

Documentation & Guide

By Harshith Sudar
21BCS097

Installing Conpot

Conpot is an open-source ICS/SCADA honeypot that allows you to simulate industrial control systems. Here's a step-by-step guide on how to install Conpot:

1. First, make sure you have Python installed on your system. Conpot requires Python 3.6 or later.

2. Make sure you have docker installed before or install it by following the given steps :

Installing Docker

Uninstall old versions

Before you can install Docker Engine, you must first make sure that any conflicting packages are uninstalled.

Distro maintainers provide unofficial distributions of Docker packages in APT. You must uninstall these packages before you can install the official version of Docker Engine.

The unofficial packages to uninstall are:

- `docker.io`
- `docker-compose`
- `docker-doc`
- `podman-docker`

Moreover, Docker Engine depends on `containerd` and `runc`. Docker Engine bundles these dependencies as one bundle: `containerd.io`. If you have installed the `containerd` or `runc` previously, uninstall them to avoid conflicts with the versions bundled with Docker Engine.

Run the following command to uninstall all conflicting packages:

```
$ for pkg in docker.io docker-doc docker-compose podman-docker containerd runc; do sudo apt-get remove $pkg; done
```

`apt-get` might report that you have none of these packages installed.

Images, containers, volumes, and networks stored in `/var/lib/docker/` aren't automatically removed when you uninstall Docker. If you want to start with a clean installation, and prefer to clean up any existing data, read the [uninstall Docker Engine](#) section.

Installation methods

You can install Docker Engine in different ways, depending on your needs:

- Docker Engine comes bundled with [Docker Desktop for Linux](#). This is the easiest and quickest way to get started.
- Set up and install Docker Engine from [Docker's Apt repository](#).
- [Install it manually](#) and manage upgrades manually.
- Use a [convenience script](#). Only recommended for testing and development environments.

Install using the Apt repository

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker repository. Afterward, you can install and update Docker from the repository.

a) Set up Docker's Apt repository.

Add Docker's official GPG key:

```
sudo apt-get update
```

```
sudo apt-get install ca-certificates curl gnupg
```

```
sudo install -m 0755 -d /etc/apt/keyrings
```

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o  
/etc/apt/keyrings/docker.gpg
```

```
sudo chmod a+r /etc/apt/keyrings/docker.gpg
```

Add the repository to Apt sources:

```
echo \
```

```
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg]  
https://download.docker.com/linux/ubuntu \
```

```
"$(. /etc/os-release && echo "$VERSION_CODENAME)" stable" | \
```

```
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

b) Install the Docker packages.

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io  
docker-buildx-plugin docker-compose-plugin
```

c) Verify that the Docker Engine installation is successful by running the `hello-world` image.

```
$ sudo docker run hello-world
```

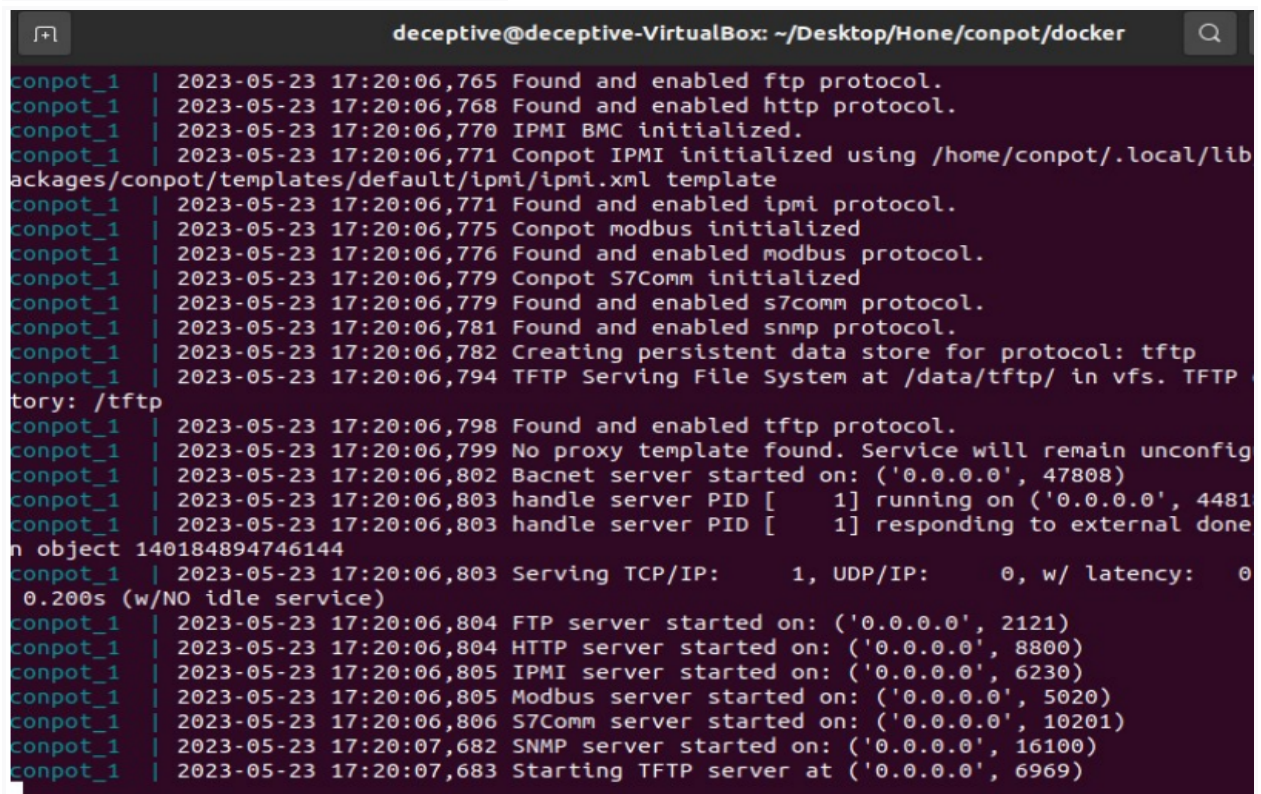
This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

You have now successfully installed and started Docker Engine.

3) Now build 'Conpot' from source and run with docker composer (Use the following code or else it may produce error)

```
$ sudo docker run -it -p 81:81 -p 103:103 -p 503:503 -p 160:160/udp --network=bridge  
honeynet/conpot
```

It should show like the below image



```
deceptive@deceptive-VirtualBox: ~/Desktop/Hone/conpot/docker  
conpot_1 | 2023-05-23 17:20:06,765 Found and enabled ftp protocol.  
conpot_1 | 2023-05-23 17:20:06,768 Found and enabled http protocol.  
conpot_1 | 2023-05-23 17:20:06,770 IPMI BMC initialized.  
conpot_1 | 2023-05-23 17:20:06,771 Conpot IPMI initialized using /home/conpot/.local/lib  
ackages/conpot/templates/default/ipmi/ipmi.xml template  
conpot_1 | 2023-05-23 17:20:06,771 Found and enabled ipmi protocol.  
conpot_1 | 2023-05-23 17:20:06,775 Conpot modbus initialized  
conpot_1 | 2023-05-23 17:20:06,776 Found and enabled modbus protocol.  
conpot_1 | 2023-05-23 17:20:06,779 Conpot S7Comm initialized  
conpot_1 | 2023-05-23 17:20:06,779 Found and enabled s7comm protocol.  
conpot_1 | 2023-05-23 17:20:06,781 Found and enabled snmp protocol.  
conpot_1 | 2023-05-23 17:20:06,782 Creating persistent data store for protocol: tftp  
conpot_1 | 2023-05-23 17:20:06,794 TFTP Serving File System at /data/tftp/ in vfs. TFTP  
tory: /tftp  
conpot_1 | 2023-05-23 17:20:06,798 Found and enabled tftp protocol.  
conpot_1 | 2023-05-23 17:20:06,799 No proxy template found. Service will remain unconfig  
conpot_1 | 2023-05-23 17:20:06,802 Bacnet server started on: ('0.0.0.0', 47808)  
conpot_1 | 2023-05-23 17:20:06,803 handle server PID [ 1] running on ('0.0.0.0', 4481  
conpot_1 | 2023-05-23 17:20:06,803 handle server PID [ 1] responding to external done  
n object 140184894746144  
conpot_1 | 2023-05-23 17:20:06,803 Serving TCP/IP: 1, UDP/IP: 0, w/ latency: 0  
0.200s (w/NO idle service)  
conpot_1 | 2023-05-23 17:20:06,804 FTP server started on: ('0.0.0.0', 2121)  
conpot_1 | 2023-05-23 17:20:06,804 HTTP server started on: ('0.0.0.0', 8800)  
conpot_1 | 2023-05-23 17:20:06,805 IPMI server started on: ('0.0.0.0', 6230)  
conpot_1 | 2023-05-23 17:20:06,805 Modbus server started on: ('0.0.0.0', 5020)  
conpot_1 | 2023-05-23 17:20:06,806 S7Comm server started on: ('0.0.0.0', 10201)  
conpot_1 | 2023-05-23 17:20:07,682 SNMP server started on: ('0.0.0.0', 16100)  
conpot_1 | 2023-05-23 17:20:07,683 Starting TFTP server at ('0.0.0.0', 6969)
```

Customization

The default profile

Conpot is shipped with a default profile(`default.xml`) which provides basic emulation of a [Siemens S7-200 CPU](#) with a few expansion modules installed. The attack surface of the default emulation includes the protocols MODBUS, HTTP, SNMP and s7comm.

While most of the configuration takes place within the XML profile, some parts are kept in separate folders within the templates directory to avoid clutter.

For further information on customization please go through the following :

<https://conpot.readthedocs.io/en/latest/usage/customization.html>

Modbus

The `<device_info />` section allows to define the device info returned to a Modbus 43 function call.

The `<slave />` section allows you to define the slaves. Every slave definition is separated into `<blocks />`.

An binary output block has the type `COILS`, binary input blocks `DISCRETE_INPUTS`. You define the starting address and size. `ANALOG_INPUTS` hold data in byte size.

In the `<values />` section you take the starting address and fill the field with values. The content is evaluated so you can easily fill it with random values.

```
<block name="a">
  <!-- COILS/DISCRETE_OUTPUTS aka. binary output, power on/power off
        Here we map modbus addresses 1 to 127 to S7-200 PLC Addresses Q0.0 to Q15.7
  -->
  <type>COILS</type>
  <starting_address>1</starting_address>
```

```

    <size>128</size>
    <values>
      <value>
        <address>1</address>
        <!-- Will be parsed with eval() -->
        <content>[random.randint(0,1) for b in range(0,128)]</content>
      </value>
    </values>
  </block>

```

`HOLDING_REGISTERS` can be considered as temporary data storage. You define them with the starting address and their size. Holding registers don't have any initial value.

SNMP

In the `<snmp />` section you define a management information base (MIB). MIBs consist of a `<symbol>` with a name attribute, and its `<value>`.

```

<symbol name="sysDescr">
  <value>Siemens, SIMATIC, S7-200</value>
</symbol>

```

In the following example we will show how to include other MIBs. As an example we will add the `ifNumber` symbol from the IF-MIB. First we have to download the IF-MIB and also the IANAifType-MIB since IF-MIB depends on this:

```
wget http://www.iana.org/assignments/ianaiftype-mib/ianaiftype-mib
```

```
wget ftp://ftp.cisco.com/pub/mibs/v2/IF-MIB.my
```

Installing Grafana

Install Grafana on Debian or Ubuntu

This topic explains how to install Grafana dependencies, install Grafana on Linux Debian or Ubuntu, and start the Grafana server on your Debian or Ubuntu system.

There are multiple ways to install Grafana: using the Grafana Labs APT repository, by downloading a `.deb` package, or by downloading a binary `.tar.gz` file. Choose only one of the methods below that best suits your needs.

Note: If you install via the `.deb` package or `.tar.gz` file, then you must manually update Grafana for each new version.

Install from APT repository

If you install from the APT repository, Grafana automatically updates when you run `apt-get update`.

Complete the following steps to install Grafana from the APT repository:

1) Install the prerequisite packages:

```
sudo apt-get install -y apt-transport-https software-properties-common wget
```

2) Import the GPG key:

```
echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
```

3) To add a repository for stable releases, run the following command:

```
echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list
```

4)To add a repository for beta releases, run the following command:

```
echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com beta main"  
| sudo tee -a /etc/apt/sources.list.d/grafana.list
```

5)Run the following command to update the list of available packages:

```
# Updates the list of available packages  
sudo apt-get update
```

6)To install Grafana OSS, run the following command:

```
# Installs the latest OSS release:  
sudo apt-get install grafana
```

7)To install Grafana Enterprise, run the following command:

```
# Installs the latest Enterprise release:  
sudo apt-get install grafana-enterprise
```

Install Grafana using a deb package or as a standalone binary

If you choose not to install Grafana using APT, you can download and install Grafana using the deb package or as a standalone binary.

Complete the following steps to install Grafana using DEB or the standalone binaries:

Navigate to the [Grafana download page](#).

Select the Grafana version you want to install.

The most recent Grafana version is selected by default.

The Version field displays only tagged releases. If you want to install a nightly build, click Nightly Builds and then select a version.

Select an Edition.

Enterprise: This is the recommended version. It is functionally identical to the open source version, but includes features you can unlock with a license, if you so choose.

Open Source: This version is functionally identical to the Enterprise version, but you will need to download the Enterprise version if you want Enterprise features.

Depending on which system you are running, click the Linux or ARM tab on the [download page](#).

Copy and paste the code from the [download page](#) into your command line and run.

Start the Grafana server

This topic includes instructions for starting the Grafana server. For certain configuration changes, you might have to restart the Grafana server for them to take effect.

The following instructions start the `grafana-server` process as the `grafana` user, which was created during the package installation.

If you installed with the APT repository or `.deb` package, then you can start the server using `systemd` or `init.d`. If you installed a binary `.tar.gz` file, then you execute the binary.

Linux

The following subsections describe three methods of starting and restarting the Grafana server: with `systemd`, `initd`, or by directly running the binary. You should follow only one set of instructions, depending on how your machine is configured.

Start the Grafana server with `systemd`

Complete the following steps to start the Grafana server using `systemd` and verify that it is running.

1) To start the service, run the following commands:

```
sudo systemctl daemon-reload
sudo systemctl start grafana-server
sudo systemctl status grafana-server
```

2) To verify that the service is running, run the following command:

```
sudo systemctl status grafana-server
```

Configure the Grafana server to start at boot using systemd

To configure the Grafana server to start at boot, run the following command:

```
sudo systemctl enable grafana-server.service
```

Sign in to Grafana

This topic describes how to sign in to Grafana.

Before you begin

- [Install Grafana](#)

Steps

To sign in to Grafana for the first time, follow these steps:

Open your web browser and go to root URL specified in [Grafana configuration file](#).

Unless you have configured Grafana differently, it is set to use `http://localhost:3000` by default.

On the signin page, enter `admin` for username and password.

Click Sign in.

If successful, you will see a prompt to change the password.

Click OK on the prompt and change your password.

Set up Grafana monitoring

Grafana supports tracing.

Grafana can emit Jaeger or OpenTelemetry Protocol (OTLP) traces for its HTTP API endpoints and propagate Jaeger and [w3c Trace Context](#) trace information to compatible data sources. All HTTP endpoints are logged evenly (annotations, dashboard, tags, and so on). When a trace ID is propagated, it is reported with operation 'HTTP /datasources/proxy/:id/*'.

Refer to [Configuration's OpenTelemetry section](#) for a reference of tracing options available in Grafana.

View Grafana internal metrics

Grafana collects some metrics about itself internally. Grafana supports pushing metrics to Graphite or exposing them to be scraped by Prometheus.

For more information about configuration options related to Grafana metrics, refer to [metrics](#) and [metrics.graphite](#) in [Configuration](#).

Available metrics

When enabled, Grafana exposes a number of metrics, including:

- Active Grafana instances
- Number of dashboards, users, and playlists
- HTTP status codes
- Requests by routing group
- Grafana active alerts
- Grafana performance

Pull metrics from Grafana into Prometheus

These instructions assume you have already added Prometheus as a data source in Grafana.

1)Enable Prometheus to scrape metrics from Grafana. In your configuration file (`grafana.ini` or `custom.ini` depending on your operating system) remove the semicolon to enable the following configuration options:

```
# Metrics available at HTTP URL /metrics and /metrics/plugins/:pluginId
```

```
[metrics]
```

```
# Disable / Enable internal metrics
```

```
enabled          = true
```

```
# Disable total stats (stat_totals_*) metrics to be generated
```

```
disable_total_stats = false
```

2)Restart Grafana. Grafana now exposes metrics at <http://localhost:3000/metrics>.

3)Add the job to your `prometheus.yml` file. Example:

```
- job_name: 'grafana_metrics'
```

```
  scrape_interval: 15s
```

```
  scrape_timeout: 5s
```

```
  static_configs:
```

```
    - targets: ['localhost:3000']
```

4)Restart Prometheus. Your new job should appear on the Targets tab.

5)In Grafana, hover your mouse over the Configuration (gear) icon on the left sidebar and then click Data Sources.

6)Select the Prometheus data source.

7)On the Dashboards tab, Import the Grafana metrics dashboard. All scraped Grafana metrics are available in the dashboard.

Grafana data sources

Grafana comes with built-in support for many *data sources*. If you need other data sources, you can also install one of the many data source plugins. If the plugin you need doesn't exist, you can develop a custom plugin.

Each data source comes with a *query editor*, which formulates custom queries according to the source's structure. After you add and configure a data source, you can use it as an input for many operations, including:

- Query the data with [Explore](#).
- Visualize it in [panels](#).
- Create rules for [alerts](#).

This documentation describes how to manage data sources in general, and how to configure or query the built-in data sources. For other data sources, refer to the list of [datasource plugins](#).

To develop a custom plugin, refer to [Build a plugin](#).

Manage data sources

Only users with the [organization administrator role](#) can add or remove data sources. To access data source management tools in Grafana as an administrator, navigate to Configuration > Data Sources in the Grafana sidebar.

For details on data source management, including instructions on how to add data sources and configure user permissions for queries, refer to the [administration documentation](#).

Dashboards

Note: Looking for prebuilt Grafana dashboards? [Check out our full library of dashboards and more →](#)

A dashboard is a set of one or more [panels](#) organized and arranged into one or more rows. Grafana ships with a variety of panels making it easy to construct the right queries, and customize the visualization so that you can create the perfect dashboard for your need. Each panel can interact with data from any configured Grafana [data source](#).

Dashboard snapshots are static. Queries and expressions cannot be re-executed from snapshots. As a result, if you update any variables in your query or expression, it will not change your dashboard data.

Before you begin, ensure that you have configured a data source. See also:

- [Use dashboards](#)
- [Build dashboards](#)
- [Create dashboard folders](#)
- [Manage dashboards](#)
- [Public dashboards](#)
- [Annotations](#)
- [Playlist](#)
- [Reporting](#)
- [Version history](#)
- [Export and import](#)
- [JSON model](#)

Panels and visualizations

The *panel* is the basic visualization building block in Grafana. Each panel has a query editor specific to the data source selected in the panel. The query editor allows you to build a query that returns the data you want to visualize.

There are a wide variety of styling and formatting options for each panel. Panels can be dragged, dropped, and resized to rearrange them on the dashboard.

Before you add a panel, ensure that you have configured a data source.

- For more information about adding and managing data sources as an administrator, refer to [Data source management](#).
- For details about using specific data sources, refer to [Data sources](#).

This section includes the following sub topics:

- [Panel editor overview](#)
- [Configure panel options](#)
- [Configure standard options](#)
- [Visualizations](#)
- [Query and transform data](#)
- [Configure thresholds](#)
- [Configure a legend](#)
- [Configure data links](#)
- [Configure field overrides](#)
- [Configure value mappings](#)
- [Calculation types](#)
- [The panel inspect view](#)

Installing sflow-RT

Step 1: Install sFlow-RT

Follow the [download and installation instructions](#) for your platform.

Step 2: Install applications

Start off by installing the [browse-metrics](#) and [browse-flows](#) applications:

```
sudo /usr/local/sflow-rt/get-app.sh sflow-rt browse-metrics
```

```
sudo /usr/local/sflow-rt/get-app.sh sflow-rt browse-flows
```

Restart sFlow-RT to load the applications:

```
sudo systemctl restart sflow-rt
```

Note: If you are using Docker, the [sflow/prometheus](#) image includes these applications.

Step 3: Access user interface

The user interface can be accessed using a web browser. Connect to HTTP port 8008 on the host running sFlow-RT, for example <http://localhost:8008> if you are running the software on your laptop/desktop.

Step 4: Configure / deploy agents

[Agents](#) describes how to configure sFlow in existing network devices and/or deploy agents to monitor hosts, hypervisors, containers, Swarm and Kubernetes clusters. Use the sFlow-RT *Status* page to verify that sFlow telemetry is being received - the *sFlow Agents*, *sFlow Bytes* and *sFlow Packets* gauges should all have non-zero values.

Note: If sFlow is not being received, check the device configurations and ensure that any firewalls between the agents and the sFlow-RT host allow UDP port 6343 packets pass to the sFlow-RT host, see [Download and Install](#).

If you don't have immediate access to a network, [Real-time network and system metrics as a service](#) describes how to replay captured sFlow data to explore the capabilities of the software on your laptop. Alternatively, [sflow-rt/containerlab](#) includes projects that emulate leaf and spine networks, EVPN, and DDoS mitigation, that can be run on a laptop using [Docker Desktop](#).

Step 5: Explore data

Access the sFlow-RT user interface.

The *Apps* tab lists the two applications we installed, *browse-flows* and *browse-metrics*, and the green color of the buttons indicates both applications are healthy.

Click on the *browse-flows* button to open the application and trend traffic flows in real-time. Type *Keys* and a *Value* into the form and click *Submit* to start trending traffic. Click on peaks in the chart to see values at that time. Click on items in the chart legend to drill down by adding the item to the current *Filter*.

[Defining Flows](#) describes the flow analytics capability of sFlow-RT that can be explored using the *browse-flows* application.

Use your browser back button to return to the *Apps* page and click on the *browse-metrics* button to open the application and trend [metrics](#) in real-time.

User your browser back button to return to the *Apps* page and select the *API* tab. The API tab provides a link to [Writing Applications](#), an introductory article on programming sFlow-RT.

Next Steps

There are a number of [pre-built applications](#) available for sFlow-RT. For example, [DDoS protection quickstart guide](#) describes how to use sFlow-RT to detect and mitigate DDoS attacks and [Flow metrics with Prometheus and Grafana](#) describes import flow data into a time series databases to support operational dashboards.

Applications

The following command downloads and installs an application:

```
./sflow-rt/get-app.sh sflow-rt browse-flows
```

The following applications are currently available on GitHub:

| User | Application | Docker Image | Description |
|------|-------------|--------------|-------------|
| | | | |

| | | | |
|----------|-----------------------------------|-------------------------------------|---|
| sflow-rt | active-routes | sflow/active-routes | Real-time active BGP route cache |
| sflow-rt | browse-drops | sflow/prometheus | Browse and trend dropped packets |
| sflow-rt | browse-flows | sflow/prometheus | Browse and trend traffic flows |
| sflow-rt | browse-metrics | sflow/prometheus | Browse and trend metrics |
| sflow-rt | ddos-protect | sflow/ddos-protect | Real-time DDoS flood mitigation using BGP RTBH and FlowSpec |
| sflow-rt | fabric-metrics | sflow/topology | Leaf and spine fabric metrics |
| sflow-rt | ixp-metrics | sflow/ixp-metrics | Real-time monitoring of Internet eXchange Point (IXP) network metrics |
| sflow-rt | mininet-dashboard | | Real-time dashboard for Mininet |
| sflow-rt | particle | sflow/particle | Visualize real-time traffic using animated particles |
| sflow-rt | prometheus | sflow/prometheus | Export metrics to Prometheus time series database |
| sflow-rt | sflow-test | sflow/sflow-test | Test data center switch sFlow implementation |
| sflow-rt | sunburst | sflow/prometheus | Real-time protocol distribution as sunburst chart |
| sflow-rt | topology | sflow/topology | Persist and verify topology, locate addresses |

| | | | |
|----------|----------------------------|--------------------------------|--|
| sflow-rt | trace-flow | sflow/topology | Real-time traffic tracing against topology |
| sflow-rt | world-map | | Real-time traffic displayed on world map |

[Writing Applications](#) provides an introduction describing the structure of an sFlow-RT application. Post information on new applications to the [sFlow-RT group](#) to have them listed.