

Configuring history on MQTT Engine tags

The history configuration for MQTT Engine tags can be setup as needed for your specific requirements as described in Ignition's [Configuring Tag History](#) and [How the Tag Historian System Works](#) documents.

There are however some caveats regarding tag history configuration persistence and handling of historical event data that are detailed in this document.

- [History Configuration Persistence](#)
- [MQTT Engine Historical Event Processing](#)
- [Writing historical events directly to the database, via the Historian, bypassing the tag](#)
 - MQTT Engine
 - Tag History Configuration
 - Edge Client
- [Writing historical events directly to tags](#)
 - MQTT Engine
 - Tag History Configuration
 - Edge Client
- [History Configuration Examples](#)
 - Sample Mode = On Change, Min Time Between Samples = 0, Max Time Between Samples = 0
 - Sample Mode = On Change, Min Time Between Samples = 0, Max Time Between Samples = 5
 - Sample Mode = Periodic, Max Time Between Samples = 0, Sample Rate = 5

History Configuration Persistence

Currently history configuration for MQTT Engine tags is not persistent and it may be required to delete MQTT Engine tags in certain cases. Reference [Understanding how tag changes at the Edge affect MQTT Engine](#) for more details.

The recommended way of configuring history for MQTT Engine tags is to use reference tags to indirectly reference MQTT Engine tags. This allows the history configuration to be persisted on the reference tag when the underlying MQTT Engine tags are deleted.



The reference tag must be a Reference tag only. Derived, Expression and OPC tags (expose MQTT Engine tag provider through OPCUA server) will not properly store history when MQTT Engine tags are updated with historical data at a high rate of speed. This is a limitation within the Ignition platform and may be addressed in a future release.

Alternatively you can use scripting to reapply the history configuration directly to MQTT Engine tags on demand

MQTT Engine Historical Event Processing

MQTT Engine has two ways to process historical events and insert the historical data into Ignition's Tag Historian module:

- [Writing historical events directly to the database, via the Historian, bypassing the tag](#)
- [Writing historical events directly to the tag](#)



Writing historical events directly to tag is required if there are:

- Tag Events scripts that fire when applicable
- Alarms that are triggered when applicable
- Reference tags used to indirectly reference MQTT Engine tags

Writing historical events directly to the database, via the Historian, bypassing the tag

The configuration parameters required to write historical events directly to the database, via the Historian, bypassing the Tag are shown below.

MQTT Engine

Under the MQTT Engine Settings General Tab, navigate to the Miscellaneous Settings and ensure Store Historical events is selected

Store Historical Events

☒ Enable the writing of historical change events directly to the History provider instead of updating the Tag value

Tag History Configuration

History must be enabled on the MQTT Engine tag but the only property used from the tag history configuration is the Storage Provider.

All historic data will be written to the configured Storage Provider, via the Historian, using the timestamp associated with the historical data.



All other properties such as Sample Mode, Max Time Between Samples, Deadband etc are ignored when writing historic data

Edge Client

The Edge side client can publish historic data either in-order (synchronously) or asynchronously.



If you are using MQTT Transmission as the Edge side client, under the MQTT Transmission Settings for your transmitter, navigate to the History Settings to disable the In-Order History configuration parameter. Enabling the In-Order History for MQTT Transmission is unnecessary and will result in a waste of resources.

In-Order History

☐ Flush history in-order (synchronously) before live data resumes
(default: false)

Writing historical events directly to tags

The configuration parameters required to write historical events to the tag instead of directly to the Historian are detailed below.

MQTT Engine

Under the MQTT Engine Settings General Tab, navigate to the Miscellaneous Settings and ensure Store Historical events is de-selected

Store Historical Events

☐ Enable the writing of historical change events directly to the History provider instead of updating the Tag value

Tag History Configuration

History must be enabled on the tag (either directly on the MQTT Engine tag and/or reference tag) with these three settings required:

- Deadband Style set to Discrete
- Sample Mode set to On Change
- Min Time Between Samples set to 0

▼ History	
History Enabled	● true ▼
Storage Provider	● MySQL ▼
Deadband Style	● Discrete ▼
Deadband Mode	● Absolute ▼
Historical Deadband	● 0.01 ▼
Sample Mode	● On Change ▼
Min Time Between Sampl...	● 0 ▼
Min Time Units	● Seconds ▼
Max Time Between Sampl...	● 0 ▼
Max Time Units	● Seconds ▼

All historic data will be written to the tag using the timestamp associated with the historical data.



Deadband Style must be set to Discrete for all tag datatypes. This ensures that a change is registered any time the value moves +/- the specified amount from the last stored value.

If left at the default Auto setting, Ignition will determine the deadband style based on the tag datatype. See this Ignition [Configuring Tag History Deadband and Analog Compression](#) for details on how the Analog and Discrete Deadband Styles differ.



When Max Time Between Samples is greater than 0, inserts at this sample time are not honored during historical data flushes because the tag updates for historical data occur too quickly.

For [example](#) the Max Time Between Samples is 5 Seconds. Whilst the edge node is offline, the edge node tag has updates every 10 seconds. On reconnect, all these historical change events are published and written to the MQTT Engine tag within milliseconds. As a result the Max Time Between Samples does not trigger.

Edge Client

If you are not using reference tags, the Edge side client must publish historic data in-order (synchronously) before live data resumes. This is because Ignition will ignore writes to the tag if the timestamp on the tag change is older than the current value.

If you are using reference tags and have the [Ignition Allow Back-fill Data](#) disabled on the tag provider that will contain the reference tags, the Edge side client must publish historic data in-order (synchronously) before live data resumes. This is because Ignition will ignore writes to the tag if the timestamp on the tag change is older than the current value.

If you are using reference tags and have the [Ignition Allow Back-fill Data](#) enabled on the tag provider that will contain the reference tags, the Edge side client can publish historic data either synchronously or asynchronously. Using each of these different flush methods has the following implications:

- In-order history from the Edge Node
 - Historical data will be stored for the reference tag
 - If history is enabled on MQTT Engine tags, historical data will be stored the MQTT Engine tag
- Asynchronous history from the Edge Node
 - Historical data will be stored for the reference tag
 - Even if history is enabled on MQTT Engine tags, historical data will not be stored the MQTT Engine tag. New values will only be stored in the historical database if it is a non-historical value.



If you are using MQTT Transmission as the Edge side client, under the MQTT Transmission Settings for your transmitter, navigate to the History Settings to select or deselect the In-Order History configuration parameter.

As shown below, when the MQTT Transmission client comes back online and flushes history, it will flush the oldest historical events first (in order) before sending live Tag changes events to Engine.

In-Order History

☒ Flush history in-order (synchronously) before live data resumes
(default: false)

History Configuration Examples

Sample Mode = On Change, Min Time Between Samples = 0, Max Time Between Samples = 0

- Suitable for writing historical events directly to the database or writing historical events directly to the tag
- Real-time data is written to the storage provider on each tag change event whilst the tag quality is Good
- Historical data is written to the storage provider or tag using the timestamp associated with the historical data
 - Handles historical metrics flushed from the Edge that have very high resolution; e.g., the historical metrics have timestamps 1 ms apart for a single tag

▼ History	
History Enabled	● true
Storage Provider	● MySQL
Deadband Style	● Discrete
Deadband Mode	● Absolute
Historical Deadband	● 0.01
Sample Mode	● On Change
Min Time Between Sampl...	● 0
Min Time Units	● Seconds
Max Time Between Sampl...	● 0
Max Time Units	● Seconds

⚡ Resultset 1							
tagid	intvalue	floatvalue	stringvalue	datevalue	dataintegrity	t_stamp	
11	56	NULL	NULL	NULL	Tag value changing - db inserts on change	192	1711644430959
11	55	NULL	NULL	NULL		192	1711644426485
11	54	NULL	NULL	NULL		192	1711644424485
11	53	NULL	NULL	NULL		192	1711644422484
11	52	NULL	NULL	NULL	Tag changes at every every 2 seconds	192	1711644421484
11	51	NULL	NULL	NULL	inserted as historical data after edge	192	1711644419483
11	50	NULL	NULL	NULL	node reconnect	192	1711644417482
11	49	NULL	NULL	NULL		192	1711644415480
11	48	NULL	NULL	NULL		192	1711644413479
11	46	NULL	NULL	NULL	Tag quality bad on edge node disconnect	500	1711644411602
11	47	NULL	NULL	NULL		192	1711644411477
11	46	NULL	NULL	NULL		192	1711644409475
11	45	NULL	NULL	NULL		192	1711644407473
11	44	NULL	NULL	NULL	Tag value changing - db inserts on	192	1711644405471
11	43	NULL	NULL	NULL	change	192	1711644403471
11	42	NULL	NULL	NULL		192	1711644401470
11	41	NULL	NULL	NULL		192	1711644399467

Sample Mode = On Change, Min Time Between Samples = 0, Max Time Between Samples = 5

- Suitable for writing historical events directly to the database or writing historical events directly to the tag
- Real-time data is written to the storage provider on each tag change event whilst the tag quality is Good
- Real-time data is written to the storage provider on the Max Time Between Samples interval if no tag change events whilst the tag quality is Good
- Historical data is written to the storage provider or tag using the timestamp associated with the historical data
 - When the Max Time Between Samples > 0, there will be no inserts during a history flush as the tag updates occur too quickly
 - Handles historical metrics flushed from the Edge that have very high resolution; e.g., the historical metrics have timestamps 1 ms apart for a single tag

▼ History		
History Enabled	● true	▼
Storage Provider	● MySQL	▼
Deadband Style	● Discrete	▼
Deadband Mode	● Absolute	▼
Historical Deadband	● 0.01	
Sample Mode	● On Change	▼
Min Time Between Sampl...	● 0	
Min Time Units	● Seconds	▼
Max Time Between Sampl...	● 5	
Max Time Units	● Seconds	▼

⚡ Resultset 1							
tagid	intvalue	floatvalue	stringvalue	datevalue	dataintegrity	t_stamp	
11	29	NULL	NULL	NULL	192	1711645822694	
11	29	NULL	NULL	NULL	192	1711645817687	
11	29	NULL	NULL	NULL	192	1711645812683	Tag value not changing - db inserts every 5 seconds
11	29	NULL	NULL	NULL	192	1711645807678	
11	29	NULL	NULL	NULL	192	1711645802484	
11	29	NULL	NULL	NULL	192	1711645749027	
11	28	NULL	NULL	NULL	192	1711645739009	Tag changes at edge every 10 seconds
11	27	NULL	NULL	NULL	192	1711645729000	inserted as historical data after edge node reconnect
11	26	NULL	NULL	NULL	192	1711645718981	
11	25	NULL	NULL	NULL	192	1711645708970	
11	24	NULL	NULL	NULL	500	1711645634253	Tag quality bad on edge node disconnect
11	24	NULL	NULL	NULL	192	1711645629451	
11	24	NULL	NULL	NULL	192	1711645624444	
11	24	NULL	NULL	NULL	192	1711645619438	
11	24	NULL	NULL	NULL	192	1711645614432	
11	24	NULL	NULL	NULL	192	1711645609425	
11	24	NULL	NULL	NULL	192	1711645604420	Tag value not changing - db inserts every 5 seconds
11	24	NULL	NULL	NULL	192	1711645599414	
11	24	NULL	NULL	NULL	192	1711645594406	
11	24	NULL	NULL	NULL	192	1711645589399	
11	24	NULL	NULL	NULL	192	1711645584394	
11	24	NULL	NULL	NULL	192	1711645579390	
11	24	NULL	NULL	NULL	192	1711645574385	
11	24	NULL	NULL	NULL	192	1711645569378	
11	24	NULL	NULL	NULL	192	1711645564372	
11	24	NULL	NULL	NULL	192	1711645557790	
11	23	NULL	NULL	NULL	192	1711645555991	Tag value changing < Max Time Between Samples - db inserts on change
11	22	NULL	NULL	NULL	192	1711645553806	
11	21	NULL	NULL	NULL	192	1711645550165	
11	20	NULL	NULL	NULL	192	1711645549355	
11	20	NULL	NULL	NULL	192	1711645544350	
11	20	NULL	NULL	NULL	192	1711645539343	Tag value not changing - db inserts every 5 seconds
11	20	NULL	NULL	NULL	192	1711645534338	
11	20	NULL	NULL	NULL	192	1711645529332	
11	20	NULL	NULL	NULL	192	1711645524326	

Sample Mode = Periodic, Max Time Between Samples = 0, Sample Rate = 5

- Suitable only for writing historical events directly to the database
- Real-time data is written to the storage provider on the Sample Rate whilst the tag quality is Good
- Historical data is written to the storage provider using the timestamp associated with the historical data

▼ History		
History Enabled	<input checked="" type="radio"/> true	▼
Storage Provider	<input checked="" type="radio"/> MySQL	▼
Deadband Style	<input checked="" type="radio"/> Discrete	▼
Deadband Mode	<input checked="" type="radio"/> Absolute	▼
Historical Deadband	<input checked="" type="radio"/> 0.01	
Sample Mode	<input checked="" type="radio"/> Periodic	▼
Max Time Between Sampl...	<input checked="" type="radio"/> 0	
Max Time Units	<input checked="" type="radio"/> Seconds	▼
Sample Rate	<input checked="" type="radio"/> 5	
Sample Rate Units	<input checked="" type="radio"/> Seconds	▼

⚡ Resultset 1							
tagid	intvalue	floatvalue	stringvalue	datevalue	dataintegrity	t_stamp	
11	41	NULL	NULL	NULL	192	1711655211031	
11	38	NULL	NULL	NULL	192	1711655206029	Tag value changing every 2 seconds - db inserts every 5 seconds
11	35	NULL	NULL	NULL	192	1711655201028	
11	33	NULL	NULL	NULL	192	1711655194071	
11	32	NULL	NULL	NULL	192	1711655192069	
11	31	NULL	NULL	NULL	192	1711655190068	
11	30	NULL	NULL	NULL	192	1711655188067	
11	29	NULL	NULL	NULL	192	1711655186066	
11	28	NULL	NULL	NULL	192	1711655184064	Tag changes at edge every 2 seconds inserted as historical data after edge node reconnects
11	27	NULL	NULL	NULL	192	1711655182062	
11	26	NULL	NULL	NULL	192	1711655180061	
11	25	NULL	NULL	NULL	192	1711655178059	
11	24	NULL	NULL	NULL	192	1711655176057	
11	23	NULL	NULL	NULL	192	1711655174056	
11	22	NULL	NULL	NULL	192	1711655172055	
11	21	NULL	NULL	NULL	192	1711655171055	
11	18	NULL	NULL	NULL	500	1711655171020	Tag quality bad after edge node disconnect
11	20	NULL	NULL	NULL	192	1711655169053	
11	19	NULL	NULL	NULL	192	1711655167052	
11	17	NULL	NULL	NULL	192	1711655166018	Tag value changing every 2 seconds - db inserts every 5 seconds
11	15	NULL	NULL	NULL	192	1711655161017	
11	12	NULL	NULL	NULL	192	1711655156015	
11	10	NULL	NULL	NULL	192	1711655151015	

Ignition Allow Back-fill Data

Starting in Ignition 8.1.4 Ignition supports a feature to 'back-fill' historical data for [tag providers](#).

Normally if a tag is updated with a value that is older than the current value, Ignition simply throws it away. In doing so, no tag change event will ever fire and the tag will not be updated. The back-fill option allows the value to be stored to the historical database if history is enabled on that tag. However even with this option enabled, it is important to note that no tag change event will fire. So, tag change event scripts, transaction groups, and any other subsystem that relies on tag change events will not be notified of the event if it is older than the current tag value (i.e. is historical).

To enable this option, browse to the Ignition Gateway Web UI **Config** **Tag Providers** **Realtime**. Select 'edit' for the Tag Provider that will contain Reference Tags that will point to MQTT Engine tags. Scroll to the bottom and click the 'Show advanced properties' checkbox. This will show the 'Allow Back-fill Data' option. Set it to true as shown below and then click 'Save Changes'.

