

SNMP & Monitoring w/ Grafana

What is Monitoring / Network Monitoring?

Network monitoring is the process of continuously observing network devices, services, and traffic to understand their availability, health, and performance.

A network administrator might want to know:

- Is a device online?
- Are its network interfaces up?
- How much network traffic is it handling?
- Is it experiencing errors or dropped packets?
- How much CPU, memory, or storage is it using?
- Has any of this changed over time?

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: `<hidden-for-public-export>`

`Ctrl-Shift-V` to paste in terminal

Why monitor?

Without monitoring, administrators may not know something is wrong until a user reports it.

Monitoring allows administrators to:

- Detect problems earlier
- Identify unusual behavior
- Compare current performance against normal performance
- Track changes over time
- Make decisions about capacity and upgrades

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: `<hidden-for-public-export>`

`Ctrl-Shift-V` to paste in terminal

ip command

ip addr show

ip -s link

```
itclub@monk:~ $ ip -s link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    RX:  bytes  packets  errors  dropped  missed  mcast
         62544      772       0        0        0        0
    TX:  bytes  packets  errors  dropped  carrier  collsns
         62544      772       0        0        0        0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP mode DEFAULT group default qlen 1
    link/ether d8:3a:dd:f8:f2:cf brd ff:ff:ff:ff:ff:ff
    RX:  bytes  packets  errors  dropped  missed  mcast
        411962531  307160       0         7        0    4325
    TX:  bytes  packets  errors  dropped  carrier  collsns
        6213265    66544       0         0        0        0
3: wlan0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1
    link/ether d8:3a:dd:f8:f2:d1 brd ff:ff:ff:ff:ff:ff
    RX:  bytes  packets  errors  dropped  missed  mcast
           0         0         0         0         0         0
    TX:  bytes  packets  errors  dropped  carrier  collsns
           0         0         0         0         0         0
itclub@monk:~ $
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: <hidden-for-public-export>

Ctrl-Shift-V to paste in terminal

SNMP

SNMP stands for ***Simple Network Management Protocol***.

Package installation:

```
sudo apt install snmp snmpd snmp-mibs-downloader
```

Verify installation:

```
sudo systemctl status snmpd
```

```
sudo ss -ulpn | grep snmpd
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: <hidden-for-public-export>

Ctrl-Shift-V to paste in terminal

Object Identifiers (OIDs)

An **OID** is a globally unique identifier that represents a specific piece of information in the SNMP-managed device.

```
1 - iso
├── .3 - org
│   ├── .6 - dod
│       ├── .1 - internet
│           ├── .2 - mgmt
│               ├── .1 - mib-2
│                   ├── .1 - system
│                       ├── .1 - sysDescr
│                       ├── .3 - sysUpTime
│                       ├── .5 - sysName
│                   └── .2 - interfaces
```

For example, **.1.3.6.1.2.1.1.5.0** is the OID for the system name of the device.

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: `<hidden-for-public-export>`

Ctrl-Shift-V to paste in terminal

Management Information Bases (MIBs)

A Management Information Base (MIB) defines and describes objects that can be managed through SNMP. MIBs give meaningful names to OIDs.

Instead of remembering:

```
.1.3.6.1.2.1.1.5.0
```

we can use:

```
SNMPv2-MIB::sysName.0
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: `<hidden-for-public-export>`

Ctrl-Shift-V to paste in terminal

snmpd Configuration

```
sudo vim /etc/snmp/snmpd.conf
```

```
view systemonly included .1.3.6.1.2.1.1
view systemonly included .1.3.6.1.2.1.2
view systemonly included .1.3.6.1.2.1.25
view systemonly included .1.3.6.1.2.1.31
```

```
sudo systemctl restart snmpd
```

```
sudo systemctl status snmpd
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: <hidden-for-public-export>

Ctrl-Shift-V to paste in terminal

snmpget

```
snmpget -v2c -c public localhost SNMPv2-MIB::sysName.0
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: <hidden-for-public-export>

Ctrl-Shift-V to paste in terminal

snmpwalk

```
snmpwalk -v2c -c public localhost IF-MIB::ifDescr
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: <hidden-for-public-export>

Ctrl-Shift-V to paste in terminal

Tables and Indexes

```
itclub@monk:~ $ snmpwalk -v2c -c public localhost IF-MIB::ifDescr
IF-MIB::ifDescr.1 = STRING: lo
IF-MIB::ifDescr.2 = STRING: eth0
IF-MIB::ifDescr.3 = STRING: wlan0
```

```
itclub@monk:~ $ snmpwalk -v2c -c public localhost IF-MIB::ifOperStatus
IF-MIB::ifOperStatus.1 = INTEGER: up(1)
IF-MIB::ifOperStatus.2 = INTEGER: up(1)
IF-MIB::ifOperStatus.3 = INTEGER: down(2)
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: `<hidden-for-public-export>`

Ctrl-Shift-V to paste in terminal

More Queries of Interest

```
snmpwalk -v2c -c public localhost IF-MIB::ifOperStatus
snmpwalk -v2c -c public localhost IF-MIB::ifHighSpeed
snmpwalk -v2c -c public localhost IF-MIB::ifHCInOctets
snmpwalk -v2c -c public localhost IF-MIB::ifHCOutOctets
snmpwalk -v2c -c public localhost IF-MIB::ifInErrors
snmpwalk -v2c -c public localhost IF-MIB::ifOutErrors
snmpwalk -v2c -c public localhost IF-MIB::ifInDiscards
snmpwalk -v2c -c public localhost IF-MIB::ifOutDiscards
snmpwalk -v2c -c public localhost HOST-RESOURCES-MIB::hrProcessorLoad
snmpwalk -v2c -c public localhost HOST-RESOURCES-MIB::hrStorageDescr
snmpwalk -v2c -c public localhost HOST-RESOURCES-MIB::hrStorageSize
snmpwalk -v2c -c public localhost HOST-RESOURCES-MIB::hrStorageUsed
snmpwalk -v2c -c public localhost HOST-RESOURCES-MIB::hrStorageAllocationUnits
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: *<hidden-for-public-export>*

Ctrl-Shift-V to paste in terminal

The Problem With What We're Doing

Manually retrieving system information using SNMP commands like `snmpget` and `snmpwalk` is not scalable. For a single device, and for looking at the system's status at a specific point in time, it might be feasible. However, this approach quickly becomes cumbersome as the number of devices increases or when continuous monitoring is required.

We need something to:

- Query SNMP automatically
- Query it repeatedly at regular intervals over time
- Store the collected data
- Calculate changes over time
- Visualize the results

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: `<hidden-for-public-export>`

`Ctrl-Shift-V` to paste in terminal

Docker

```
docker --version
```

```
docker compose version
```

```
cat /etc/group | grep docker
```

```
sudo usermod -aG docker $USER
```

```
newgrp docker
```

```
docker run hello-world
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: *<hidden-for-public-export>*

Ctrl-Shift-V to paste in terminal

Telegraf, InfluxDB, and Grafana Configuration

```
mkdir -p ~/snmp-monitoring/telegraf
```

```
cd ~/snmp-monitoring
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: *<hidden-for-public-export>*

Ctrl-Shift-V to paste in terminal

Telegraf, InfluxDB, and Grafana Configuration

vim .env

```
INFLUXDB_USERNAME=buildit  
INFLUXDB_PASSWORD=buildit-monitoring  
INFLUXDB_ORG=buildit  
INFLUXDB_BUCKET=network  
INFLUXDB_TOKEN=buildit-monitoring-token
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: *<hidden-for-public-export>*

Ctrl-Shift-V to paste in terminal

Telegraf, InfluxDB, and Grafana Configuration

vim telegraf/telegraf.conf

```
[agent]
  interval = "10s"

inputs.snmp
  agents = ["udp://127.0.0.1:161"]
  version = 2
  community = "public"

  inputs.snmp.field
    name = "hostname"
    oid = "SNMPv2-MIB::sysName.0"

  inputs.snmp.field
    name = "uptime"
    oid = "SNMPv2-MIB::sysUpTime.0"

  inputs.snmp.table
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: <hidden-for-public-export>

Ctrl-Shift-V to paste in terminal

Telegraf, InfluxDB, and Grafana Configuration

vim telegraf/Dockerfile

```
# Download the standard MIB definitions
FROM debian:bookworm AS mibs

RUN sed -i 's/Components: main/Components: main non-free/' \
    /etc/apt/sources.list.d/debian.sources \
    && apt-get update \
    && apt-get install -y --no-install-recommends snmp-mibs-downloader

# Telegraf image
FROM telegraf:latest

USER root

COPY --from=mibs /var/lib/mibs /var/lib/mibs

RUN mkdir -p /usr/share/snmp/mibs \
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: <hidden-for-public-export>

Ctrl-Shift-V to paste in terminal

Telegraf, InfluxDB, and Grafana Configuration

vim compose.yml

```
services:
  influxdb:
    image: influxdb:2
    container_name: influxdb
    restart: unless-stopped
    ports:
      - "127.0.0.1:8086:8086"
    environment:
      DOCKER_INFLUXDB_INIT_MODE: setup
      DOCKER_INFLUXDB_INIT_USERNAME: ${INFLUXDB_USERNAME}
      DOCKER_INFLUXDB_INIT_PASSWORD: ${INFLUXDB_PASSWORD}
      DOCKER_INFLUXDB_INIT_ORG: ${INFLUXDB_ORG}
      DOCKER_INFLUXDB_INIT_BUCKET: ${INFLUXDB_BUCKET}
      DOCKER_INFLUXDB_INIT_ADMIN_TOKEN: ${INFLUXDB_TOKEN}
    volumes:
      - influxdb-data:/var/lib/influxdb2
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: *<hidden-for-public-export>*

Ctrl-Shift-V to paste in terminal

Telegraf, InfluxDB, and Grafana Configuration

```
docker compose up -d
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: *<hidden-for-public-export>*

Ctrl-Shift-V to paste in terminal

Grafana Web Interface

1. Open your web browser and navigate to `http://localhost:3000`.
2. Log in using the default username and password:
 - Username: `admin`
 - Password: `admin`
3. You will be prompted to change the password after the first login.
4. Add `InfluxDB` as a data source in Grafana using the credentials and bucket name specified in your `.env` file.
 - Query language: `Flux`
 - URL: `http://influxdb:8086`
 - Organization: `buildit`
 - Token: `buildit-monitoring-token`
 - Default Bucket: `network`
5. Click `Save & Test`
6. Create a Dashboard
7. Add Panels (queries in Flux)

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: `<hidden-for-public-export>`

`Ctrl-Shift-V` to paste in terminal

Test Your Dashboard

Use CloudFlare's speed test to generate network traffic:

```
curl -o /dev/null  
https://speed.cloudflare.com/__down?bytes=100000000
```

Make Python eat our memory:

```
python3 -c "import time; x = b'x' * 3_000_000_000;  
time.sleep(60)"
```

Project Page: <https://suddenlysixam.club/guides/snmp-grafana/>

Raspberry Pi Password: *<hidden-for-public-export>*

Ctrl-Shift-V to paste in terminal