Industrial Ethernet Training

Network Automation and Monitoring with Weidmueller Industrial Ethernet Devices via SNMP

Abstract:
This application note explains how to configure Weidmueller Industrial Ethernet devices for SNMP-based management using Ansible and how to implement comprehensive monitoring with Zabbix. It provides a structured workflow for automated configuration, secure SNMP operation, and centralized supervision of network devices in industrial environments.

Hardware reference

No.	Component name	Article No.	Hardware / Firmware version
1	IE-Training Kit-01	2874670000	
2			
3			

IE-Training Kit content

No.	Component name	Article No.	Hardware / Firmware version
1	IE-SR-4TX	2751270000	v2.3.1
2	IE-SW-AL08M-8TX	2682280000	v1.18
3	IE-SW-AL05LM-5TX	2682250000	v1.20
4			

Software reference

No.	Software name	Software version
1	Ansible	v2.7.14
2	Zabbix	v7.4
3	WSL	v2.6.3
4	net-snmp	v5.9.1

File reference

No.	Name	Description	Version
1	IE-SW-AL08M-8TX_v118_2026-07-02.mib	SNMP private MIB	1.18
2	IE-SW-AL05LM-5TX_v120_2026-06-08.mib	SNMP private MIB	1.20

Contact
Weidmüller Interface GmbH & Co. KG
Klingenbergstraße 26
32758 Detmold, Germany
www.weidmueller.com

For any further support please contact your local sales representative:
https://www.weidmueller.com/countries

Content

1 Warning and Disclaimer

Warning
Controls may fail in unsafe operating conditions, causing uncontrolled operation of the controlled devices. Such hazardous events can result in death and / or serious injury and / or property damage. Therefore, there must be safety equipment provided / electrical safety design or other redundant safety features that are independent from the automation system.

Disclaimer
This Application Note / Quick Start Guide / Example Program does not relieve you of the obligation to handle it safely during use, installation, operation and maintenance. Each user is responsible for the correct operation of his control system. By using this Application Note / Quick Start Guide / Example Program prepared by Weidmüller, you accept that Weidmüller cannot be held liable for any damage to property and / or personal injury that may occur because of the use.

Note
The given descriptions and examples do not represent any customer-specific solutions, they are simply intended to help for typical tasks. The user is responsible for the proper operation of the described products. Application notes / Quick Start Guides / Example Programs are not binding and do not claim to be complete in terms of configuration as well as any contingencies. By using this Application Note / Quick Start Guide / Example Program, you acknowledge that we cannot be held liable for any damages beyond the described liability regime. We reserve the right to make changes to this application note / quick start guide / example at any time without notice. In case of discrepancies between the proposals Application Notes / Quick Start Guides / Program Examples and other Weidmüller publications, like manuals, such contents have always more priority to the examples. We assume no liability for the information contained in this document. Our liability, for whatever legal reason, for damages caused using the examples, instructions, programs, project planning and performance data, etc. described in this Application Note / Quick Start Guide / Example is excluded.

Security notes
In order to protect equipment, systems, machines and networks against cyber threats, it is necessary to implement (and maintain) a complete state-of-the-art industrial security concept. The customer is responsible for preventing unauthorized access to his equipment, systems, machines and networks. Systems, machines and components should only be connected to the corporate network or the Internet if necessary and appropriate safeguards (such as firewalls and network segmentation) have been taken.

2 Introduction

2.1 What is SNMP?

The Simple Network Management Protocol (SNMP) is an industry-standard protocol used to monitor and manage network devices. It enables software tools to query device status, read performance metrics, receive alerts, and modify certain configuration values through a defined data structure known as the Management Information Base (MIB).

SNMP Versions (General Overview)

Weidmueller Industrial Ethernet devices support SNMP v1, v2c and v3. The web user interface of every device offers a help section describing these modes, their security characteristics and how to activate them.

SNMPv1

- Easy to set up but not recommended
- Unencrypted community strings

SNMPv2c

- Broad compatibility and easy to use
- Still uses plaintext community strings

SNMPv3

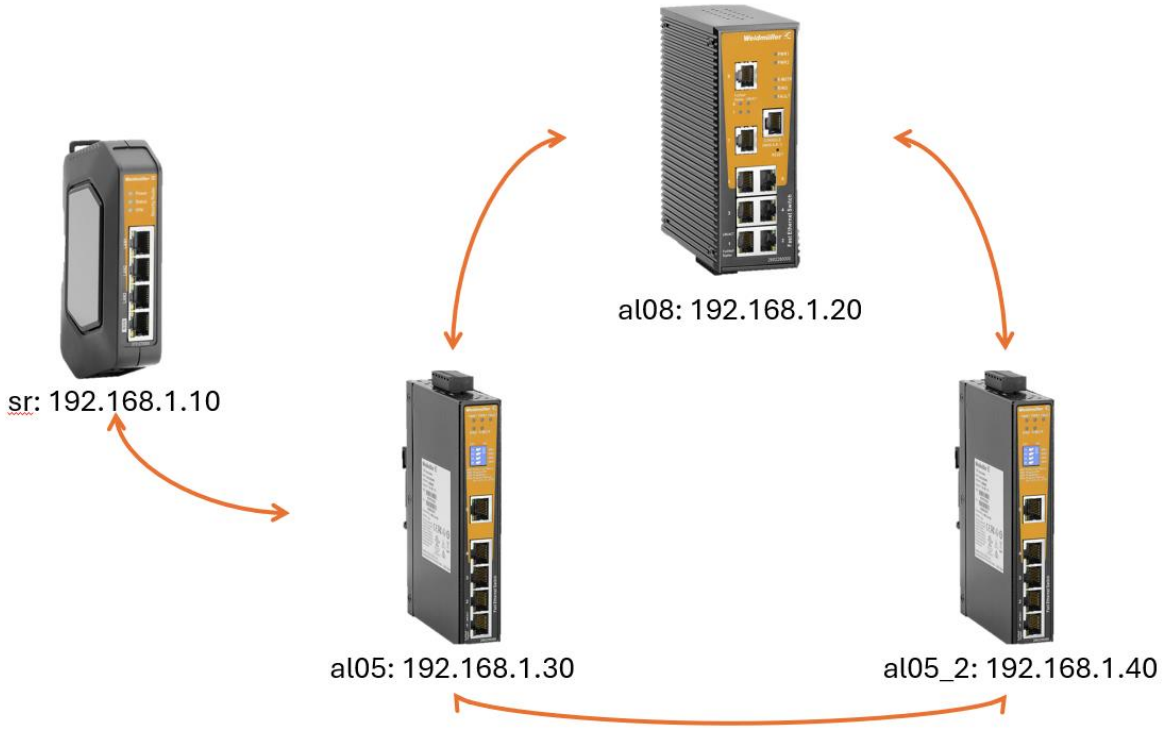
- User-based authentication (USM)
- Supports message encryption
- More complex to configure initially
- Recommended to use

Note
Every feature shown is also applicable on every managed Weidmueller Industrial Ethernet supporting SNMP.

®

2.2 Network Topology Overview

In this application example, we use a ring topology with O-Ring enabled. All Industrial Ethernet devices in the ring are reachable from the workstation running Ansible and Zabbix. To see how the Weidmueller Industrial Ethernet Trainingkit gets configured, refer to “Setting up default configuration of IE Training Kit” and “Setting up an O-Ring network structure with Weidmueller switches” Application Note (visit https://eshop.weidmueller.com/ → search “IE-Trainingskit”→ Product Downloads).



The abbreviations (sr, al08, al05, al05_2) are used throughout the following chapters to reduce name length and keep configuration steps, playbooks, and monitoring examples easier to read.

2.3 Prerequisites

Before starting with the configuration and monitoring tasks, the following prerequisites must be fulfilled:

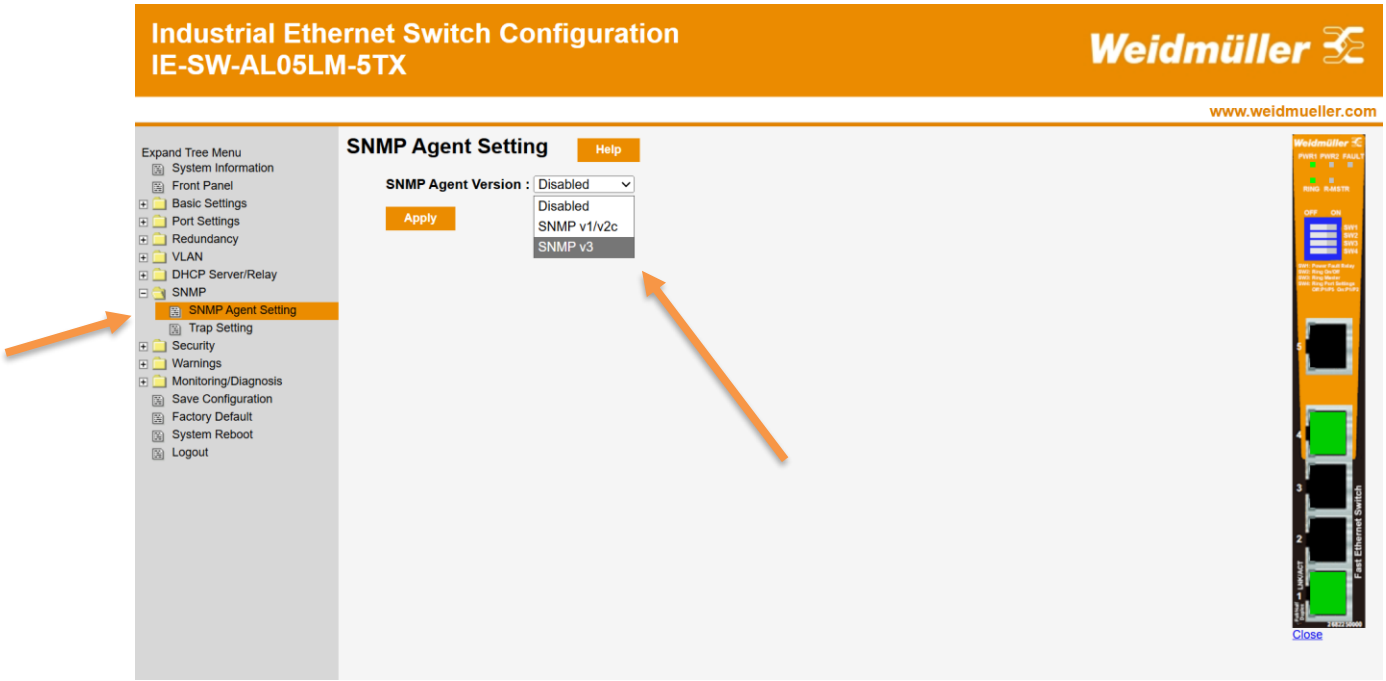
- Industrial Ethernet Kit connected via Ethernet
- Private MIB files for the managed devices

The private MIB file for an Industrial Ethernet device can be found on its Weidmueller e-shop page under “Software Support”. Additionally, it is recommended to install a SNMP MIB Browser to retain overview of the MIB tree.

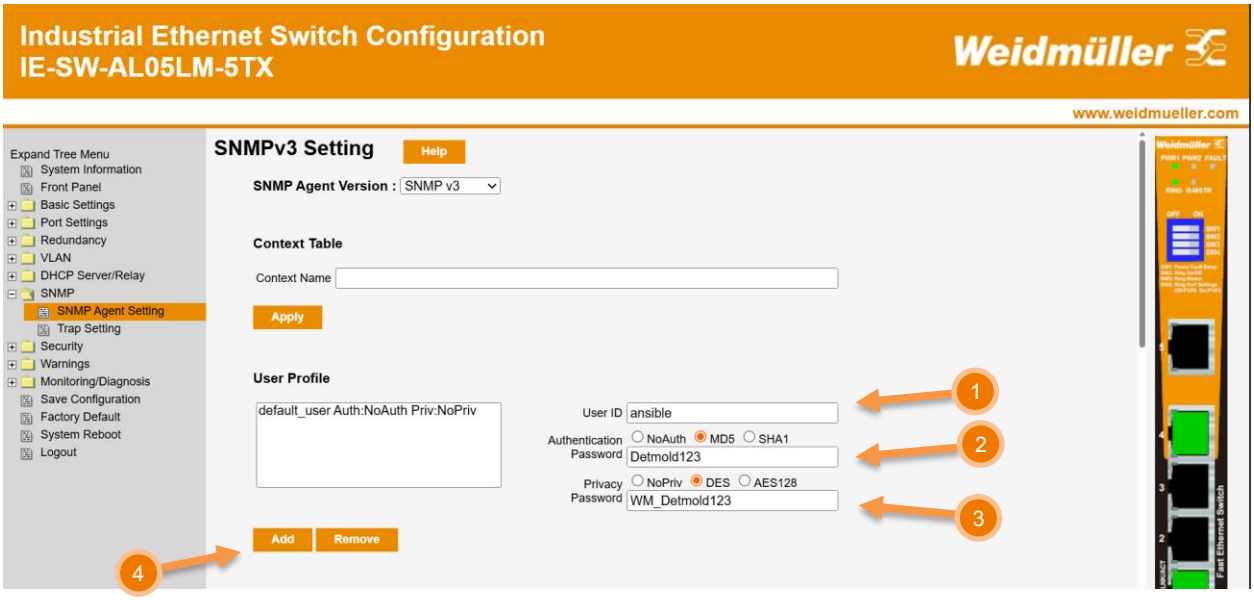
2.4 Enabling SNMP on Industrial Ethernet Devices

Repeat the following configuration steps on all Industrial Ethernet switches.

1. Open the web interface of the device you would like to configure. Direct to “SNMP” → “SNMP Agent Setting”. Enable the SNMP Agent and select SNMPv3 as it is the safest and most often recommended version to use.

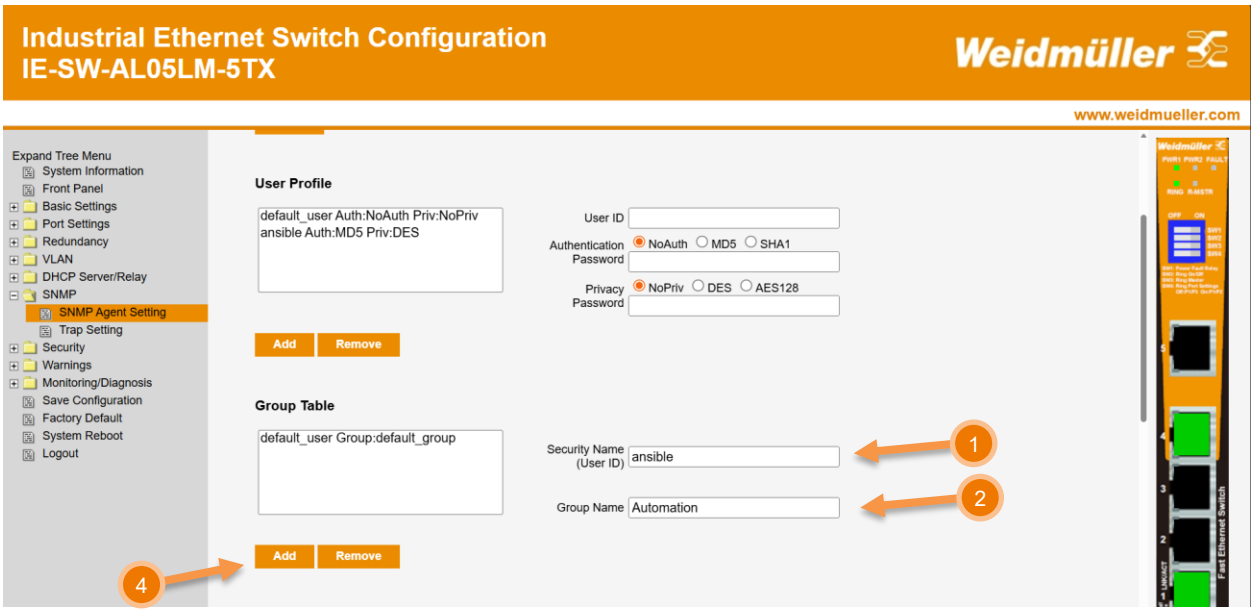


2. Configure a user profile. Set a name (1), authentication password & hash function (2) and privacy password & encryption (3). Press “Add” to finish (4).

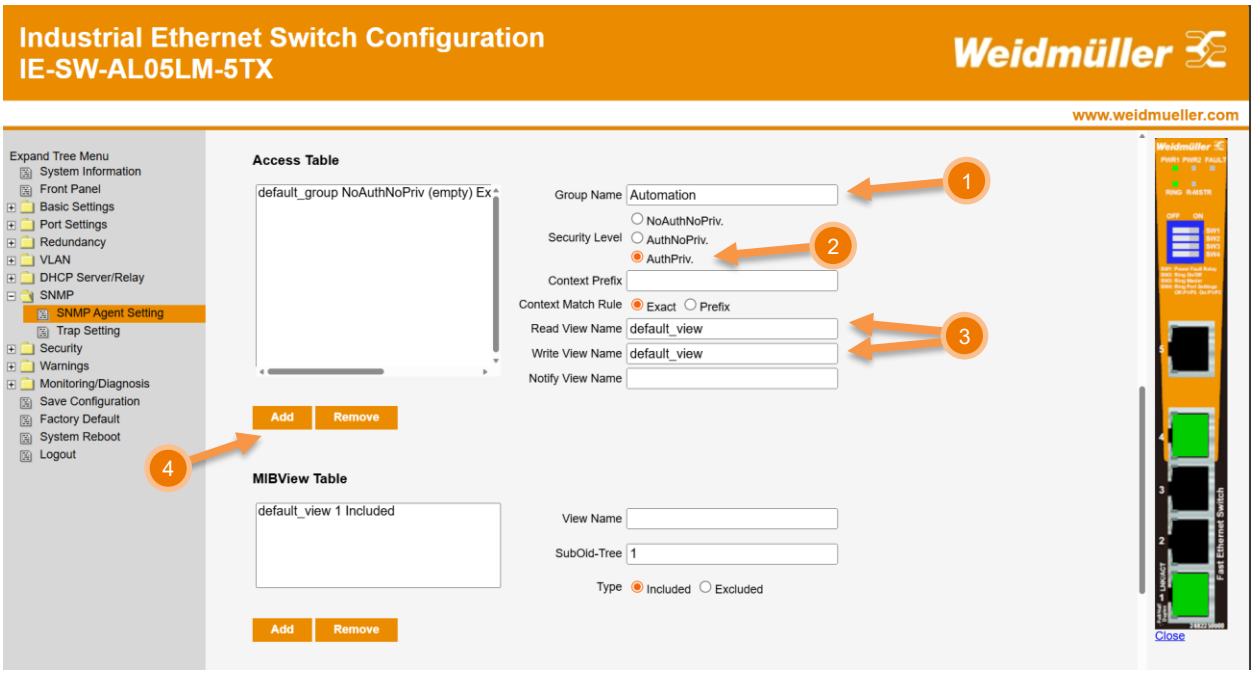


The context table is not considered as a necessary setting in this application note as it is for organizing access control to the MIB tree among the users and groups. The SNMPv3 connection will work fine without a configuration of the context table.

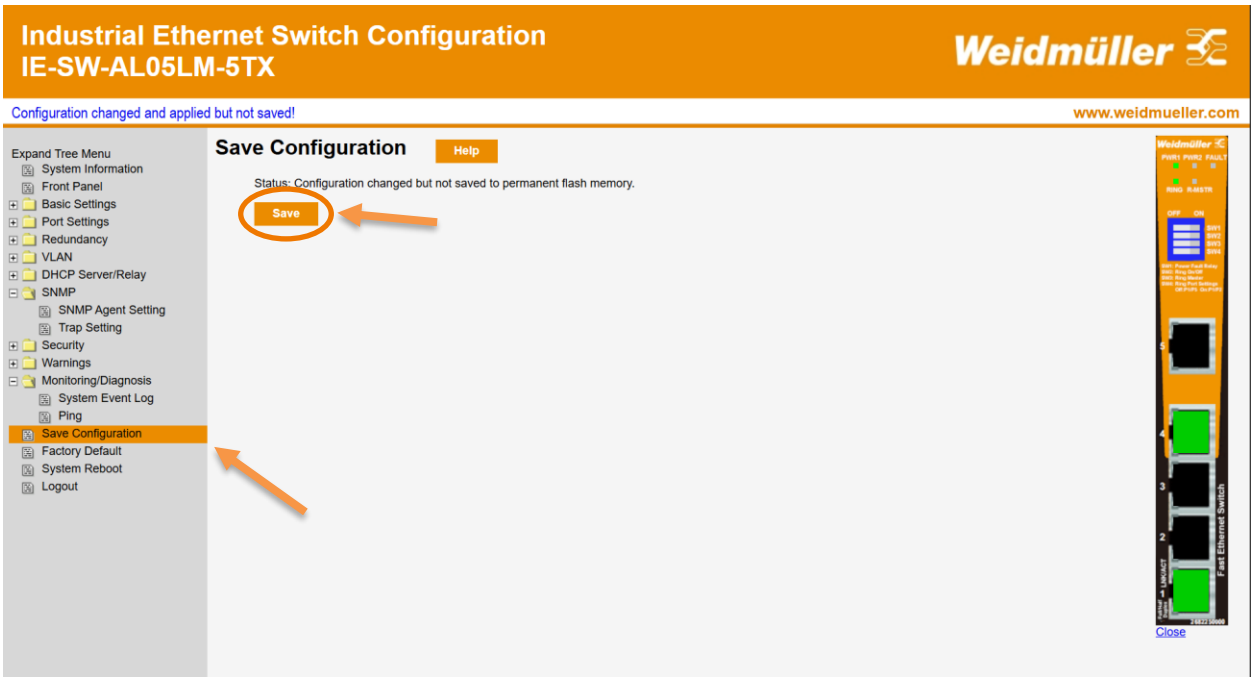
3. Add the just created user to a group. Thus, type in the user ID (1) and set a group name (2). Press “Add” to finish (3).



4. Configure the access table to manage which OIDs of the MIB tree are visible to the group. Type in the group name (1), select “AuthPriv.” (2) and type in “default_view” into read/ write name as this group should be able to view the entire MIB tree (3). Press “Add” to finish (4).



5. Navigate to “Save Configuration” and press “Save” to write the configuration permanently onto the running image.



3 Network Automation with Ansible

With the foundational SNMP concepts established, this section now moves into the practical configuration phase using Ansible. Ansible is a widely used automation framework designed for declarative, infrastructure-as-code management. The goal is to roll out standardized settings across all Industrial Ethernet devices in an automated way.

3.1 Setup & Requirements

To follow the configuration steps, prepare the following environment:

- Weidmueller Industrial Ethernet devices with configured SNMPv3 agent
- Linux distribution of your choice or Windows Subsystem for Linux (WSL). For this application WSL2, running Ubuntu, has been chosen.
- Python 3.10
- Ansible Core 2.17.14
- Ansible SNMP collection: ansible.snmp 3.0.0
- Python binding: net-snmp 5.9.1
- Setup snmptranslate

Visit the documentation of Ansible, the SNMP collection and Microsoft WSL2 to gain further information about the installation process (docs.ansible.com, galaxy.ansible.com & learn.microsoft.com).

Beforehand, create a virtual environment (venv) in your project folder, then install Ansible and other dependencies to prevent version complications (docs.python.org).

Setting up snmptranslate leads to the use of textual instead of numeric OIDs (net-snmp.org).

Testing SNMP Before Using Ansible

Before running any Ansible playbook, verify that the device responds to SNMP queries. This avoids troubleshooting Ansible while the real issue is SNMP communication. Run the following recommended checks in your bash console on your control node:

1. **Ping the device**

```
$ ping 192.168.1.X
```

2. **Test SNMP**

   Install the package and run the following query:

```
$ sudo apt install snmp
$ snmpget -v 3 -l authPriv -u <user-ID> -a <hash> -A <auth. pw> -x <encryption>
  -X <priv. pw> 192.168.1.X .1.3.6.1.2.1.1.5.0
```

3.2 Configuring Device Location Using Ansible

This chapter explains how to configure the system location of multiple Industrial Ethernet switches using Ansible and SNMPv3. The system location is a descriptive text stored on the switch. It can identify the physical installation position of the device, for example: “Cabinet 104”, “Cell 103.02”, “Production Line 4”, etc.

In this project, Ansible reads the current system location from each switch, compares it with the desired value defined in the inventory, and changes the value only when a difference exists. This approach follows the Infrastructure as Code principle: the desired configuration is stored in files instead of being configured manually on every device.

The workflow consists of three main steps:

1. Define the Industrial Ethernet switches in the inventory.
2. Assign device-specific OIDs and desired location values.
3. Write and permanently save the configuration only when required.

3.2.1 Overview of the Project Structure

The project consists of two files:

```
project/
├─ inventory.ini
└─ set_sysLoc.yml
```

The files have the following purposes:

File	Purpose
inventory.ini	Defines the switches, management IP addresses, SNMPv3 connection parameters, model-specific OIDs, and desired system location values.
set_sysLoc.yml	Contains the reusable automation logic that reads, compares, changes, and saves the system location.

A central design principle of this project is the separation between configuration data and automation logic. The inventory.ini file contains all values that may differ between devices. The playbook contains only the general procedure required to apply these values. Therefore, a user normally does not need to modify the playbook. To change the system location of a device, only the corresponding value in inventory.ini must be changed.

3.2.2 Inventory

An Ansible inventory defines the devices that Ansible manages. In this application, the targets are Industrial Ethernet switches that are accessed through SNMPv3.

The inventory defines:

- the logical Ansible name of each switch,
- the management IP address,
- the switch model group,
- the private OID used for the system location,
- the private OID used to save the configuration,
- the desired system location,
- the SNMPv3 username,
- the SNMPv3 authentication parameters,
- the SNMPv3 encryption parameters.

inventory.ini:

```
# Industrial Ethernet switches

[al05_switches]

al05     ansible_host=192.168.1.30  switch_setting_system_location_value='Cell
103.01'

al05_2  ansible_host=192.168.1.40  switch_setting_system_location_value='Cell
103.02'
```

```
[al08_switches]
al08  ansible_host=192.168.1.20 switch_setting_system_location_value='Cabinet
103'



# Common target group
[targets:children]
al05_switches
al08_switches



# OIDs for AL05 devices
[al05_switches:vars]
switch_setting_system_location_oid=IE-SW-AL05LM-5TX-
MIB::switchSettingSystemLocation
save_cfg_mgt_action_oid=IE-SW-AL05LM-5TX-MIB::saveCfgMgtAction



# OIDs for AL08 devices
[al08_switches:vars]
switch_setting_system_location_oid=IE-SW-AL08M-8TX-
MIB::switchSettingSystemLocation
save_cfg_mgt_action_oid=IE-SW-AL08M-8TX-MIB::saveCfgMgtAction



# Common SNMP and OID parameters
[targets:vars]
ansible_connection=ansible.snmp.v3_usm
ansible_snmp_timeout=900000

ansible_snmp_auth_pass=Detmold123
ansible_snmp_auth_proto=MD5
```

```
ansible_snmp_priv_pass=Detmold123
ansible_snmp_priv_proto=DES

ansible_snmp_sec_name=ansible
ansible_snmp_sec_level=authPriv

switch_setting_system_location_iid=0
switch_setting_system_location_type=OCTETSTR

save_cfg_mgt_action_iid=0
save_cfg_mgt_action_value=1
save_cfg_mgt_action_type=INTEGER
```

Device groups
The switches are divided into two model-specific groups: [al05_switches] and [al08_switches]. This grouping is required because different switch models use different private MIB modules. The AL05 switches use: IE-SW-AL05LM-5TX-MIB and the AL08 switch uses: IE-SW-AL08M-8TX-MIB. Although the parameters perform the same logical function, the full symbolic OIDs are different. Grouping devices by model avoids repeating the same OID for every individual switch.

Host definitions
The following line defines one device:

```
al05   ansible_host=192.168.1.30   switch_setting_system_location_value='Cell
103.02'
```

It contains three elements.

- **Inventory hostname**: al05
  This is the logical name used by Ansible. The inventory hostname does not have to be the DNS hostname of the device. It is an internal identifier used in playbook output and Ansible commands.
- **Management IP address**: ansible_host=192.168.1.30
  The ansible_host variable specifies the real network address that Ansible uses to contact the switch.

- **Desired system location**: switch_setting_system_location_value='Cell 103.02'
  This variable defines the desired system location. It represents the target state of the device. If the switch currently contains a different location, Ansible changes it to this value. Values containing spaces should be enclosed in quotation marks.

Common target group
The two model groups are combined into one parent group:

```
[targets:children]
al05_switches
al08_switches
```

Ansible automatically includes all devices from both child groups. This allows one generic playbook to manage different device models.

Model-specific OIDs
The OIDs for AL05 switches are defined once for the complete AL05 group:

```
[al05_switches:vars]
switch_setting_system_location_oid=IE-SW-AL05LM-5TX-
MIB::switchSettingSystemLocation
save_cfg_mgt_action_oid=IE-SW-AL05LM-5TX-MIB::saveCfgMgtAction
```

The equivalent OIDs are defined separately for the AL08 group:

```
[al08_switches:vars]
switch_setting_system_location_oid=IE-SW-AL08M-8TX-
MIB::switchSettingSystemLocation
save_cfg_mgt_action_oid=IE-SW-AL08M-8TX-MIB::saveCfgMgtAction
```

The variable “switch_setting_system_location_oid” points to the parameter that stores the system location.
The variable “save_cfg_mgt_action_oid” points to the parameter that triggers the permanent saving of the switch configuration.

The same variable names are used for both model groups. Only the assigned OIDs differ.
As a result, the playbook does not need to know which switch model it is managing. Ansible automatically selects the correct group variable for each device.

3.2.3 Playbook

An Ansible playbook is a YAML file that defines the operations Ansible performs. The playbook used in this project does not contain device-specific IP addresses, location values, or model-specific OIDs. These values are obtained from the inventory.

The playbook performs the following sequence:

- Configure the system location,
- Save the configuration after a change.

After execution, the playbook remains unchanged and can be reused for all supported Industrial Ethernet switches.

set_sysLoc.yml:

```
- name: Set system location on managed switches
  hosts: targets
  gather_facts: false


  tasks:
    - name: Set system location
      ansible.snmp.set:
        oids:
          - oid: "{{ switch_setting_system_location_oid }}"
            iid: "{{ switch_setting_system_location_iid }}"
            value: "{{ switch_setting_system_location_value }}"
            type: "{{ switch_setting_system_location_type }}"
      notify: Save switch configuration


  handlers:
    - name: Save switch configuration
      ansible.snmp.set:
        oids:
          - oid: "{{ save_cfg_mgt_action_oid }}"
            iid: "{{ save_cfg_mgt_action_iid }}"
            value: "{{ save_cfg_mgt_action_value | int }}"
            type: "{{ save_cfg_mgt_action_type }}"
```

3.3 Execution

Run the following command on your control node:

```
$  ansible-playbook -i inventory.ini set_sysLoc.yml
```

To run this command verbose and
directly see the result strings append a “-vvv” flag.

The expected output will report:

- **changed** → Location successfully updated
- **ok** → Location already had the desired value
- **failed** → Wrong SNMPv3 credentials or incorrect OID for device
- **unreachable** → Connectivity issue
- **skipped** → Desired value already set

Example output:

```
(.venv) admin@node:~/iac/ansible$ ansible-playbook -i inventory.ini
set_sysLoc.yml


PLAY [Set system location on managed switches]
*********************************************************************
*********************************************************************


TASK [Read current system location]
*********************************************************************
*********************************************************************
*******
ok: [al05_2]
ok: [al05]
ok: [al08]


TASK [Show current and desired system location]
*********************************************************************
*********************************************************************
ok: [al05] => {
    "msg": [
        "Switch: al05",
        "Current location: Cell 103.02",
        "Desired location: Cell 103.02"
    ]
}
ok: [al05_2] => {
    "msg": [
        "Switch: al05_2",
        "Current location: Cell 103.03",
        "Desired location: Cell 103.03"
    ]
}
```

```
ok: [al08] => {
    "msg": [
        "Switch: al08",
        "Current location: Cabinet 103",
        "Desired location: Cabinet 103"
    ]
}

TASK                    [Set                    system                    location]
*******************************************************************************
*******************************************************************************
***************
skipping: [al05]
skipping: [al05_2]
skipping: [al08]

PLAY                                                                        RECAP
*******************************************************************************
*******************************************************************************
******************************
al05                         : ok=2    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
al05_2                       : ok=2    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
al08                         : ok=2    changed=0    unreachable=0    failed=0
skipped=1    rescued=0    ignored=0
```

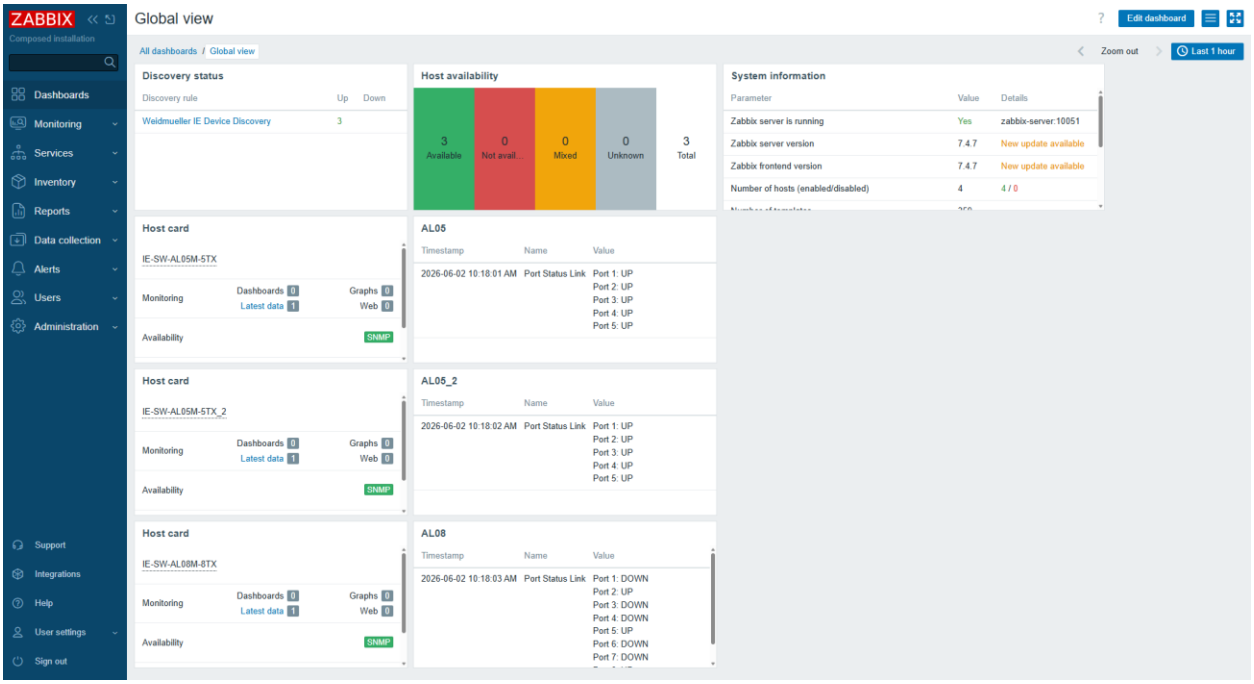
4 Network Monitoring with Zabbix

In the previous section, we prepared all Industrial Ethernet devices for SNMP-based management by enabling SNMPv3 and assigned standardized configuration parameters. We now move from configuration to monitoring and introduce Zabbix, the Network Management System that will be used to collect, visualize, and evaluate device data.

Zabbix is an open-source infrastructure monitoring system designed to collect metric data from networks, devices, applications and servers. It supports numerous collection protocols including IPMI, JMX, HTTP, SSH, MQTT and SNMP. Its core components are:

- **Server** – process data, triggers, events alerts, discovery, polling cycle configuration
- **Database** – stores metrics, history, trend data; can be MySQL/ MariaDB or PostgreSQL + TimescaleDB extension
- **Frontend** – a web interface for configuration and visualization

The following describes how to build a dashboard that shows the current state of the device's interfaces.



4.1 Setup Backend & Requirements

To follow the configuration steps, prepare the following environment:

- Weidmueller Industrial Ethernet devices with configured SNMPv3 agent
- Linux distribution of your choice or Windows Subsystem for Linux (WSL). For this application WSL2, running Ubuntu, has been chosen.
- Docker Container Engine
- Cloned Zabbix Docker Compose repository

Visit the documentation of Zabbix to gain further information about the installation process (www.zabbix.com).

For this application Zabbix v7.4 and the MySQL Alpine Docker Compose have been chosen. Some configuration steps, related to Zabbix, may differ in newer versions.

Keep in mind that a new username for SNMPv3 has been chosen. It switched from “ansible” to “zabbix”. You may proceed all steps without changing your credentials to “zabbix”.

Test SNMP Connection inside Zabbix Server Container

1. **Identify the Zabbix Server Container:**
   List all running containers and note the Zabbix server container name.

```
$ docker ps
```

2. **Open a Shell Inside the Zabbix Server Container**

```
$ docker exec -u 0 -it <zabbix-server-container> /bin/sh
```

3. **Ping the desired device**

```
$ ping 192.168.1.X
```

4. **Test SNMP connectivity**

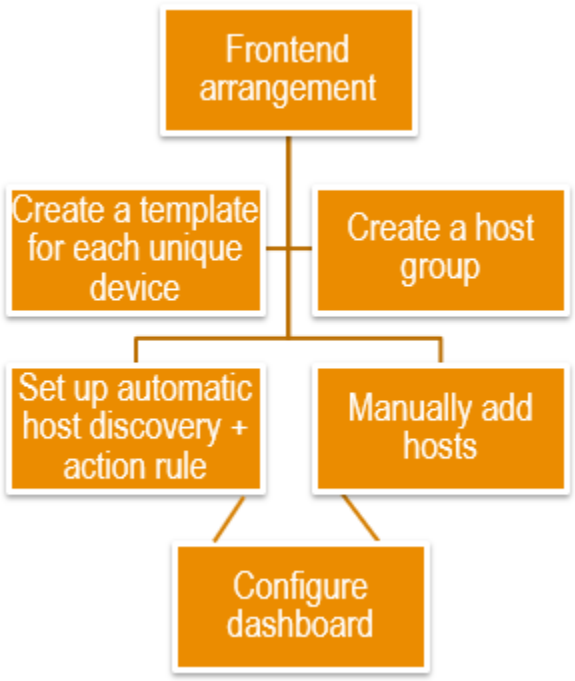
```
$ apk update
$ apk add net-snmp net-snmp-tools
$ snmpget -v 3 -l authPriv -u <user-ID> -a <hash> -A <auth. pw> -x <encryption>
  -X <priv. pw> 192.168.1.X .1.3.6.1.2.1.1.5.0
```

4.2 Setup Frontend

This chapter shows how to configure Zabbix via its web-based frontend to monitor Weidmueller Industrial Ethernet devices, including creating host groups, building a reusable template per device type and setting up the monitoring item with the OID of `portStatusLink` using preprocessing so the dashboard shows readable interface state information. Additionally, there are two options explained for onboarding Industrial Ethernet network devices:

a. **Manual Setup**
   Ideal for a small and static number of devices.
b. **Automatic Host Discovery**
   Recommended when monitoring many devices or when the network topology changes over time.

Finally, yet significantly needs the Zabbix dashboard some configuration so that all information gets visualized on a centralized window.



Zabbix Web Frontend

To access the frontend of Zabbix, you have two options. Either you install a browser in WSL and open “http://localhost”, or you use a browser on windows and access the WSL instance via its IP address.

Accessing Zabbix through Windows Browser
Execute the following command, copy the IP-address, named “inet” and open it in Windows browser.

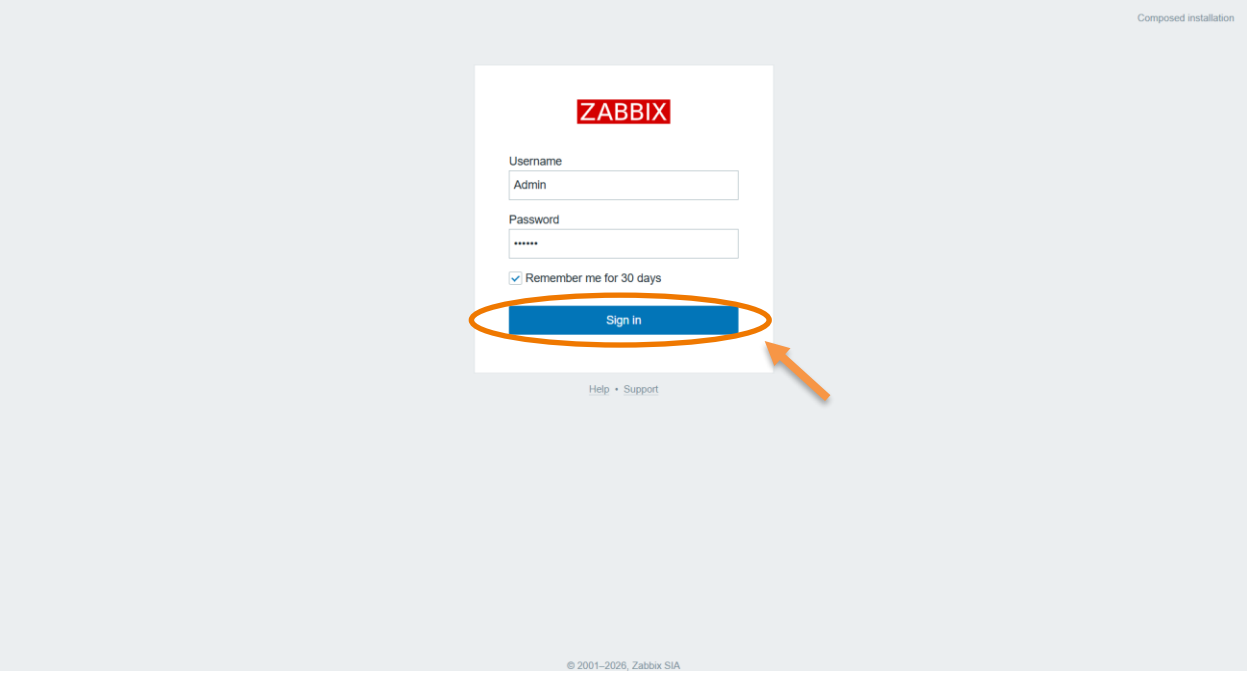
```
$  ip a | grep eth0
```

Login

Sign in into the frontend. Standard credentials:

- Username: Admin
- Password: zabbix

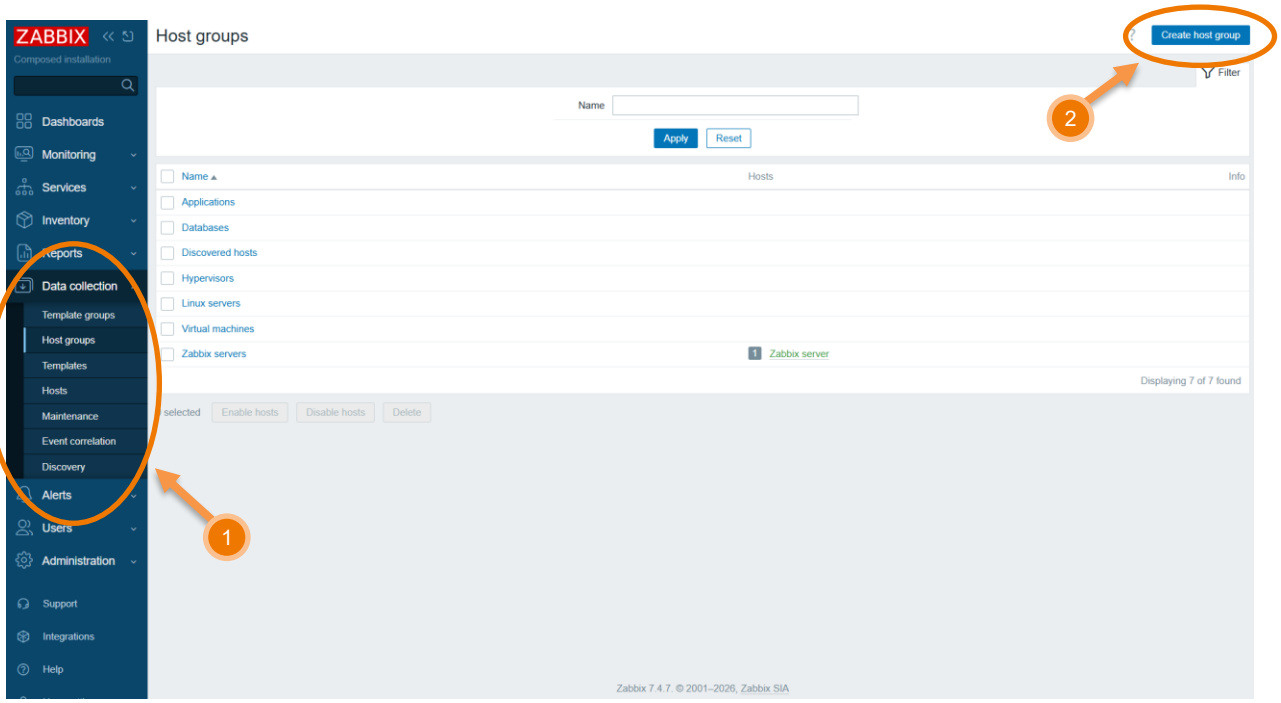
Press “Sign in”.



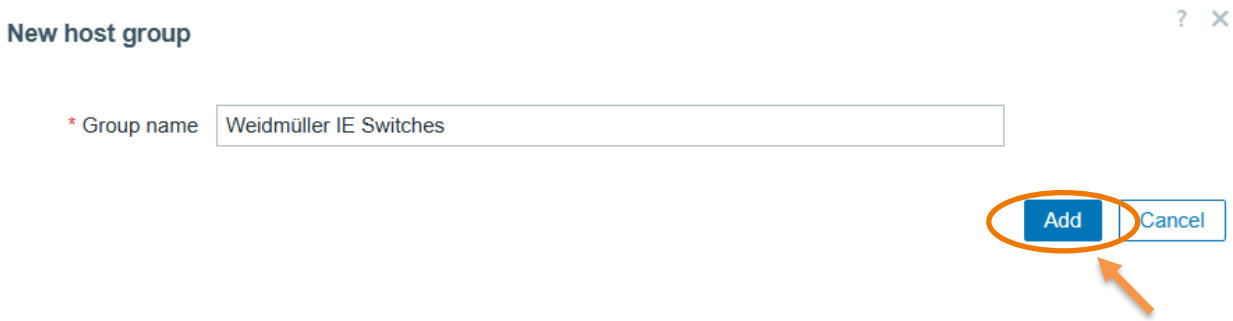
Create a Host Group

A host group organizes devices logically.

1. Open “Data collection” → “Host groups” (1) and click “Create host group” (2).



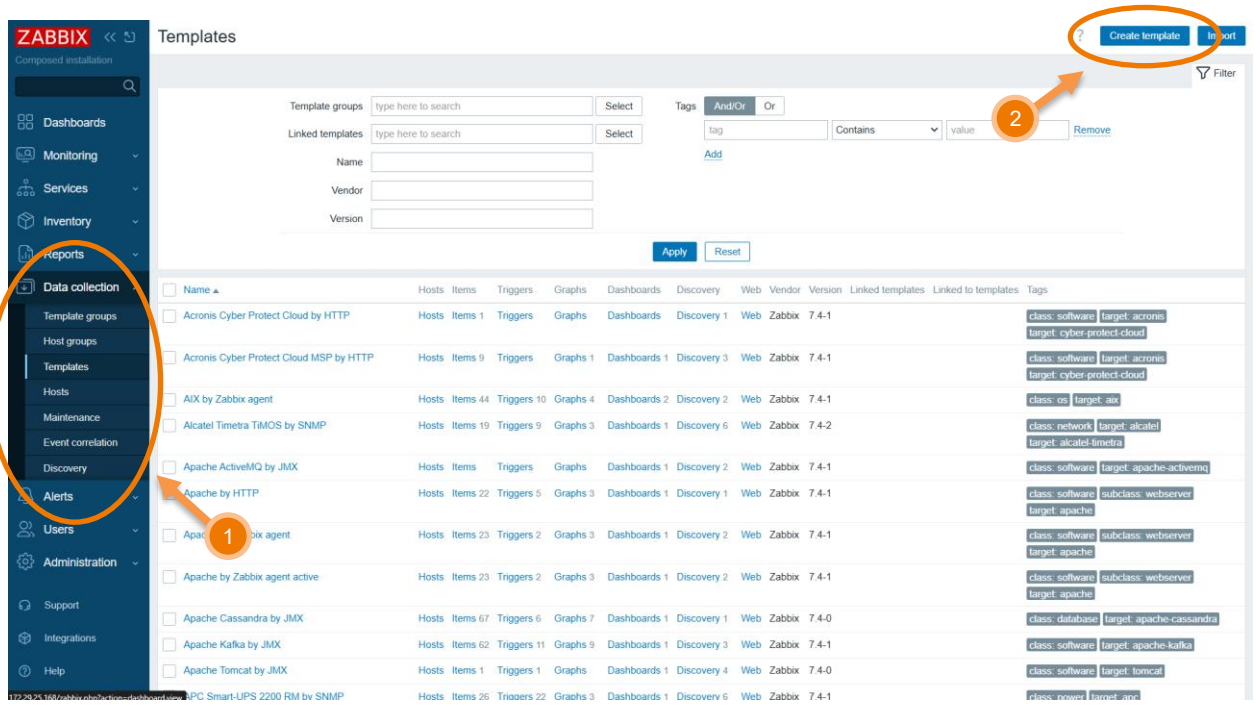
2. Enter a name. Press “Add”.



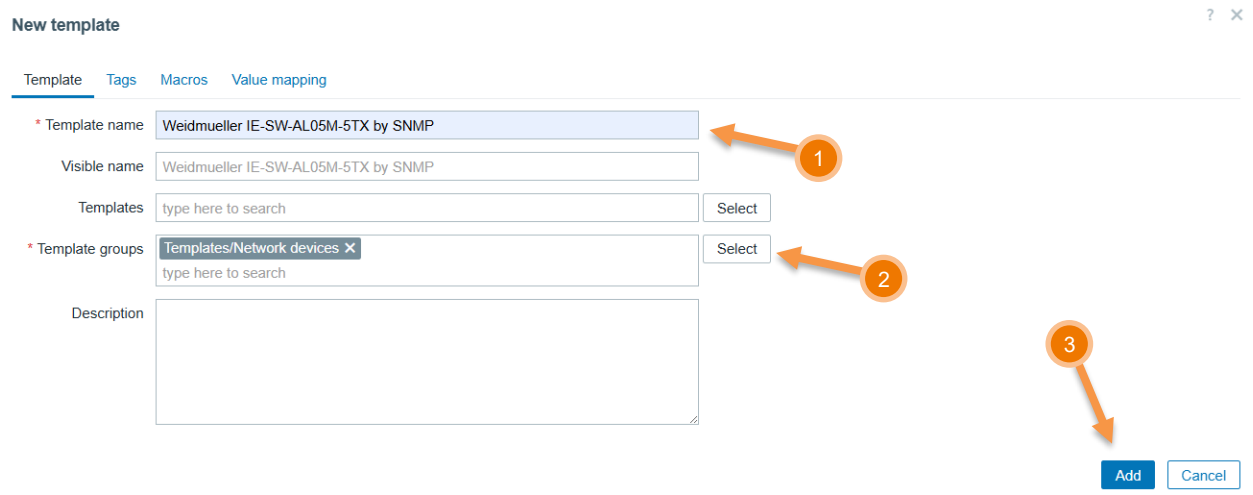
Create a Template for Each Unique Device

Templates standardize monitoring across devices and allow Zabbix to automatically assign items, triggers and graphs. This reduces configuration effort by avoiding per-host manual setup. Create templates for each device family that does not share the same private MIB tree. In the following, the creation of the template for IE-SW-AL05LM-5TX is shown. Reproduce same steps for
IE-SW-AL08M-8TX to complete template creation.

1. Navigate to “Data collection” → “Templates” (1). Click “Create template” (2).

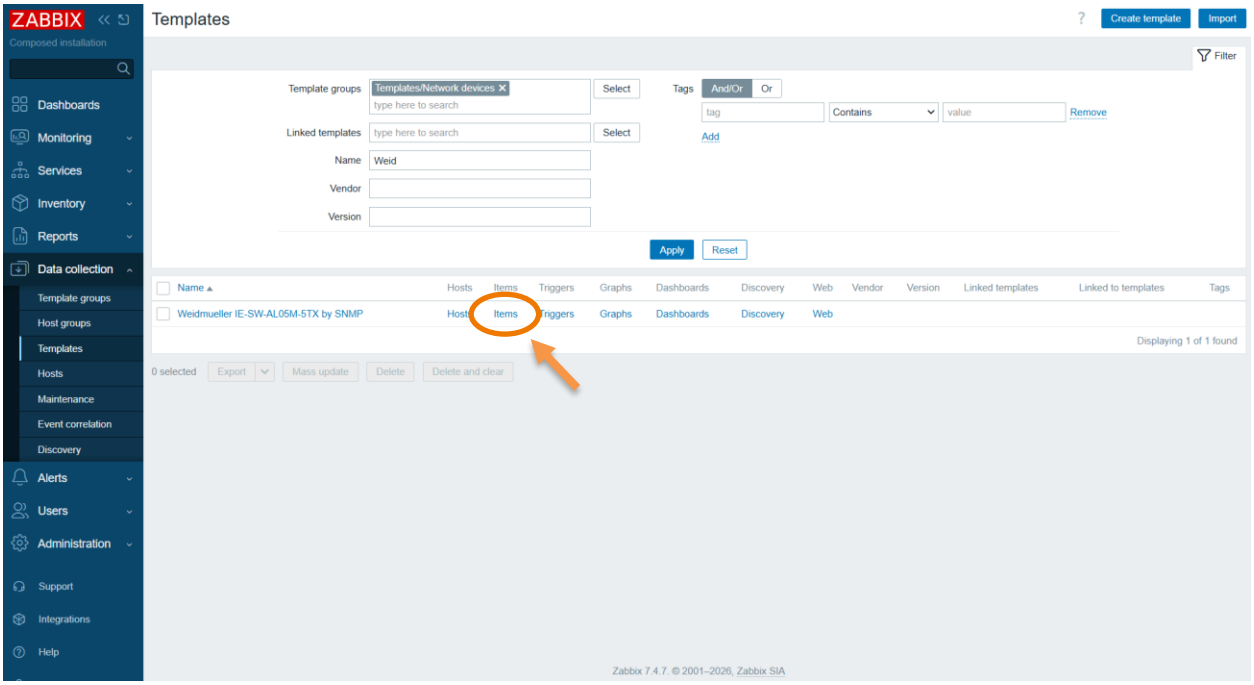


2. Fill in a template name (1) and select a template group (2). Press “Add” (3).

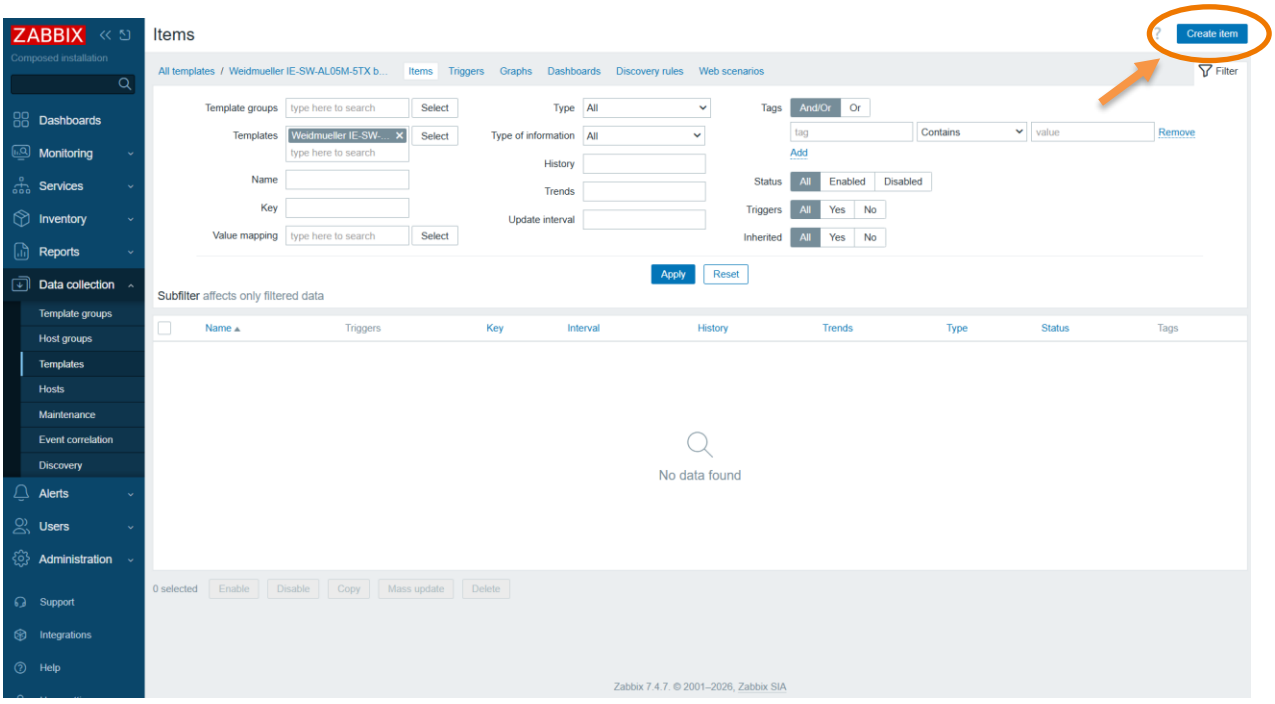


3. Add monitoring items to the template. SNMPv3 device information is mapped to items so Zabbix can collect and store the data. You can repeat this process for relevant OIDs.

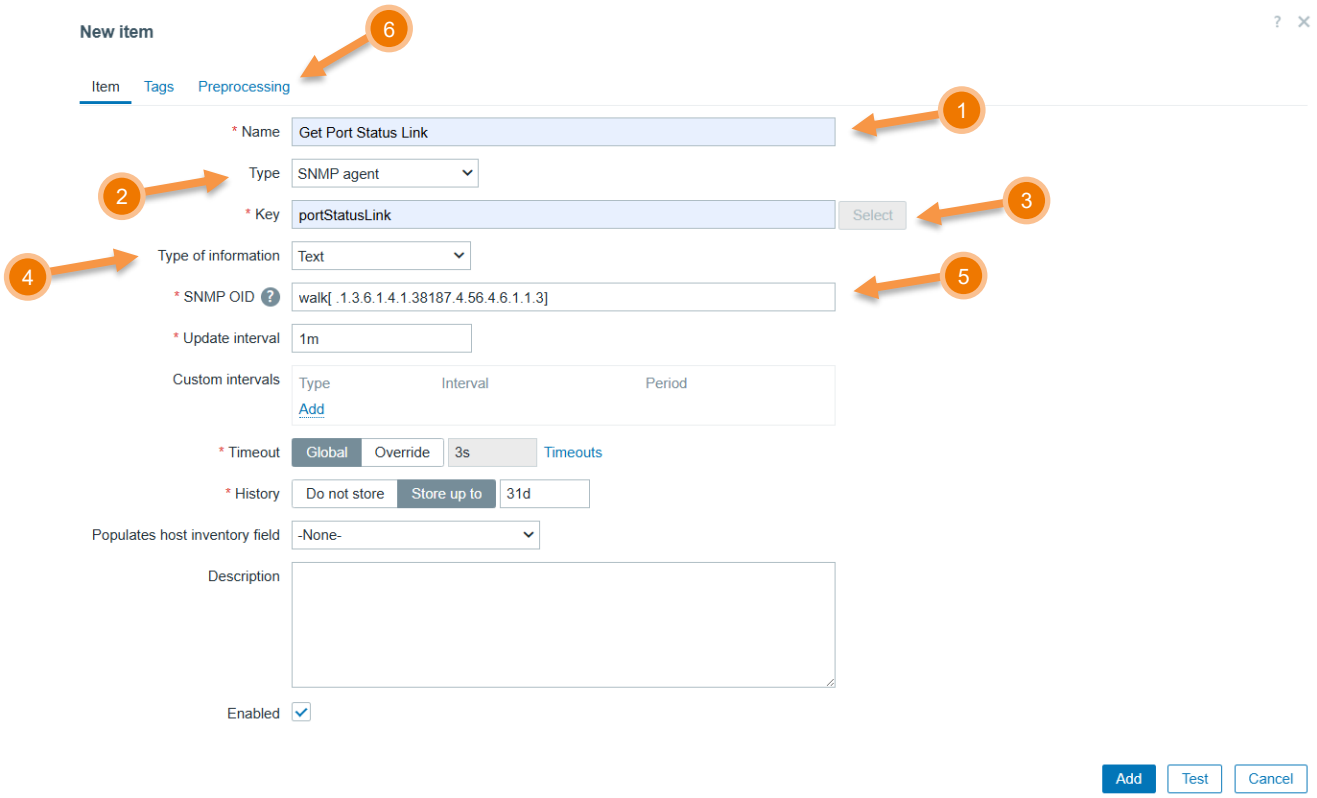
   Press “Items”.



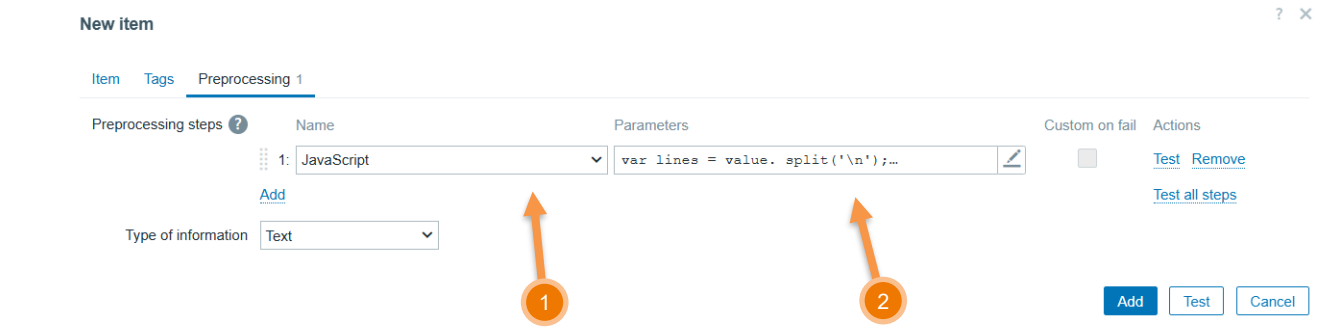
4. Press “Create item”.



5. Fill out the details: Name (1), SNMP agent as type (2), a unique key for this item (3), “Text” as the type of information (4) and your desired OID (5). Press “Preprocessing” for the next step (6).



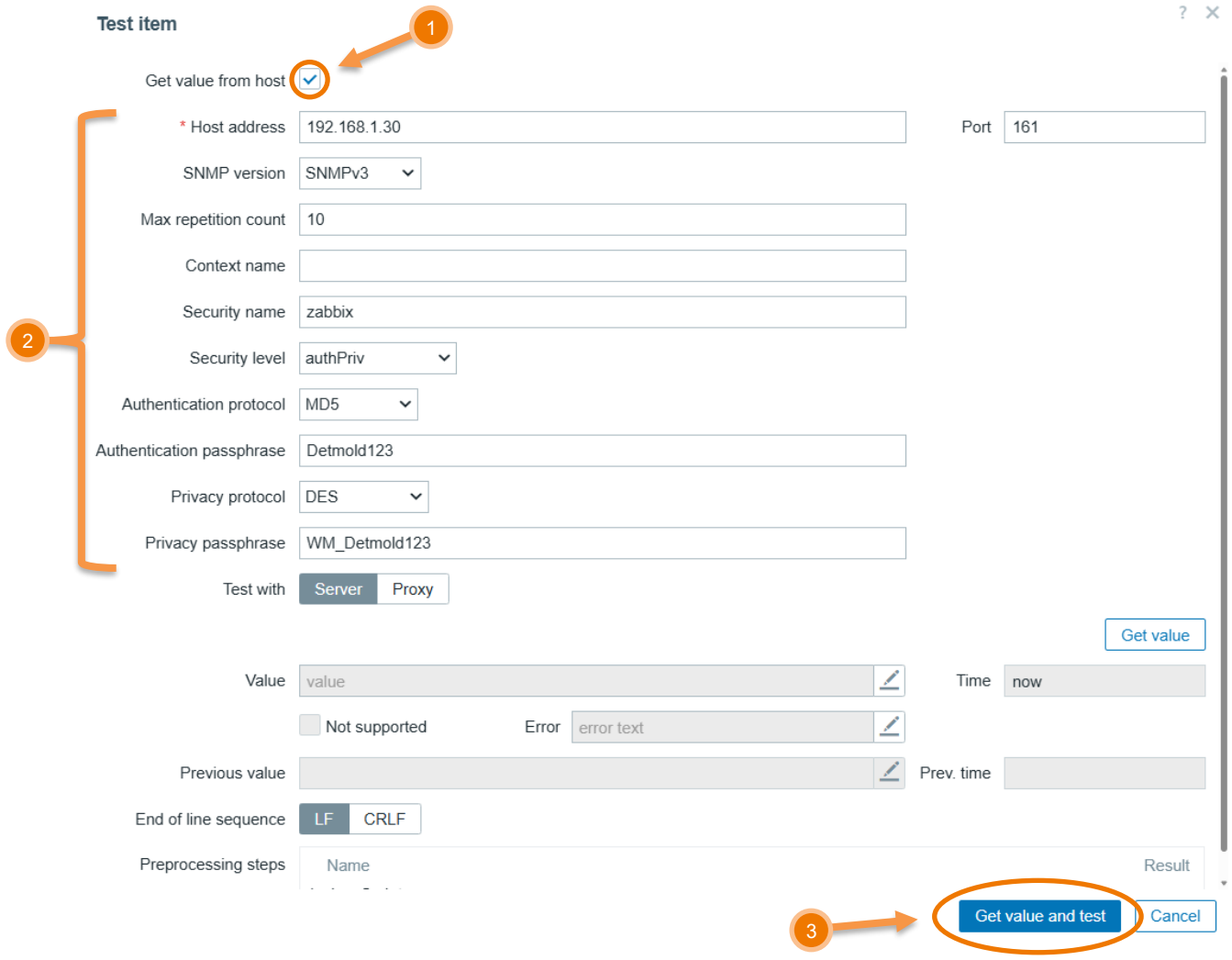
6. In “Preprocessing”, select JavaScript (1) and paste the following script (2) to transform raw SNMP output into a clean formatted string that the UI can display and the database store.



format_portStatusLink.js

```
var lines = value. split('\n');

var result = "";

for (var i = 0; i < lines. length; i++) {

    var m = lines[i].match(/\.([0-9]+)\s*=\s*INTEGER:\s*([0-9]+)/);

    if (m) {
        var port = m[1];        // Port index (last OID number)
        var rawVal = m[2];      // Integer value (1 or 2)

        var state = (rawVal === "1") ? "UP" :
                    (rawVal === "2") ? "DOWN" :
                    rawVal;     // fallback if unexpected

        result += "Port " + port + ": " + state + "\n";
    }
}

return result.trim();
```

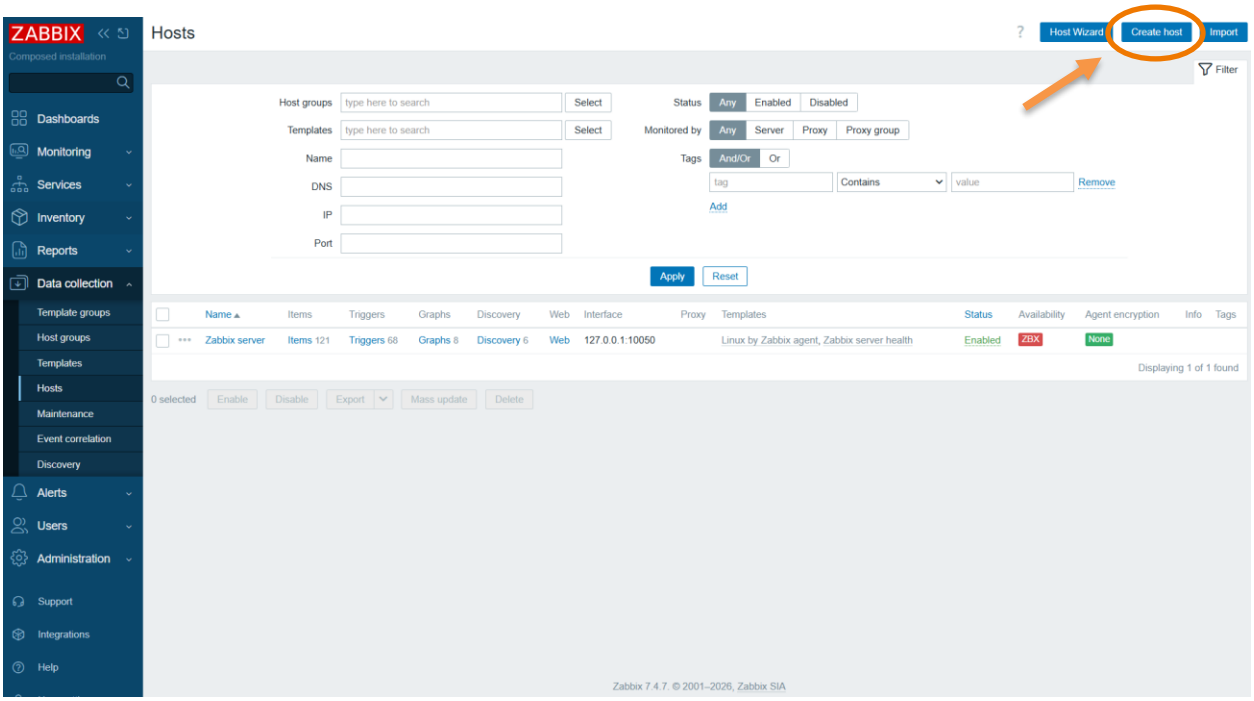
7. Press “Test” to validate correctness. Thus tick “Get value from host” (1) and fill in the SNMPv3 credentials (2). After that, click “Get value and test” (3). If the given result is as expected, proceed with repeating this process for the IE-SW-AL08M-8TX and its specific OIDs.



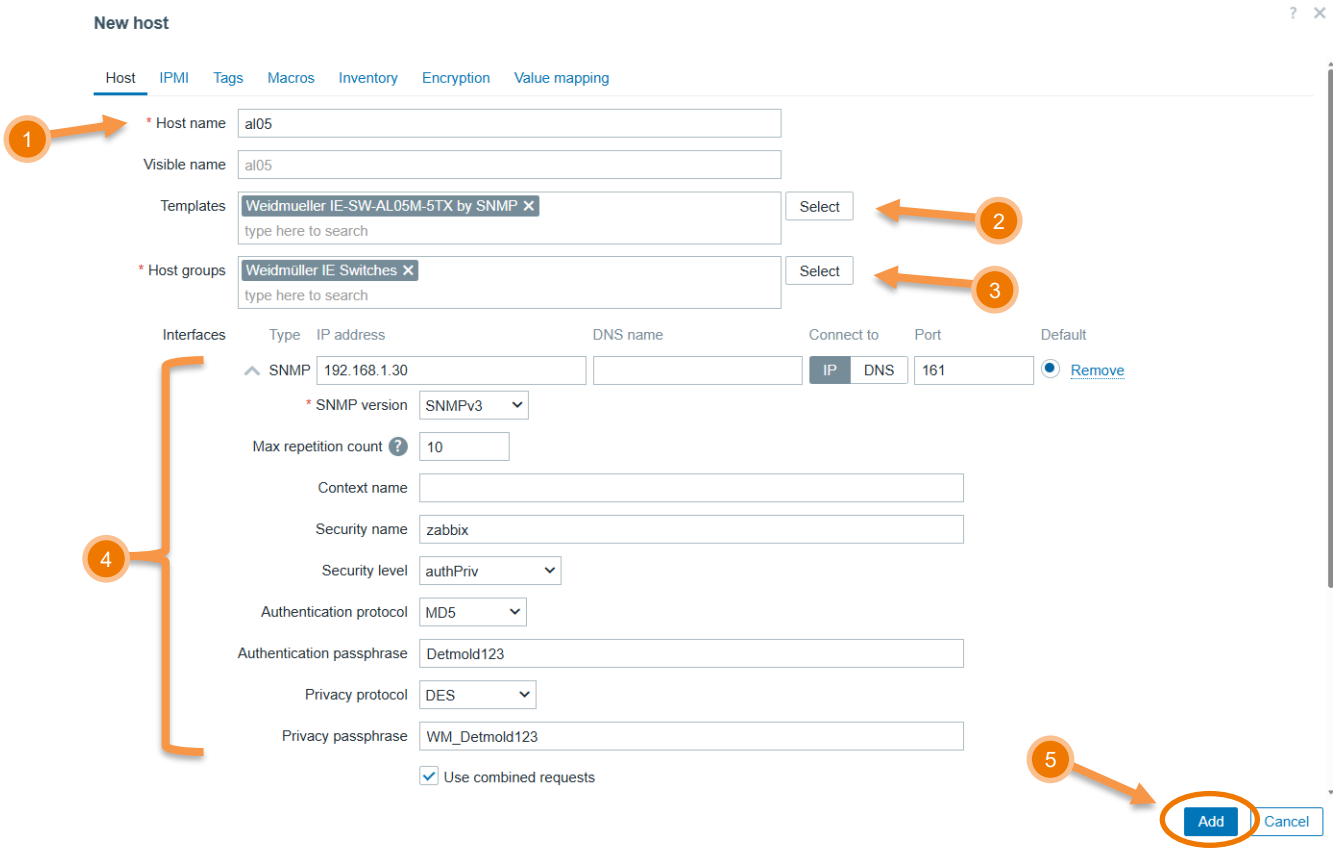
Manually Adding Hosts

To add every Weidmueller Industrial Ethernet device this procedure must be repeated for each one. In the following, the manual attachment of IE-SW-AL05LM-5TX to Zabbix is shown.

1. Navigate to “Data collection” → “Hosts”. Press “Create host”.



2. Type in a name (1), add the previously created template (2), select the desired host group (3) and fill in the SNMPv3 credentials (4) by pressing the blue underlined “Add”. Finally, click “Add” (5).

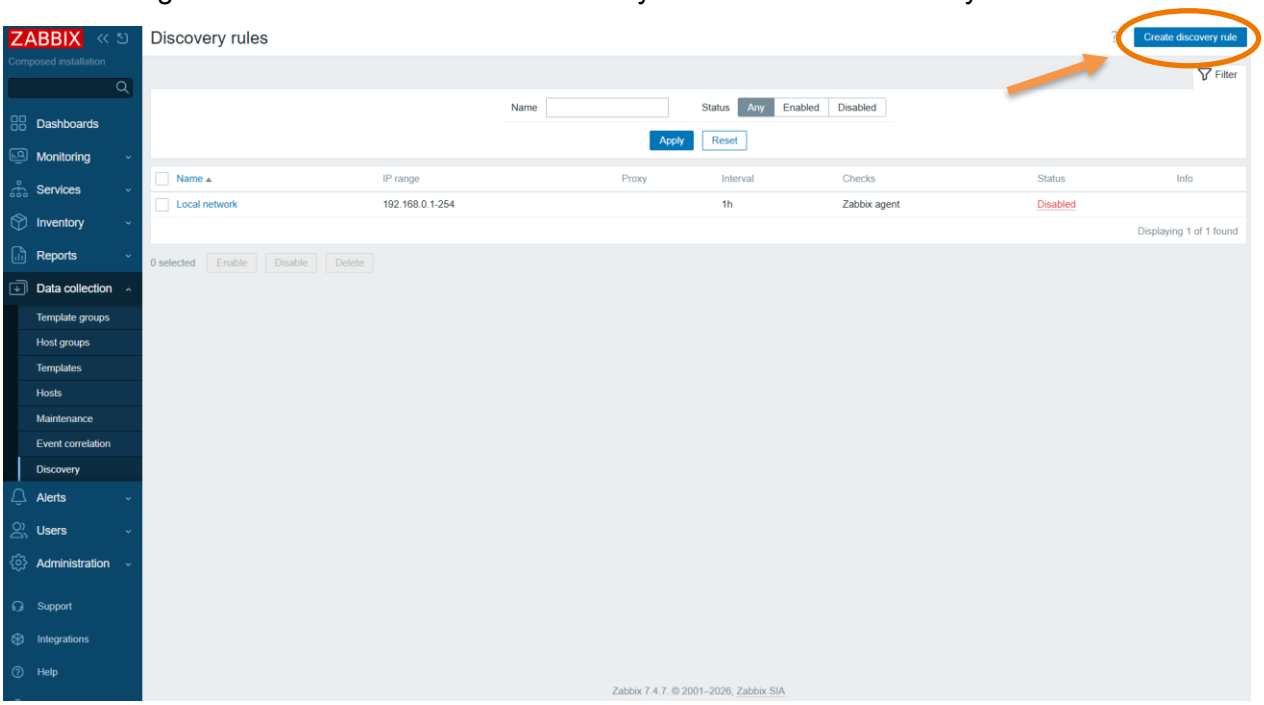


You can now proceed to **Configure Dashboard**.

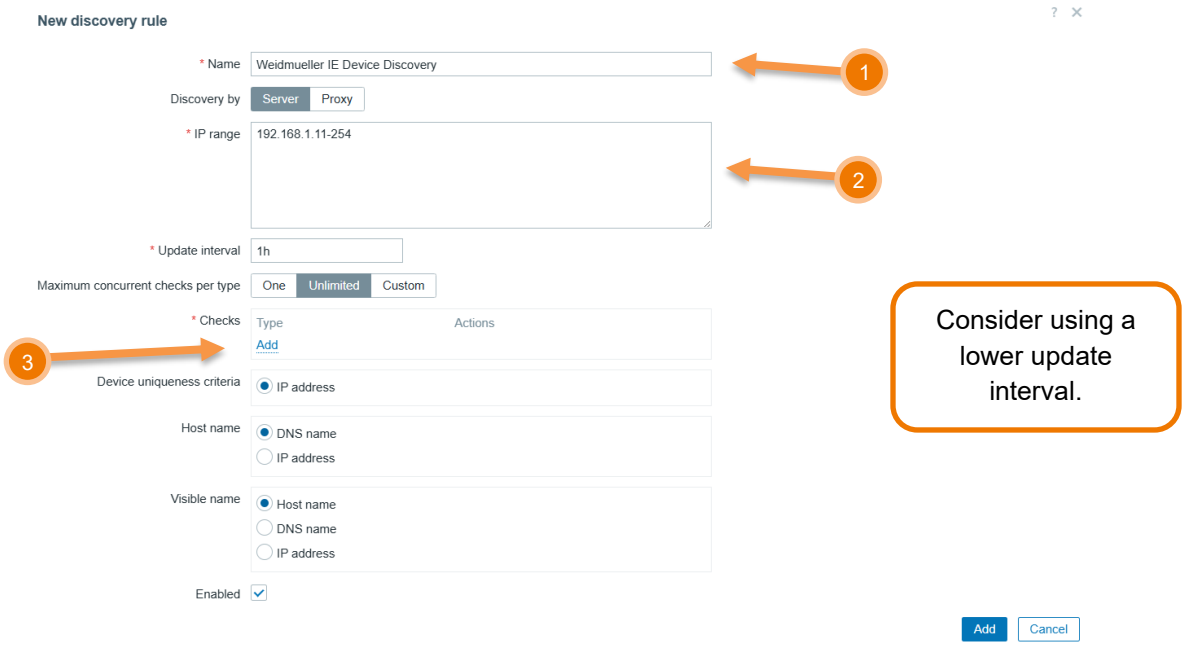
Configuring Automatic Host Discovery

Automatic host discovery reduces manual effort by continuously scanning your network for new Weidmueller Industrial Ethernet devices and creating hosts automatically based on defined rules.

1. Navigate to “Data collection” → “Discovery”. Click “Create discovery rule”.

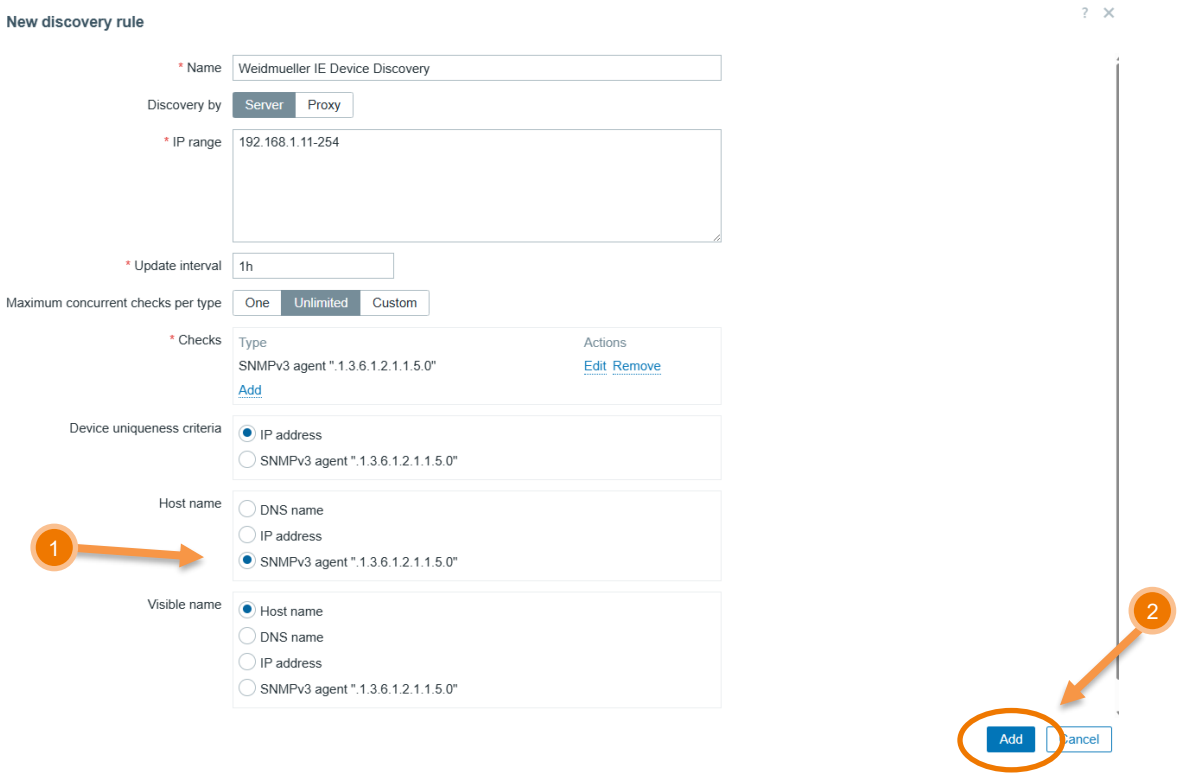


2. Type in a name for the rule (1) and set a desired IP range (2). In this application a range from 192.168.1.11-254 is set. Press the blue underlined “Add” (3).



3. Set check type to SNMPv3 agent (1). It is beneficial to query the `sysName` of a discovery Industrial Ethernet device as it could be saved as the host name in Zabbix. Thus, type in .1.3.6.1.2.1.1.5.0 in the “SNMP OID” text box (2). Fill in the SNMP credentials (3). Click “Add” to continue (4).

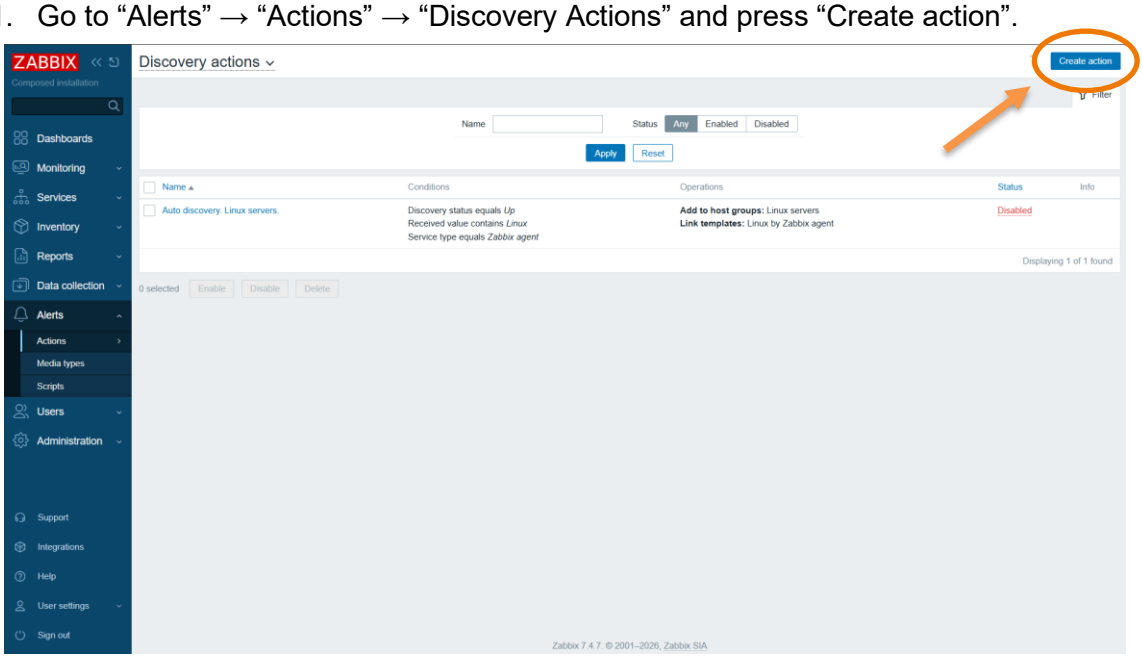
4. Choose “SNMPv3 agent .1.3.6.1.2.1.1.5” as host name to link the queried result to the discovered Industrial Ethernet device (1). Press “Add” to proceed (2).



Configuring Action Rules based on discovered Devices

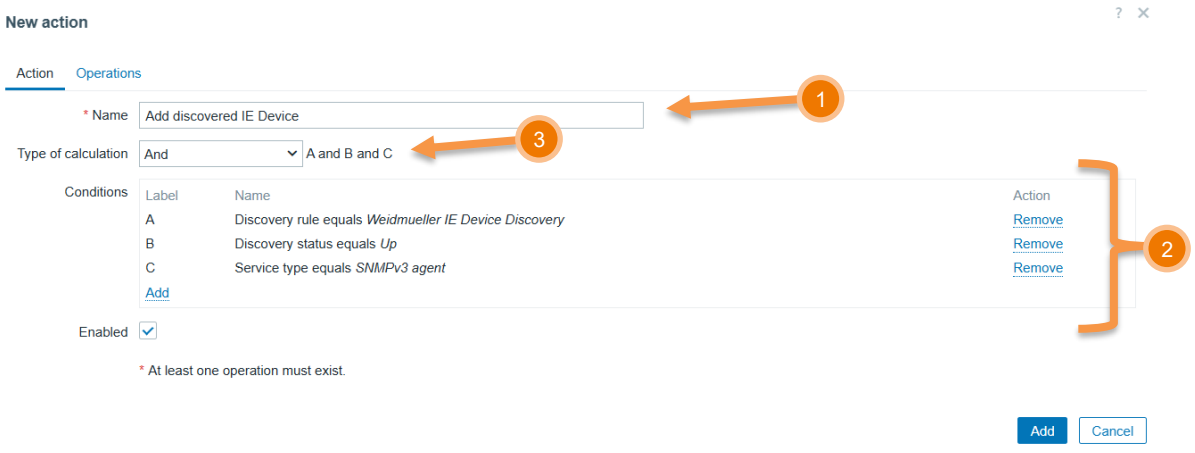
With device discovery in place, the next step is configuring Action Rules that automatically process newly found Industrial Ethernet devices and assign them to the correct groups and templates. The required actions are to automatically add each discovered device as a host, assign it to the correct host group and link the appropriate template based on the device's discovered `sysName`.

1. Go to "Alerts" → "Actions" → "Discovery Actions" and press "Create action".



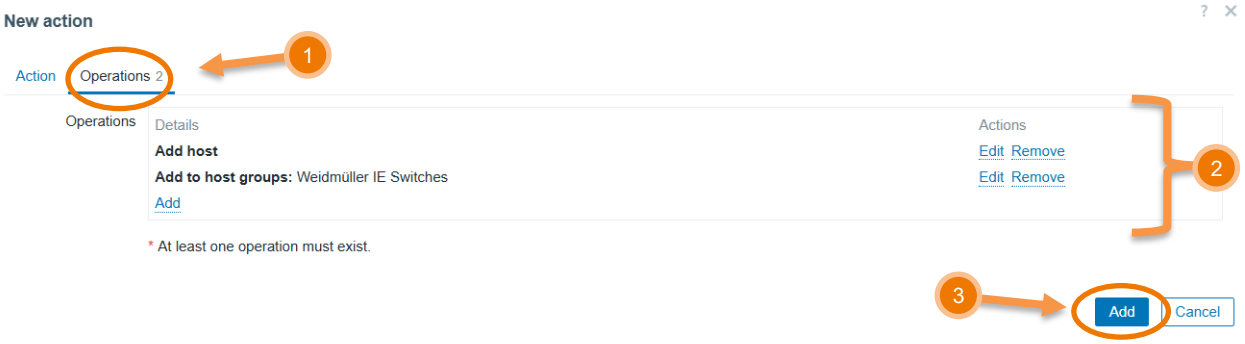
2. The goal of this new action rule is that it should add discovered IE devices as hosts to Zabbix. Additionally, the host should be organized into a host group. Type in a name for this action (1) and add the following conditions by pressing the blue underlined "Add" (2):
   a. Discovery rule = Previously determined discovery rule
   b. Discovery status = Up
   c. Service type = SNMPv3 agent

   Finally choose "And" as type of calculation (3).



3. Now that the conditions have been configured, the resulting operations have to be determined too. Switch to “Operations” (1). Add the following operations by pressing the blue underlined “Add” (2):
    a. Add host
    b. Add to host group = Previously created host group for your Industrial Ethernet devices

    Finish the configuration by pressing “Add” (3)

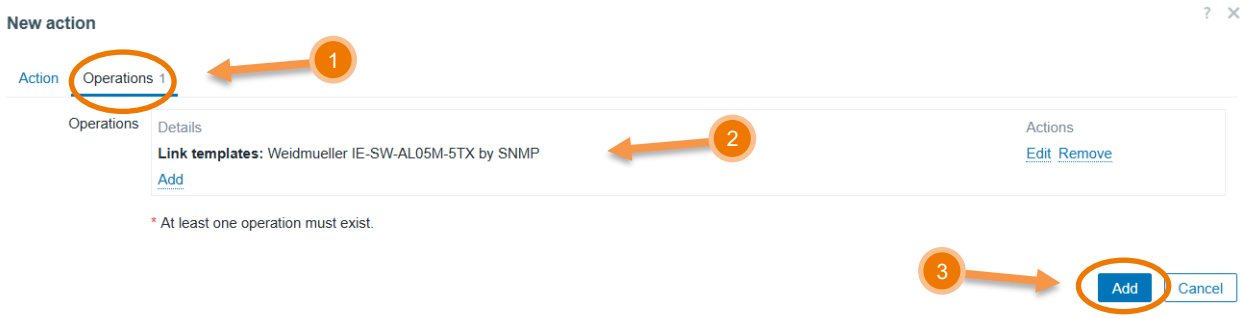


The next step is to link the appropriate template based on the device’s `sysName`. The following example shows the rule for the IE-SW-AL05-5TX, but the same procedure must be repeated for the IE-SW-AL08-8TX as well.

4. Add another action rule by pressing “Create action” (see step 1). Type in a name for this action (1) and add the following conditios by clicking the blue underlined “Add” (2):
    a. Discovery rule = Previously created discovery rule
    b. Discovery status = Up
    c. Service type = SNMPv3 agent
    d. Received value contains “AL05”

    Choose “And” as type of calculation (3).

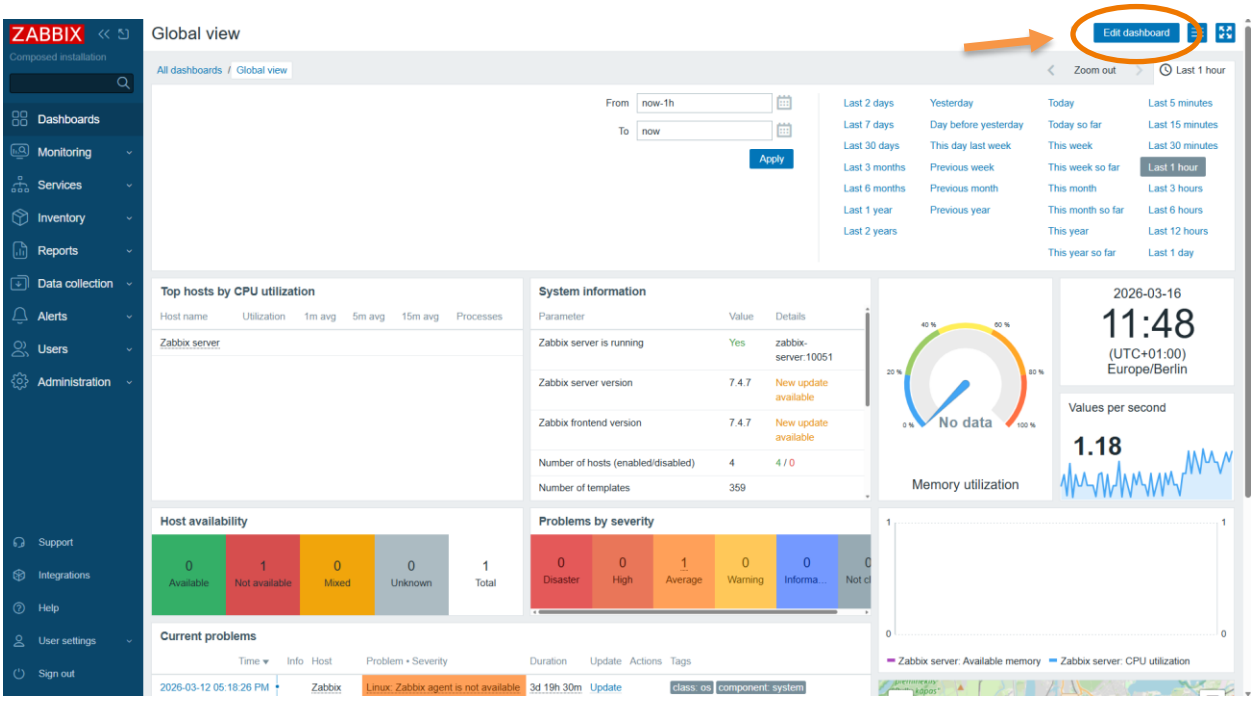
5. Switch to “Operations” (1) and add “Link templates” by clicking the blue underlined “Add” (2). Finally, press “Add” (3).



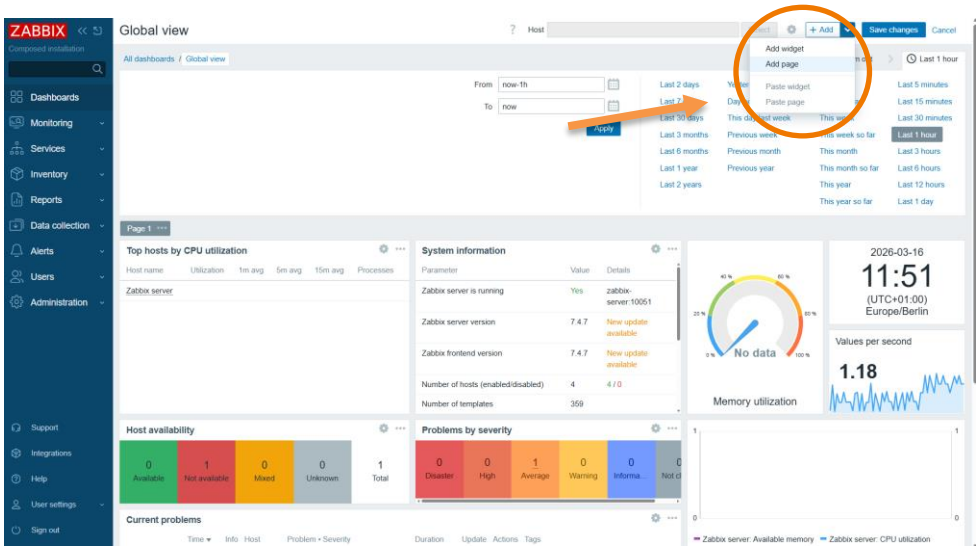
Configuring the Dashboard

Whether the Weidmueller Industrial Ethernet devices were added manually or through automatic discovery, the next step is to configure the dashboard, so the collected data becomes clearly visible and easy to interpret. Thus, you must open a new empty page and add desired widgets. The following procedure needs to be repeated for each Industrial Ethernet device.

1. Head over to “Dashboard” and press “Edit dashboard”.

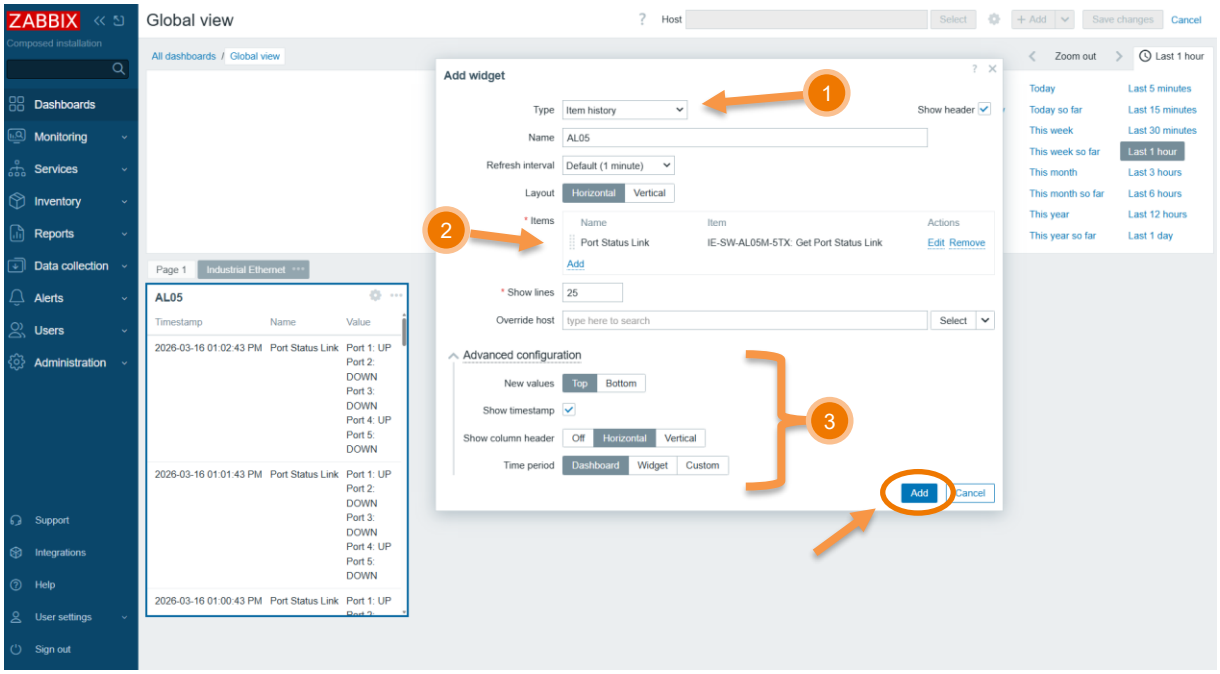


2. Open the drop-down menu by clicking the arrow, facing downwards. Click “Add page”.



3. Type in a name and press “Apply”.

4. Click on the dashboard and choose “Item history” (1). Type in a name, add the desired item (2) by searching for the Industrial Ethernet device to be added and set in advanced configuration “Show timestamp” and “Show column header” to “Horizontal” (3).



Useful Widgets

An overview of all available Industrial Ethernet devices can be reached over the “Host availability” widget. In the configuration windows, tick “Interface type” SNMP and select your host group.