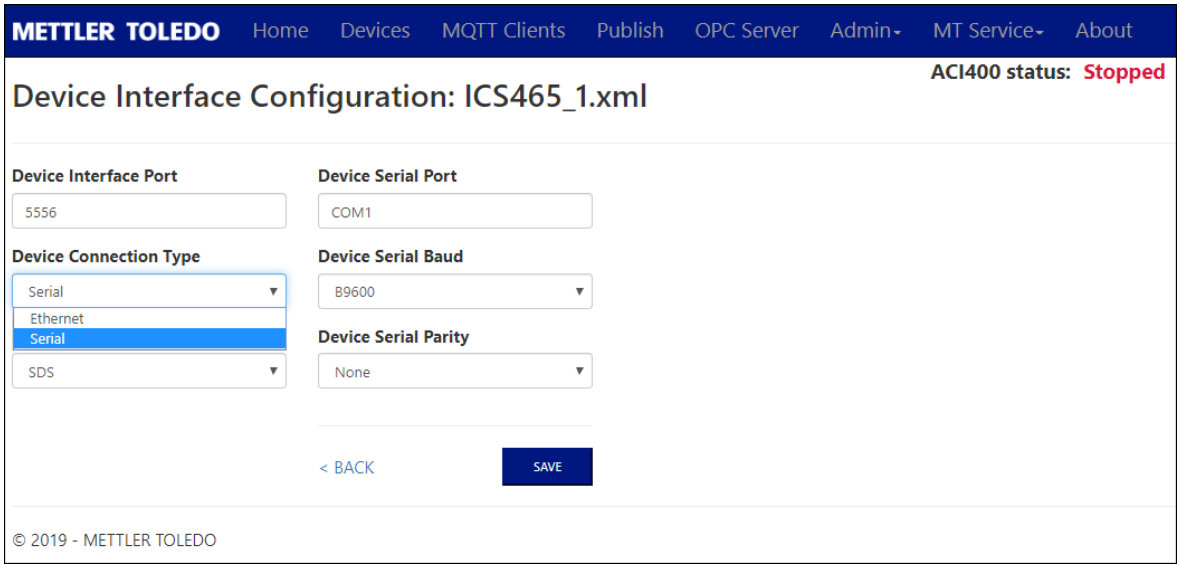

Figure 22: Device Connection Type - Serial Selected

When Device Connection Type is **Serial**, the following parameters are required:

- Device Protocol
- Device Serial Port
- Device Serial Baud
- Device Serial Parity

Device Protocol

The **Device Protocol** is the communication protocol the ACI400 will use to communicate with the weighing device. When Device Connection Type is Serial, the following Device Protocols are functional:

- **SICS** (METTLER TOLEDO Standard Interface Command Set)
- Print

NOTICE

SDS (Shared Data Server) and RPA455 are shown in the drop-down list, but ACI400 cannot connect to a device using SDS or RPA455 over a Serial connection.

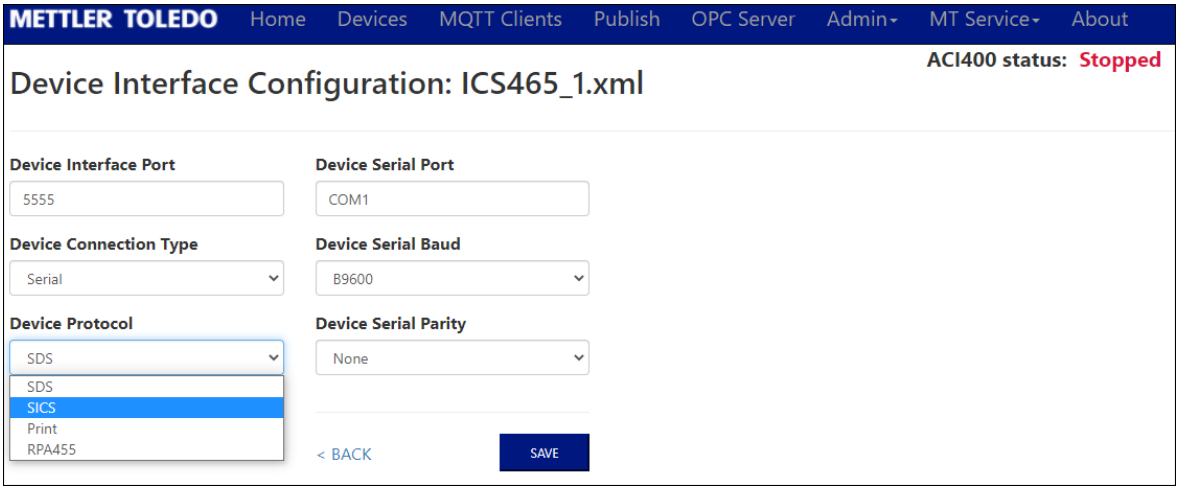

Figure 23: Device Protocol Selections

Device Serial Port

The **Device Serial Port** setting identifies the serial/COM port on the connected weighing device used to communicate with the ACI400. Enter the appropriate serial port ID.

Device Serial Baud

Use the **Device Serial Baud** setting to enter the baud rate of the serial port identified in the Device Serial Port field.

Options are **B9600** [default], B19200, B38400, B57600 and B115200. When using SICS as the Device Protocol, recommended settings are B9600 or B19200.

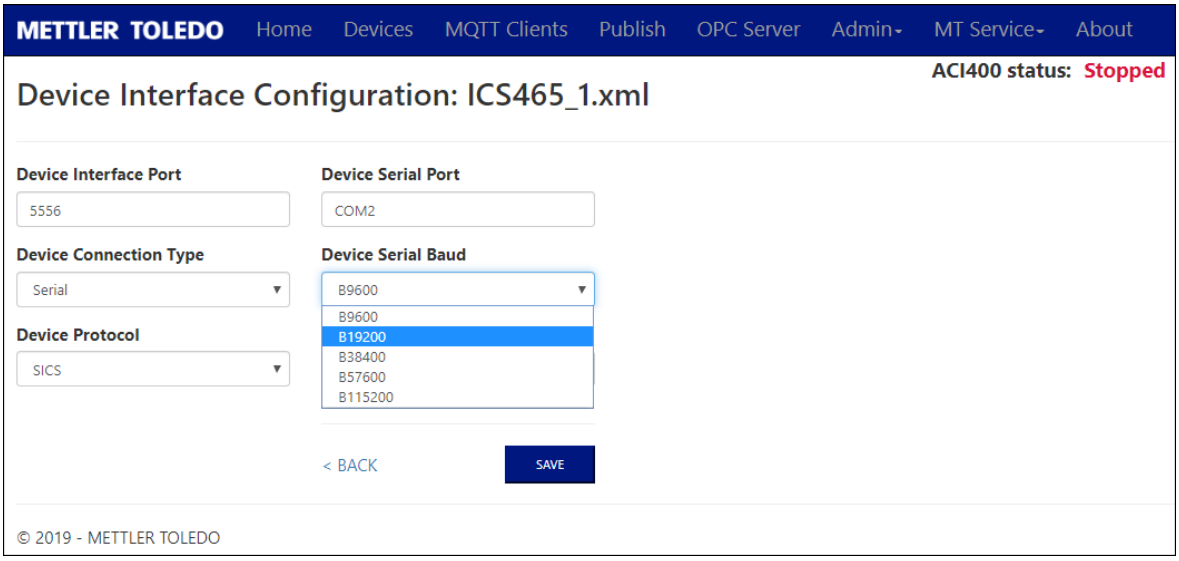

Figure 24: Device Serial Port Baud Options

Device Serial Parity

Enter the parity setting of the serial port identified in the **Device Serial Parity** field.

Options are **None** [default], Odd, Even, Mark and Space.

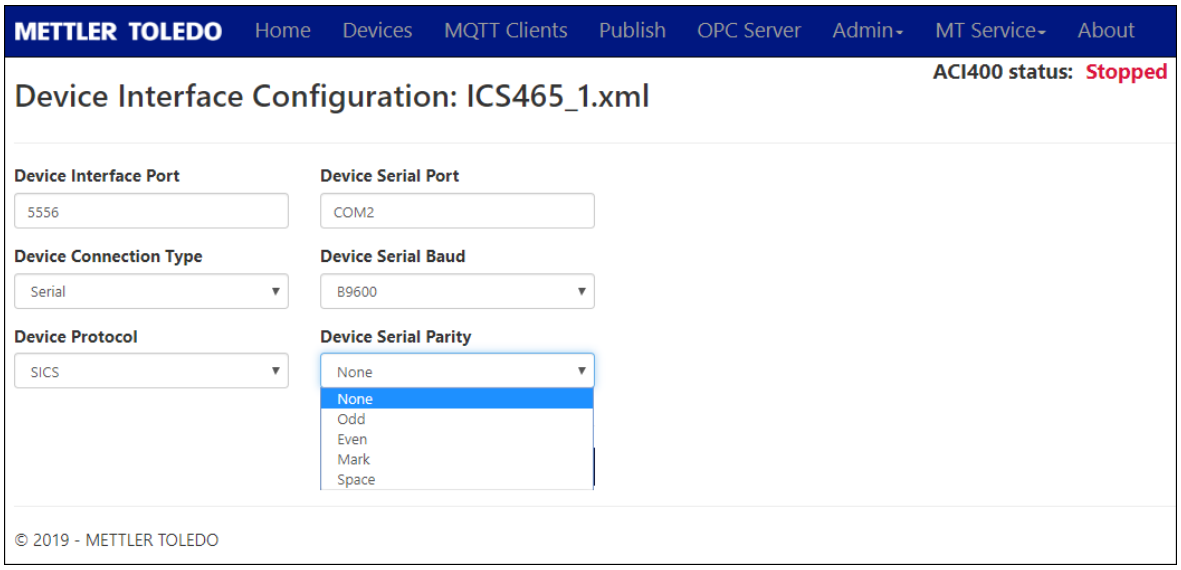

Figure 25: Device Serial Port Parity Options

3.1.5 Saving Changes

Once the required serial settings have been entered, click SAVE, then click <BACK to return to the **Launch Devices** screen.

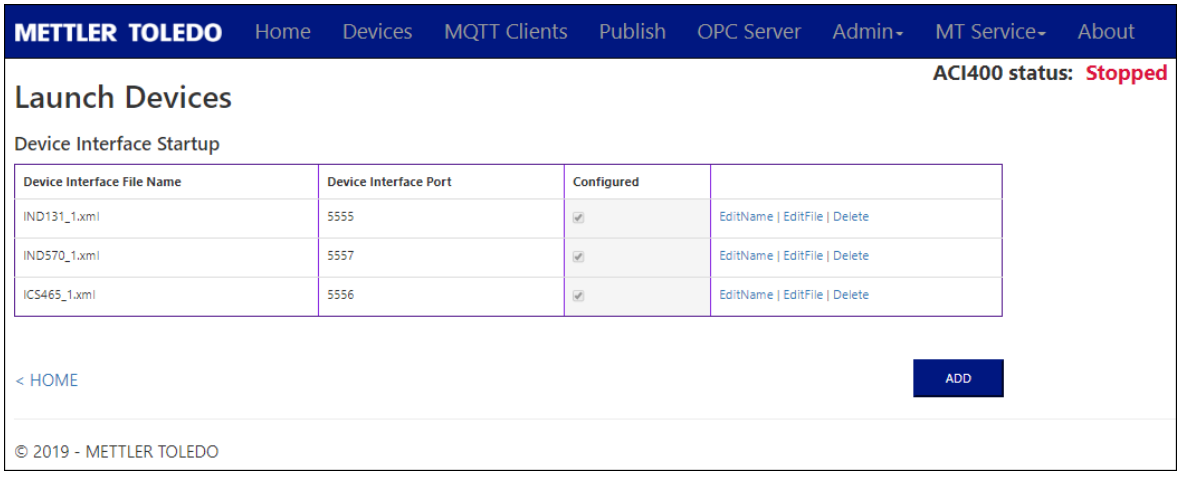

Figure 26: Launch Devices - Configured Devices List

3.2 MQTT Clients

In order for an ACI400 MQTT client to send (publish) data collected from a connected Device to an MQTT broker (receiver of published client data), an MQTT Client Configuration file must be created and configured. To create an MQTT client connection, navigate to the **MQTT Clients** tab.

The default MqttConfix.xml file (shown in Figure 3 11) is provided as a reference for Client Configuration File creation. This default file can be edited or deleted.

To edit this default file, select **EditName** or **EditFile** option to the far right of the MQTT Client Startup table. To remove the default file (or any MQTT Client Startup file) select the **Delete** option to the far right of the file name (Figure 3 11). The steps below guide the user through creation and setup of a new MQTT Client Startup file.

1 On the **Launch MQTT Clients** screen, click ADD to create a new **Client Configuration File**.

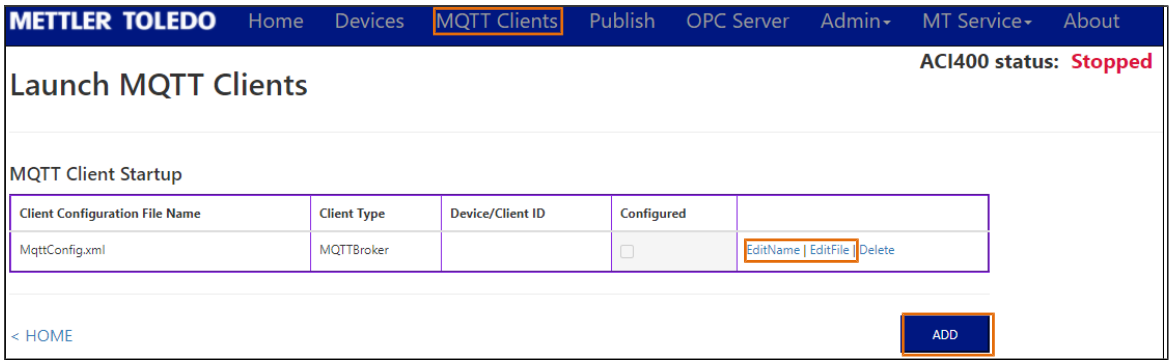

Figure 27: MQTT Client Configuration File - Add

2 Type in a descriptive name for the MQTT Client Configuration File - for example, **MQTT_Config_1_Azure** - and click CREATE.

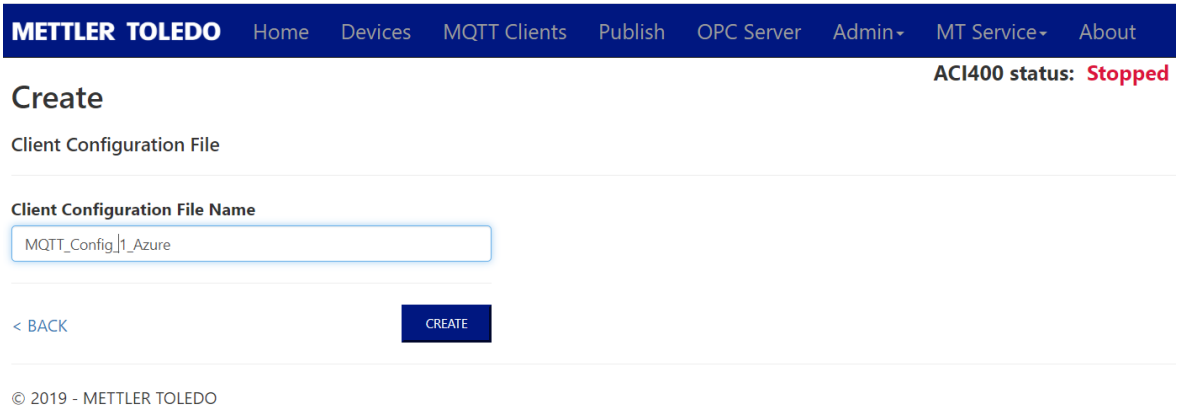

Figure 28: MQTT Client Configuration File - Create

3 The browser will return to the **Launch MQTT Clients** screen.

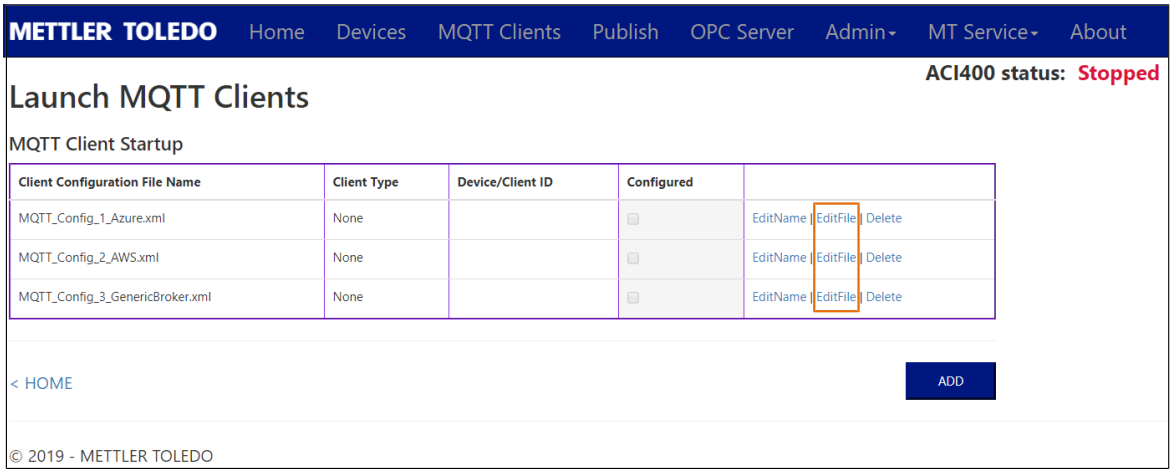

Figure 29: MQTT Client Configuration File - Edit File

4 After creating the Client Configuration file, click on the **EditFile** option to the right of the file name. The **MQTT Configuration** page will display. Descriptions of all MQTT Client Configuration settings begin at --- MISSING LINK ---.

NOTICE

Note that the Configured checkbox (shown in [MQTT Clients ▸ Page 24]**) is a visual aid only and not selectable. After a Client Configuration File is created, it must be configured in order to be functional. The Configured checkbox will be greyed out until all required parameters within the file are edited.**

5 To remove a file, select **Delete** ([MQTT Clients ▸ Page 24]).

NOTICE

Deleting a file from the configuration web pages does not completely remove the file from the ACI400 internal file directory. If a new Client Configuration File is created with a previously used file name, the new file will point to the previously created file and its associated settings. Ensure that required changes are made to any file that reuses a previous file name.

MQTT Client Configuration File

Some of the information required to fully set up an MQTT Client Configuration file is generated by end user's chosen MQTT broker (cloud service).

The following section lists basic descriptions of the **Client Configuration File** setup required for each MQTT Client type supported in the ACI400.

The first step in MQTT Configuration file setup is to select the **Client Type**. Selections include:

None	For no connection.
MQTTBroker	For connection to an [MQTT broker ▸ Page 26] not specifically listed in the drop-down
Amazon	For connection to [Amazon Web Services ▸ Page 28] (AWS)
Azure	For connection to [Microsoft Azure ▸ Page 29]

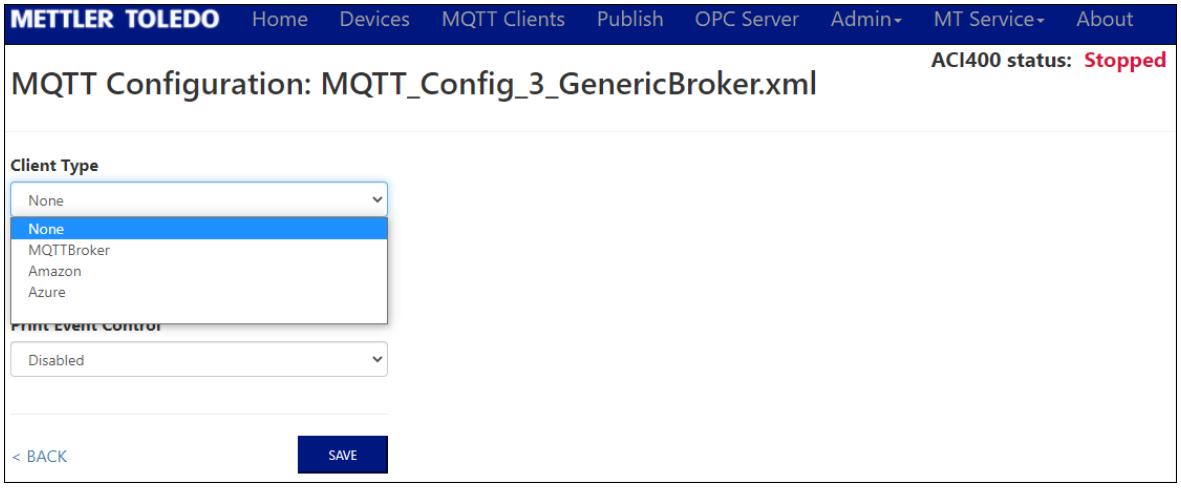

Figure 30: MQTT Client Type Selection

Client Configuration File Parameters - Client Type MQTTBroker

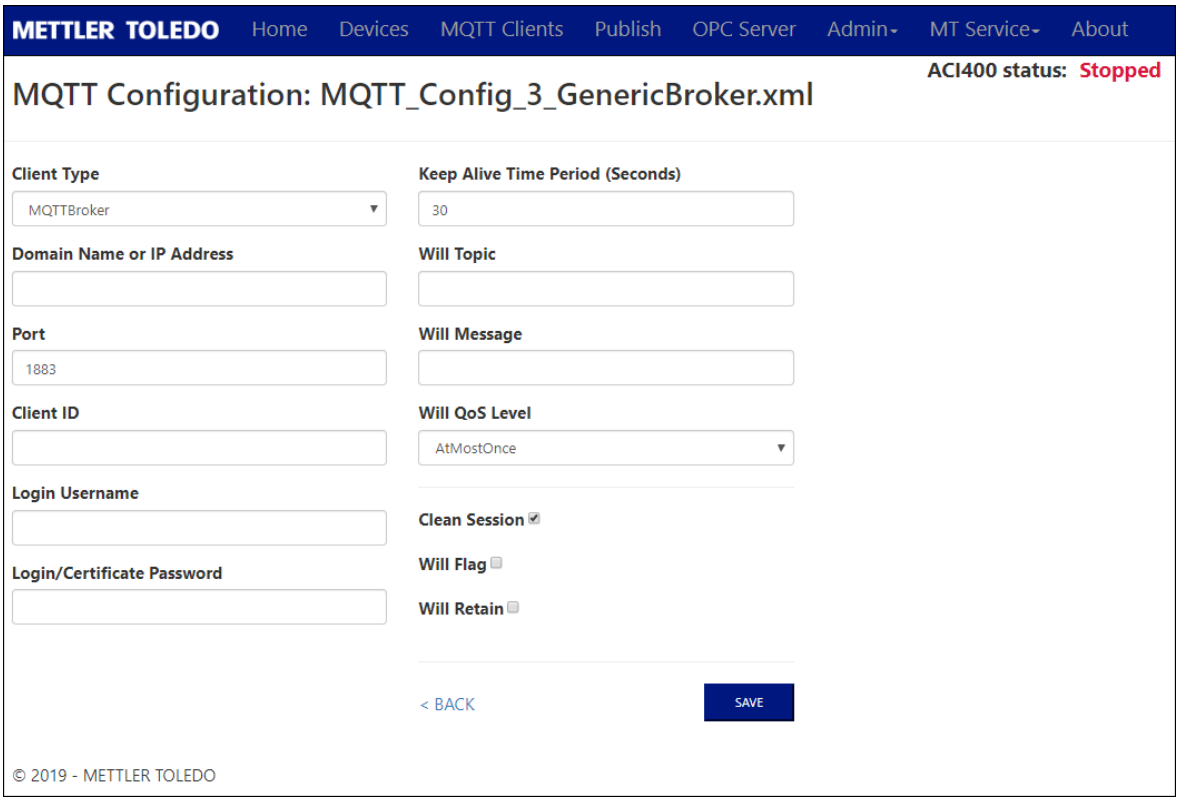

Figure 31: MQTT Client Type MQTTBroker

MQTTBroker Client Configuration Parameters

Parameter	Description
Client Type	MQTTBroker
Domain Name or IP Address	Endpoint or web address of the MQTT broker to which the ACI400 MQTT client will connect
Port	Access port at the MQTT Broker. Standard broker access ports include 1883 and 8883.
Client ID	Client ID identifies each MQTT client that connects to an MQTT broker. The broker uses the Client ID to identify the connecting client and the current state of that client. The Client ID can be blank if the client state does not need to be held by the broker. An empty Client ID results in a connection without any state. If the Client ID is blank, the Clean Session Flag must checked (set to ON/TRUE) or the MQTT broker will reject the connection.
Device Event Control	Refer to Device Event Control, below.
Print Event Control	Refer to Print Event Control, below.
Login Username	ACI400 MQTT clients can send a user name and password for the purpose of client authentication and authorization at the broker.
Login/Certification Password	ACI400 MQTT clients can send a user name and password for the purpose of client authentication and authorization at the broker.

Keep Alive Time Period	**Keep Alive** is a time interval in seconds that defines the longest period of time that the broker and client connection can endure without exchanging a message. The ACI400 MQTT client will send a regular ping request messages to the broker as defined by the **Keep Alive Time Period**. The broker responds with a ping response. This method allows both sides to determine if the other one is still available.
Optional Data Type	ACI400 makes 2 optional data fields available that can be appended to published messages. These optional fields are activated by setting the Optional Data Type to **CustomerConfiguration**. These data structures are configured in the ACI400 web pages at **Admin > Customer Configuration** ([MQTT Clients ▸ Page 27])

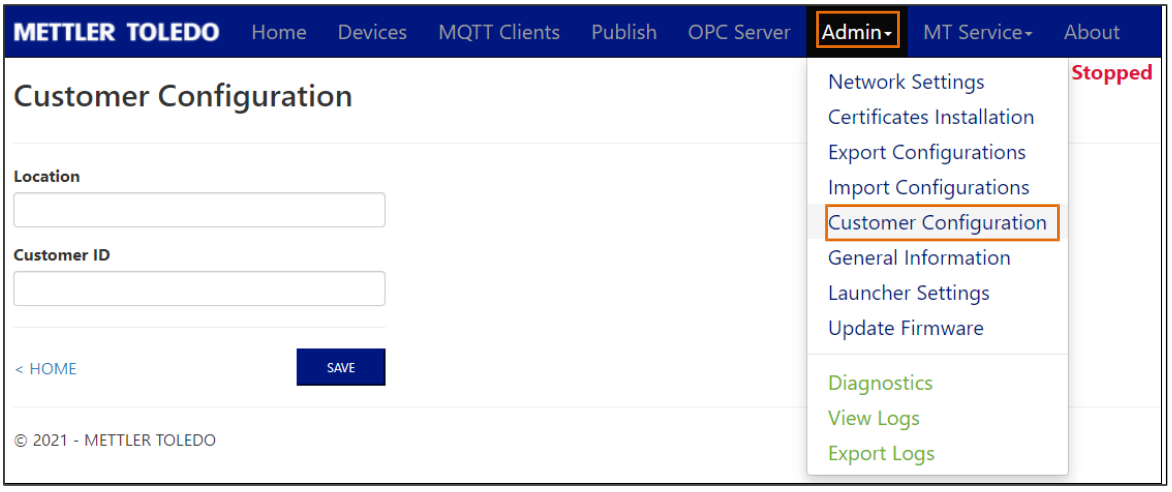

Figure 32: Access to Optional Data Type (Customer Configuration) Setup

Last Will and Testament (LWT)

The following settings control the function of Last Will and Testament (LWT) feature of the generic MQTTBroker client. An MQTT broker uses the last will and testament message to notify subscribing clients of an unexpected shut down of the publishing client.

NOTICE

The following settings only apply to the LWT feature. An additional QoS (Quality of Service) setting is available in the Publish Item setup for all other payloads (messages) selected for publication to a broker. (Refer to Print Event Control.)

LWT Settings

Will Topic	Topic (in general) refers to a string that an MQTT broker uses to filter messages for each client connection. **Will Topic** allows the broker to specifically filter and send the Will Message to clients who have subscribed to the Will Topic.
Will Message	When a client first connects, the **Will Message** is sent to the broker and stored by the broker. In the event that the client disconnects ungracefully, the broker will send this message as a notification to all subscribers of data sent from the disconnected client.

Will QoS Level	**Will Quality of Service Level** determines the reliability of the Will Message delivery between the MQTT client and the MQTT broker. • **AtMostOnce** (QoS Level 0) – This is the simplest, lowest-overhead method of sending a Will Message. The ACI400 client simply publishes the message, and there is no acknowledgement by the broker. There is no guarantee of delivery. • **AtLeastOnce** (QoS Level 1) – This method guarantees that a Will Message is transferred successfully to the broker. In this service level, the ACI400 client sends a Will Message and expects the broker will acknowledge receipt of the message. If the broker does not acknowledge receipt, the ACI400 MQTT client will resend the Will Message multiple times until it receives acknowledgement from the broker. There will be no indication that it is a repeat of the Will Message, so it is possible for a Will Message to reach the broker more than once. • ExactlyOnce (QoS Level 2) – This is the highest level of service in MQTT in which there is a sequence of four messages between the sender (ACI400 client) and the receiver (broker), a kind of handshake to confirm that a specific Will Message has been sent and that the acknowledgement has been received. Both ends of this transaction are assured that the Will Message was sent/received exactly once.
Clean Session	The **Clean Session** flag tells MQQT whether or not the MQQT client wants to establish a persistent session. • In a persistent session (CleanSession flag = OFF), the broker stores all subscriptions for the client and all missed messages for the client that subscribed with a Quality of Service (QoS) level 1 or 2. • If the session is not persistent (CleanSession flag = ON), the broker does not store anything for the client and purges all information from any previous session.
Will Flag	If the **Will Flag** is ON, the Will QoS and Will Retain fields are also used during connection. The Will Topic and Will Message fields must be populated, as these data structures will be present in the message payload. If the Will Flag is OFF, the Will QoS and Will Retain field must be blank, as the Will Topic and Will Message fields must not be present in the message payload.
Will Retain	When **Will Retain** is ON, the ACI400 client instructs the broker to keep the last Will Message on a topic even is there are no active subscriptions.

Client Configuration File Parameters - Client Type = Amazon

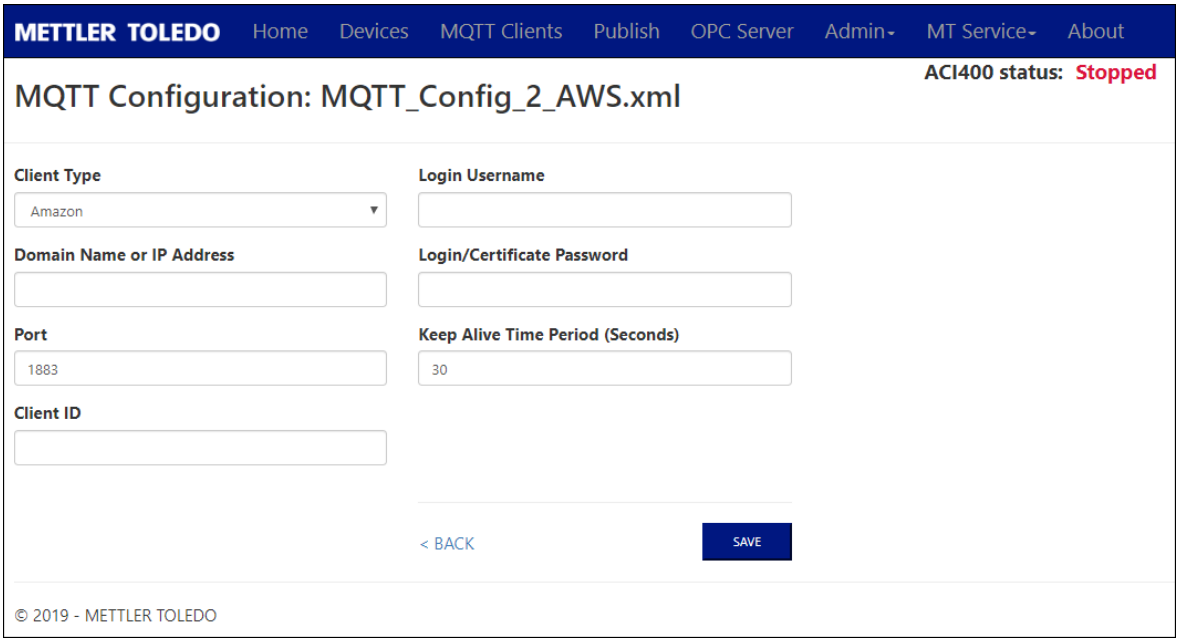

Figure 33: Client Type = Amazon

Parameter	Explanation
Client Type	Amazon (Amazon Web Services, AWS)
Domain Name or IP Address	Endpoint or web address of the targeted AWS IoT Core.
Port	Endpoint or web address of the targeted AWS IoT Core.
Client ID	Aazure
Device Event Control	Refer to [Device Event Control ▸ Page 30]
Print Event Control	Refer to [Print Event Control ▸ Page 30]
Login Username	Login key required by AWS IoT Core when the ACI400 client attempts to make a connection. The Login Username entered in the ACI400 must match with what is stored in AWS.
Login Certificate/ Password	Security key required by AWS when the ACI400 client attempts to make a connection to AWS IoT Core. This security key is generated by AWS and must be entered into the MQTT configuration file in the ACI400.
Keep Alive Time Period	Frequency the ACI400 Amazon client will ping the broker to keep the connection alive.
Optional Data Type	ACI400 makes 2 optional data fields available that can be appended to published messages. These optional fields are activated by setting Optional Data Type to CustomerConfiguration. These data structures are configured in the ACI400 web pages at [MQTT Clients ▸ Page 27].

Client Configuration File Parameters - Client Type = Azure

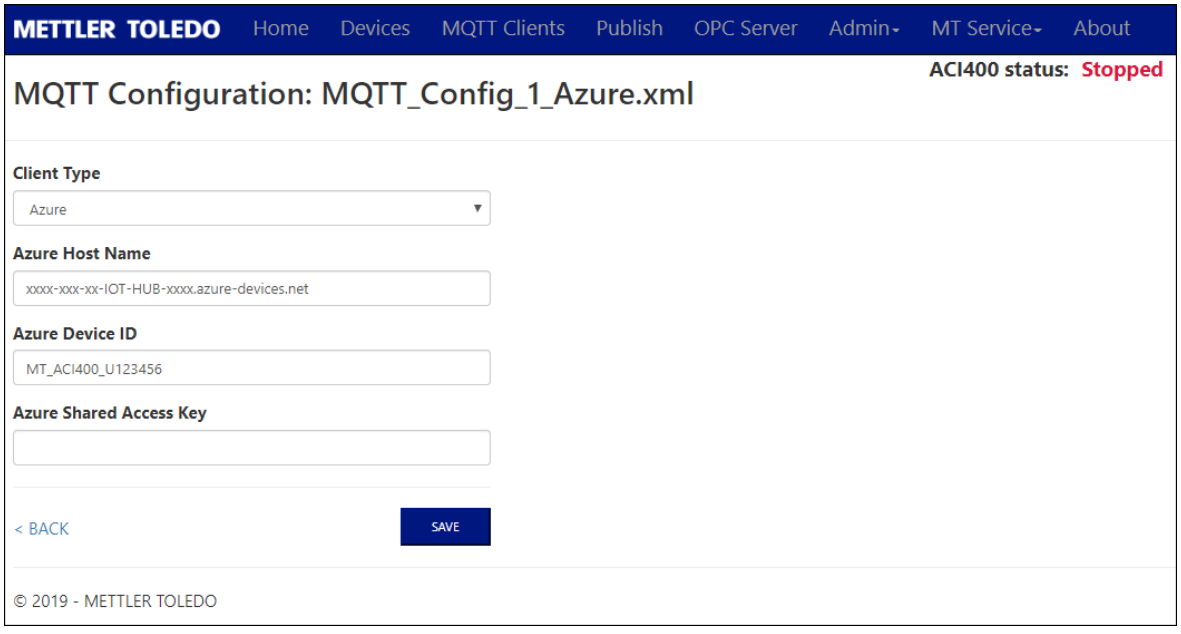

Figure 34: Client Type = Azure

Azure Client Configuration Parameters

Parameter	Explanation
Client Type	Azure (Microsoft Azure)
Azure Host Name	Endpoint or web address of the targeted Azure IoT Hub.
Azure Device ID	Unique device name assigned to the ACI400 Azure client and used by Azure for identification purposes when the ACI400 client connects to Azure IoT Hub.

Azure Shared Access Key	Security key required by Azure when the ACI400 client attempts to make a connection to Azure IoT Hub. This security key is generated by Azure and must be entered into the MQTT configuration file in the ACI400.
Optional Data Type	ACI400 makes 2 optional data fields available that can be appended to published messages. These optional fields are activated by setting Optional Data Type to CustomerConfiguration. These data structures are configured in the ACI400 web pages at [Admin > Customer Configuration ▸ Page 27].
Device Event Control	Refer to [MQTT Clients ▸ Page 30]
Print Event Control	Refer to [MQTT Clients ▸ Page 30]

Device Event Control

The **Device Event Control** parameter allows the ACI400 MQTT client to pass "Alert" type events from connected Devices to the broker.

- Setting Device Event Control as **ForwardAllEvents** will allow all events coming from a connected device to be passed through the ACI400 MQTT client to the broker
- When Device Event Control is **Disabled**, the ACI400 client will not pass any Device events tot he broker.

Print Event Control

The Print Event Control setting regulates the publishing of the PrintEventPayload by an MQTT client. The Print Event Control setting is only relevant when Print is the selected Device Interface Protocol (refer to [Device ConnectionType - Ethernet ▸ Page 20] and [Device Connection Type - Serial ▸ Page 21]

- • When Print Even Control is set as ForwardAllPrintOutput, the ACI400 MQTT client will publish the Print-EventPayload when a Print request is received by the client. A Print request to the client can be triggered when the ACI400 client receives a Print command from the broker or when a connected device executes a local Print function (e.g. an operator presses the PRINT or TRANSMIT button on the connected device or the device executes an Auto-Print).
- • When Print Event Control is Disabled, ACI400 clients will not act on any Print triggers and the PrintEvent-Payload will not be published.

See also

- MQTT Clients ▸ Page 24
- MQTT Clients ▸ Page 27
- MQTT Clients ▸ Page 30
- MQTT Clients ▸ Page 30

3.2.1 Saving Changes

After modifying the required settings for the MQTT Client Configuration File, click ADD, then click <BACK to return to the Launch MQTT Clients screen.

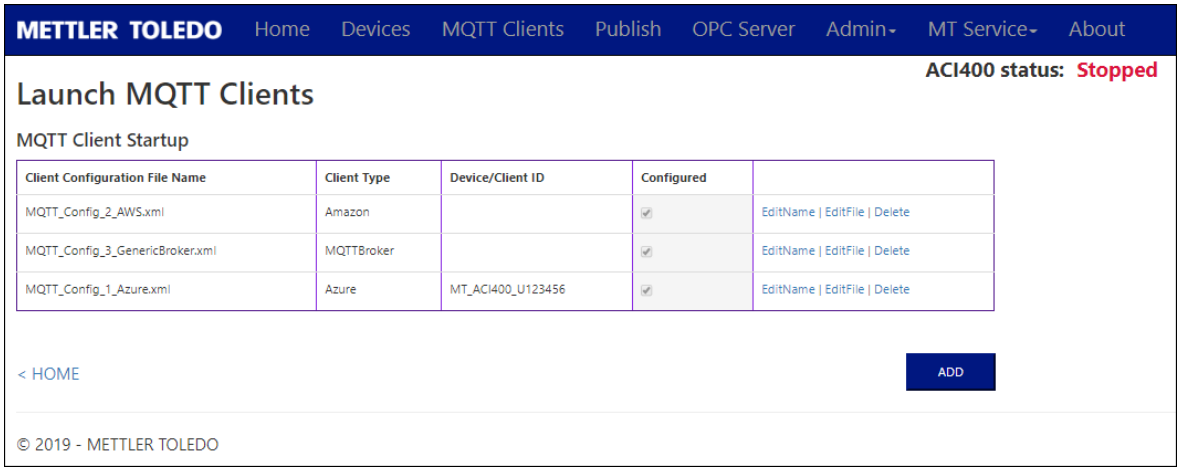

Figure 35: Launch MQTT Clients - MQTT Client Configuration Files List

3.3 Publish

The next step in configuring MQTT communication in the ACI400 is selecting payloads to publish.

To begin payload selection, navigate to the **Publish** tab. Click ADD on the **Publish Configuration** screen to create a new Publish Item.

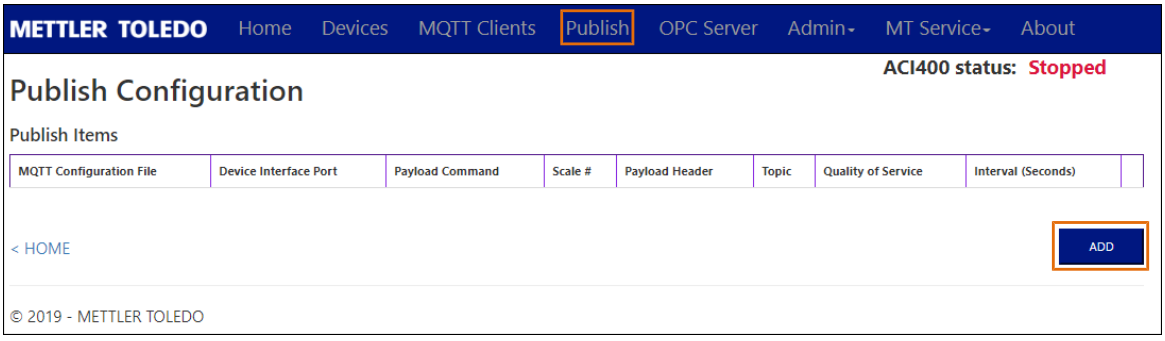

Figure 36: Publish Configuration - Add

3.3.1 Create Publish Item

The **Published Item** file brings together a Device configuration with an MQTT Client configuration, creating the secure path for data from a specific connected device to travel to a specific cloud service.

MQTT Configuration File

The **MQTT Configuration File** selection identifies the MQTT broker (or targeted cloud service) to which a payload will be published/transmitted.

Select the appropriate **MQTT Configuration File** from the list of files created in the ACI400.

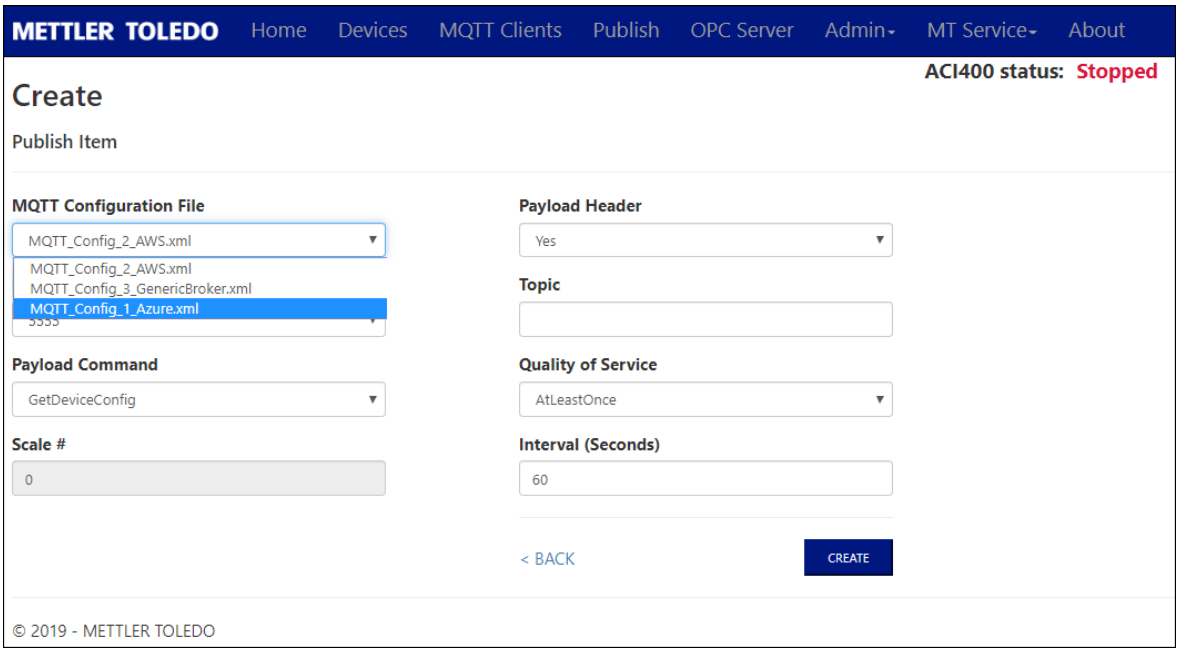

Figure 37: MQTT Client Configuration File Selection

Device Interface Port

During Device creation (Launch Devices - Configured Devices List), a unique Device Interface Port number was assigned to each connected weighing device. Here in the creation of a Publish Item, the Device Interface Port setting determines which connected device will populate the data fields of the published payload (selected in the Payload Command setting).

Select the appropriate Device Interface Port (i.e. the appropriate connected device) from the dropdown list.

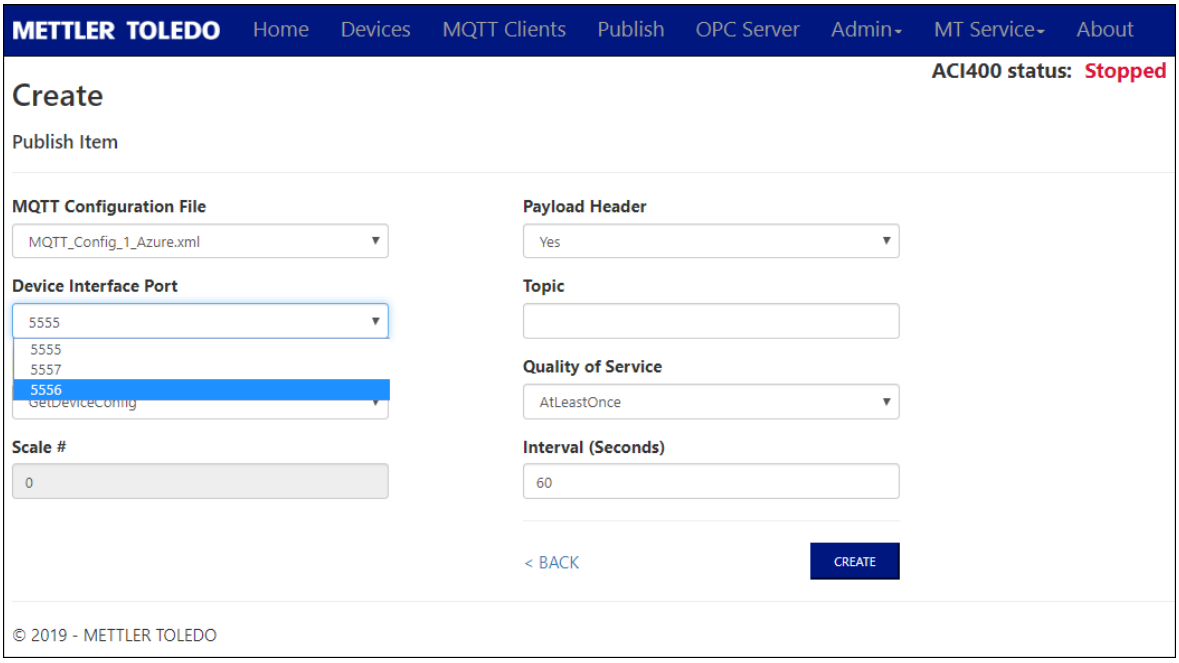

Figure 38: Device Interface Port Selection

Payload Command

Payloads are predefined data structures that an ACI400 client can publish as JSON messages to an MQTT broker.

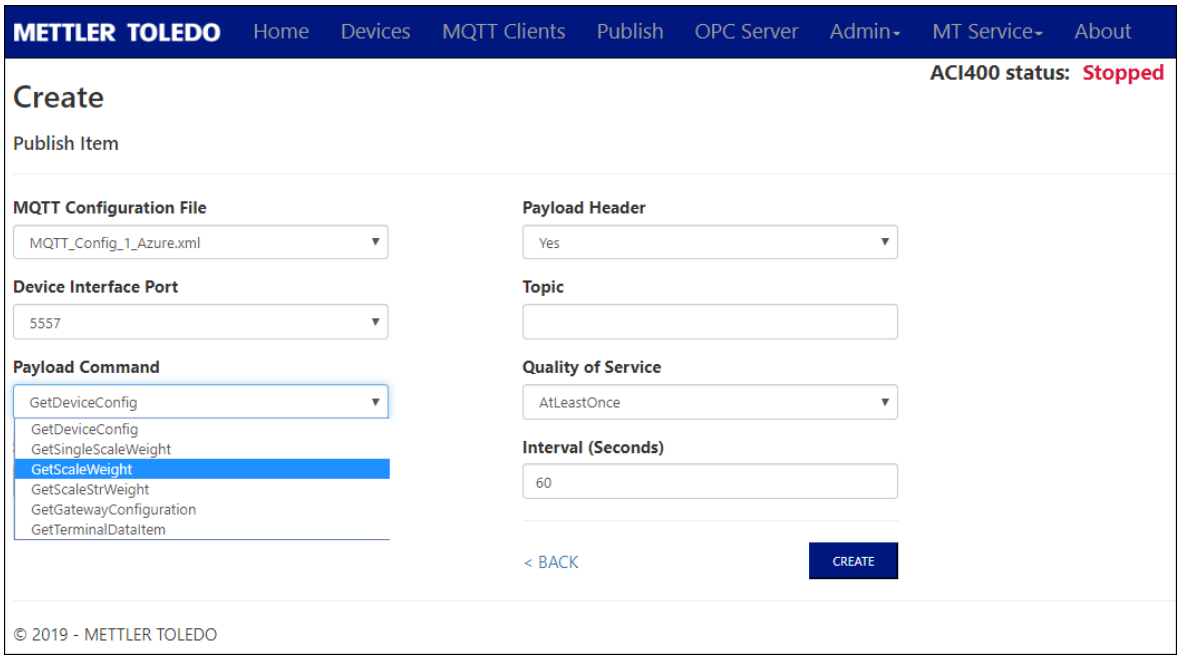

Figure 39: Payload Command Selection

Available **Payload Command** selections are:

Payload Command Selections

GetDeviceConfig	GetScaleStrWeight
GetSingleScaleWeight	GetGatewayConfiguration
GetScaleWeight	GetTerminalDataItem

Details on the structures of all available MQTT payloads are listed in [MQTT Payload Structures and General Commands ▸ Page 48]

NOTICE

Note that the PrintEventPayload is not listed in the dropdown. The PrintEvent-Payload is published as a part of the Print protocol function and is not managed by the Publish Items function. Please refer to Chapter 5,[Device Interface Protocols ▸ Page 46]**, for additional details on the Print protocol methodology.**

Scale

For connected devices with multiple scale channels, the **Scale #** determines the scale channel on a connected weighing device that populates the data fields of a published payload.

The Scale # field is only active when the GetSingleScaleWeight payload is selected. The **GetSingleScaleWeight** payload only applies when a connected device supports multiple scale channels. The IND780 and IND689 terminals are examples of a multi-scale devices.

If appropriate, enter the **Scale #**.

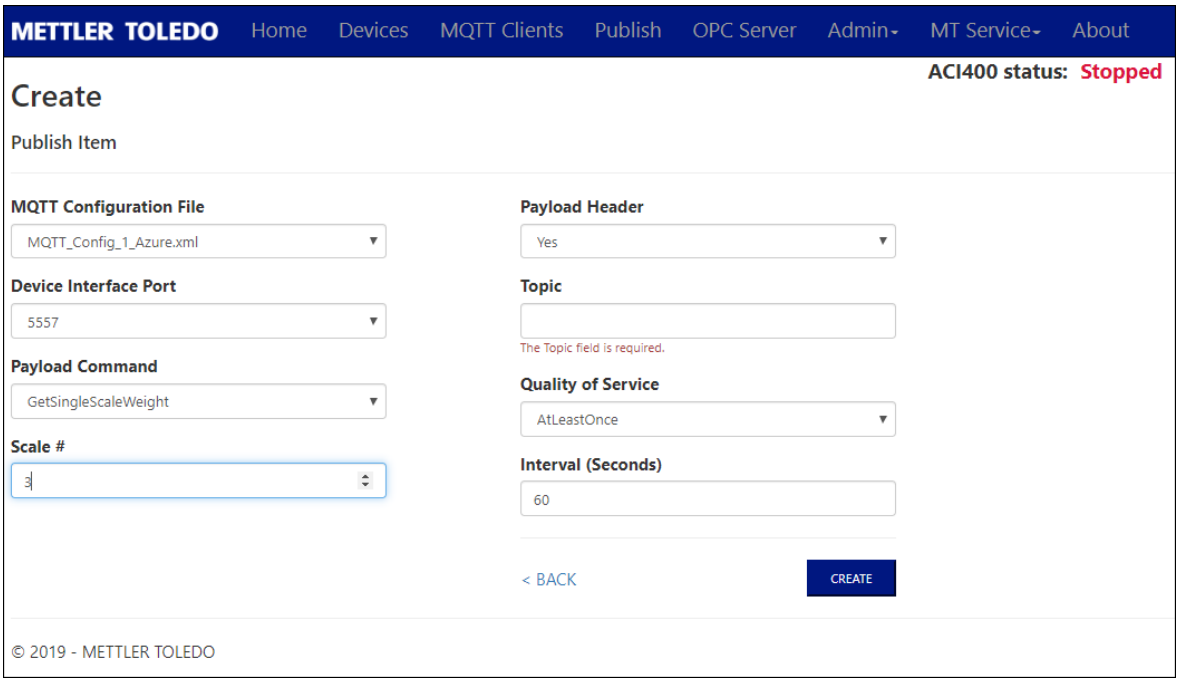

Figure 40: Scale # Selection

Payload Header

The **Payload Header** setting determines whether the published payload will include header information. YES enables the header content in the published payload. NO disables header content.

Payload Header information is configured under the MT Service tab (and protected by a unique security login). Payload Header data includes the following fields, and by design, is intended for use only by METTLER TOLEDO organizations.

- Division
- Strategic Business Unit
- SPG (Strategic Product Group)
- Message Version
- Ship To ID

Topic

The **Topic** field is required in order to publish a payload.

In MQTT, the word "topic" refers to a text string that the MQTT broker uses to filter messages received from each client connection. There is no standard for topic structure, but the following are general guidelines for topic structure:

- A topic can be single- or multi-level. Each topic level is separated by a forward dash. For example, mettler-toledo/IND570/blendroom1
- The forward slash alone is a valid topic
- A topic must contain at least one character
- The topic string permits empty spaces
- • Topics are case-sensitive. For example: mettlertoledo/IND570/blendroomA and MettlerToledo/ind570/blendroomA are two different topics

Quality of Service

There are three qualities of service for payload/message delivery:

Qualities of Service for Payload/Message Delivery

AtMostOnce (QoS Level 0)	This is the simplest, lowest-overhead method of sending a message. The ACI400 client simply publishes the message, and there is no acknowledgement by the broker. There is no guarantee of delivery. Published messages are delivered according to the best efforts of the underlying TCP/IP network and message loss or duplication can occur. This level could be used where it does not matter if an individual reading is lost as the next one will be published soon after.
AtLeastOnce (QoS Level 1)	This method guarantees that a message is transferred successfully to the broker, but duplicates may occur. In this service level, the ACI400 client sends a message and expects the broker will acknowledge receipt of the message. If the broker does not acknowledge receipt, the ACI400 MQTT client will resend the message multiple times until it receives acknowledgement from the broker. There will be no indication that it is a repeat of the original message, so it is possible for a message to reach the broker more than once.
ExactlyOnce (QoS Level 2)	This is the highest level of service in MQTT in which there is a sequence of four messages between the sender (ACI400 client) and the receiver (broker), a kind of handshake to confirm that a specific message has been sent and that acknowledgement has been received. Both ends of this transaction are assured that the message was sent/received exactly once. This level is typically used in instances where duplicate or lost messages would lead to critical data loss.

Interval (Seconds)

Interval is the frequency (in seconds) at which the client publishes a message with the selected payload.

3.3.2 Saving Changes

Once **Publish Item** setup is complete, click SAVE, then click <BACK to return to the Publish Configuration screen. To edit an existing Publish Item, select Edit. To remove an existing Publish Item, select **Delete**.

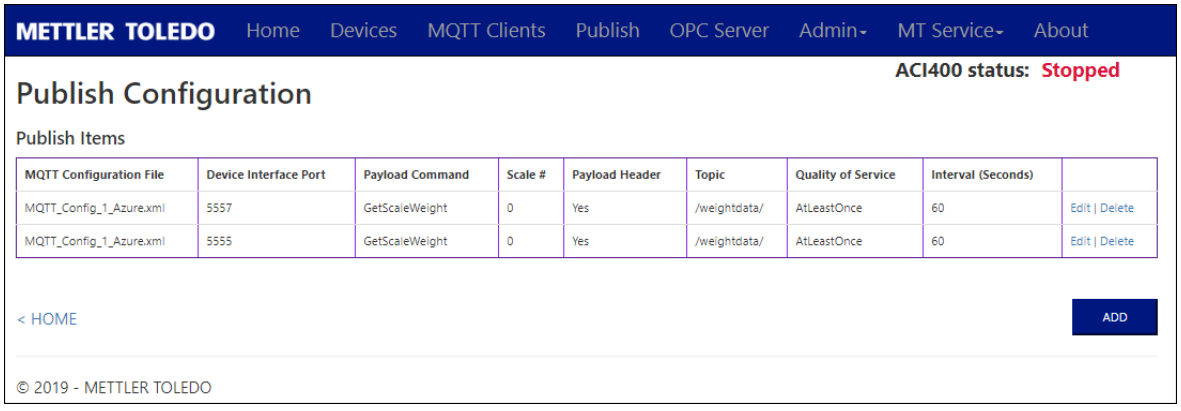

Figure 41: Publish Items List

3.4 Starting MQTT Communication Services

Once all required configuration for Devices, MQTT Clients, and Publish tabs is complete, return to the **Home** tab to start all MQTT related communication services.

10-15 seconds after communication services are started, the status of ACI400-to-Device and ACI400 MQTT Client-to-MQTT Broker communications can be viewed in the webserver at **Admin > Diagnostics**.

NOTICE

Note that ACI400 status remains Running while the user is on the Diagnostic screen.

Details on the Diagnostic information can be found in Appendix B, [Appendix B: Admin Functions and Diagnostics ▸ Page 86].

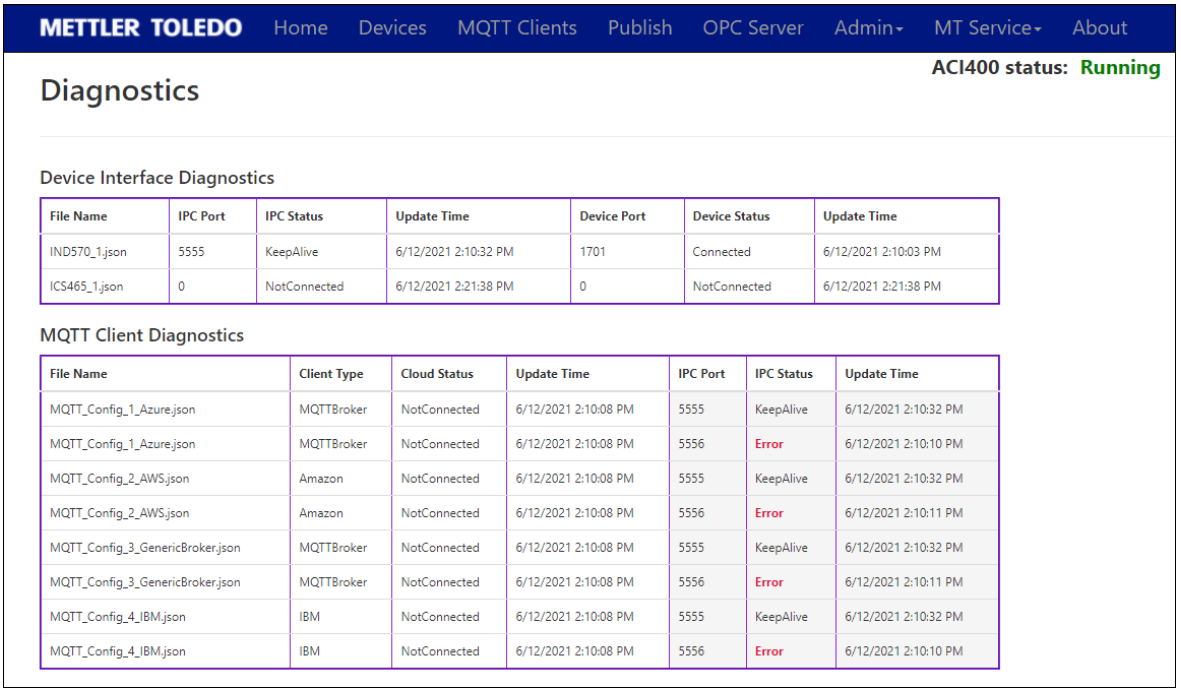

Figure 42: Communication Services Diagnostics

4 OPC UA Communication Configuration

To enable communication via the OPC UA Server in ACI400, navigate to the following web browser tabs in the order listed:

- Devices
- OPC Server

See also

OPC UA Communication Configuration ▸ Page 37

4.1 Devices

Configuration options available through the **Devices** tab identify the weighing device(s) with which the ACI400 will communicate, and to select communication details such as protocol and connection types.

NOTICE

Refer to [Device Interface Protocols ▸ Page 46] **for guidance on recommended device interface configurations by connected device type and desired communication services.**

The default DeviceInterfaceConfig.xml file (shown in [Devices ▸ Page 37]) is provided as a reference for Device Interface File creation. This default file can either be edited or deleted.

To edit this default file, select the **EditName** or **EditFile** option to the far right of the Device Interface Startup table. To remove the default file (or any Device Interface File) select the Delete option to the right of the file name ([Devices ▸ Page 37]).

Follow the steps below to create and set up a new Device Interface File:

1 Click the ADD button on the Launch Devices screen to begin setup of a new device.

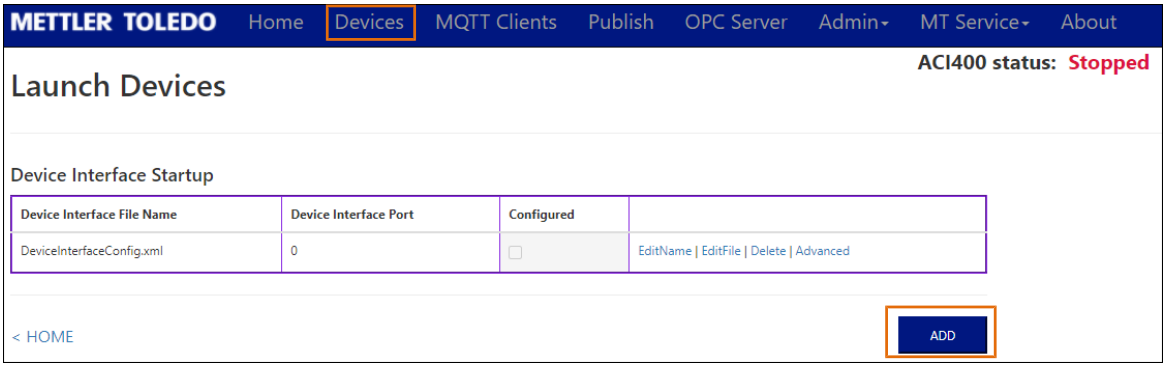

Figure 43: Device Interface File - Add

2 Type in a descriptive name for the Device Interface File Name - for example, IND570_1 - and click CREATE. The browser will return to the Laucnh Devices screen

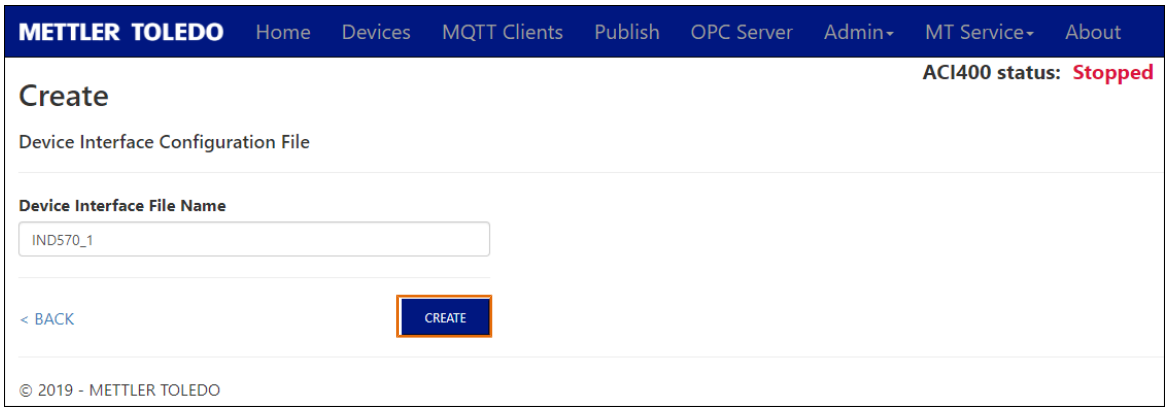

Figure 44: Device Interface File - Create

3 After creating the Device Interface File Name, select the EditFile option to the far right of the file name. The Device Interface Configuration parameters page will display. Descriptions of all Device Interface Configuration settings start at [Device Interface Port ▸ Page 38]

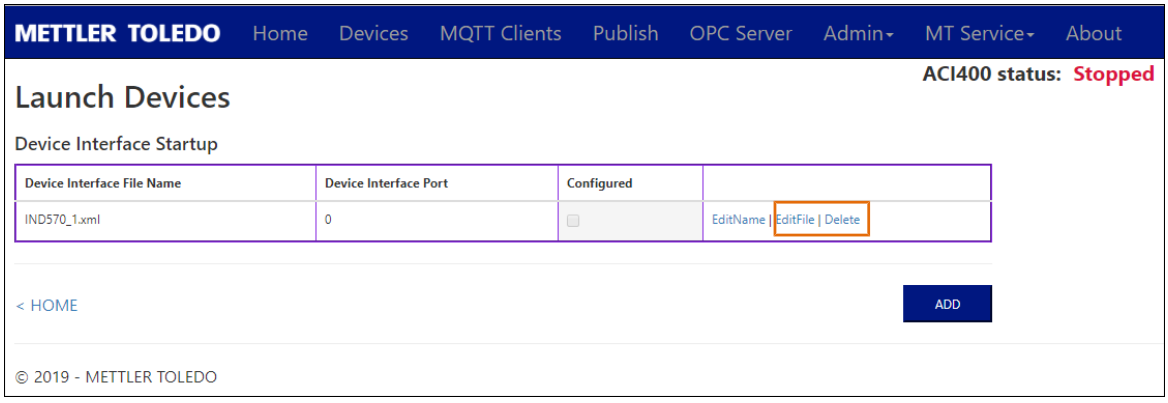

Figure 45: Device Interface File Configuration - Edit/Delete File Options

NOTICE

The Configured checkbox visible in [Devices ▸ Page 38] **is a visual aid only and not selectable. Once a Device Interface File is created, it must be configured to be functional. The Configured checkbox will be greyed out until all required parameters are set in the Device Interface File.**

See also

Device Interface Port ▸ Page 38

4.1.1 Device Interface Port

ACI400 uses the **Device Interface Port** to identify and route communication to the correct connected device.

The Device Interface Port number must be unique among **all** connected devices. The Device Interface Port number must also be outside the range of well-known restricted port numbers used by common networking features and software applications. Restricted port numbers generally fall between 0 and 1024. METTLER TOLEDO suggests a safe range for Device Interface Port settings between 5555 and 65535.

5555 is the default Device Interface Port number.

NOTICE

Port number 4840 should not be used as it is the default TCP/IP port for communication with the ACI400 OPC UA server.

The Device Port Number is used not only in communication between ACI400 and a connected device, but also in the [OPC UA server output ▸ Page 41] and [MQTT Payloads ▸ Page 48]. METTLER TOLEDO recommends that unique Device Interface Port numbers be assigned to **all** devices connected to **all** ACI400s in use on a network.

4.1.2 Device Connection Type

The **Device Connection Type** determines the physical connection that the ACI400 will use to communicate with the weighing device. There are two options for Device Connection Type: **Ethernet [default]** or **Serial**.

Select the connection type to use. The screen will update with a set of properties appropriate for the selected connection type.

Device Connection Type - Ethernet

When **Device Connection Type** is **Ethernet**, the following parameters are required:

- Device Protocol
- Device IP Address
- Device Port

4.1.3 Device ConnectionType - Ethernet

When **Device Connection Type** is **Ethernet**, the following parameters are required:

- Device Protocol
- Device IP Address
- Device Port

Device Protocol

The **Device Protocol** is the communication protocol the ACI400 will use to communicate with the weighing device. Options are **SDS** (Shared Data Server) [default], **SICS** (METTLER TOLEDO Standard Interface Command Set), **Print**, and **RPA455**.

Refer to [Device Interface Protocols ▸ Page 46], for additional details on the available Device Protocols.

Device IP Address

Device IP Address is the IP address of the weighing device with which the ACI400 will communicate. The default setting is **192.168.0.1**.

Enter the IP address for the connected weighing device. The ACI400 and weighing device must have similar network interface settings (IP address, subnet) in order to communicate.

Network settings for the ACI400 can be modified under the **Admin** tab on the Network Settings page.

Device Port

The **Device Port** is the TCP/IP port the connected weighing device uses to communicate with the ACI400. Enter the appropriate TCP/IP port number for the connected device.

When **SDS** (Shared Data Server) is selected as the **Device Protocol over Ethernet**, the **Device Port** must always be 1701.

When **SICS** is selected as the **Device Protocol over Ethernet**, the port number will vary. For example, an ICS terminal uses port 4305 as the default Ethernet communication port.

Refer to the documentation of the connected device for details on the correct port setting.

Saving Changes

After entering the required Ethernet settings, click SAVE, then click <BACK to return to the Launch Devices screen.

4.1.4 Device Connection Type - Serial

When Device Connection Type is **Serial**, the following parameters are required:

- Device Protocol
- Device Serial Port
- Device Serial Baud
- Device Serial Parity

Device Protocol

The **Device Protocol** is the communication protocol the ACI400 will use to communicate with the weighing device. When Device Connection Type is Serial, the following Device Protocols are functional:

- **SICS** (METTLER TOLEDO Standard Interface Command Set)
- Print

NOTICE

SDS (Shared Data Server) and RPA455 are shown in the drop-down list, but ACI400 cannot connect to a device using SDS or RPA455 over a Serial connection.

Device Serial Port

The **Device Serial Port** setting identifies the serial/COM port on the connected weighing device used to communicate with the ACI400. Enter the appropriate serial port ID.

Device Serial Baud

Use the **Device Serial Baud** setting to enter the baud rate of the serial port identified in the Device Serial Port field.

Options are **B9600** [default], B19200, B38400, B57600 and B115200. When using SICS as the Device Protocol, recommended settings are B9600 or B19200.

Device Serial Parity

Enter the parity setting of the serial port identified in the **Device Serial Parity** field.

Options are **None** [default], Odd, Even, Mark and Space.

4.1.5 Saving Changes

Once the required serial settings have been entered, click SAVE, then click <BACK to return to the **Launch Devices** screen.

4.2 OPC Server

Once at least one device has been configured, navigate to the **OPC Server** tab.

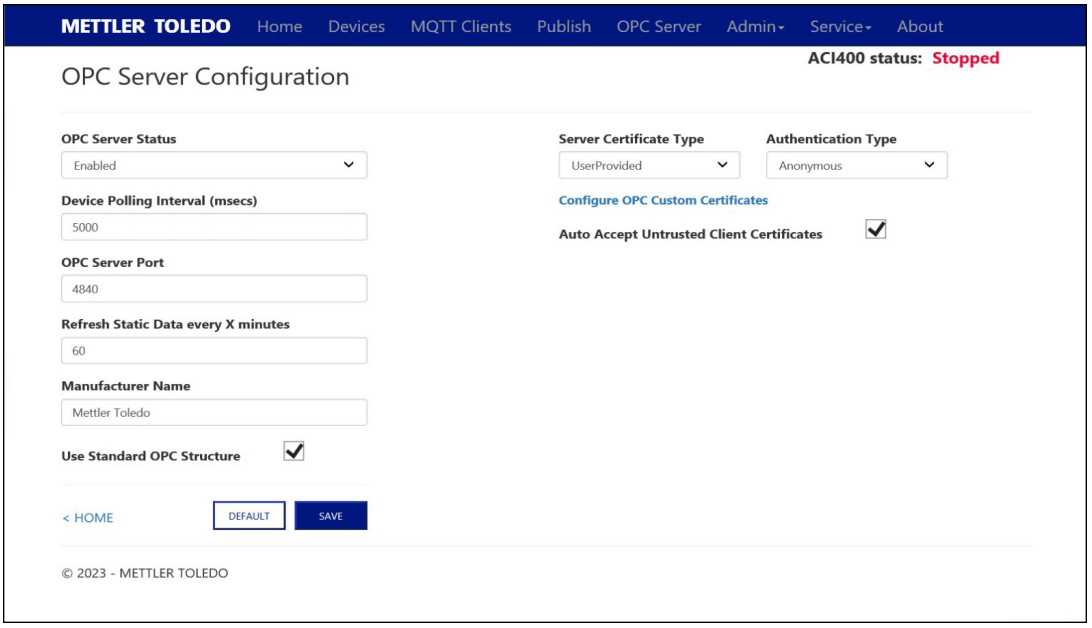

Figure 46: OPC Server Configuration

OPC Server Status

The **OPC Server Status** parameter turns the OPC UA server ON and OFF.

Set OPC Server Status to **Enabled** to run the ACI400 OPC UA server.

Device Polling Interval

In order to optimize the update of data from all connected devices, some OPC UA data objects have higher update priority than others. The data with higher priority are dynamic data objects that change frequently and sometimes rapidly during standard device operation.

The **Device Polling Interval** determines the frequency (in milliseconds) that the OPC UA Server:

- Requests higher priority data from connected weighing devices
- Refreshes the output of higher priority data to an OPC UA client

Do not set the Device Polling Interval lower than 500 milliseconds. 2000-5000 milliseconds is the preferred range for the Device Polling Interval.

NOTICE

RECOMMENDATION: The true device refresh rate is reported on the Diagnostic screen (navigate to Admin > Diagnostics, or refer to Appendix B, [OPC UA Server Diagnostics Field Definitions ▸ Page 95]**). In the OPC Server Diagnostics block, confirm the Average Fast Refresh time. This is the maximum possible refresh rate of a connected device. A safe setting for the Device Polling Interval is the Average Fast Refresh Rate + 500 to 1000 milliseconds.**

OPC Server Port

The TCP/IP Port # on the ACI400 dedicated to communication with the OPC UA server. The default port is **4840**. This setting can be changed to meet the needs of the user's OPC UA client and network setup.

Refresh Static Data Every X Minutes

The **Refresh Static Data** setting determines the frequency at which the ACI400:

- Requests lower priority data from connected weighing devices
- Refreshes the output of lower priority data to an OPC UA client

Manufacturer Name

Manufacturer ID that will appear in the OPC UA server data object 'Manufacturer.' **Mettler Toledo** is the default entry.

User Standard OPC Structure

This checkbox changes the OPC Structure from the original one used in the ACI400 to a version modified to comply with the latest OPC Foundation structures. Please note that the two structures use different node indexes:

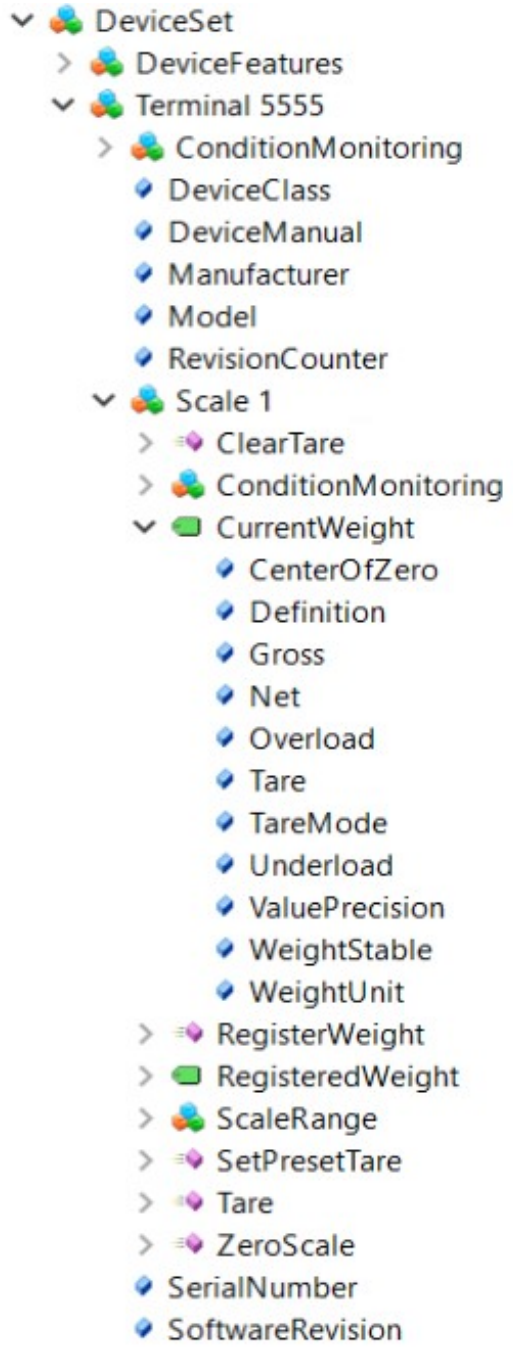

Figure 47: Original OPC Structure

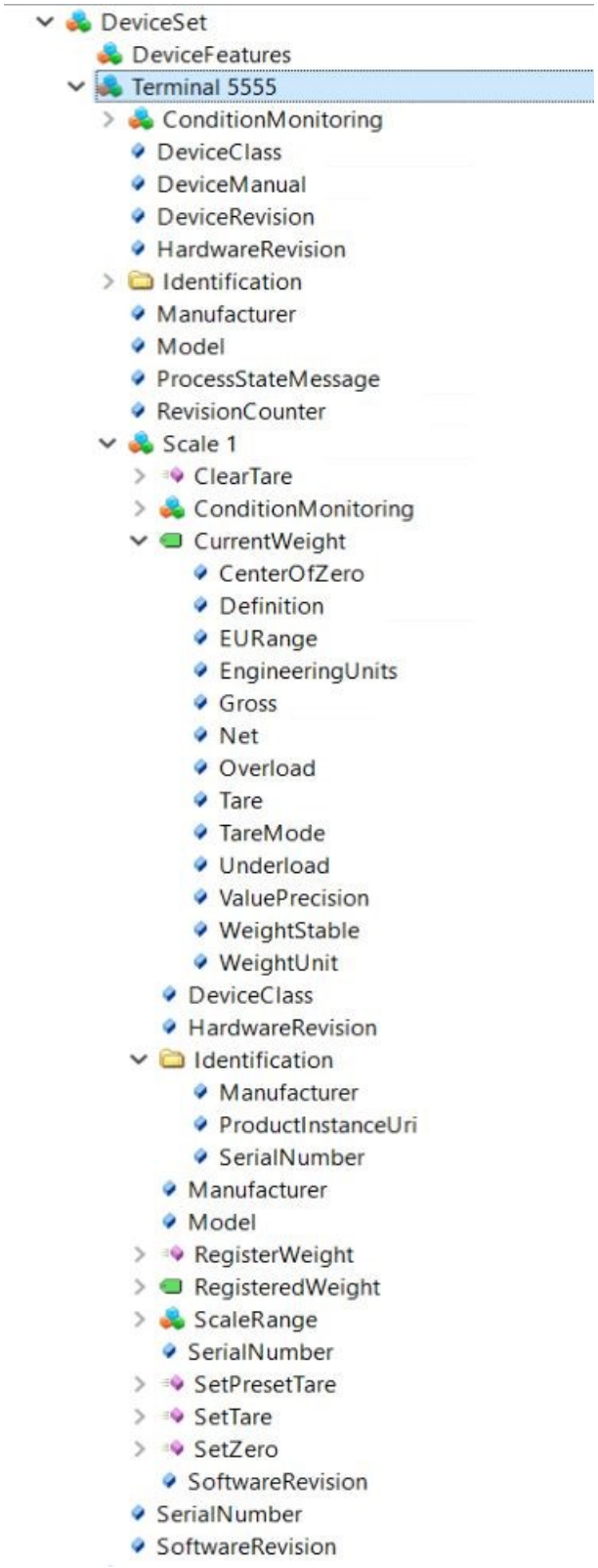

Figure 48: New Standard OPC Structure

Saving Changes

After entering all required OPC Server Configuration settings, click SAVE, then click <HOME to return to the Home page.

Default

Clicking on the **Default** button will return all **OPC Server Configuration** settings to the factory default settings.

OPC Server Certificate Type

Options are **UserProvided** and **Generated**.

User Provided -- enables on the ability to add server and client certifications to the ACI400.

Generated -- use the self-signed certificates auotmatically generated by the system.

Authentication Type

Options are **Anonymous**, **UsernamePassword**, and **Certificate**.

Anonymous authentication does not require a user name, password or certificate to connecto the server.

UsernamePassword authentication uses a name and password challenge.

Certification authentication relies on the certificate to confirm identity.

Configure OPC Custom Certificates

Check this box to display the **OPC Server Custom Certificates** page:

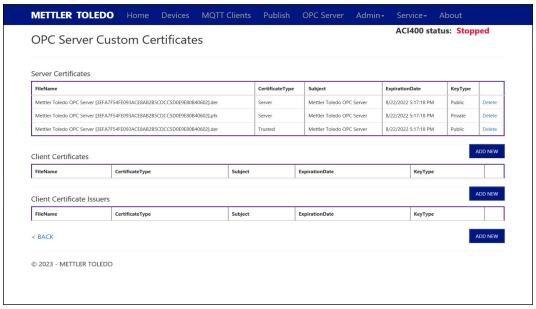

Figure 49: OPC Server Custom Certificates

Server Certificates allow the encryption of a connection.

Client Certificate allows the ACI400 OPC UA server to recognize a client in order to authenticate using certificate authentication instead of a user name and password.

Client Certificate Issuers provides the Root certificate indicating who generated the client certificate.

For each type, an ADD NEW button allows the addition of a new certificate or issuer.

See also

OPC UA Server Diagnostics Field Definitions ▸ Page 95

4.3 Starting OPC UA Services

After completing all required configuration in the Devices and OPC Server tabs, return to the Home tab to start all OPC UA-related communication services.

10 to 15 seconds after communication services are started, the status of the ACI400-to-Device and ACI400 OPC UA server communications can be viewed in the web interface at **Admin > Diagnostics**.

Diagnostic information details can be found in [Appendix B: Admin Functions and Diagnostics ▸ Page 86].

NOTICE

Note that ACI400 status remains Running while the user is on the Diagnostic screen.

Figure 50: Communication Services Diagnostics

5 Device Interface Protocols

5.1 SICS

The Standard Interface Command Set is present in many METTLER TOLEDO weighing devices and even some competitive devices. However, not every device that supports SICS supports all of the required SICS commands for successful communication with the ACI400. In order for a weighing device to communicate successfully and completely with the ACI400 TEXT, a weighing device must support the following SICS commands.

Required Terminal Support for SICS Commands

Command	Provides	Note
I2	Model information	
I3	Firmware information	
I4	Serial number	
I6	Range, division information	If I6 command is unavailable, full scale setup data (e.g. ranges, divisions) may be missing from output.
SI	Scale unit, Net weight, Stability, Overload/Underload	
SXI	Gross, Net, Tare, Stability, Overload/Underload, Invalid status	SI and TA commands are valid alternatives for some SXI data.
SIX1	Center of Zero	If SIX1 command is not available, Center of Zero status is not available.

5.2 SDS (Shared Data Server)

When SDS is selected, the ACI400 will use a predefined set of shared data server read/write commands to communicate to a connected device. The SDS device protocol is only available for use with the following METTLER TOLEDO products:

- IND780 and IND780xx indicators
- IND570 and IND570xx indicators
- IND560 and IND560xx indicators
- IND560x indicators

SDS - Read/Write Function

The ACI400 does not support on-demand read-write of Shared Data variables.

5.3 Print

NOTICE

Device Connection Type Print only works in combination with MQTT communication. Device Connection Type Print is NOT compatible with OPC UA server output.

To be compatible with the ACI400 Print protocol, a connected Device MUST include an ETX/CR/LF sequence at the end of the ASCII output. This control character sequence is required for the ACI400 to capture, package and transmit the ASCII output from the connected Device.

Device Connection Type - Print

Device Connection Type Print allows communication between the ACI400 and a connected weighing device through a simple ASCII Print, Transfer or Demand Output type connection.

When the ACI400 receives a transmission from a connected weighing device via the Print protocol, the ACI400 packages the received data into a JSON message "as received". No data parsing is available. The created JSON message is then available for transmission as a payload from an ACI400 MQTT client to an MQTT broker.

The following code shows a basic example of the JSON conversion of a simple ASCII Print output from a connected IND570 indicator.

```
{
"PayloadType" : "PrintEventData",
"PayloadVersion" : 1,
"PrintData" : " 147.0 kg \r\n\0.0 kg T \r\n 147.0 kg N\r\n\r\n",
"TimeStamp" : "2021-10-07T10:51:50.2653155-04:00",
"DataSource: 3,
PayloadError : 0
}
```

5.4 RPA455

The RPA455 protocol is only used when connecting an RPA455 scale to the ACI400. Separate documentation on the RPA455 protocol is supplied on an as-needed basis to users of the RPA455.

6 MQTT Payload Structures and General Commands

6.1 Available Payloads

ACI400 IIoT Edge device publishes MQTT messages in JSON format.

The following payloads are available for publishing by all ACI400 MQTT Clients. The payload structures provided in the following sections are shown in JSON format and do not include available header information. Refer to [Payload Headers ▸ Page 56] for details on available header structures.

Some of the payload examples do contain or the Optional Data Type (Customer Configuration) fields. Refer to [MQTT Communication Configuration ▸ Page 18] for information on the Optional Data Type fields.

NOTICE

Payload data fields and JSON formatting are shown in black text. Example data for each data field is shown in blue text. Payload data examples are based on a simulated ACI400 connection to an IND570 indicator using SDS-Shared Data Server (Device Protocol setting) over Ethernet (Device Connection Type setting).

6.1.1 Device Configuration Payload

At the startup of ACI400 communication services, the **GetDeviceConfig** payload is automatically requested and published once by each created MQTT Client for each Device Interface linked to the client.

The **GetDeviceConfig** payload can also be selected for scheduled publication through the Publish Item configuration (refer to [Create Publish Item ▸ Page 31])


```
{
"PayloadType" : "DeviceConfiguration",
"PayloadVersion" : 1,
"ModelName" : "IND570",
"FirmwareID" : "2.01.0008",
"SerialNumber" : "B916545478",
"LegalForTrade" : false,
"NumberOfScales" : 1,
"ScaleInfo" : [ {
"ScaleType" : "Analog",
"ScaleNumber" : 1,
"IncludedInSum" : false,
"ScaleCapacity" : "500.000000",
"ScaleIncrement" : "0.010000",
"ScaleUnit" : "kg",
"PrimaryUnit" : "kg",
"NumberOfRanges" : "2",
"LowIncSize" : "0.010000",
"MidIncSize" : "0.010000",
"HighIncSize" : "0.500000",
"NumberOfLoadcells" : "0",
"LoadcellInfo" : [ ],
"Id" : 0,
"HasScales" : true,
"HasScaleType" : true,
"TimeStamp" : "2021-06-12T18:27:30.8968701-04:00",
"DataSource" : 0,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
} ],
"Id" : 0,
"HasScales" : true,
"HasScaleType" : false,
```

```
"TimeStamp" : "2021-06-12T18:27:31.7062296-04:00",
"DataSource" : 1,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false },
"CustomerConfigData" : {
"Location" : "Worthington-OH",
"CustomerID" : "MTWT",
"DeviceID" : [ {
"Port" : 5570,
"ID" : "-"
} ],
"TimeZone" : {
"TimeZoneName" : "(UTC-05:00) Eastern Time (US & Canada)",
"DaylightSavingsSupported" : true,
"DaylightSavingsActive" : true
}
}
}
```


See also

Publish ▸ Page 31

6.1.2 Single Scale Weight Payload

The **GetSingleScaleWeight** payload can be used for collecting a single scale's data from both single scale and multi-scale devices. In the case of the multi-scale device like the IND780 terminal, scale data can be requested from any one of the four main scales plus the sum scale.

This payload can be selected for scheduled publication through the Publish Item configuration (refer to [Create Publish Item ▸ Page 31]).


```
{
"PayloadType" : "SingleScaleWeight",
"PayloadVersion" : 1,
"SerialNumber" : "B916545478",
"NumberOfScales" : 1,
"ScaleInfo" :
[ {
"ScaleNumber" : 1,
"Tare" : 15.0,
"Net" : 132.0,
"Gross" : 147.0,
"Units" : "kg",
"OverCapacity" : false,
"UnderZero" : false,
"CenterOfZero" : false,
"Motion" : false,
"WeightDataOk" : true,
"NetMode" : true
} ],
"Id" : 0,
"HasScales" : false,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:27:37.4101221-04:00",
"DataSource" : 1,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
}
```


See also

Publish ▸ Page 31

6.1.3 Scale Weight Payload

The **GetScaleWeight** payload can be used to collect scale data from all scales available on a connected device. This payload provides both double float and string formatted weight data for each scale.

This payload can be selected for scheduled publication through the Publish Item configuration (refer to [Create Publish Item ▸ Page 31]).


```
{
"PayloadType" : "ScaleWeight",
"PayloadVersion" : 1,
"SerialNumber" : "B916545478",
"LegalForTrade" : false,
"NumberOfScales" : 1,
"ScaleInfo" : [ {
"ScaleNumber" : 1,
"IncludedInSum" : false,
"Tare" : 15.0,
"TareString" : "15.0",
"Net" : 132.0,
"NetString" : "132.0",
"Gross" : 147.0,
"GrossString" : "147.0",
"OverCapacity" : false,
"UnderZero" : false,
"CenterOfZero" : false,
"Motion" : false,
"CurrentZeroCounts" : 431.0,
"WeightDataOk" : true,
"SelectedScale" : false,
"NetMode" : true,
"Units" : "kg"
} ],
"Id" : 0,
"HasScales" : true,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:27:42.3032325-04:00",
"DataSource" : 1,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false, "CustomerConfigData" : {
"Location" : "Worthington-OH",
"CustomerID" : "MTWT",
"DeviceID" : [ {
"Port" : 5555,
"ID" : "IND570-B916545478"
} ],
"TimeZone" : {
"TimeZoneName" : "(UTC-05:00) Eastern Time (US & Canada)",
"DaylightSavingsSupported" : true,
"DaylightSavingsActive" : true
}
}
}
```


6.1.4 Scale String Weight Payload

The **GetScaleStrWeight** payload can be used to collect scale data from all scales available on a connected device. This payload provides only string formatted weight data for each scale.

This payload can be selected for scheduled publication through the Publish Item configuration (refer to [Create Publish Item ▸ Page 31]).


```
{
"PayloadType" : "ScaleStrWeight",
"PayloadVersion" : 1,
"SerialNumber" : "B916545478",
"LegalForTrade" : false,
```

```
"NumberOfScales" : 1,
"ScaleInfo" : [ {
"ScaleNumber" : 1,
"TareWeight" : "15.0",
"NetWeight" : "132.0",
"GrossWeight" : "147.0",
"Units" : "kg",
"OverCapacity" : false,
"UnderZero" : false,
"CenterOfZero" : false,
"Motion" : false,
"WeightDataOk" : true,
"NetMode" : true
} ],
"Id" : 0,
"HasScales" : true,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:28:19.7050574-04:00",
"DataSource" : 1,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
}
```


6.1.5 Gateway Configuration Payload

The **GetGatewayConfiguration** payload provides comprehensive ACI400 configuration data. This payload is automatically published by each configured MQTT Client at startup of the ACI400 communication services.

This payload can also be selected for scheduled publication through the Publish Item configuration (refer to [Create Publish Item ▸ Page 31]).


```
{
"PayloadType" : "GatewayConfiguration",
"PayloadVersion" : 1,
"DeviceName" : "Mettler Toledo ACI400",
"OperatingSystem" : "Microsoft Windows 10.0.17763",
"ApplicationSoftware" : "2021.04.16.3",
"IpAddress" : "192.168.0.100",
"NumberOfTerminals" : 2,
"Region" : "MT-MOUS",
"CustomerLocationNumber" : "400999991",
"DevInterfacePort" : 0,
"OpcObjectName" : "Gateway 0",
"Id" : 0,
"HasScales" : false,
"HasScaleType" : false,
"TimeStamp" : "2021-06-08T18:10:19.0339767-04:00",
"DataSource" : 0,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false,
"CustomerConfigData" : {
"Location" : "Worthington-OH",
"CustomerID" : "MTWT",
"DeviceID" : [ {
"Port" : 5555,
"ID" : "IND570-B916545478"
}, {
"Port" : 5570,
"ID" : "-"
} ],
"TimeZone" : {
"TimeZoneName" : "(UTC-05:00) Eastern Time (US & Canada)",
"DaylightSavingsSupported" : true,
"DaylightSavingsActive" : true
}
```

```
}
}
```

6.1.6 Terminal Data Item Payload

The **GetTerminalDataItem** payload is the main payload used to populate data in the ACI400 internal OPC UA Server.

This payload is also available for external publishing and can be selected for scheduled publication through the Publish Item configuration (refer to [Create Publish Item ▸ Page 31]).


```
{
"PayloadType" : "TerminalDataItem",
"PayloadVersion" : 1,
"NumberOfScales" : 1,
"Scales" : [ {
"CurrentWeight" : {
"CenterOfZero" : false,
"TareMode" : 0,
"WeightStable" : true,
"Overload" : false,
"Underload" : false,
"InsideZero" : false,
"Invalid" : false,
"Tare" : 15.0,
"Net" : 132.0,
"Gross" : 147.0,
"EngineeringUnits" : "kg"
},
"RegisteredWeight" : {
"WeightStable" : true,
"Overload" : false,
"Underload" : false,
"Tare" : 14.97,
"Net" : 0.0,
"Gross" : 14.97,
"EngineeringUnits" : "kg"
},
"ScaleRange" : {
"Range" : {
"Low" : 0.0,
"High" : 0.0
},
"ScaleDivision" : 0,
"DisplayedScaleDivision" : 0,
"RangeEngineeringUnit" : {
"NamespaceUri" : null,
"UnitId" : 2,
"DisplayName" : {
"Locale" : null,
"Text" : "kg"
},
"Description" : null
},
"EngineeringUnits" : {
"NamespaceUri" : null,
"UnitId" : 2,
"DisplayName" : {
"Locale" : null,
"Text" : "kg"
},
"Description" : null
}
},
"ConditionMonitoring" : {
"ConnectionOK" : 1,
"AlarmPresent" : 1,
```

```
"Smart5Alarm" : {
"Level" : "5",
"Id" : "2013",
"Description" : "Scale Network Failure",
"Action" : "Check Load Cell Cable",
"Timestamp" : "06,12,2021,06,27,51",
"Reference" : "B327714530"
}
},
"ScaleNumber" : 1,
"OpcObjectName" : "Scale 1",
"ClearTareCommand" : {
"PlCommand" : 1013,
"PayloadType" : "ClearTareScaleCommand",
"PayloadVersion" : 1,
"ClearTareStatus" : "",
"PassThroughData" : null,
"SdPrimaryCommand" : "wc0102",
"SicsSecondaryCommands" : [ ],
"SicsPrimaryCommand" : "TAC",
"ScaleNumber" : 1,
"CommandSuccess" : false,
"HasScales" : false,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:27:50.499972-04:00",
"DataSource" : 0,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
},
"TareCommand" : {
"PlCommand" : 1012,
"PayloadType" : "TareScaleCommand",
"PayloadVersion" : 1,
"TareStatus" : "",
"PassThroughData" : null,
"SdPrimaryCommand" : "wc0101",
"SicsSecondaryCommands" : [ "T" ],
"SicsPrimaryCommand" : "TI",
"ScaleNumber" : 1,
"CommandSuccess" : false,
"HasScales" : false,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:27:50.4999812-04:00",
"DataSource" : 0,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
},
"ZeroScaleCommand" : {
"PlCommand" : 1014,
"PayloadType" : "ZeroScaleCommand",
"PayloadVersion" : 1,
"ZeroStatus" : "",
"PassThroughData" : "",
"SdPrimaryCommand" : "wc0104",
"SicsSecondaryCommands" : [ "Z" ],
"SicsPrimaryCommand" : "ZI",
"ScaleNumber" : 1,
"CommandSuccess" : false,
"HasScales" : false,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:27:50.4999931-04:00",
"DataSource" : 0,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
```

```
},
"SetPresetTareCommand" : {
"PlCommand" : 1015,
"PayloadType" : "SetPresetTareCommand",
"PayloadVersion" : 1,
"SetPresetTareStatus" : "",
"TareWeight" : 0.0,
"TareUnit" : null,
"SdPrimaryCommand" : "wk0104 0",
"SicsSecondaryCommands" : [ ],
"SicsPrimaryCommand" : "TA 0",
"ScaleNumber" : 1,
"CommandSuccess" : false,
"HasScales" : false,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:27:50.4999962-04:00",
"DataSource" : 0,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
},
"StoreCommand" : {
"PlCommand" : 1017,
"PayloadType" : "StoreCommand",
"PayloadVersion" : 1,
"Tare" : "",
"Net" : "",
"Gross" : "",
"Motion" : false,
"Units" : "",
"Overload" : false,
"Underload" : false,
"StoreCmdStatus" : "",
"SdPrimaryCommand" : "zz0199",
"SicsSecondaryCommands" : [ "zz0199" ],
"SicsPrimaryCommand" : "zz0199",
"ScaleNumber" : 1,
"CommandSuccess" : false,
"HasScales" : false,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:27:50.5000147-04:00",
"DataSource" : 0,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
}
} ],
"DevInterfacePort" : 0,
"OpcObjectName" : "Terminal 0",
"SerialNumber" : "B916545478",
"RevisionCounter" : 0,
"Manufacturer" : {
"Locale" : null,
"Text" : "Mettler Toledo"
},
"Model" : {
"Locale" : null,
"Text" : "IND570"
},
"DeviceManual" : "www.mt.com",
"SoftwareRevision" : "2.01.0008",
"DeviceClass" : "Weighing Device",
"ConditionMonitoring" : {
"ConnectionOK" : 1,
"AlarmPresent" : 0,
"Smart5Alarm" : {
"Level" : "5",
```

```
"Id" : "2013",
"Description" : "Scale Netwwork Failure",
"Action" : "Check Load Cell Cable",
"Timestamp" : "06,12,2021,06,27,51",
"Reference" : "B327714530"
}
},
"DeviceHealth" : 5,
"Id" : 0,
"HasScales" : true,
"HasScaleType" : false,
"TimeStamp" : "2021-06-12T18:27:51.5040424-04:00",
"DataSource" : 1,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
}
```


6.1.7 Event Configuration Payload

At the startup of ACI400 communication services, the **EventConfiguration** payload is automatically requested and published one time by each active ACI400 MQTT Client.


```
{
"PayloadType" : "EventConfiguration",
"PayloadVersion" : 1,
"DeviceEventControl" : 1,
"PrintEventControl" : 1,
"Id" : 0,
"HasScales" : false,
"HasScaleType" : false,
"TimeStamp" : "2021-06-13T07:16:16.0874104-04:00",
"DataSource" : 3,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
}
```


6.1.8 Print Event Payload

The **PrintEventData** payload is forwarded by the ACI400 MQTT client when the client receives a Print trigger from the connected device. In the client setup, Print Event Control must be set to ForwardAllPrintOutput to allow the ACI400 client to publish the PrintEventData payload.


```
{
"PayloadType" : "PrintEventData",
"PayloadVersion" : 1,
"PrintData" : "147.0 kg \r\n 0.0 kg T \r\n 147.0 kg N\r\n\r\n",
"TimeStamp" : "2021-06-13T07:20:38.2653155-04:00",
"DataSource" : 3,
"PayloadError" : 0
}
```


PrintEventData Payload with Customer Configuration Data


```
{
"PayloadType" : "PrintEventData",
"PayloadVersion" : 1,
"PrintData" : "147.0 kg \r\n 0.0 kg T \r\n 147.0 kg N\r\n\r\n",
"TimeStamp" : "2021-06-13T07:20:38.2653155-04:00",
"DataSource" : 3,
"PayloadError" : 0
"CustomerConfigData" : {
"Location" : "Worthington-OH",
"CustomerID" : "MTWT",
"DeviceID" : [ {
"Port" : 5570,
```

```
"ID" : "-"
} ],
"TimeZone" : {
"TimeZoneName" : "(UTC-05:00) Eastern Time (US & Canada)",
"DaylightSavingsSupported" : true,
"DaylightSavingsActive" : true
}
}
}
```

6.2 Payload Headers

6.2.1 Headers

Payload headers can be enabled or disabled on a per payload basis when configuring a Publish Item (refer to [Create Publish Item ▶ Page 31]). If a header is enabled, the implemented header structure is controlled by the MQTT Client Type selected in the MQTT Configuration File (refer to [Create Publish Item ▶ Page 31]).

If the Payload Header setting is enabled (YES) and the client type is set as Amazon, then the Amazon header is used. All other MQTT client types will use the Standard METTLER TOLEDO header when the Payload Header setting is enabled.

6.2.2 METTLER TOLEDO Header

The configurable fields for the METTLER TOLEDO header are (Div, SBU, SPG, MessageVersion and ShipToId). The values for these fields are loaded from the MtHeaderConfig file which can be configured through the MT Header configuration page under the MT Service section of the Web Service. The remaining items (Timestamp, PayloadType and DeviceId) fields are dynamically loaded when the header is created.

METTLER TOLEDO Payload Header Fiels

TimeStamp	The current (Universal) time the payload header was created
PayloadType	DeviceConfig, SingleScaleWeight, ScaleWeight, ScaleStrWeight, GatewayConfiguration, TerminalDataItem
DeviceID	The BIOS serial number of the ACI400 Unit

METTLER TOLEDO Header Content

```
{
"RoutingHeader" : {
"Div" : "Division",
"SBU" : "SBU",
"SPG" : "SPG",
"MessageVersion" : "1.0",
"Timestamp" : "2019-10-24T22:05:39.1929054Z",
"PayloadType" : "SingleScaleWeight",
"DeviceId" : "BTDWNQ2",
"ShipToId" : "ID"
},
"Payload" :
…
(payload data – refer to [Available Payloads ▶ Page 48])
…
}
```

6.2.3 Amazon Header

The Amazon header is only used when the Client type is Amazon and the Header setting is enabled in the Publish Item configuration.

The Header setting must be Enabled with all payloads sent to the **Amazon Device Shadow** application. Without the header, the data will not be accepted. This is specific requirement for this Amazon application. Payloads published to other Amazon applications, may not have this requirement.

Amazon Header Content

```
{
"state": {
"reported":
…
(payload data – refer to [Available Payloads ▶ Page 48] for payload structures)
…
}
```

6.2.4 Sample Payload with METTLER TOLEDO Header


```
{
"RoutingHeader" : {
"Div" : "Division",
"SBU" : "SBU",
"SPG" : "SPG",
"MessageVersion" : "1.0",
"Timestamp" : "2019-10-24T22:05:39.1929054Z",
"PayloadType" : "SingleScaleWeight",
"DeviceId" : "BTDWNQ2",
"ShipToId" : "ID"
},
"Payload" : {
"PayloadType" : "SingleScaleWeight",
"SerialNumber" : "B747032814",
"NumberOfScales" : 1,
"ScaleInfo" : [ {
"ScaleNumber" : 1,
"Gross" : 39.31,
"Tare" : 0.0,
"Net" : 39.31,
"Units" : "kg",
"OverCapacity" : false,
"UnderZero" : false,
"CenterOfZero" : false,
"Motion" : false,
"WeightDataOk" : true,
"NetMode" : false
} ],
"Id" : 0,
"HasScales" : false,
"TimeStamp" : "2019-10-24T18:05:39.1750496-04:00",
"DataSource" : 1,
"PayloadError" : 0,
"IgnoreStaticData" : false,
"HasSumScale" : false
}
}
```


6.2.5 Sample Payload with Amazon Header


```
{
"state": {
"reported": {
"PayloadType": "SingleScaleWeight",
"SerialNumber": "B747032814",
"NumberOfScales": 1,
"ScaleInfo": [
{
"ScaleNumber": 1,
"Gross": 39.31,
"Tare": 0,
"Net": 39.31,
"Units": "kg",
"OverCapacity": false,
```

```
"UnderZero": false,
"CenterOfZero": false,
"Motion": false,
"WeightDataOk": true,
"NetMode": false
}
],
"Id": 0,
"HasScales": false,
"TimeStamp": "2019-10-24T16:36:50.8931835-04:00",
"DataSource": 1,
"PayloadError": 0,
"IgnoreStaticData": false,
"HasSumScale": false
}
}
}
```


6.3 Payload Request Command and Direct Device Commands

ACI400 MQTT clients are cable of responding to on-demand requests for payloads from brokers. The MQTT clients can also pass through some simple scale commands from the broker to the connected devices.

The Payload Request function permits all payload requests and device commands to be handled using a common topic and similar message structure.

6.3.1 Payload Request Structure

The payload request function is essentially a subscription command from the broker to the ACI400 client. It allows the broker to send an on-demand request for the ACI400 to publish any of the available non-event payloads. The Payload Request command consists of a topic followed by a JSON formatted message string.

NOTICE

The PrintEvent payload and AlertEvent payloads are not available through the Payload Request structure

Payload Request Topic Structure

The payload request topic uses the following consistent structure:

`MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest`

Payload Request Topic Elements

MettlerToledo	Case sensitive start of topic. Must be used as shown or ACI400 will not respond to payload requests or pass-through device commands.
MqttClientFileName	Should match the exact name of the MQTT Client Configuration File created under the MQTT Clients tab of the ACI400 websever. Refer to Chapter 3, section 3.2 for additional details.
Port Number	Port number of the connected device that will receive the payload request and provide the data for the payload. Refer to Chapter 3, section 3.1.1 for additional details.
GetPayloadRequest	Case sensitive end of topic. Must be used as shown or ACI400 will not respond to payload requests or pass-through device commands.

Payload Request Message Structure

The payload request message is a JSON string consisting of the payload Request Type, Request Data and Pass Through data fields. It uses the following consistent structures and syntax:

`{"RequestType":"DeviceConfiguration","RequestData":{}}`

Payload Request Message Elements

RequestType	The name of the requested payload.
RequestData	The portion of the string which contains any parameters required to complete the payload request. The Request Data can include any number of items (name/value pairs) and the items can be in any order. An example of Request Data is a preset Tare value that must be sent along with the SetPresetTareCommand request type. If there are no parameters required for execution of a specific Request Type, the RequestData portion of the JSON string should be blank.
PassThroughData	This portion of the string serves to pass-through information that the customer needs to have associated with the CustomPrint payload being returned. This optional data does not actually pass through the ACI400 to the connected device; rather, the ACI400 saves this data and appends it to the published CustomPrint payload.

6.3.2 Payload Request and Device Commands Without "Request Data" Parameters

Device Configuration Payload

Device Configuration Payload Topic and Message

Topic:	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message:	{"RequestType":"DeviceConfiguration","RequestData":{}}

Terminal Data Item

Terminal Data Item Topic and Message

Topic:	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message:	{"RequestType":"TerminalDataItem","RequestData":{}}

Scale Weight

Scale Weight Topic and Message

Topic:	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message:	{"RequestType":"ScaleWeight","RequestData":{}}

ScaleStrWeight

ScaleStrWeight Topic and Message

Topic:	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message:	{"RequestType":"ScaleStrWeight","RequestData":{}}

Gateway Configuration

Gateway Configuration Topic and Message

Topic:	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message:	{"RequestType":"GatewayConfiguration","RequestData":{}}

6.3.3 Payload Request and Device Commands With "Request Data" Parameters

SingleScaleWeight

SingleScaleWeight Topic and Message

Topic	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message	{"RequestType":"SingleScaleWeight","RequestData":{"ScaleNumber":"1"}}

ZeroScaleCommand

ZeroScaleCommand Topic and Message

Topic	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message	{"RequestType":"ZeroScaleCommand","RequestData":{"ScaleNumber":"1"}}

TareScaleCommand

TareScaleCommand Topic and Message

Topic	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message	{"RequestType":"TareScaleCommand","RequestData":{"ScaleNumber":"1"}}

ClearTareScaleCommand

ClearTareScaleCommand Topic and Message

Topic	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message	{"RequestType":"ClearTareScaleCommand","RequestData": {"ScaleNumber":"1"}}

SetPresetTareCommand

SetPresetTareCommand Topic and Message

Topic	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message	{"RequestType":"SetPresetTareCommand","RequestData": {"TareWeight":"9.25","TareUnit": "lb","ScaleNumber":"1"}}

StoreCommand

StoreCommand Topic and Message

Topic	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message	{"RequestType":"StoreCommand","RequestData":{"ScaleNumber":"1"}}

CustomPrintCommand

CustomPrintCommand Topic and Message

Topic	MettlerToledo/<MqttClientFileName>/<Port Number>/GetPayloadRequest
Message	{"RequestType":"CustomPrint","RequestData": {"PassThrough":"1234567890ABCDEFGHIJK LMNOPQRSTUVWXYZ","ScaleNumber":"1"}}

Publish Example of the CustomPrintCommand with PassThrough Data

Payload Request Topic:

```
MettlerToledo/MQTT_Config_1_Azure/5555/GetPayloadRequest
```

Payload Request Message

```
{"RequestType":"CustomPrint","RequestData":{"PassThrough":"ID":"A1004Gh6",
"Customer":"Acme Inc.", "Contract":"AI556", "TransactionType":"Append"}}
```

ACI400 client response:

```
{
"RequestType" : "ScaleWeight",
"RequestData" : {
"ID" : "A1004Gh6",
"Customer" : "XYZ Inc.",
"Contract" : "AI556",
"TransactionType" : "Append",
"ScaleNumber" : "1"
}
}
```

7 Service

The **Service** menu provides access to the **User Account Manager** page.

7.1 User Account Management

NOTICE

User name and password

It is recommended that the customer change the default user name and password. Make sure that the password is kept safe, apart from the system.

The User Account Manager page, under the Service+ menu, allows the default password to be changed, and additional users defined, each with their own access rights.

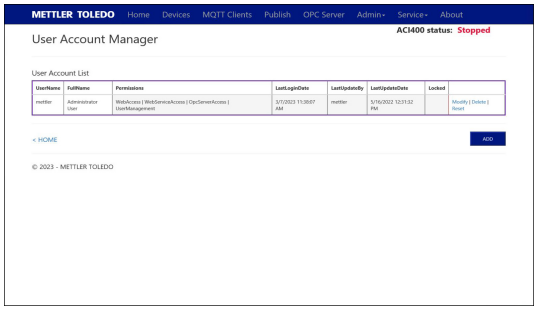

Figure 51: User Account Manager

Modify opens the **Modify User Account** page, where an existing account can be modified.

Delete deletes the selected account.

Reset is used to reset an account's password.

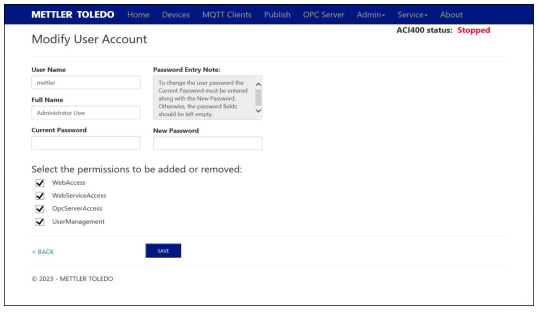

Figure 52: Modify User Account

The **User Name**, **Full Name**, and password fields are modified for each account.

Permissions are set for each account, and include **WebAccess** (the most basic form of access), **WebServiceAccess**, **OpcServerAccess** and **UserManagement**.

Web Access: Grants access to the ACI400 web server information, such as device settings, MQTT configuration, and Publishing.

Web Service Access: Grants access to the service menu, including this user account manager.

OPC Server Access: Grants access to the OPC menu, for OPC UA server configuration.

User Management: Grants access to the User Account Manager page under the service access.

Password Requirements

Passwords must meet the following minimum requirements, and include **at least**:

- 8 characters
- 1 lower case and 1 upper case character
- 1 number

- 1 special character

8 MQTT Data Dictionary

MQTT Data Dictionary

Name	Example	Description
TimeStamp	2021-09-29T11:59:31.6783348-04:00	Local time with offset to UTC. Example: for Eastern Standard time -5:00 is used as the difference between local time and UTC. For Eastern Daylight Saving Time -4:00 is used
Timestamp	2021-09-29T15:59:31.7799488Z	Universal Time (Zulu time), used in Routing Header
Timestamp	09,29,2021,12,01,12	Part of Smart5Alarm tag
RoutingHeader	"RoutingHeader": { "Div": "Division", "SBU": "SBU", "SPG": "SPG", "MessageVersion": "1.0", "Timestamp": "2021-09-29T16:19:34.5749563Z", "PayloadType": "SingleScaleWeight", "DeviceId": "U815933", "ShipToId": "ID" },	MT Defined Information including Timestamp in universal (Zulu) time, Payload type, DeviceID
Message Version	1.0	Version information for the JSON communications

Name	Example	Description
PayloadType	SingleScaleWeight	Defines what information is being sent. Defined Payload Types are: **DeviceConfiguration** - information on the devices connected to the gateway **TerminalDataItem** - includes the information on terminal and connected scales **GatewayCongifuration** - information on the gateway **ScaleStatistics** - information on the individual scales, statistics and calibration information **ScaleWeight** - weight and status of all connected scales for a terminal **ScaleStrWeigh** - data from ScaleWeight with weight values expressed in character strings **SingleScaleWeight** - data from ScaleWeight for an individual scale for terminals supporting more than one scale **CustomPrint** - print data coming from the terminal packaged within the JSON message **ZeroScale** - response to ZeroScale command, set scale to zero **TareScale** - response to TareScale command, tare weight on the scale **ClearTare** - response to ClearTare command, clear the current tare value **SetPresetTare** - response to SetPresetTare command, send a tare value to the scale and set that value as the current tare **StoreCommand** - response to the StoreCommand, which captures stable weight as "registered weight"
PayloadVersion	1	Version information for the JSON Payload structure
ModelName	IND570	Model Name of connected device such as a terminal
FirmwareID	2.01.0002	software version of the terminal
Serial Number	B852003633	Serial Number of connected device
LegalForTrade	false	Is device legal for trade (metrologically approved)? true = enabled for legal for trade false = not enabld for legal for trade
NumberOfScales	1	Number of scales attached to the device such as a terminal. Some terminals support more than one scale

Name	Example	Description

ScaleInfo	"ScaleInfo": [
	{
	"ScaleType": "Analog",
	"ScaleNumber": 1,
	"IncludedInSum": false,
	"ScaleCapacity": "50.000000",
	"ScaleIncrement": "1.000000",
	"ScaleUnit": "kg",
	"PrimaryUnit": "kg",
	"NumberOfRanges": "1",
	"LowIncSize": "1.000000",
	"MidIncSize": "0.010000",
	"HighIncSize": "0.010000",
	"NumberOfLoadcells": "0",
	"LoadcellInfo": [],
	"Id": 0,
	"HasScales": true,
	"HasScaleType": true,
	"TimeStamp":
	"2021-09-29T11:59:31.6783
	348-04:00",
	"DataSource": 1,
	"PayloadError": 0,
	"IgnoreStaticData": false,
	"HasSumScale": false
	}
	],
		Array of data about each connected scale. The ScaleInfo can change depending on the message type.

ScaleType	Analog	Only valid when terminal supports Shared Data Variables. Not valid with SICS connections. Uses shared data csXX01 command to return the following values 65 = Analog 68 = DigiNet 73 = IDnet 78 = None 80 = MTX 82 = SICSpro 83 = SICS 84 = POWERCELL 85 = Summing
ScaleNumber	1	Which number is assigned to a scale in a terminal that supports multiple scales. Normally 1-4 with 5 being a Sum Scale
IncludedInSum	false	Is the weight value of this scale (when multiple scales are connected to a terminal) included in the SUM scale value of that terminal true = yes, weight value included in the Sum Scale value false = no, weight value is not included in the Sum Scale value

Name	Example	Description
ScaleCapacity	50	Capaciy in primary units of the scale. For example if 50 and the primary units are kg then the capcity of the scale is 50kg
ScaleIncrement	1	Weight increment of the scale in primary units. For example 1 and the primary units are kg then each scale increment is 1kg
ScaleUnit	kg	The currenly selected unit measurement for the scale. Can be different than primary unit when multiple units are enabled
PrimaryUnit	kg	The primary unit used when calibrating the scale
NumberofRanges	1	Number of weighing ranges configured for this scale. Normally it is 1 to 3 ranges
LowIncSize		Range 1 increment
MidIncSize		Range 2 increment
HighIncSize		Range 3 increment
NumberofLoadcells	4	Only valid when terminal supports Shared Data Variables. Not valid with SICS connections. Returns the number of load cells associated with that particular scale.
id	0	Reserved value, not yet defined
HasScales	true	Indicates if there are multiple scales attached to the terminal for terminals that support multiple scales true = yes, there are multiple scales connected to this terminal false = no, this terminal is a single scale terminal
DataSource	1	Indicates which MT protocol is being used to connect to the terminal or scale 1 = shared data 2 = SICS 3 = Print protocol 4 = RPA455 protocol

Name	Example	Description
PayloadError	0	Possible errors are: 0 = No errors 1 = Response error 2 = Not available 3 = Bad checksum 4 = Termination error 11 = Max users logged in 12 = Bad group number 13 = Field access denied 14 = Field access error 15 = Bad field name 16 = Not supported 17 = Too many callbacks 18 = Reg callback failure 19 = Bad syntax 20 = Bad number of fields 21 = Bad timeout 22 = No saved parameters 23 = Bad number of streams 24 = Bad Stream 25 = Callback field access error 26 = Denied 27 = Failed 99 = Unknown indicator error
IgnoreStaticData		This comes from an OPC UA setting and is used in the TerminalDataItem payload true = only send dynamic data. Static data is refreshed every X number of minutes based on a setting in the OPC Server Configuration false = send both static and dynamic data
HasSumScale	false	Does the terminal have a Sum Scale setup true = yes false = no
CurrentWeight	"CurrentWeight": { "CenterOfZero": false, "TareMode": 0, "WeightStable": true, "Overload": false, "Underload": false, "InsideZero": false, "Invalid": false, "Tare": 0.0, "Net": 439.0, "Gross": 439.0, "EngineeringUnits": "kg" },	Structure that provides information on the weight. It is part of a Scale structure with additional information about the scale

Name	Example	Description
CenterofZero	true	Is the scale at center of zero. Only used with terminals supporting shared data variables communications true = yes false = no
TareMode	0	Only used with terminals supporting SICS communications. Terminals with Shared Data communications do not support this value. "N" is None which has a value 0. "M" is Tare which has a value 1. "P" is PresetTare which has a value of 2.
NetMode	true	Is the scale in net mode or in gross weight mode true = scale in net mode false = scale not in net mode. Scale is in gross weight mode
WeightStable	true	Is the weight stable, in a no motion condition true = weight is stable (no motion) false = weight is not stable (motion condition)
Motion	false	Opposite of the weight stable variable. true = motion condition (weight is not stable) false = no motion (weight is stable)
Overload	false	Is the scale in an overload condition true = yes, in an overload condition false = no, not in an overload condition
Underload	false	Is the scale in an underload condition true = yes, in an underload condition false = no, not in an underload condition
InsideZero	false	No longer used. Current tag is "CenterofZero"
Invalid	false	If data shown in the current weight data is valid or not and appears in the Terminal-DataItem True - value is invalid False - value is valid
Tare	0.0	Value of the tare weight
TareString	"0.0"	String value of the tare weight
Net	439.0	Value of the net weight
NetString	"439"	String value of the net weight
Gross	439.1	Value of the gross weight
GrossString	"439.1"	String value of the gross weight
EngineeringUnits		OPC UA structure for weight unit
CurrentZeroCounts	10095	Cell counts used to set base of zero weight

Name	Example	Description
WeightDataOK		If data shown in the weight fields is valid. Only used with terminals supporting Shared Data communications True - values are valid False - values are invalid (examples: error on scale, overload, underload)
SelectedScale	false	Is this the selected scale on terminals supporting multiple scales. Only for terminals using shared data protocol. true = yes, this is the currently selected scale false = no, this is not the currently selected scale
LegalForTrade	false	Is the scale in a metrologically approved (legal for trade) mode true = yes, scale is in a legal for trade mode false = no, scale is not in a legal for trade mode
Registered Weight		Array of data for scale weight that is captured as a sample by a store command
Scale Range		Range switchover points from one range to the next
ScaleDivision		Number of divisions used by the calibrated scale
DisplayedScaleDivisions		The number of scale divisions displayed on the terminal
RangeEngineeringUnit		Data Structure with OPC UA structured unit definitions for the scale
ConditionMonitoring		Data Structure with condition monitoring information, timestamps and Smart5 structure
Smart5Alarm		Data Structure with Smart5 information
DeviceName	Mettler Toledo ACI400	Name of the ACI400 device
OperatingSystem	Microsoft Windows 10.0.1.17763	Operating system information for ACI400 device
ApplicationSoftware	2021.04.16.3	ACI400 application version
IpAddress	192.168.0.100	IP address of the ACI400 device
ComandSuccess	true	Status in reply from command sent to ACI400 to execute a command (e.g. Tare Scale, Zero Scale, etc.) true = command was successfully executed false = command was not successfully executed
PassThroughData		Part of the custom print data. Customer sends string to ACI400 along with print command and ACI400 print response also includes this PassThroughData

Name	Example	Description
StoreCmdStatus		Response depends on the terminal communication method Shared Data communication "0" = Zero completed successfully. "1" = Zero in progress. "2" = Scale in motion during zero. "3" = Illegal scale mode during zero. "4" = Scale out of zeroing range. "5" = IDNET zero command timeout. "6" = Pushbutton zero disabled. "7" = Command timeout error. "8" = Scale communications disabled. SICS communication - response depends on terminal type "S" = Zero setting performed under stable conditions. "D" = Zero setting performed under dynamic conditions. "I" = Command understood but not executable. "+" = Upper limit of zero setting range exceeded. "-" = Lower limit of zero setting range exceeded. "A" = Command executed successfully. "I" = Command understood but not executable. "+" = Upper limit of zero setting range exceeded. "-" = Lower limit of zero setting range exceeded.
CustomPrintData		Print data from terminal that is passed through the ACI400. Print template output
TransactionCounter	2	Number of transactions performed on that terminal
NumOfZeroSuccesses	3	Number of zero commands successfully executed on a scale
NumOfZeroFailures	2	Number of zero commands not successfully executed on a scale
LastCalibrationDate	1600-12-31T19:01:37.8074354-05:00	The date and time of the last calibration performed on a scale
NextCalibrationDAte	1600-12-31T19:02:41.3088-05:00	The date and time of the next calibration to be performed on a scale
LastCalibrationTest	12/31/1600 7:02:41 PM	The date and time of the last calibration test that was performed
TestIntervalDays	10	The number of days between tests.
TestIntervalWeighments	100	The number of weighments between tests

Name	Example	Description
Timestamp	2021-09-29T15:59:31.7799488Z	Universal Time (Zulu time), used in Routing Header

9 OPC UA Data Dictionary

The illustration below lists the nodes described in the **OPC UA Data Dictionary** table.

Figure 53: Original (left) and Standard (right) OPC UA Data Structure Tree

OPC UA Data Dictionary

Name	Type	Description	Read/Method
DeviceClass	String	Indicates for what purpose a certain Device is used.	Read
DeviceHealth	Int32	Device health status.	Read
DeviceManual	String	URL address for the user manual.	Read
Manufacturer	Localized Text	Name of the company that manufactured the Device.	Read
Model	Localized Text	Provides the model name of the Device.	Read
Revision Counter	Int32	Incremental counter indicating the number of times the Device has been modified.	Read
Scale 1		A scale terminal can control more than one scale. The parameter indicates which scale the values comes from. Data from additional scales can be found in the server under "Scale 2", "Scale 3" or "Scale 4" if they are present.	
Clear Tare		The tare value is set to zero or deleted.	Method

Name	Type	Description	Read/Method
CenterOfZero	Boolean	Indicates that the scale is in an unloaded state. The tolerance of zero is one quarter (1/4) of a measurement interval or division. For example, consider a 1000-kilogram floor scale with a measurement interval is 500 grams; the zero indication range is 0.000 +/-125g. Note that zero commands must not be used to tare the scale because in many weights-and-measures regulated applications, the zero range is limited and does not extend to the full range of the scale. Unlike tare, zero cannot be cleared; it must be reset or adjusted.	Read
Gross	Double	Gross defines the current gross value of weight as a 0:Double precision floating point number. It is the current value that is measured at the weight sensor at the current timestamp. This might be a fluctuating value.	Read
Invalid	Boolean	Data invalid.	Read
Net	Double	Net defines the net value of weight as a 0:Double precision floating point number. It is the current value that is measured at the weight sensor at the current timestamp. This might be a fluctuating value.	Read
Overload	Boolean	Indicates maximum allowed measured weight value is exceeded.	Read
Tare	Double	Tare defines the tare value of weight as a 0:Double precision floating point number. It is the current value that is measured at the weight sensor at the current timestamp. This might be a fluctuating value.	Read
TareMode	Int32	0 = No tare 1 = Tare value triggered via method call "Tare". 2 = Tare value pre-set via method call "SetPre-SetTareCommand".	Read
Underload	Boolean	Indicates measured weight value is below allowed value.	Read
WeightStable	Boolean	Indicates weight is stable and not in motion. The sensitivity can be adjusted in most weighing devices. It refers to gross, net and tare weight.	Read
WeightUnit	String	The weighing unit in which the scale is set. Eg. kg, g, mg.	Read
ScaleRange		Not in use.	
SetPreSetTare Command	Double	The value which should be set as tare value.	Method
Tare	Boolean	Triggers the tare operation. The current weight value is used as the tare.	Method
ZeroScale	Boolean	Sets zero point. It sets the gross value of the current weight value to zero.	Method
SerialNumber	String	Unique production number of device set by the manufacturer.	Read
SoftwareRe-vision	String	Revision number of the software and firmware of the device.	Read
DeviceClass	String	Indicates for what purpose a certain Device is used.	Read

9.1 Tips for Dealing with Data

What is the Current Weight Node?

Node" is the OPC UA term for a single consistent data point that may include more than one value. The Current Weight node manages three essential weight values: net, tare and gross. The gross weight is the sum of the net and the tare weights. However, the relationship between these three values bears the risk of errors resulting from time differences in the data capturing procedure in case the three values are not captured simultaneously. In other words, Current Weight provides a consistent data set.

The OPC UA Current Weight node can eliminate this potential error source by transferring all three values as one node, but the user must open the Current Weight node to transfer the correct values. However, an OPC UA Client might not support the transfer of three values in one node. Therefore, it is helpful to choose one that does. METTLER TOLEDO offers these three values as individual nodes in addition to Current Weight nodes. This built-in flexibility also respects the fact that many users do not subscribe to all three values for weight processing in their IT system.

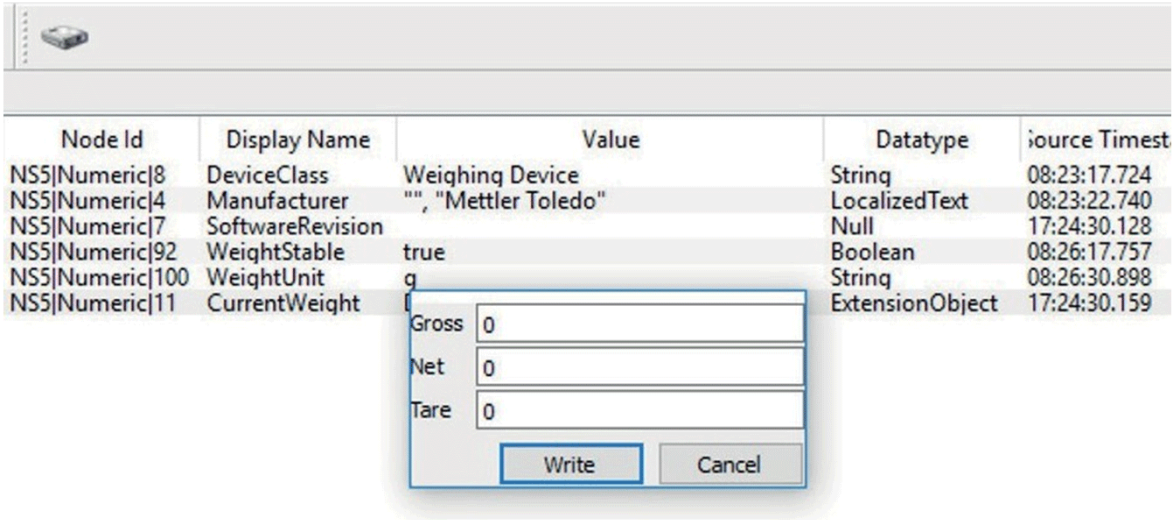

Figure 54: Client View with Display of net, Tare and Gross Weight via Current Weight

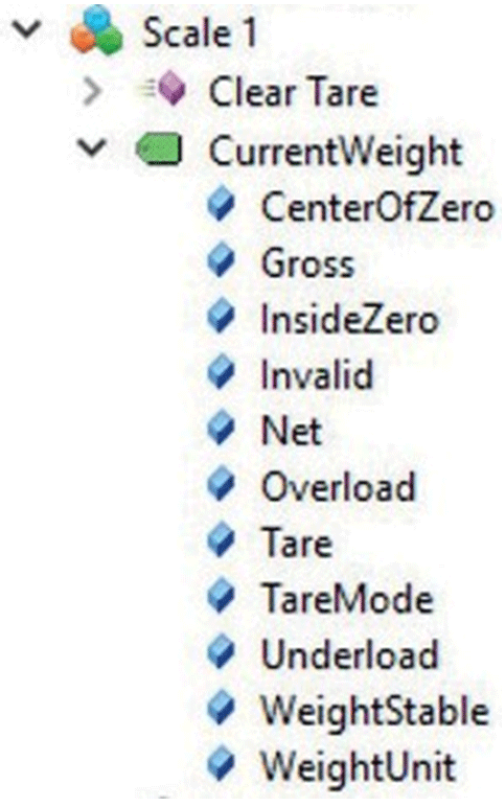

Figure 55: Client View with Individual Nodes for Gross, Net and Tare

What is the Registered Weight Node?

Weighing is often performed in an unstable environment with vibrations and draft. This can result in permanently fluctuating net, tare or gross values. This can pose a problem when the user wants to get the correct value for further processing in their IT system.

Fluctuating value issues can be solved by using the Registered Weight node which freezes the next stable value in the server after using a method call initiated by the client. The Registered Weight node keeps its value until the client calls the method RegisterWeight again.

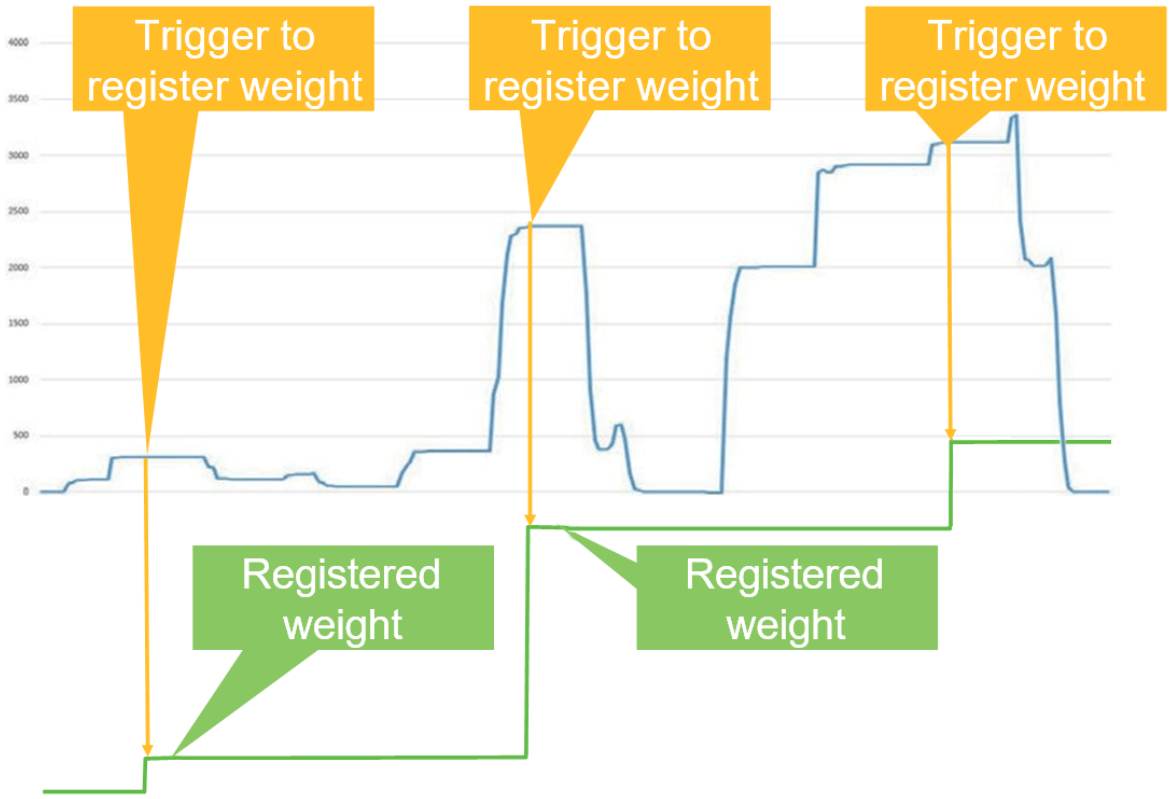

Figure 56: Context Between Current Weight and Registered Weight

Tips to Remember about Method Functionality

method is sub-process initiated by the client and executed by the weighing device. The use of a method relieves the software engineer from having to deal with multiple data nodes, handshakes and states to control a weighing process. The execution of a method in a Simple Weighing application can last from a fraction of a second to several minutes. However, the execution of method calls for „Tare", „SetPresetTare" and „Zero" can last longer than expected. This can be due to an unstable environment (vibrations or draft) or the sloshing or vaporization of liquids that de- lays detection of the criteria for a stable weight value.

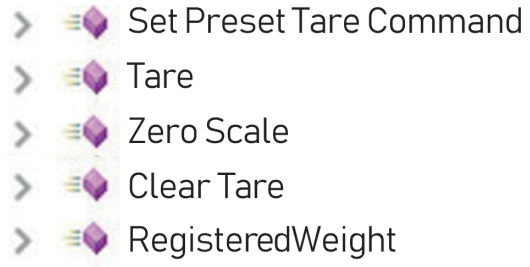

Figure 57: Methods for Simple Weighing

10 Appendix A: Installation and Accessories

10.1 Installation Notes

- Do not open or modify the ACI400 IIoT Edge device. The device uses components that comply with FCC and CE regulations. Modification of the device will void these certifications.
- Install the device securely. Be careful handling the device to prevent injury and do not drop.
- Keep the device away from liquids and flammable materials.
- Do not clean the device with liquids. The chassis can be cleaned with a cloth.
- ACI400 is intended for indoor operation only. Installation in server room conditions or a well ventilated area is recommended.
- Install the device only with shielded network cables.
- Service and repair of the device must be performed by qualified METTLER TOLEDO service personnel.

10.2 Operating Environment

Operating temperature must be between 0 and 50°C with a non-condensing relative humidity of 10-90%. The device can be stored at temperatures between 0 and 60°C.

10.3 Power

Only use the ACI400 IIoT Edge device with the provided UL-Listed external power supply. This external power supply has a rated output of 12 VDC, min. 3A min.

10.4 Kit Contents

Each ACI400 comes with the following items:

- ACI400 IIoT Edge device
- External power supply with adapter plugs for US, UK, Europe and Australia
- Mounting Bracket
  - Kit # 30551833 contains a bracket for DIN rail mounting
  - Kit # 30551834 contains a bracket for wall mounting
  - Both kits contain rubber pads for flat surface mounting
- QR code and URL link to electronic documentation

10.5 Dimensions

10.5.1 ACI400

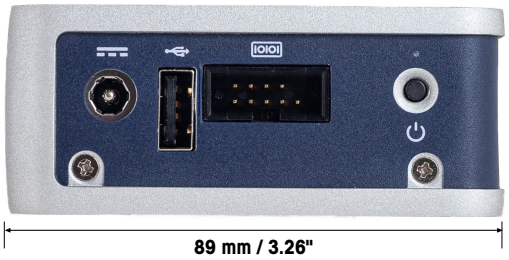

Figure 58: ACI400 Dimensions, Side/Bottom

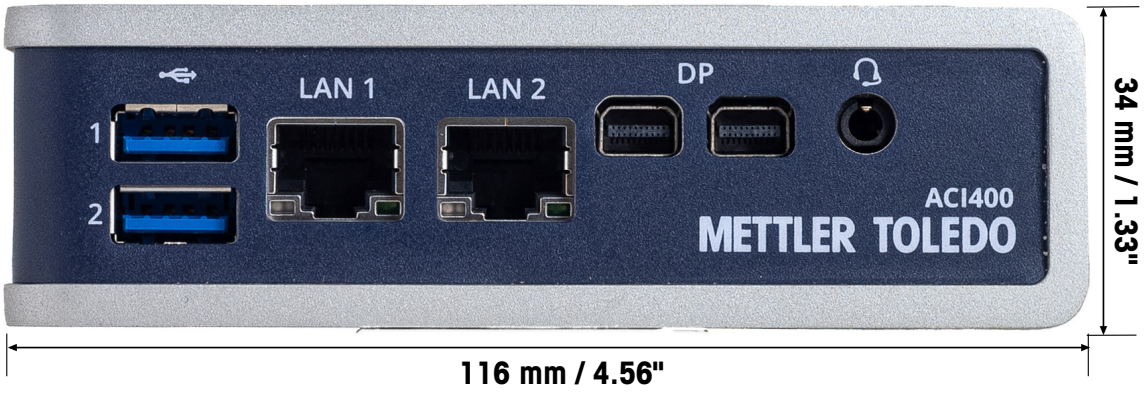

Figure 59: ACI400 Dimensions, Front

10.5.2 ACI400 with Brackets

Wall/Ceiling Mounting Bracket

Figure 60: ACI400 Dimensions with Wall/Ceiling Bracket

DIN Rail Mounting Bracket

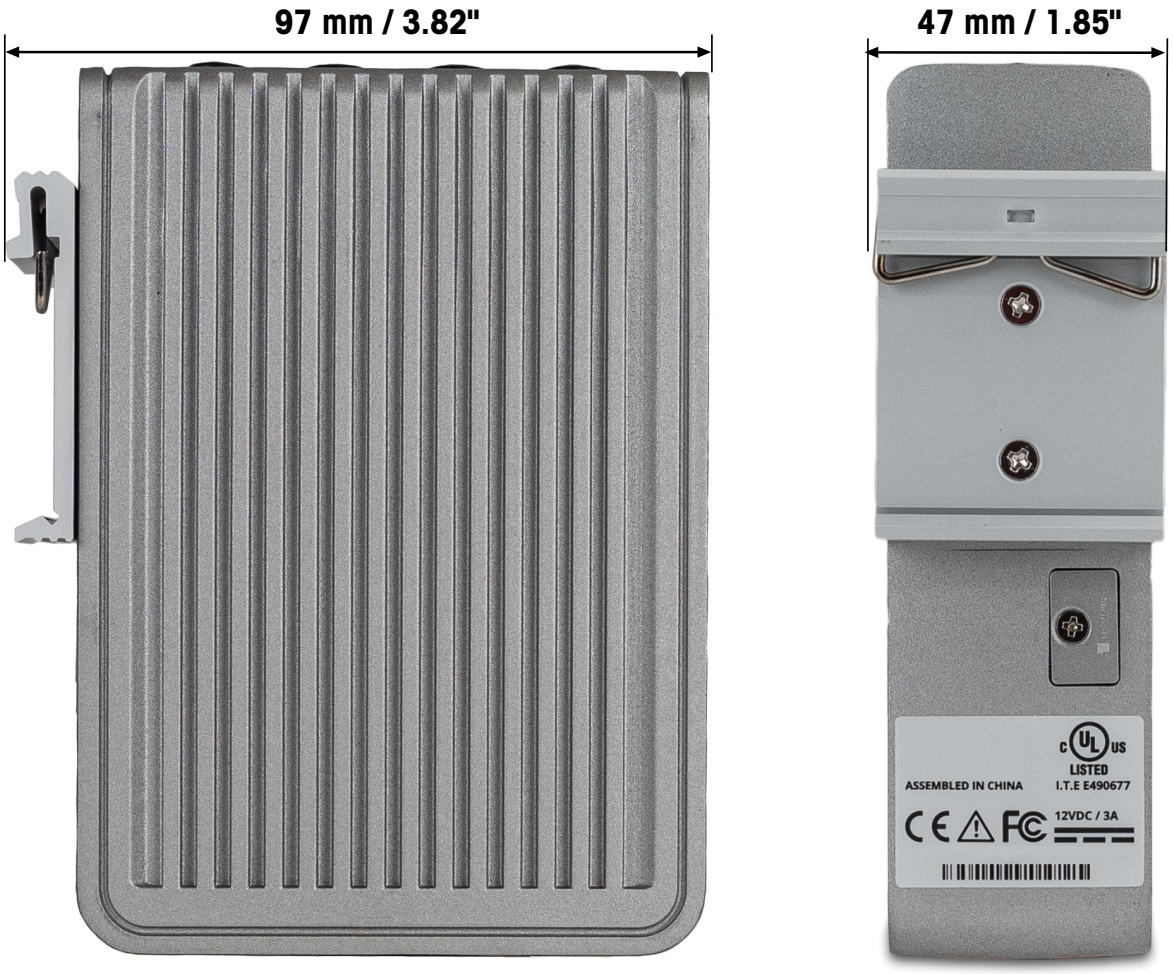

Figure 61: ACI400 Dimensions with DIN Rail Bracket

10.6 Mounting

NOTICE

Allow at least 51mm / 2" of space around all sides of the device for proper cooling. If device is mounted to a vertical surface, the heatsink fins should allow air to rise unobstructed. Alternative orientations may result in reduced operational temperature range.

Two mounting options are available for the ACI400 IIoT Edge device: Wall-mount or DIN-rail mount. Each ACI400 IIoT Edge kit includes one of two possible mounting hardware options.

- Kit # 30551833 contains a bracket for DIN rail mounting
- Kit # 30551834 contains a bracket for wall mounting
- Both kits contain rubber pads for flat surface mounting

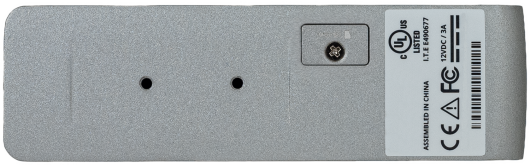

Figure 62: Threaded Holes for Mounting

10.6.1 Wall/Ceiling Mounting

To mount the ACI400 on a wall or ceiling:

1 Use the mountain bracket included with the ACI400 as a template to mark screw holes in the mounting location. In the figure below, holes are marked for vertical and horizontal mounting.

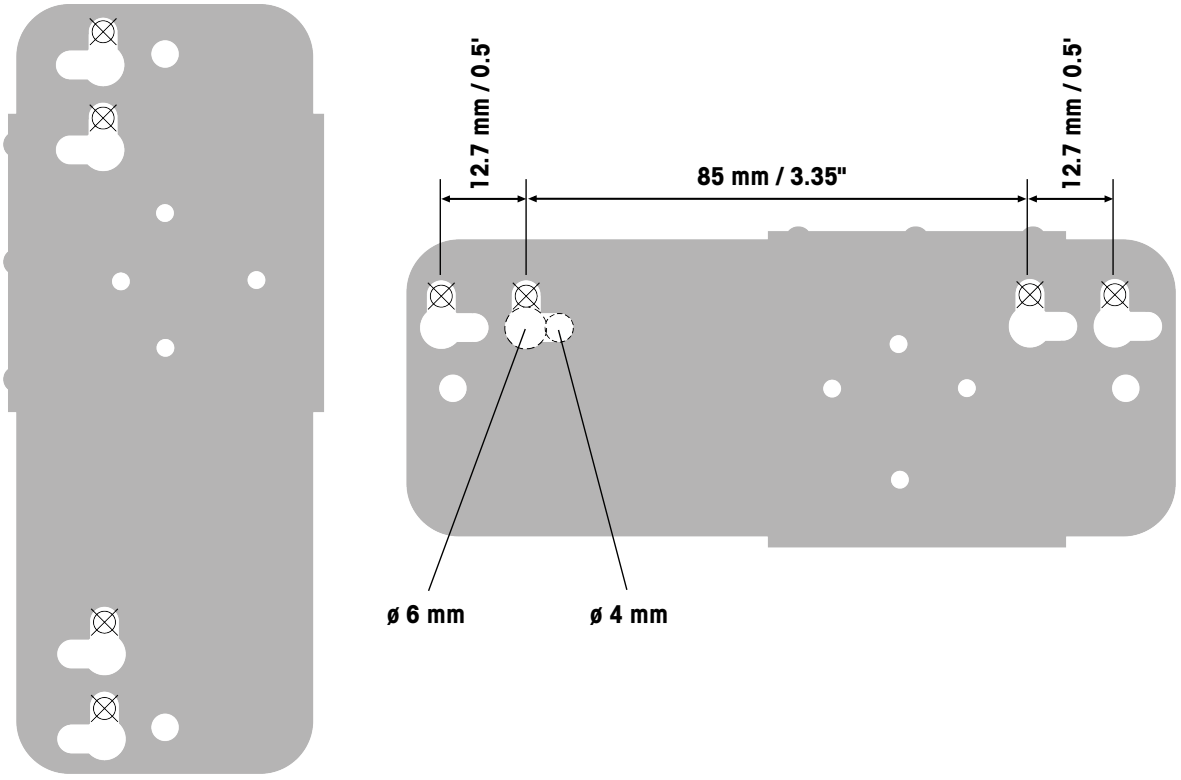

Figure 63: Marking Mounting Holes

2 Mount the device in the desired location using appropriate fasteners. The mounting surface should be of metal construction and have a minimum thickness of 1mm. Use mounting screws of at least 4 mm length, and add an additional 1mm of screw length for each millimeter of plate or bracket thickness beyond 1.5 mm.

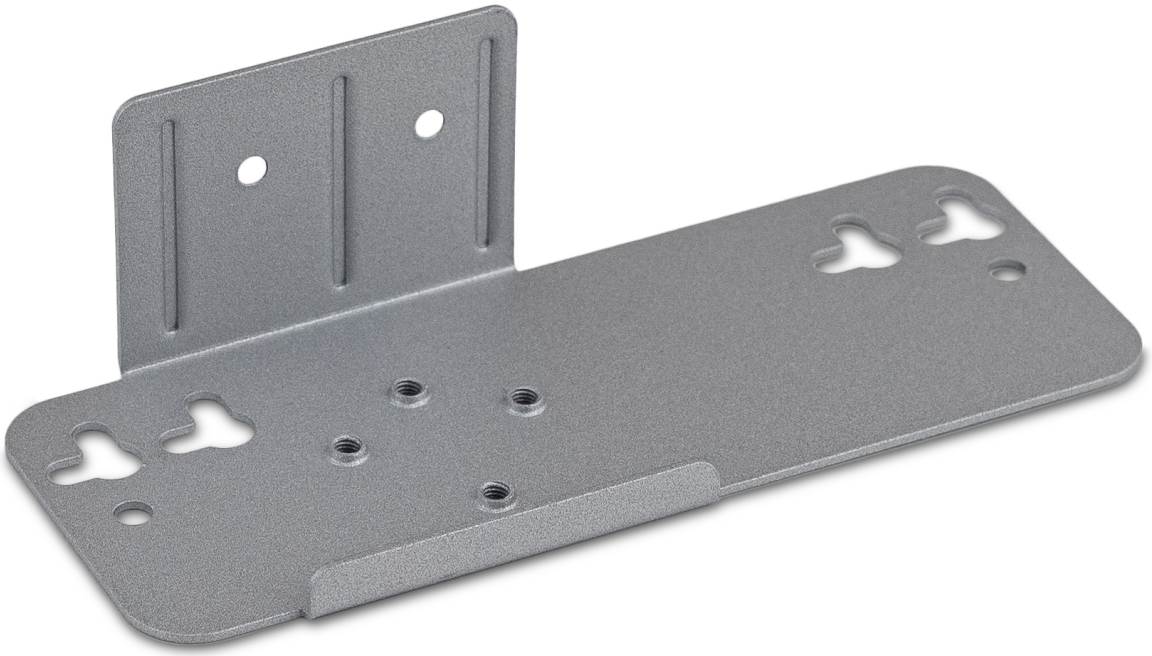

Figure 64: ACI400 Mounting Bracekt

3 Next, use the two provided M3 x 0.5 mm screws to attach the ACI400 to the mounting bracket, using the threaded holes on the back of the chassis (Threaded Holes for Mounting). The figure below shows the ACI400 attached to a bracket in the preferred orientation for optimum cooling.

Figure 65: Mounting Bracket Attached to ACI400

10.6.2 DIN Bracket

1 Attach the DIN rail Bracket tot he ACI400 chassis using the two provided M3 x 0.5 screws.

Figure 66: ACI400 DIN Bracket

Figure 67: DIN Bracket Attachment

2 Clip the device securely onto the DIN rail.

Figure 68: ACI400 Installed on DIN Rail

10.7 Connections

With the ACI400 mounted in position, make the required connections. The device has the following external connections on its front and bottom (or side depending on mounting orientation) surfaces:

10.7.1 Front Connections

- 2 USB 3.0 ports
- 2 RJ45 LAN ports
- 2 mini-Display Ports (not used)
- 1 Combo Audio jack (not used)

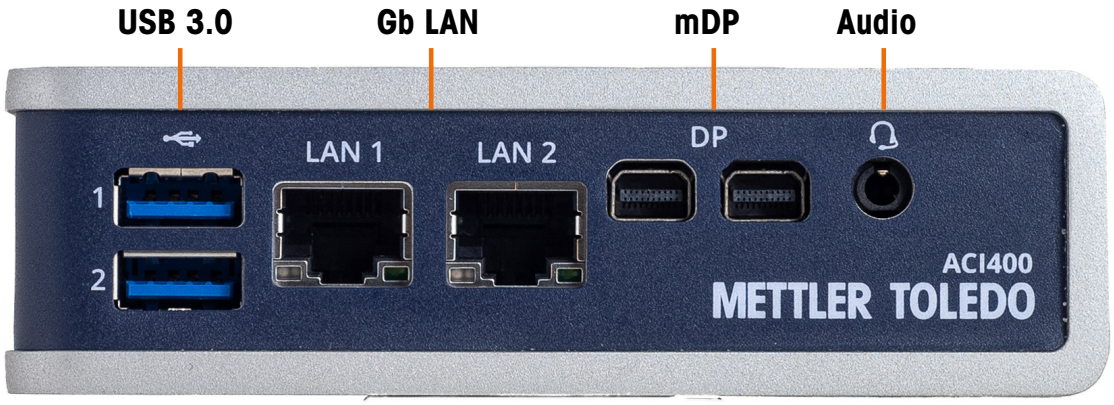

Figure 69: Front Connections

10.7.2 Bottom/Side Connections

- 1 USB 2.0 port
- 1 9-pin RS232 serial COM port
- 1 DC in power jack

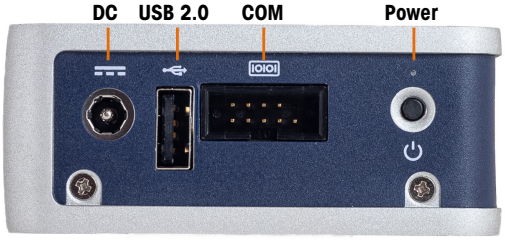

Figure 70: Connections, Bottom/Side

10.8 Accessories and Parts

ACI400 Accessories

Accessory of Spare Part Description	Material Number
USB to RS232 Converter Cable (FTDI)	64088427
RS232 Serial Cable	30539590
DIN Mounting Bracket	30551859
Wall Mounting ACI400 IIoT Edge	30551860
External Power Supply, 12 VDC output	30576643

11 Appendix B: Admin Functions and Diagnostics

11.1 Admin Functions

Use the following login information to access the Admin tab of the ACI400 webserver:

- USERNAME: **mettler**
- PASSWORD: **mettlertoledo**

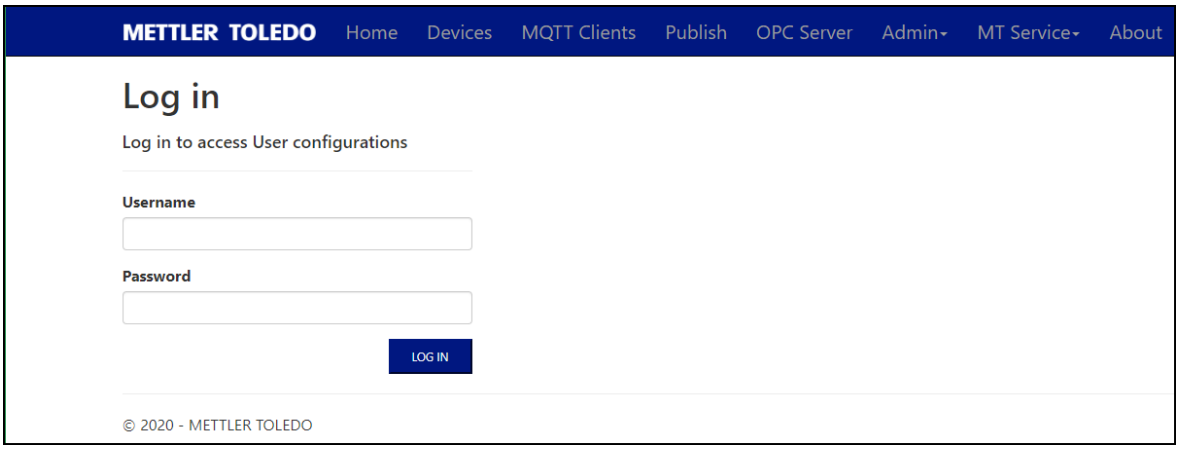

Figure 71: Log in Dialog

See also

Login ▸ Page 16

11.1.1 Overview of Admin Functions

The Admin menu drop-down includes two sections of options. Accessing any of the first options, listed in BLUE, stops all ACI400 communication service. To ensure restart of all communication functions, always return to the **Home** tab before exising the ACI400 web browser.

NOTICE

Device communication is NOT interrupted by accessing the Diagnostic options. listed in GREEN.

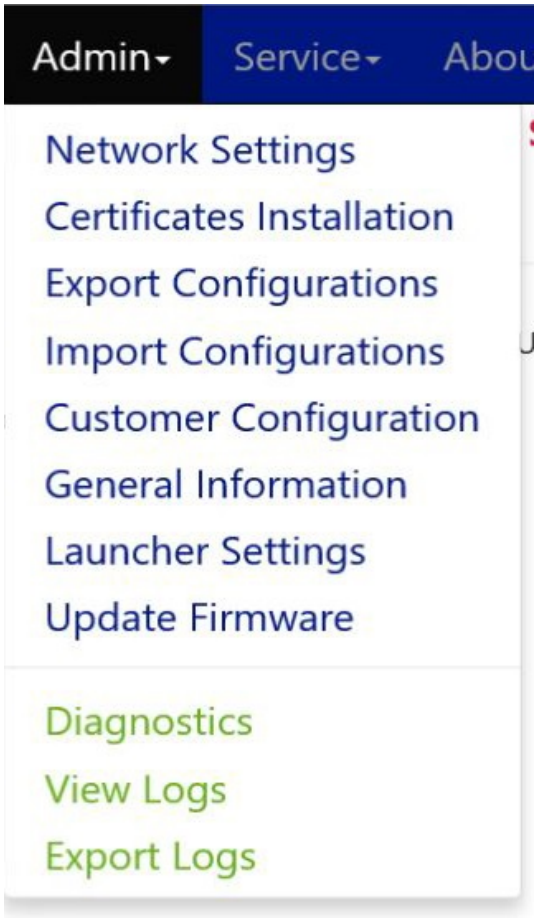

Figure 72: Admin Drop-Down List

NOTICE

ATTENTION! If the Admin tab drop-down list does not appear, please ensure your web browser is up to date. You can also try deleting the web browser cache and all cookies. If the problem persists, please try an alternative browser.

11.1.2 Network Settings

To Change and settings of the LAN1 and LAN2 ports, click **Edit Settings** in the right column of the [ACI400 LAN table ▸ Page 87]. After entering the appropriate settings, click **SET** to [save changes ▸ Page 88].

The Operational Status column in the ACI400 LAN table only indicates if a network cable is plugged into the LAN port. It does not reflect the status of activity on the LAN port.

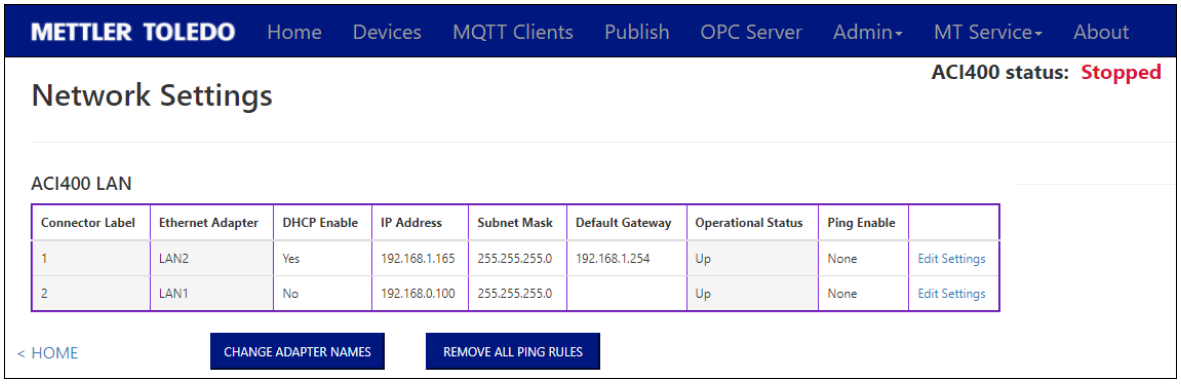

Figure 73: Network Settings Screen

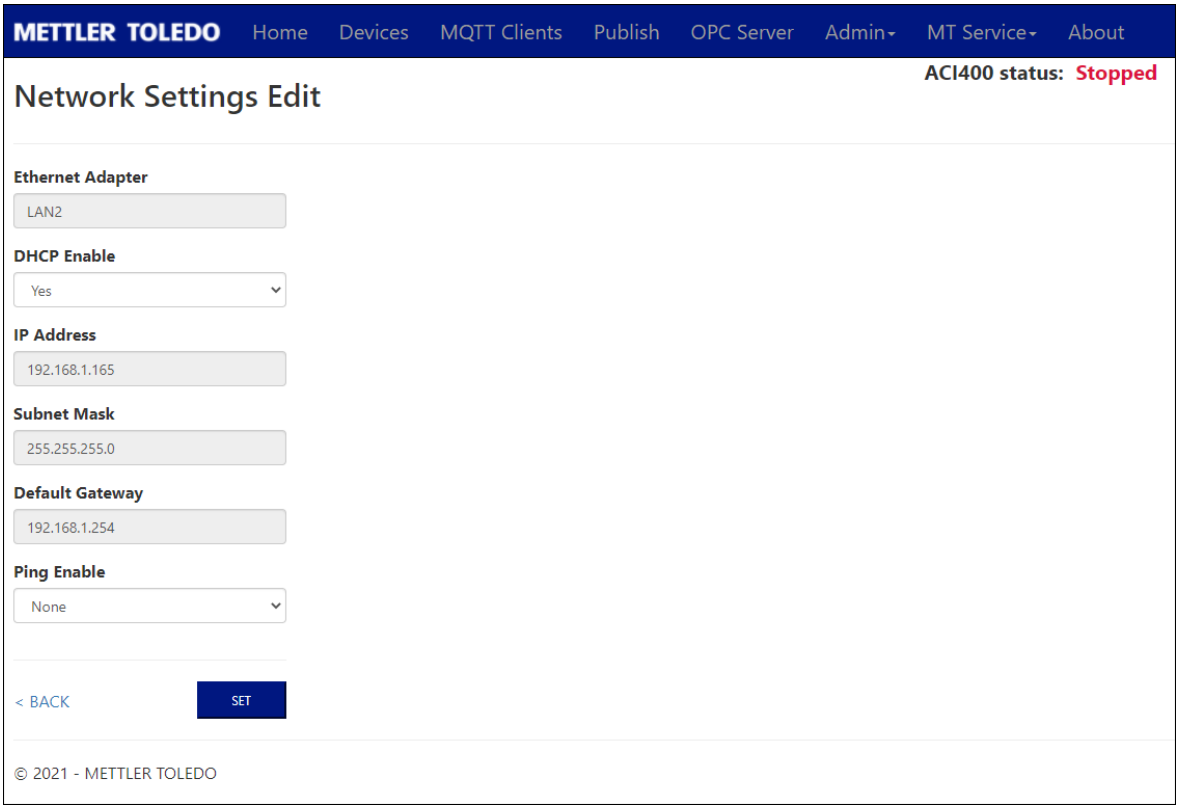

Figure 74: Editing Network Settings

11.1.3 Certificates Installation

MQTT Client Certificates ensure that the handshake between the ACI400 and target client is authenticated before a secure connection is established. MQTT Client Certificates are typically generated by the targeted Cloud service when an ACI400 is registered for connection to the Cloud service.

Certificate uploads **MQTT**, **AWS** and **Azure**, are located using a standard Windows browser window. Select the Choose File button in the Certificates Installation screen, and navigate to the location where the downloaded MQTT Client Certificate is stored.

Once the file is located, click **Open**. Click **Upload**. Completion of upload is indicated in the Load Status bar

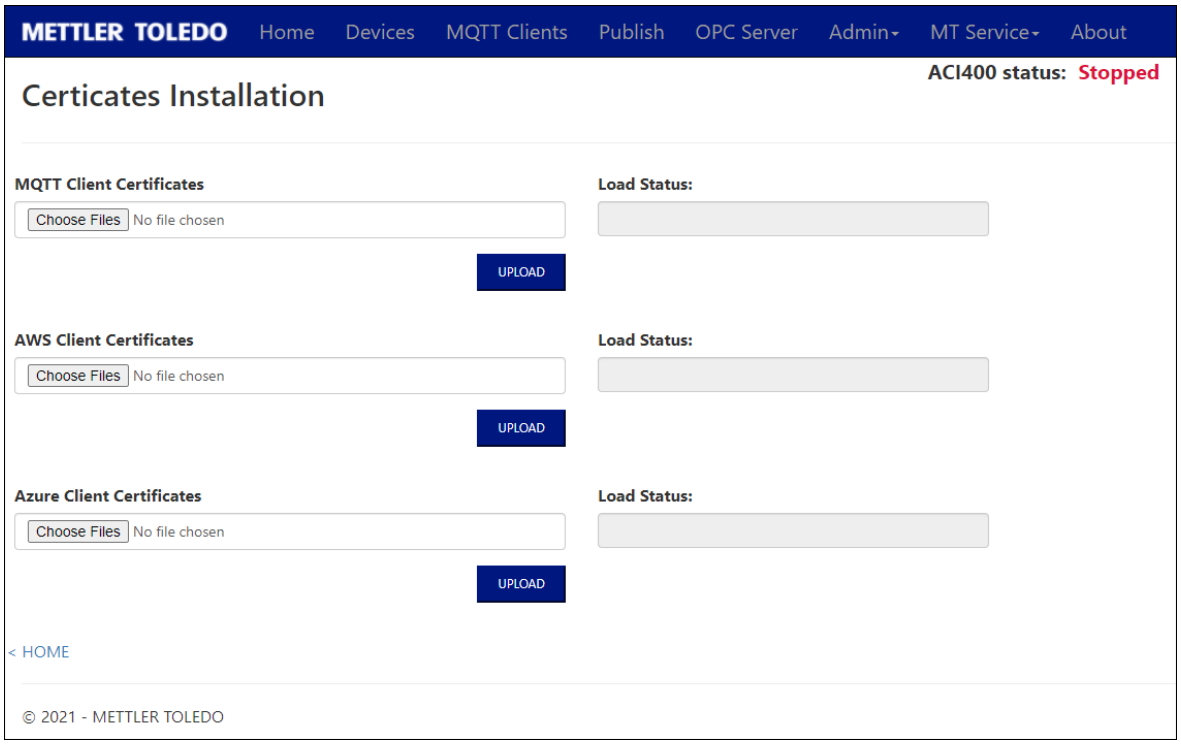

Figure 75: Cewrtificates Upload Screen

11.1.4 Software Update

In order to perform a software update for the device, an installer file (with the suffix **.msi**) must be available, either on the host PC or on a USB drive connected to it.

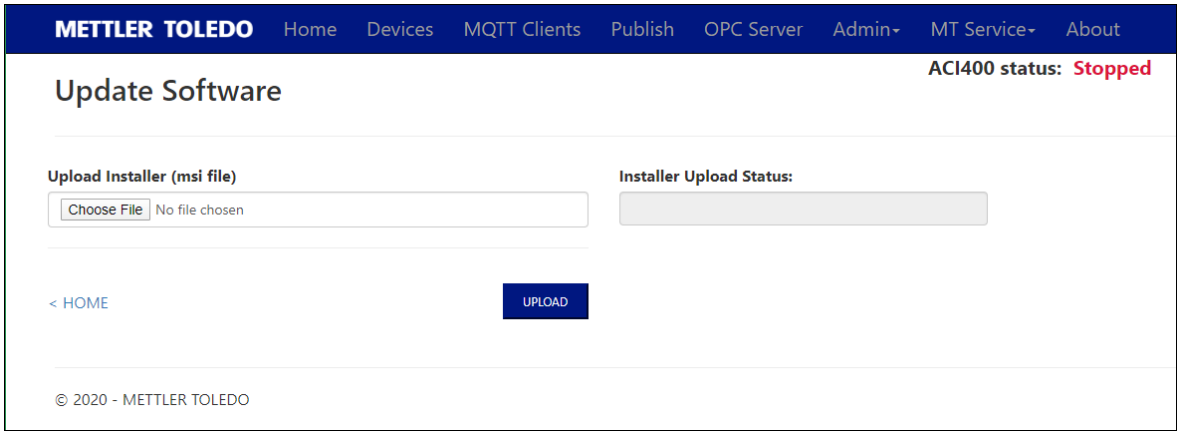

Figure 76: Update Software Screen

When the **Choose File** button is clicked in the **Update Software** screen, a standard Windows browser window opens, permitting navigation to the location where the update file is available.

Once the file is located, click **Open**. The ACI400 will verify that the file is of the correct type, and that the version is newer than the one already installed, and then being the update process. Completion of the upload is indicated in the **Installer Upload Status** bar.

11.1.5 Exporting and Importing Configurations

11.1.5.1 Exporting Configurations

Configuration information is exported in the form of a set of .xml files. The [Export Configurations ▶ Page 90] screen lists all the configurations available. One, several, or all can be selected.

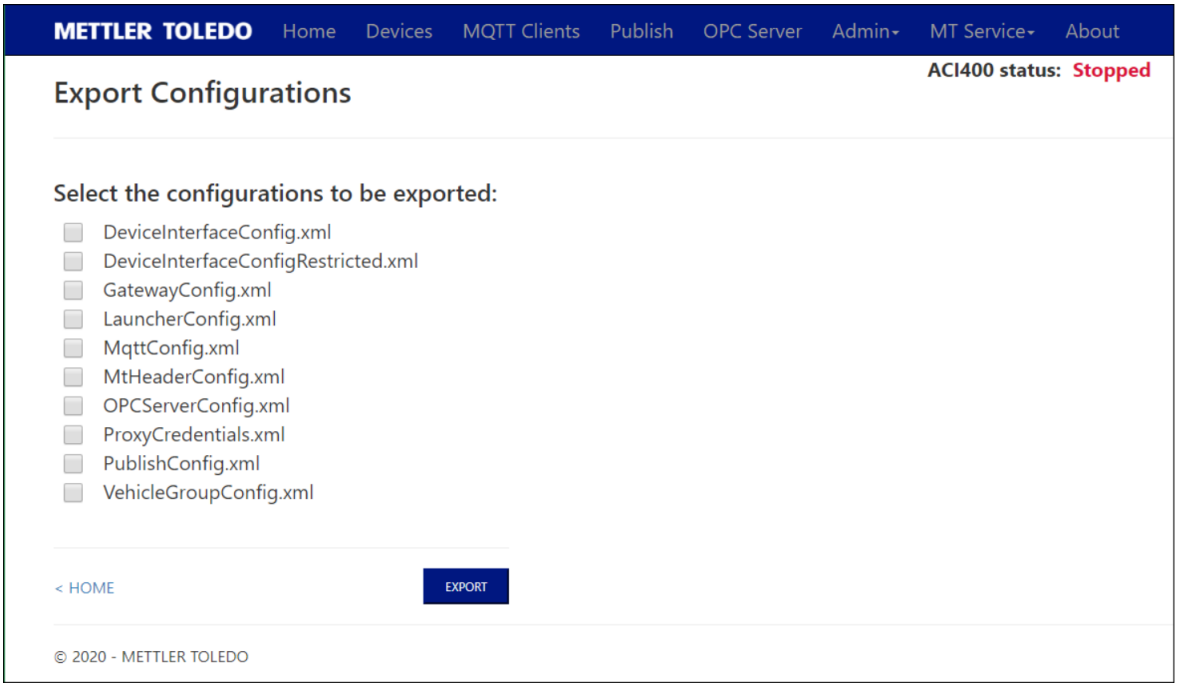

Figure 77: Export Configurations Screen

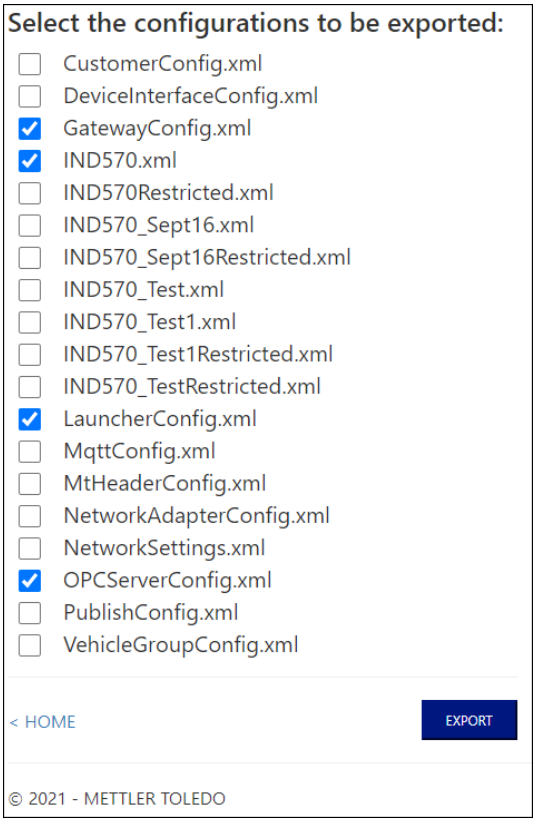

Figure 78: Configuration Files Selected for Export

Once files are selected, click the **Export** button to perform the export. When the export is complete, a .zip file appears, containing the selected .xml files. By default, this file is saved toe h **Downloads** folder on the user's PC.

11.1.5.2 Importing Configurations

Exported .xml files can be imported bo the ACI400 from which they were exported.

NOTICE

While exported configurations can be uploaded into different ACI400 devices for the purpose of cloning, METTLER TOLEDO discourages this practice as duplice IP addresses and Device Interface Ports on a network will be problematic.

The [Import Configurations screen ▶ Page 91] includes a **Choose Files** button. When clicked, this button opens a Windows browser window.

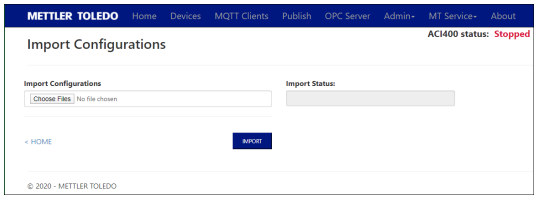

Figure 79: Import Configurations Screen

Browse to the location where the .xml files are stored, and select the desired file or files, using CTRL-Click to select multiple files.

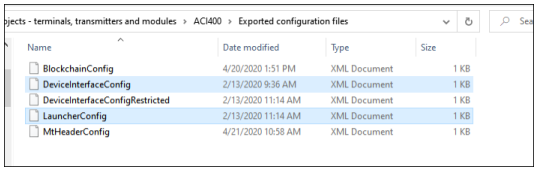

Figure 80: Browsing for Files to Import

When the desired files are selected, click **Import**. When the process is complete, the device will prompt to move them from the temporary folder to the configuration folder.

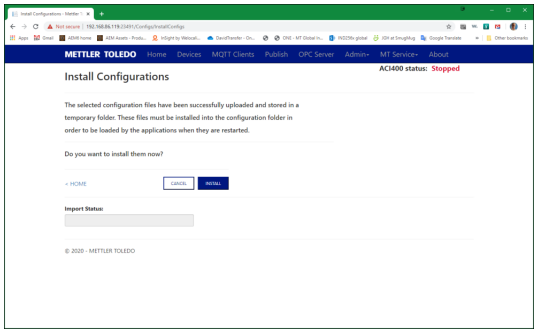

Figure 81: Configuration Files Imported, Ready to Install

Click **Install** to complete the import of configurations. The **Import Status** field will indicate that the installation is complete. The restored or new configuration files will be used when the device is restarted.

Power cycle the ACI400 to ensure all imported files are activated.

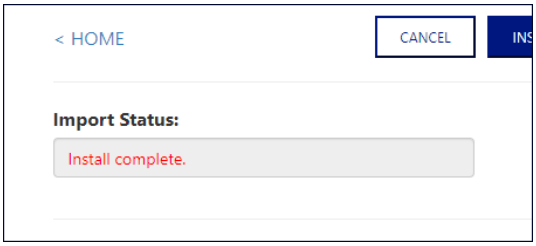

Figure 82: Configuration Installation Complete

11.1.6 Customer Configuration

The Customer Configuration setup allows a user to identify two fiels of data that can be transmitted with each published MQTT payload.

To append these Customer Configuration data fields to ta published payload, the **Optional Data Type** field in the MQTT Client setup must be set to **CustomerConfiguration**. Refer to [MQTT Communication Configuration ▸ Page 18] for details on enabling the **Optional Data Type**.

11.1.7 General Information

The **General Information** tab provides basic information about the ACI400 IIoT Edge Device, such as current firmware version, operating system, serial number and date/time settings.

The **ACI400 Name** field appears in the **GetGatewayConfiguration** MQTT Payload. This field can be modified by the user to provide specific device identification as required.

11.1.8 Launcher Settings

To ensure successful connectivity with targeted OPC UA clients and Cloud services, it may be necessary to configure a delay before the ACI400 OPC UA server or MQTT client is started, and process monitoring begins. Required delays and intervals can be defined in seconds.

Figure 83: Launcher Settings Screen

11.2 Diagnostics and Log Functions

The final three options under the **Admin** tab are intended for troubleshooting. Accessing Diagnostic and Log functions does NOT interrupt ACI400 communication services.

11.2.1 Diagnostics

The Diagnostics page provides diagnostic information about ACI400-to-Device connections and ACI400 connections to external data services.

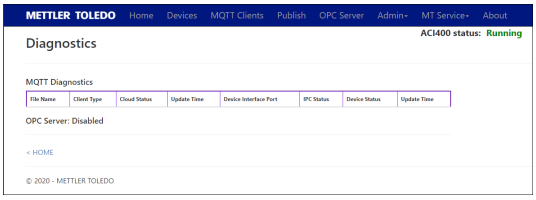

Figure 84: Diagnostics Screen

11.2.1.1 Device Interface Diagnostic Field Definitions

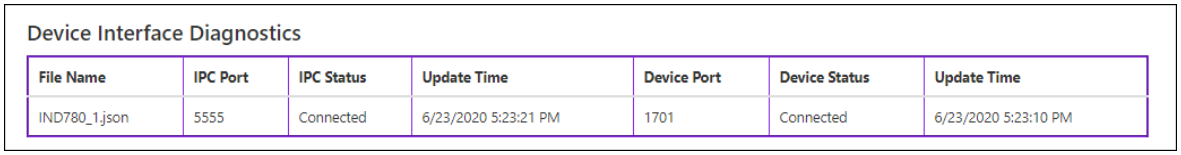

Figure 85: Device Interface Diagnostics Fields

Parameter	Description
File Name	Matches the Device Interface File Name entered in the Device Interface Startup list under the Devices tab. The Devices diagnostics file has a ".json" extension instead of the ".xml" used for the configuration file.
IPC Port	Device Interface Port entered in the Device Interface configuration file. Values should be in the range of 5555 – 65535.
IPC Status	IPC status is the status of the connection between an IPC client and the IPC server. Possible values: 1. NotConnected – No IPC connection has been made between an IPC client (MQTT client or the OPC server) and the IPC server which is the Device Interface. 2. Connected – A connection has been made successfully to the Device Interface. Once a connection exists, then if publishing items exist in the Publishing list the MQTT client will start making requests to the Device Interface. The OPC server typically always makes requests to the Device Interface on a timed basis. 3. KeepAlive – A connection has been made, but no request or event data is moving between the applications. When no request/reply or event data has been sent or received for a time interval (typically 30 seconds) the IPC client will send a status request to the Device Interface to verify that the connection is still viable. 4. Error – A data or status request has been made by an IPC client and no response was returned from the Device Interface. After a time interval the IPC client will disconnect and try to reconnect to the Device Interface. 5. Disconnected – The IPC client has disconnected from the Device Interface.
Update Time	Date and time of the last IPC Status change (not the last time an update occurred). This is done to keep the number of file updates to a minimum.
Device Port	The TCP/IP port number used by the Device Interface to connect to the configured indicator. The port is assigned in the Device Interface configuration file and must match the indicator port setup. (Only used for Ethernet communications.)
Device Status	Device Status is the status of the connection between the ACI400 Device Interface and the configured terminal/indicator. Possible values: 1. NotConnected – No connection has been made between Device Interface and the terminal/indicator. 2. Connected – A connection has been made successfully from the ACI400 Device Interface to the indicator. 3. Error – The ACI400 Device Interface failed to connect to the terminal/indicator or a command has been issued to the terminal/indicator from the Device Interface and no response was received 4. Disconnected – The ACI400 Device Interface has disconnected from the terminal/indicator.
Update Time	Date and time of the last Device Status change (not the last time an update occurred). This is done to keep the number of file updates to a minimum.

11.2.1.2 MQTT Client Diagnostics Field Definitions

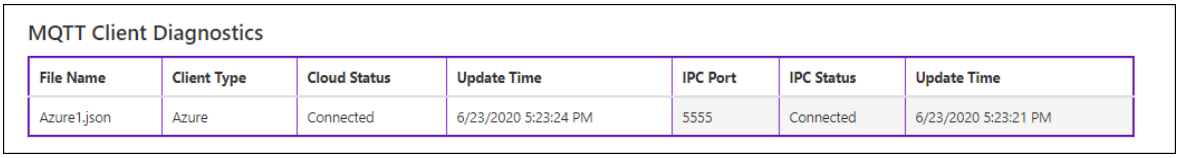

Figure 86: MQTT Client Diagnostics Fields

MQTT Client Diagnostics Fields

Parameters	Description
File Name	Matches the **Client Configuration File Name** entered in the MQTT Client Startup list under the MQTT Clients main menu item. The diagnostics file has a ".json" extension instead of the ".xml" used for the configuration file.
Client Type	Client Type specifies the type of cloud connector used to connect to the cloud services. Possible values: 1. None – No Client Type assigned. 2. MQTTBroker – This selects the standard MQTT broker connector which is typically used for testing. 3. Amazon – This selects the Amazon connector which contains specific logic required to connect to the AWS IoT managed cloud platform at Amazon. 4. Azure – This selects the Azure connector which contains specific logic required to connect to Azure Portal services. 5. IBM – This selects the IBM connector which contains specific logic required to connect to the IBM cloud and the Watson IoT platform.
Cloud Status	Cloud Status is the status of the connection between the MQTT client and the selected cloud service. Possible values: 1. NotConnected – No connection has been made to the selected cloud service. 2. Connected – A connection has been made successfully to the selected cloud service. 3. Error – An error occurred sending data to the cloud service. After a time interval the connector will try to re-establish the connection to the cloud service. 4. Disconnected – The connector has disconnected from the cloud service. 5. TE_Resend – (Azure connector only). Azure connector has encountered a transient error while sending data to the Azure cloud service. The connector will try to resend the data one time.
Update Time	Date and time of the last Cloud Status change (not the last time an update occurred). This is done to keep the number of file updates to a minimum.
IPC Port	The Device Interface Port entered in the Device Interface configuration file. It is the port used in the inter-process communications (IPC) between the MQTT client and the Device Interface application. Values should be in the range of 1025 – 65535.

IPC Status	IPC status is the status of the TCP/IP connection between an MQTT client and the Device Interface. Possible values: 1. NotConnected – No IPC connection has been made between the MQTT client and the Device Interface. 2. Connected – A connection has been made successfully to the Device Interface. Once a connection exists, if publishing items exist in the Publishing list the MQTT client will start making requests to the Device Interface. 3. KeepAlive – A connection has been made, but no request or event data is moving between the applications. When no request/reply or event data has been sent or received for a time interval (typically 30 seconds) the MQTT client will send a status request to the Device Interface to verify that the connection is still viable. This generally happens when there are no items in the publishing list for the MQTT client. 4. Error – A data or status request has been sent by an MQTT client and no response was returned from the Device Interface. After a time interval the MQTT client will disconnect and try to reconnect to the Device Interface. 5. Disconnected – The MQTT client has disconnected from the Device Interface.
Update Time	Date and time of the last IPC Status change (not the last time an update occurred). This is done to keep the number of file updates to a minimum.

11.2.1.3 OPC UA Server Diagnostics Field Definitions

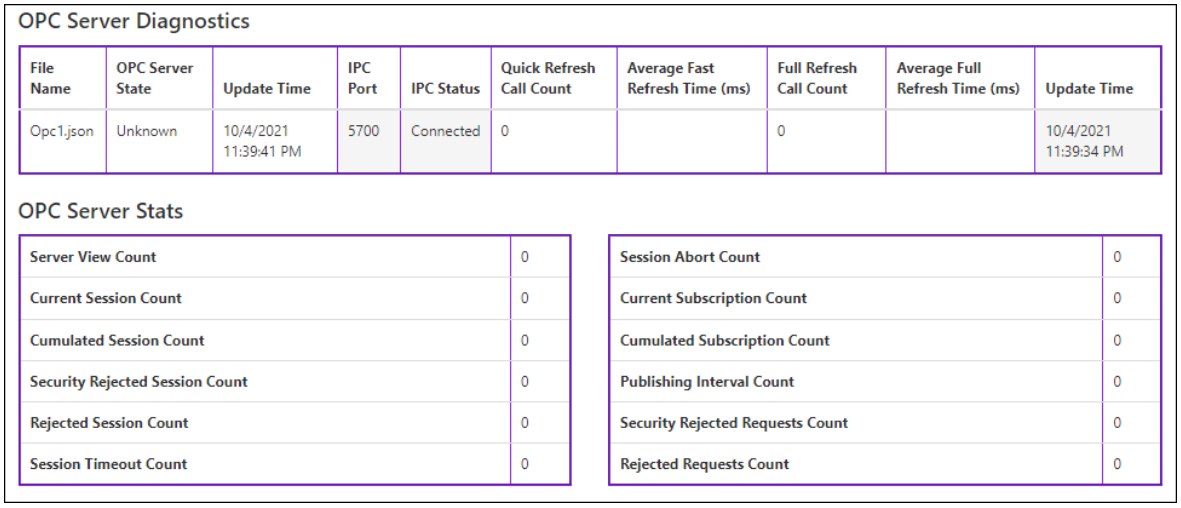

Figure 87: OPC UA Diagnostics Fields

OPC UA Diagnostics Fields

Parameter	Description
File Name	Fixed name for the OPC server diagnostics file.
OPC Server State	Current state of the OPC UA server.
Update Time	Date and time of the last OPC Server State update.
IPC Port	Device Interface Port entered in the Device Interface configuration file. It is the port used in the inter-process communications (IPC) between the OPC server and the Device Interface application. Values should be in the range of 1025 – 65535.

IPC Status	The IPC status is the status of the TCP/IP connection between the OPC server and the Device Interface. Possible values: 1. NotConnected – No IPC connection has been made between the OPC server and the Device Interface. 2. Connected – A TCP/IP connection has been made successfully to the Device Interface. Once a connection exists, the OPC server typically makes requests to the Device Interface on a timed basis. 3. KeepAlive – A TCP/IP connection has been made, but no request data is moving between the applications. When no request/reply has been sent or received for a time interval (typically 30 seconds) the OPC server will send a status request to the Device Interface to verify that the connection is still viable. 4. Error – A data or status request has been sent by the OPC server and no response was returned from the Device Interface. After a time interval the OPC server will disconnect and try to reconnect to the Device Interface. 5. Disconnected – The OPC server has disconnected from the Device Interface.
Quick Refresh Call Count	Number of times the ACI400 has refreshed the data fields governed by the OPC Server Device Polling Interval setting. Refer to [OPC Server ▸ Page 41].
Average Fast Refresh Time (ms)	Realized average rate the Device Polling Interval is executing (refer to [OPC Server ▸ Page 41]). Use the value in this Average Fast Refresh Time field to appropriately set the OPC UA Device Polling Interval on the OPC Server screen/tab. Time set in Device Polling Interval field should be 500-1000 msec greater than value displayed in this Average Fast Refresh Time field.
Full Refresh Call Count	Number of times the ACI400 has refreshed the data fields governed by the OPC Server Refresh Static Data field. Refer to [OPC Server ▸ Page 41].
Average Full Refresh Time (ms)	Realized average rate at which the Refresh Static Data is executing. Refer to [OPC Server ▸ Page 41].
Update Time	Realized average rate at which the Refresh Static Data is executing.
OPC Server Stats	Server statistics are generated by the embedded (Softing) OPC server code.

11.2.2 Viewing and Exporting Logs

The ACI400 device logs information which allows its functions and problems to be diagnosed.

11.2.2.1 View Logs

The View Logs screen lists all available logs, but only one can be selected - and disapleyed - at a time.

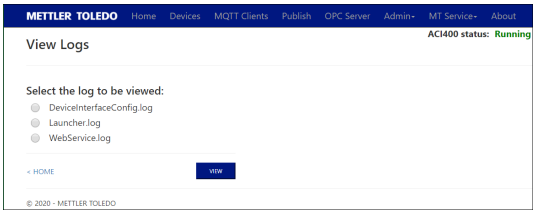

Figure 88: View Logs Screen

To view a log's contents, clcik on the radio button next to the log's name, then click **View**.

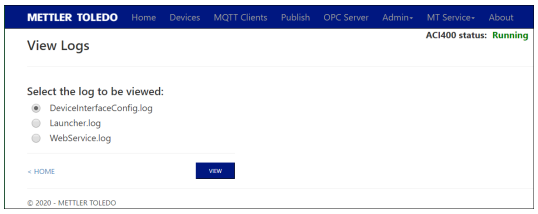

Figure 89: Selecting a Log to View

When the **View** button is clicked, the log's contents display in a pane to the right of the window. Place the cursor over the contents and use a mouse wheel or the keyboard's up and down arrows to scroll through the displayed log.

[View Logs ▸ Page 97] provides a sample view of a Device Interface log maintained by the ACI400.

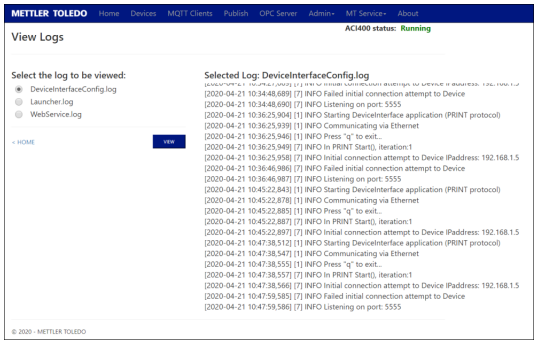

Figure 90: View Logs - DeviceInterfaceConfig Log Contents Displayed

11.2.2.2 Export Logs

Like the configuration file export function, the **Export** button in the **Export Logs** screen sends the selected log's contents, in the form of a .zip file, to the host PC's **Downloads** folder. Any or all of the logs can be selected for export.

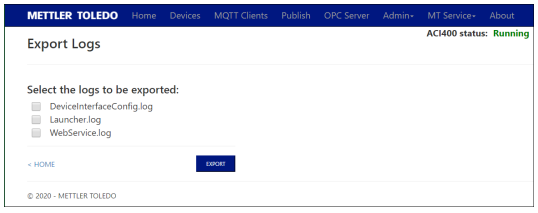

Figure 91: Export Logs Screen

To protect your product's future:
METTLER TOLEDO Service assures the quality, measuring accuracy and preservation of value of this product for years to come.

Please request full details about our attractive terms of service.

www.mt.com/service

www.mt.com
For more information

Mettler-Toledo (Changzhou) Measurement Technology Co., Ltd.
111 Tai Hu Xi Road
213125 Changzhou Jiangsu Province
People´s Republic Of China
www.mt.com/contacts

3057841101

30578411