

# Metric collection for Parameters

## Overview

**Metric collection** allows you to store historical information for Parameter values received from managed devices. When enabled, CONTROL automatically captures and stores data points over time, enabling trend analysis and historical reporting.

## Enabling Metric Collection

To activate metric collection for a Parameter:

1. Navigate to the [Parameter configuration screen](#)
2. Enable the  metric option

Empty values

When this option is set the system will user, process and apply empty values configured for this parameter. This option don't apply for changes applied through the API interface.

Metric

When a parameter is defined as a matric the system will store historical information of each value received from the device.

Create object

Used to activate the creation of objects on the managed devices.

Create object method:

Maximum index number

Method to be used to create the objects in the managed devices.

Custom variable name for Number of objects value:

Used when the create object method is set to 'Number of objects' and the managed device uses a non-standard variable to report that number.

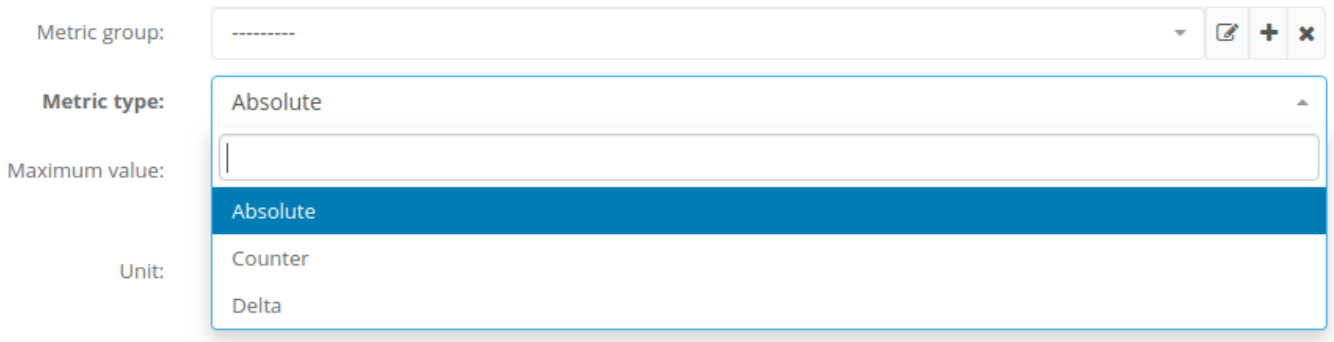
Notification:

Configure the managed device to send notifications and/or updates when this parameter changes.

**Note:** Metric collection is primarily designed for Parameters that return numeric values. For non-numeric Parameters, see the [Non-numeric Parameter Metrics](#) section below.

## Metric Types

CONTROL supports three types of metrics for Parameters: **Absolute** (default), **Counter**, and **Delta**. Each type determines how received values are processed and stored.



The screenshot shows a configuration window for a metric. It includes a 'Metric group' dropdown menu, a 'Metric type' dropdown menu with 'Absolute' selected, a 'Maximum value' input field, and a 'Unit' dropdown menu with 'Counter' selected. The 'Metric type' dropdown is open, showing 'Absolute', 'Counter', and 'Delta' as options.

## Absolute Metrics

This metric type stores the exact value received from the device after applying:

- Unit conversions (when configured)
- [Processing scripts](#) (when configured)

Use this type when you need to track the actual reported values over time (e.g., temperature, voltage, signal strength).

## Counter Metrics

This metric type stores the **rate of change** between consecutive values. The calculation is:

$$\text{stored\_value} = (\text{current\_value} - \text{previous\_value}) / \text{time\_elapsed}$$

The calculation is performed after applying:

- Unit conversions (when configured)
- [Processing scripts](#) (when configured)

Use this type for continuously incrementing counters where you want to track the rate (e.g., bytes per second, requests per minute).

## Delta Metrics

This metric type stores the **difference** between consecutive values. The calculation is:

$$\text{stored\_value} = \text{current\_value} - \text{previous\_value}$$

The calculation is performed after applying:

- Unit conversions (when configured)
- [Processing scripts](#) (when configured)

Use this type when you need to track the absolute change between readings (e.g., change in disk usage, change in user count).

# Visualization of Parameter Metrics

TBC

# Non-numeric Parameter Metrics

TBC

---

Revision #2

Created 2026-02-13 22:27:53 UTC by ipena@zequenze.com

Updated 2026-02-14 01:08:16 UTC by ipena@zequenze.com