USER MANUAL (IN-APP, COMPLETE)

Data Bridge

Online: mergenhub.envest.com.tr/docs/manual-data-bridge

ON THIS PAGE

Agents
Agent Examples & Endpoints
Sending a report to an external system (report agent)

Agents

The **Data Bridge** bridges tag values northbound to external systems. It runs as a separate Windows service (`MergenHubDataBridge`) and is fed from the core over REST (X-MergenHub-Token). Everything is an **agent** — nothing is enabled by default, including the REST Server.

Page: **Settings › Web & Connectivity › Data Bridge**. On the home page each bridge type is a separate card: **REST server** (the external system pulls), **MQTT push** and **REST push** (MergenHub sends), **Grafana** and **Power BI · Excel** (reading with an access key). Each card opens on its own page with that type's bridge list and editor; for the Grafana and Power BI pages see Grafana and Power BI.

REST Server	Inbound: an external system reads/writes tags through the `/iotgateway/browse|read|write` endpoints (industry-standard JSON `{id,s,r,v,q,t}`). If disabled, the endpoint returns 404.
MQTT Client	Outbound: publishes selected tags to a broker (`[{id,v,q,t}]`); bidirectional through a write topic. Cloud integration without opening a field port.
REST Client	Outbound: sends tag values to a REST endpoint (POST/PUT/PATCH; Basic / Bearer / API-Key authentication; periodic or on change). Can be tried without saving using **Send test request**.

Publish trigger: periodic (`publishRateMs`) or on change only (`publishOnChangeOnly`). Connection status (connected/published/last error) is monitored live on each agent's Status tab.

> **Note** API routes (`/iotgateway`), the internal exe name and setting keys keep the **IoT Gateway** identity for technical compatibility; the Windows service name is `MergenHubDataBridge` and the product name is **Data Bridge**.

WHO MANAGES IT, WHO READS THE REST SERVER ENDPOINT (3.10.1)

Seeing the Data Bridge tab and managing the agents needs the **Data Bridge** permission. When an agent is saved, every tag you selected must be open to you.

Authenticated mode (default)	The **Allow anonymous access** box starts unticked on a new agent. The endpoint asks for credentials (session or HTTP Basic): the reading account needs Connectivity: View and resource access to the tags it reads, the writing account needs **Tag write and services**. A tag the account cannot see does not appear in the list and cannot be read.
Anonymous mode	Only **Root** can tick the box (and only Root changes the tag list of an anonymous agent). The endpoint asks for no credentials: the tags in the agent's tag list (all tags if the list is empty) are read without signing in. Use it only on a closed network and with a narrow tag list.
Writing	The **Allow write** box and the **Write topic** of an MQTT agent are locked if you do not have Tag write and services. Every write made through the REST Server endpoint goes to the audit log (who or "anonymous", tag = value, client address).

> **After upgrading** Saved agents keep working as they are (an agent in anonymous mode stays anonymous). An integration account that reads all tags in authenticated mode now sees only the tags it can access: give the account resource access. When you change an agent's target (server, port, user) the saved password is not sent to the new target; enter it again.

See also Agent Examples Permissions page by page

Agent Examples & Endpoints

REST SERVER — READ/WRITE

```
# Değer oku (id = tag yolu)
GET /iotgateway/read?ids=Modbus_1.1001.Debi
→ [{"id":"Modbus_1.1001.Debi","v":42.5,"q":true,"t":1712345678}]

# Değer yaz
POST /iotgateway/write
[{"id":"Modbus_1.1001.Setpoint","v":50}]
```

MQTT CLIENT AGENT (CONFIGURATION)

```
{
  "name": "Broker1", "type": "MqttClient", "enabled": true,
  "brokerHost": "10.0.0.20", "brokerPort": 1883,
  "topic": "saha/veri", "qos": 1,
  "publishRateMs": 1000, "publishOnChangeOnly": true,
  "tags": ["Modbus_1.1001.Debi", "Modbus_1.1001.Basinc"],
  "writeTopic": "saha/yaz"
}
```

REST CLIENT AGENT — ENVELOPE SENT & AUTHENTICATION

```
PATCH https://sunucu.example.com/api/veri
Content-Type: application/json
X-API-Key: ********        ← Auth: API Key (başlık adı seçilebilir)

{"timestamp": 1723094400000,
 "values": [
   {"id":"Kanal.Cihaz.Debi","v":42.5,"q":true,"t":1723094399870}
 ]}
```

Authentication types: **Basic** (user + password), **Bearer** (token), **API Key** (custom header) — you do not need to type the header by hand; additional headers can also be entered line by line. The **"How does it work? + example"** section in the form generates the equivalent `curl` command from the values you entered.

> **Tip Send test request** sends a single request to the target with the real envelope WITHOUT saving the settings; the HTTP status code, duration and response body appear on screen — you can verify the integration in minutes.

> **Note** The publish payload has the form `[{"id","v","q","t"}]`. If `writeTopic` is empty, writing is disabled. For TLS use port 8883 + a CA/client certificate.

Sending a report to an external system (report agent)

Instead of tag values an agent can also send the table of an **Excel report**: the ERP, BI or other software does not have to come and read the report; MergenHub sends it itself at the time you set. The report is recalculated with current values at every send. The data sent is identical to the form read with the access key of the report.

Location: **Settings** → **Data Bridge** tab.

SETTING UP: STEP BY STEP

1. First create the report: the **Reports** → **Counter tracking lists** tab or the **Counters & Metrics** → **Excel reports** tab (see Counter tracking list).
2. On the **Data Bridge** tab press **+ Add Agent**; enter a **Name** and as **Type** choose **REST Client** (sends to an address) or **MQTT Client** (publishes to a broker topic). Fill in the target as usual: for REST the **URL**, method and authentication; for MQTT the broker, port and **Topic**.
3. In the **What to send** box choose **Report (Excel report)**. The report fields open below it.
4. In the **Report** box choose the report to send.
5. **Format**: **Flat records** (recommended; each row is "column key → value", the easiest to process in an ERP) or **Table** (columns + rows).
6. **Each row** and **Range**: leave **Report default** to use the report's own period, or change it for this agent (Hour, Day, Week, Month, Year; This month, Last month, This year, Last year, Last … periods). When **Last … periods** is selected, enter the number of periods in the box next to it.
7. **When to send** → **When**: **At an interval** (**Interval (minutes)**; 1 minute at the most frequent), **Every hour** (**Minutes past the hour**: 0 = on the hour, 5 = every hour at :05) or **Every day** (**Time**; server local time).

8. The **Preview the data to send** button shows the data that would go out (row count, size and content) in a window; it sends nothing anywhere. On a REST Client agent, **Send test request** really sends the same data to the target once.
9. Press **Save**. The **Status** tab shows **Last publish**, **Next send** and, if any, **Last error**.

EXAMPLES

Every night, the daily values of yesterday and today	**Each row**: Day · **Range**: Last … periods, 2 · **When**: Every day, 00:05
Every hour, the hourly values of the last 24 hours	**Each row**: Hour · **Range**: Last … periods, 24 · **When**: Every hour, 0
Every fifteen minutes, this month's summary	**Each row**: Report default · **Range**: This month · **When**: At an interval, 15

DATA SENT

```
{ "timestamp": 1794480000000, "agent": "ERP gece aktarımı",
  "report": { "name": "Sayaç Takip Listesi", "grain": "Day", "start": …, "end": …,
    "columns": [ { "key": "counter:Kuyu1:Total", "header": "Fark", "unit": "m³", "bands":
["ÜRETİM","Kuyu 1"] }, … ],
    "rows": [ { "start": …, "end": …, "startIso": "2026-01-10T00:00:00+03:00",
                "values": { "counter:Kuyu1:EndIndex": 147875, "counter:Kuyu1:Total": 312.5,
"sum:Fabrika/ÜRETİM:Total": 540 } }, … ],
    "totals": { … } } }
```

`timestamp` is the moment of sending (Unix ms), `agent` is the agent's name, `report` is the report itself. Column keys are stable (they stay the same as long as the counter code does not change). In the **Table** format `values` is an array in column order.

On change only	When ticked, the report is not sent again if it is identical to the previous one sent (the timestamp does not count).
If sending fails	It is retried one minute later; an hourly or daily agent does not wait for the next hour / day. The reason is written in the **Last error** row on the **Status** tab.
Time that passes while the service is stopped	A missed send is not made up later; when the service starts, the current report goes out at the next time. Keeping the range wide (e.g. Last … periods, 2) also carries the day in between.
On saving	An agent that sends **At an interval** sends once as soon as it is saved; an **Every hour** or **Every day** agent waits for its first time.
Tags tab	Not used in an agent that sends a report.
Limits	The report limits apply: at most 800 rows (periods), 160 columns. Mind the message size limit of your MQTT broker; narrow the range for a large report.
Permission	An agent can only send a report that the user who saved it can see; a user with restricted resource access cannot send out through an agent a report they cannot see.
Data Bridge service	It reads the report from the core with the local internal service token. If **Last error** mentions the "internal service token", the service is not using the same data folder as the core (it cannot find the token file).

See also Agents · Counter tracking list and access key