AIR. NOTHING ELSE.

INTEGRATION GUIDE

MAS-100 Sirius®/ Sirius Ex Microbial Air Sampler



English

THANK YOU VERY MUCH!

Dear customer

congratulations and thank you very much for choosing a product from MBV AG. We are pleased to make a contribution to your success.



MBV AG develops and produces precision instruments for active and passive microbial air and gas monitoring for more than 20 years. Our uncompromising quest for the best possible quality has convinced globally leading companies to rely on our solutions for their most demanding applications. Our superior product quality is backed by global availability of application and service support.

For best performance of your product please read this instruction manual carefully. The manual will provide you with the necessary information for correct and safe use of your new instrument.

If you need additional technical or application information, do not hesitate to contact us at welcome@mbv.ch or find more information on our homepage www.mbv.ch.

MANUFACTURER

MBV AG
Industriestrasse 9
CH8712 Stäfa
Switzerland

NOTE:

Up-to-date product information – such as Service Information Letters (SIL) – are available in the expert center section of the MBV AG homepage: www.mbv.ch

Moreover, important product after-sales information about changes, issues, instructions as well as news are provided in the corresponding expert center of MAS-100 Sirius and MAS-100 Sirius Ex.

DOCUMENT HISTORY

VERSION	RELEASED	CHANGE HISTORY
1.0	2026-06-02	Initial release (LAR/CKE)

TRADEMARKS

The following product names are used for identification purposes only. They remain the exclusive property of their respective owners.

MBV AG	
MAS-100 Sirius®	Switzerland and other countries

Copyright © 2026 MBV AG. All rights reserved.

This document may not be reproduced, in whole or in part, in any form or by any means electronic or mechanical (including photocopying, recording, or any other process), without the written permission of MBV AG, Switzerland. Information contained in this document may be subject to change without notice.

TABLE OF CONTENTS

	Manufacturer	2
	Document history	3
	Trademarks	3
1.	**GENERAL INFORMATION**	**7**
1.1.	Scope of this document	7
1.2.	Target Group	7
1.3.	Applicable Instrument	7
1.4.	Applicable API Version	7
1.5.	Related documents	7
1.6.	Contact and support	7
2.	**SYSTEM OVERVIEW**	**8**
2.1.	Instrument Description	8
2.2.	Key Capabilities Relevant to Integration	8
3.	**NETWORK AND CONNECTIVITY**	**9**
3.1.	Network interfaces	9
3.2.	Wifi	9
3.3.	IP Configuration	9
3.4.	Base URL	9
3.5.	Network Security Recommendations	9
4.	**INTERFACES AND COMMUNICATION PROTOCOLS**	**10**
4.1.	Protocol Stack Summary	10
4.2.	Content type	10
4.3.	Port Assignment	10
4.4.	No additional protocols	10
5.	**REST API**	**10**
5.1.	API Specification	10
6.	**DATA CONTENT AND FLOW**	**12**
6.1.	Core Workflow: Initiating a Sampling Run	12
6.2.	Sampling API	12
	6.2.1. Check Device Readiness	12
	6.2.2. Discover Available Sampling Parameters	12
	6.2.3. Start a Sampling Run	13

6.2.4. Abort a running sampling 13
6.2.5. Retrieve the Last Sampling Resul 13
6.2.6. QR-Code Confirmation (Attended Mode) 14
6.3. Retrieving Results and Reports 14
6.3.1. Query the Audit-Trail Report 14
6.3.2. Export Signed Report Archive 15
6.3.3. Delete All Report Entries 15
6.3.4. Retrieve Archived Measurement QR Code 16
6.4. Device State and Monitoring 16
6.4.1. Device and Version Information 16
6.4.2. Active Hardware Errors 16
6.4.3. Device Clock 16
6.4.4. Reset Error Screen 16
6.5. Settings (Read-Only Reference) 16
6.5.1. Read a Setting 16
6.5.2. Sequential Sampling Mode (SQS) 17

7. DATA FORMAT AND STRUCTURE 17
7.1. Primary Format: JSON 17
7.2. Sampling Result (JSON) 17
7.3. Signed Report Archive (ZIP / XML) 18
7.4. QR Code (SVG) 18
7.5. Timestamps and Timezone Handling 18

8. CYBERSECURITY 18
8.1. Encryption in Transit 19
8.2. Network Isolation 19
8.3. Audit Trail integrity 19
8.4. Concurrency 19

9. MULTI-INSTRUMENT MESH DISCOVERY 19
9.1. Discover Neighbours 19
9.2. Data Store Consistency 19

10. ERROR HANDLING 20
10.1. HTTP Status Codes 20
10.2. Error Message Format 20
10.3. Hardware Error Structure 20

11. POLLING AND CONCURRENCY PATTERNS 20
11.1. Busy Detection 20

11.2. Recommended Poll Intervals 21

12. ENDPOINT REFERENCE 21

12.1. Device Information & Status 21

12.2. Sampling 22

12.3. Reports & Archive 22

12.4. Settings (Read) 22

12.5. Multi-Instrument Mesh 22

1. GENERAL INFORMATION

1.1. SCOPE OF THIS DOCUMENT

This document provides a comprehensive guide to use the hardware and software of the following instruments for active microbial air sampling.

It addresses use cases involving integration with a LIMS/EM system, data retrieval, and remote sampling.

1.2. TARGET GROUP

Software integrators connecting a superordinate system to the MAS-100 Sirius and MAS-100 Sirius Ex microbiological air samplers via its REST API. This edition covers the interfaces required for automated sampling, result retrieval, and report management. Service, calibration, and anemometer interfaces are out of scope for this document.

1.3. APPLICABLE INSTRUMENT

This LIMS integration guide is applicable to MAS-100 Sirius and MAS-100 Sirius Ex air samplers manufactured by MBV AG, Switzerland.

1.4. APPLICABLE API VERSION

The current release as per SW V2.0.5

1.5. RELATED DOCUMENTS

The following related documents provide additional information and are available via the dedicated Sirius download page www.mbv.ch/sirius/files:

DOCUMENT	SCOPE
Global Safety Datasheet	Important safety information related to the use of the instrument.
User Manual MAS-100 Sirius®/ Sirius Ex	Information for operation of the instrument.
IQ/OQ MAS-100 Sirius®/ Sirius Ex (MBV article number 201938)	Installation qualification and operational qualification of the instrument
Declaration of Conformity (DoC) for MAS-100 Sirius®/ Sirius Ex	Information on compliance with regulatory requirements.

1.6. CONTACT AND SUPPORT

For integration support please contact the instrument manufacturer:

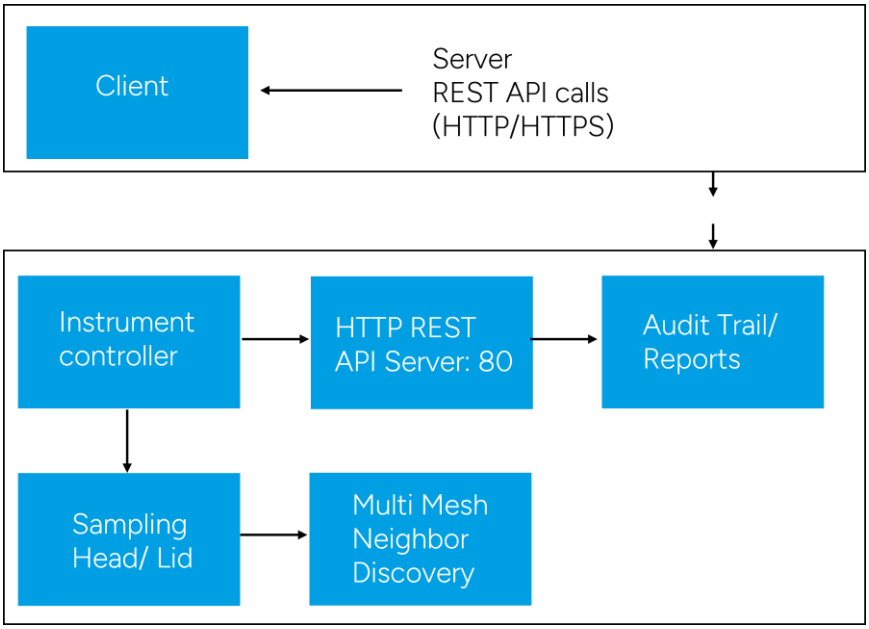
MBV AG
Industriestrasse 9
CH8712 Stäfa
Switzerland

Customer Support: www.mbv.ch/en/services/support/

2. SYSTEM OVERVIEW

2.1. INSTRUMENT DESCRIPTION

The Sirius is a precision microbiological air sampler designed for cleanroom monitoring and environmental control programs. It draws a calibrated volume of air through a settling plate at a precisely controlled flow rate, enabling accurate colony-forming-unit (CFU) measurements.



2.2. KEY CAPABILITIES RELEVANT TO INTEGRATION

CAPABILITY	DETAILS
Sampling volumes	50, 100, 250, 500, 1 000 L (configurable up to 5 000 L)
Default flow rates	100, 200 SLPM (calibrated per lid type; available points vary by instrument)
Lid types	Lid A, Lid B
Pre-sampling delay	0–3000 seconds
Result metadata	Timestamps (UTC + local TZ), serial numbers, calibration point, flow mode, operator identity, sampling location, plate barcode
Report export	Digitally signed ZIP/XML for audit trail
Multi-instrument mesh	Instruments share a data store over LAN
API interface	HTTP REST on port 80; all payloads JSON

3. NETWORK AND CONNECTIVITY

3.1. NETWORK INTERFACES

The instrument has multiple network interfaces:

INTERFACE	TYPE	TYPICAL USE
usb0 / usb1	USB-Browser User Interface	Service / bench use
wlan0	Wi-Fi	Mobile access

3.2. WIFI

The wlan0 interface provides wireless connectivity for mobile access scenarios. Wi-Fi is suitable for attended or portable deployments and may also be used for super-ordinate connections.

NOTE: Consult the instrument's hardware and configuration manual for supported standards and security mode options.

3.3. IP CONFIGURATION

IP addresses are assigned per interface. Retrieve current addresses with:

```
GET /api/information
```

```
{
  "deviceName": "Sirius",
  "serialNumber": "SRS-00123",
  "hardwareVersion": "2.1",
  "baseSwVersion": "3.2.1",
  "instrumentLabel": "Room A Sampler",
  "publicSiteKey": "AAABBBCCC...base64...",
  "networkInterfaces": [
    { "name": "wlan0", "ip": "192.168.1.51", "mask": "255.255.255.0",
"hwAddress": "AA:BB:CC:DD:EE:01" }
  ]
}
```

3.4. BASE URL

```
http://<instrument-ip>/       (port 80, plain HTTP)
```

All functional endpoints are prefixed with /api/.

3.5. NETWORK SECURITY RECOMMENDATIONS

- Assign the instrument to a dedicated VLAN or network segment accessible only from the server.
- Use firewall rules to restrict inbound access to ports 80 and 9000 to the server IP.
- Treat the instrument segment as untrusted from the main network's perspective. Traffic flowing out of the instrument VLAN/SSID toward the corporate network is the higher-risk direction.

4. INTERFACES AND COMMUNICATION PROTOCOLS

4.1. PROTOCOL STACK SUMMARY

LAYER	TECHNOLOGY
Transport	TCP/IP over wired Ethernet or Wi-Fi
Application protocol	HTTP/1.1 (port 80)
API style	REST — resource-oriented, stateless between requests
Data encoding	JSON (application/json) for all structured data
Scalar responses	text/plain for simple values and acknowledgements
Binary responses	application/zip (signed report archive), image/svg+xml (QR codes)

4.2. CONTENT TYPE

All request bodies must use Content-Type: application/json. Responses use:

- application/json — structured result objects
- text/plain — simple acknowledgements and scalar values (e.g., "ready", "true")
- application/zip — signed report export
- image/svg+xml — measurement QR code

4.3. PORT ASSIGNMENT

PORT	PROTOCOL	USAGE
80	HTTP	Default; used when TLS is not configured

4.4. NO ADDITIONAL PROTOCOLS

The Sirius instrument exposes its full control and data interface exclusively via HTTP REST. There are no MQTT, OPC-UA, MODBUS, or proprietary binary protocols exposed to integrators.

Active Directory authentication requires network connectivity. During temporary network outages, users cannot log in using Active Directory credentials until the connection is restored.

5. REST API

5.1. API SPECIFICATION

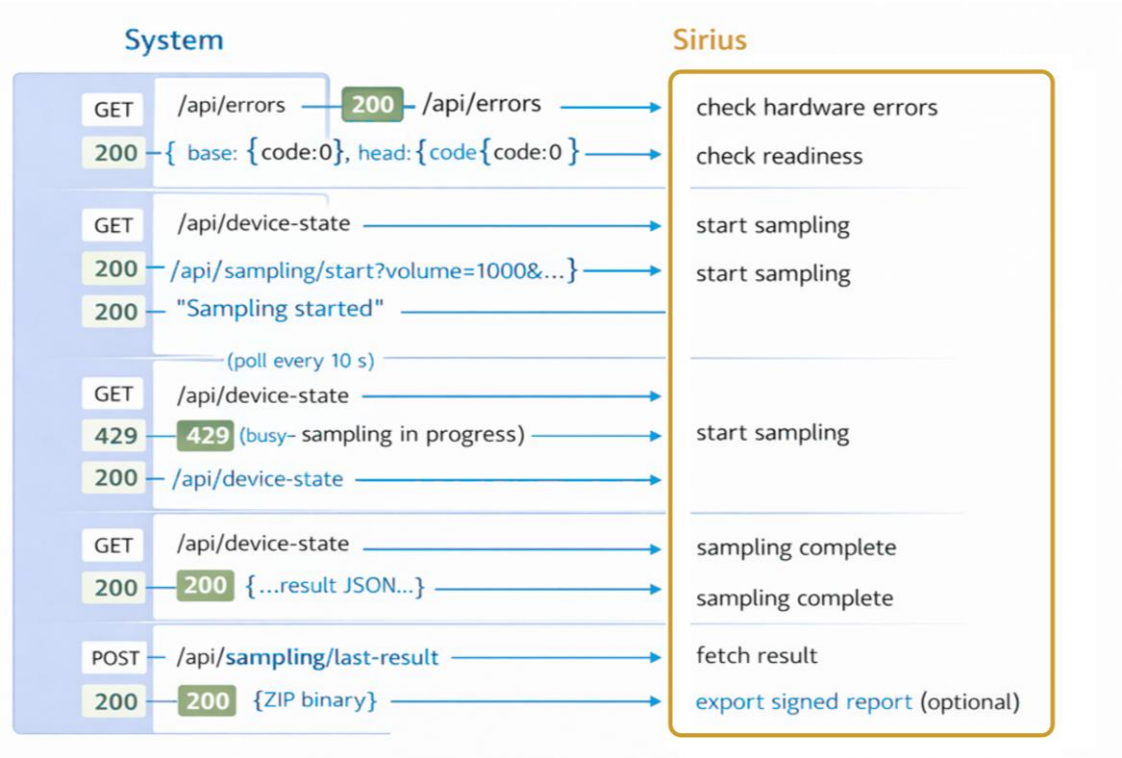
The Sirius REST API endpoints are documented in this guide and the Complete Endpoint Reference.

Summary of API characteristics:

PROPERTY	VALUE
Style	REST over HTTP
Base path	/api/
Authentication	None — the instrument is operated in userless mode for remote control
Request body	application/json
Response body	application/json, text/plain, application/zip, image/svg+xml
HTTP methods used	GET, POST, PUT
Versioning	No explicit version prefix; tied to firmware version
Pagination	limit + offset parameters on report filter endpoint
Error format	HTTP status code + ErrMsg:<description> plain text body

6. DATA CONTENT AND FLOW

6.1. CORE WORKFLOW: INITIATING A SAMPLING RUN



Operating mode: The instrument is configured for userless remote control. No login, bearer token, or admin role is used by the integrating system. All endpoints shown in this guide are called without an Authorization header.

6.2. SAMPLING API

6.2.1. CHECK DEVICE READINESS

```
GET /api/device-state
```

RESPONSE	MEANING
200 "ready"	Instrument is idle; sampling can be started
429	Instrument is busy (sampling, calibration, self-test, etc.)
503	Head not connected or instrument not powered

6.2.2. DISCOVER AVAILABLE SAMPLING PARAMETERS

Before starting a sampling run, query the instrument for valid lid types and flow rates. The available options depend on the calibration performed on the specific instrument.

```
GET /api/flow-calibration/calibration-points
```

```
[
  { "slot": 0, "lidType": "Lid A", "operationalUnit": "SL/min", "targetFlowRate": 100.0 },
  { "slot": 1, "lidType": "Lid B", "operationalUnit": "SL/min", "targetFlowRate": 200.0 },
]
```

Populate the sampling-parameter drop-downs from this response. Do not hardcode lid or flow_rate values.

6.2.3. START A SAMPLING RUN

```
GET /api/sampling/start?delay=<s>&vol-
ume=<SL>&flow_rate=<SL/min>&lid=<type>
```

All parameters are optional; omitting them applies the instrument's configured defaults.

PARAMETER	TYPE	DESCRIPTION	EXAMPLE
delay	integer (s)	Pre-sampling idle period (0–3 000)	60
volume	integer (SL)	Total volume to sample	1000
flow_rate	integer (SL/min)	Target flow rate	100
lid	string	Lid type identifier from calibration points	"Lid A"

Responses:

CODE	BODY	MEANING
200	"Sampling started"	Run initiated
400	"ErrMsg:Invalid delay" etc.	Parameter validation failure
429	"ErrMsg:Instrument is busy"	Another operation in progress
500	error string	Internal error

6.2.4. ABORT A RUNNING SAMPLING

```
GET /api/abort-sampling
```

Returns 200 "ready" when the abort is accepted. The instrument immediately returns to idle.

6.2.5. RETRIEVE THE LAST SAMPLING RESUL

```
GET /api/sampling/last-result
```

Returns a JSON object; see Appendix A for the full field reference.

Critical fields for ingestion:

FIELD	TYPE	DESCRIPTION
startTime	int64 (Unix ms, UTC)	Sampling start timestamp
endTime	int64 (Unix ms, UTC)	Sampling end timestamp
localStartTime	string (ISO-8601)	Start time in the instrument's configured timezone
timezone	string	IANA timezone name (e.g. "Europe/Zurich")
targetVolume	integer	Requested volume (SL)
actualVolume	float	Volume actually sampled (SL)
flowRate	float	Mean achieved flow rate (SL/min)
lidType	string	Lid identifier used
calibrationSlot	integer	Calibration point index
plateSerialNumber	string	Nutrient plate barcode
lidSerialNumber	string	Lid barcode (if scanned)
samplingLocation	string	Location label
baseSerialNumber	string	Instrument base serial number
headSerialNumber	string	Instrument head serial number
errorCode	integer	0 = success; non-zero = failure
passed	boolean	Overall sampling validity

Sample ID: The baseSerialNumber combined with the startTime field for the UTC start time point is the primary key for linking a Sirius sampling result to a sample record.

6.2.6. QR-CODE CONFIRMATION (ATTENDED MODE)

Some site configurations require an operator to confirm a QR code on the instrument display before the run is validated. If the instrument is waiting for this:

```
GET /api/confirm-sampling-qr
```

Returns 200 "ready" when confirmed.

6.3. RETRIEVING RESULTS AND REPORTS

6.3.1. QUERY THE AUDIT-TRAIL REPORT

```
POST /api/report/filter
Content-Type: application/json

{
  "language":       "en",
  "fromTime":       1700000000000,
  "toTime":         1700086400000,
  "categories":     0,
  "firstName":      "",
  "lastName":       "",
  "uniqueUserName":"",
  "userTypes":      0,
  "limit":          100,
```

```
  "offset":         0
}
```

Request fields:

FIELD	TYPE	DESCRIPTION
language	string	Locale for translated message text ("en", "de", "fr")
fromTime	int64 (Unix ms)	Inclusive range start
toTime	int64 (Unix ms)	Inclusive range end
categories	int64 (bitmask)	Event category filter; 0 = all categories
userTypes	int64 (bitmask)	User-role filter; 0 = all roles
limit	int64	Page size
offset	int64	Page offset for pagination

Response:

```
{
  "entries": [
    {
      "id": 4201,
      "time": 1700042000000,
      "localtime": "2023-11-15T10:13:20+01:00",
      "timezone": "Europe/Zurich",
      "category": 1,
      "userFirstName": "Jane",
      "userLastName": "Smith",
      "uniqueUserName": "jsmith",
      "userType": 2,
      "message": "Sampling completed successfully",
      "parameterType": 0,
      "stringParameter": ""
    }
  ]
}
```

Sample ID: Here, the localtime is the primary key for linking a Sirius sampling result to a sample record.

6.3.2. EXPORT SIGNED REPORT ARCHIVE

```
POST /api/report/export
Content-Type: application/json

{ "language": "en" }
```

Returns a binary ZIP file containing a digitally signed XML report. The signature can be verified using the instrument's public site key (publicSiteKey from /api/information). The Content-Disposition header provides the suggested filename.

Save this file in your document store before calling the delete endpoint.

6.3.3. DELETE ALL REPORT ENTRIES

```
POST /api/report/delete
```

Permanently deletes all entries. This action is irreversible — always export first.

6.3.4. RETRIEVE ARCHIVED MEASUREMENT QR CODE

```
GET /api/measurement-archive/qr-code/{index}
```

Returns the QR code for the archived sampling at position {index} as an SVG string. The archive stores up to 100 recent records.

6.4. DEVICE STATE AND MONITORING

6.4.1. DEVICE AND VERSION INFORMATION

```
GET /api/information
```

Returns identity, software versions, and network addresses. Call this once at startup to cache the instrument serial number.

6.4.2. ACTIVE HARDWARE ERRORS

```
GET /api/errors
```

```
{
  "base": { "code": 0,    "description": "No error" },
  "head": { "code": 1024, "description": "Head communication timeout"
}
}
```

Error code 0 means no active error. Log non-zero error codes; they may prevent sampling from starting. You may check this endpoint before every automated sampling run.

6.4.3. DEVICE CLOCK

```
GET /api/device-manager/clock
```

Returns the current UTC Unix timestamp (integer, seconds). Use this to verify the instrument clock matches the server time before timestamping results.

6.4.4. RESET ERROR SCREEN

```
GET /api/error/reset
```

Returns 200 "ready" when the error screen has been dismissed and the instrument returns to idle.

6.5. SETTINGS (READ-ONLY REFERENCE)

Settings follow a hierarchical key path. For integration, read-only access to sampling defaults is sufficient.

6.5.1. READ A SETTING

```
GET /api/settings/read/{key}
```

Common relevant keys:

KEY	DESCRIPTION
sampling/defaultVolume	Default sampling volume (L)
sampling/defaultDelay	Default pre-sampling delay (s)
sampling/defaultLid	Default lid type
sampling/defaultFlowRate	Default target flow rate
sampling/sqsEnabled	Whether Sequential Sampling mode is active
sampling/sqsCycles	Number of SQS cycles when enabled
locale/timezone	IANA timezone applied to local timestamps
instrument/label	Human-readable instrument label

6.5.2. SEQUENTIAL SAMPLING MODE (SQS)

When SQS is enabled, a single call to /api/sampling/start triggers multiple consecutive sampling cycles. The actualVolume in the result reflects one cycle; the total sampled volume is actualVolume × sqsCycles. The superordinate system should read sampling/sqsEnabled and sampling/sqsCycles to interpret results correctly.

7. DATA FORMAT AND STRUCTURE

7.1. PRIMARY FORMAT: JSON

All structured API request and response bodies use JSON (application/json). There is no XML request/response interface for the API itself.

7.2. SAMPLING RESULT (JSON)

The primary data object returned by GET /api/sampling/last-result is a flat JSON object. Example (abbreviated):

```
{
  "startTime": 1700042000000,
  "endTime":   1700042360000,
  "localStartTime": "2023-11-15T10:13:20+01:00",
  "localEndTime":   "2023-11-15T10:19:20+01:00",
  "timezone": "Europe/Zurich",
  "targetVolume": 1000,
  "actualVolume": 999.8,
  "flowRate": 100.0,
  "lidType": "Lid A",
  "calibrationSlot": 0,
  "plateSerialNumber": "PLATE-2023-001234",
  "lidSerialNumber": "LID-A-5678",
  "samplingLocation": "Room A",
  "samplingNote": "",
  "baseSerialNumber": "SRS-00123",
  "operatorFirstName": "Jane",
  "operatorLastName": "Smith",
  "operatorUserName": "jsmith",
  "errorCode": 0,
  "headWarnings": [],
  "passed": true,
  "ambientPressure": 101325.0,
  "ambientTemperature": 21.4,
  "ambientHumidity": 45.2
}
```

Also see user manual MAS-100 Sirius chapter 3.4 for the complete field reference.

7.3. SIGNED REPORT ARCHIVE (ZIP / XML)

The export endpoint (POST /api/report/export) returns a ZIP archive containing a digitally signed XML document. This is the audit-trail format for regulated environments (e.g., GMP 21 CFR Part 11 / Annex 11).

PROPERTY	VALUE
Container format	ZIP
Document format	XML
Signature	Digital signature
Verification	Via browser interface
Suggested filename	Provided in Content-Disposition response header

Always store the signed ZIP in the superordinate system before deleting on-instrument records. The signature can be used to demonstrate data integrity during audits.

7.4. QR CODE (SVG)

The GET /api/measurement-archive/qr-code/{index} endpoint returns an SVG string representing the measurement QR code. This can be embedded in the superordinate system or displayed in a browser.

7.5. TIMESTAMPS AND TIMEZONE HANDLING

FIELD PATTERN	FORMAT	NOTES
*Time (e.g. startTime)	int64, Unix milliseconds, UTC	Use for absolute time calculations and storage
local*Time (e.g. localStartTime)	ISO-8601 string with UTC offset	Use for display and reporting in local time
timezone	IANA timezone name (e.g. "Europe/Zurich")	Instrument's configured timezone

Always store the raw UTC Unix millisecond values. Convert to local time for display using the timezone field.

8. CYBERSECURITY

For more information, please consult the user manual of the instrument.

8.1. ENCRYPTION IN TRANSIT

SCENARIO	RECOMMENDATION
Isolated lab bench / development	HTTP (port 80) acceptable only if network access is physically restricted
Wi-Fi deployments	TLS is used only in the enterprise WiFi setting

Sirius is intended for deployment within a trusted and protected network. In this architecture, network-level controls sufficient as a security measure.

8.2. NETWORK ISOLATION

The instrument is operated in userless mode for remote control — the REST API is open, so network-level controls are the primary protection:

- Restrict network access to the instrument's HTTP port to the server IP using firewall rules or a dedicated VLAN.
- Do not expose the instrument API directly to untrusted networks or the internet.
- Treat the instrument segment as untrusted from the main network's perspective.

8.3. AUDIT TRAIL INTEGRITY

- The instrument's audit trail records sampling runs, system events, and configuration changes.
- The signed export (POST /api/report/export) can be verified using the instrument's browser user interface.
- Do not delete the on-instrument report without first exporting and verifying the signature.
- The audit trail supports compliance with GMP requirements (21 CFR Part 11 / EU GMP Annex 11).

8.4. CONCURRENCY

PROPERTY	GUIDANCE
Concurrent operations	One operation executes at a time; use a queue or mutex to serialise requests

9. MULTI-INSTRUMENT MESH DISCOVERY

In environments with multiple Sirius instruments on the same network, each instrument maintains awareness of its neighbours.

9.1. DISCOVER NEIGHBOURS

```
GET /api/known-neighbors
```

Returns a list of instruments visible on the LAN. Use this to enumerate all instruments starting from a single known IP, rather than relying on a static configuration.

9.2. DATA STORE CONSISTENCY

```
GET /api/get-hash
```

Returns a hash of the shared instrument data store. Compare hashes across instruments to detect synchronisation inconsistencies.

```
GET /api/get-all-store-elements
```

Returns all elements from the shared data store. Useful for cross-instrument data reconciliation.

10. ERROR HANDLING

10.1. HTTP STATUS CODES

CODE	MEANING	RECOMMENDED ACTION
200	Success	Process response normally
400	Bad request (invalid parameter)	Log the ErrMsg: body; do not retry without fixing the request
404	Not found (e.g. no previous result)	Handle gracefully — no result yet
429	Instrument is busy	Retry after polling interval (see Section 11)
500	Internal server error	Log; alert operator; do not retry automatically
501	Not implemented (feature absent on this variant)	Skip gracefully
503	Service unavailable (head disconnected / powered off)	Alert operator; retry when available

10.2. ERROR MESSAGE FORMAT

Most 400 and 500 responses return plain text prefixed with "ErrMsg:":

```
ErrMsg:Invalid volume
ErrMsg:Instrument is busy
ErrMsg:Head communication timeout
```

Parse the substring after "ErrMsg:" for user-facing messages.

10.3. HARDWARE ERROR STRUCTURE

```
{
  "base": { "code": 1024, "description": "Battery critically low" },
  "head": { "code": 0,    "description": "No error" }
}
```

Code 0 always means no error. Check /api/errors before each automated run; block the run if either base.code or head.code is non-zero.

11. POLLING AND CONCURRENCY PATTERNS

11.1. BUSY DETECTION

The primary mechanism for detecting a busy instrument is the 429 response from GET /api/device-state. Implement a wait loop:

```
import time, requests

def wait_for_ready(instrument_ip, timeout_s=3600, poll_s=10):
    deadline = time.time() + timeout_s
    while time.time() < deadline:
        r = requests.get(f"http://{instrument_ip}/api/device-state")
        if r.status_code == 200 and r.text.strip() == "ready":
            return True
        if r.status_code not in (200, 429):
            raise RuntimeError(f"Unexpected {r.status_code}:
{r.text}")
        time.sleep(poll_s)
    return False  # timed out
```

11.2. RECOMMENDED POLL INTERVALS

PHASE	INTERVAL
Waiting for sampling to complete	10 s
Periodic instrument health check	60 s

12. ENDPOINT REFERENCE

All endpoints below are called without authentication. The instrument is operated in userless mode for remote control.

12.1. DEVICE INFORMATION & STATUS

METHOD	PATH	DESCRIPTION
GET	/api/information	Device identity, versions, network interfaces
GET	/api/errors	Active hardware errors (base + head)
GET	/api/device-state	Instrument ready state
GET	/api/error/reset	Dismiss error screen and return to idle
GET	/api/error/shut-down	Shut down from error screen
GET	/api/device-manager/refresh	Signal UI to reload
GET	/api/device-manager/clock	Current UTC Unix timestamp
GET	/api/device-manager/time-zones	List available IANA timezone strings

12.2. SAMPLING

METHOD	PATH	DESCRIPTION
GET	/api/device-state	Check if instrument is idle and ready
GET	/api/flow-calibration/calibration-points	List valid lid types and flow rates
GET	/api/sampling/start	Start a sampling run (query: delay, volume, flow_rate, lid)
GET	/api/abort-sampling	Abort ongoing sampling
GET	/api/sampling/last-result	Most recent sampling result
GET	/api/confirm-sampling-qr	Confirm QR-code prompt on display

12.3. REPORTS & ARCHIVE

METHOD	PATH	DESCRIPTION
POST	/api/report/filter	Query audit-trail entries with filters and pagination
POST	/api/report/export	Export signed ZIP/XML report archive
POST	/api/report/delete	Delete all report entries (irreversible)
GET	/api/measurement-archive/qr-code/{index}	Get archived sampling QR code as SVG

12.4. SETTINGS (READ)

METHOD	PATH	DESCRIPTION
GET	/api/settings/read/{key}	Read current setting value by key path
GET	/api/settings/original/read/{key}	Read factory/calibration default value
GET	/api/settings/sealed-state	Metrological seal state
GET	/api/settings/calibration-validity	Days remaining until calibration expiry

12.5. MULTI-INSTRUMENT MESH

METHOD	PATH	DESCRIPTION
GET	/api/known-neighbors	Instruments visible on the local network
GET	/api/get-hash	Hash of the shared instrument data store
GET	/api/get-all-store-elements	All elements in the shared data store

MBV AG makes every effort to include accurate and up-to-date information within this publication; however, it is possible that omissions or errors might have occurred. Information contained in this publication regarding device applications and the like is provided only for your convenience, MBV AG cannot make any representations or warranties, expressed or implied, written or oral, statutory or otherwise as to the accuracy or completeness of the information provided in this publication. Changes in this publication can be made at any time without notice. All mentioned trademarks are protected by law. For technical details and detailed procedures of the specifications provided in this document please contact your MBV representative.

All mentioned trademarks are protected by law. In general, the trademarks and designs referenced herein are trademarks, or registered trademarks of MBV AG, Staefa, Switzerland. Product names and company names that are not contained in the list but are noted herein may be the trademarks of their respective owners.

For disclaimer please visit: https://www.mbv.ch/en/about-us/imprint/

AIR. NOTHING ELSE.

MBV AG, Industriestrasse 9, CH-8712 Stäfa, T +41 44 928 30 80, welcome@mbv.ch, www.mbv.ch
© 2026 MBV AG Switzerland, all rights reserved