**SNIATM Swordfish Datadog Sample Integration Developer Documentation**

Contents:-

[Introduction: 2](#_Toc516072143)

[Purpose: 2](#_Toc516072144)

[Overview:- 2](#_Toc516072145)

[Datadog Agent: 2](#_Toc516072146)

[ The Collector: - 2](#_Toc516072147)

[ Dog-StatsD:- 2](#_Toc516072148)

[ The Forwarder :- 2](#_Toc516072149)

[Agent check:- 2](#_Toc516072150)

[Metrics:- 3](#_Toc516072151)

[Integration of Datadog dashboard :- 3](#_Toc516072152)

[Config files (YAML):- 3](#_Toc516072153)

[Checks (python) files:- 4](#_Toc516072154)

[Creation of Datadog dashboard: 5](#_Toc516072155)

[Time Board: 6](#_Toc516072156)

[ScreenBoard 8](#_Toc516072157)

# Introduction:

The Datadog Agent is a piece of software that runs on your hosts. Its job is to faithfully collect events and metrics and bring them to Datadog on your behalf so that you can do something useful with your monitoring and performance data. The Datadog Agent is open source, view the source code on GitHub for Agent v5 and Agent v6. To see all changes between Agent v5 and v6, consult our dedicated changes documentation

Datadog lets you collect all these metrics, events, and service states in one place. Then, visualize and correlate the data with beautiful graphs, and set flexible alerting conditions on it—all without running any storage or monitoring infrastructure yourself.

About Swordfish Dashboard and functionality

* Swordfish Dashboard Shows The capacity data and threshold values for different collections like Volumes, Storage pools and filesystems.
* Using different custom metrics, data-dog will collect all the required data and visualize it in to a Graph or Gauge.

Purpose:

The Datadog Agent is lightweight piece of software that runs on your hosts. Its job is to faithfully collect events and metrics and bring them to Datadog on your behalf so that you can do something useful with your monitoring and performance data.

# Overview:-

## Datadog Agent:

Datadog Agent is software that runs on hosts, which collects events and metrics for monitoring and performance of data. The agent has three main parts: collector, DogStatsD and forwarder.

* The Collector: - Collector runs checks on the current machine for whatever integrations you have and it captures system metrics such as memory and CPU.
* Dog-StatsD:- It is a statsd backend server you can send custom metrics to from an application.
* The Forwarder :- This retrieves data from both Dog-StatsD and the collector and then queues it up to be sent to Datadog

## Agent check:-

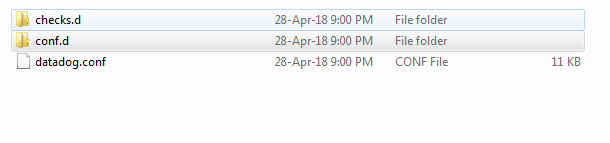
* + - Agent checks used to collect metrics from custom application or systems. It’s important to write integration for collecting all metrics available in an application.
    - In Agent Check interface, all the custom checks will inherit from /checks/ directory and it requires a check () method that takes one argument and an instance which is a dict having configuration of particular instance.
    - The check method is run once per instance defined in checks configuration.

## Metrics:-

* + - Metrics allows creating Query metrics from any time period and post metrics data which we can be graphed on Datadog Dashboards.
    - Metrics type visible on metrics summary page, developer should change the type if he want to submit metrics with a new type. There are three types of metrics. They are Gauge, Rate and Count.
    - There are many ways to send a metrics to Datadog.
    - With Datadog agent directly, by using StatsD or submit metrics directly to Datadog HTTP API.

# Integration of Datadog dashboard:-

* After all the installation of Datadog Agent. We need to go through /etc/dd-agent folder
* We have two Directories , one is “checks.d “and “conf.d”



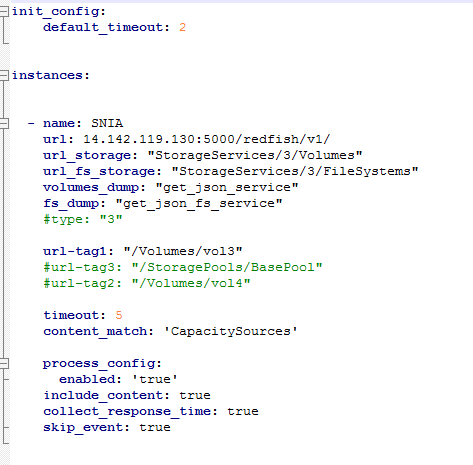
* **conf.d** contains **“.yaml“** files & checks.d contains **“python files”.**
* The agent checks and some logic are written in python and they rely on yaml file configuration.
* We can write or modify existing checks to create and collect metrics
* The name of the check file and the config file must match.
* If we named the check file “**sample.py**“, then our config file must be named **sample.yaml.**

## Config files (YAML):-

* **Objective:** developer can configure server integration by adding url and sub tags. Each check has a YAML configuration file that is placed in conf.d directory. The file name should match the name of checks module.
* From below figure .yaml file contains init\_config and instances Section.
* Init\_config section allows to have configuration option available to run checks

Instances Section is list of instances that runs against checks.

* A developer can integrate Swordfish service by defining URL and some information regarding the service.
* Developer can add tag and sub-tags of resource which makes easy access to all the data in a particular server.

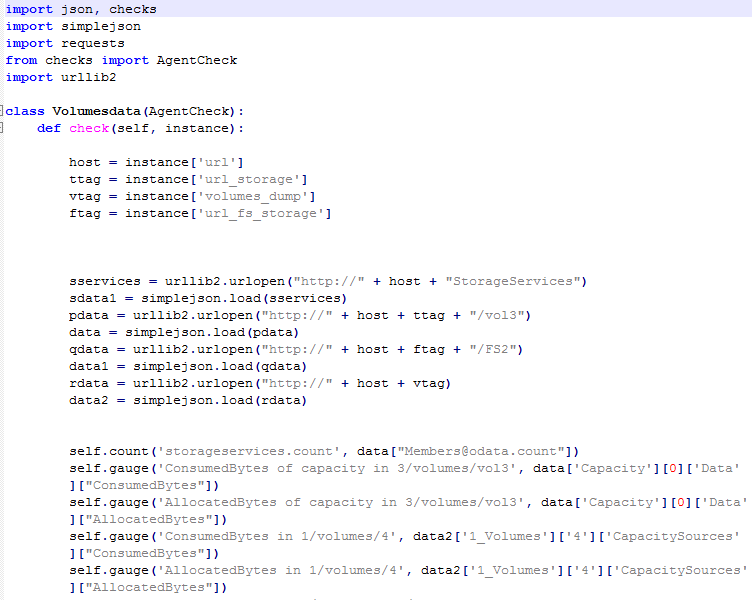


## Checks (python) files:-

* **Objective:** The main part of checks is making a request to url. Once the request passes, write custom metrics for service url which is placed in .yaml file.
* All the custom checks inherit from AgentCheck and it will take check () to take arguments. Each check instance has its own instance of class.
* The check method run once per instance.
* Checks will make a request to URL and response time.
* Developer can send events, gauges and service checks in a check.
* From the below example its sending metrics in a check

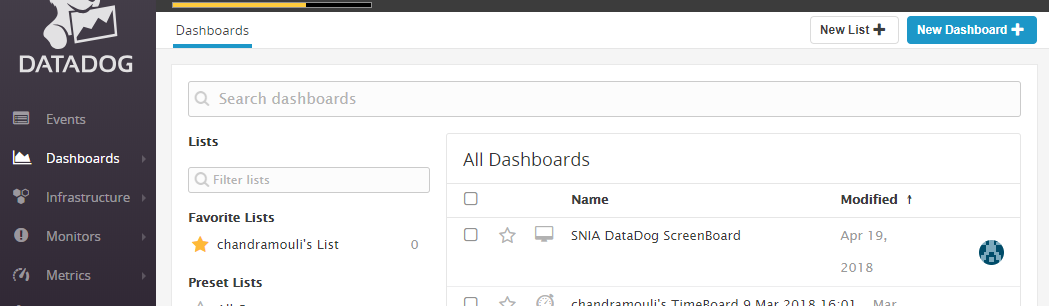
**E.g.,** Self.gauge( )

* The above method can take arguments like value, tag, hostname and metrics.
* All the custom checks are not able to import modules by default.
* Once we place the checks in checks.d folder we can test it by restarting the agent.
* Sudo service datadog-agent restart (In Ubuntu)

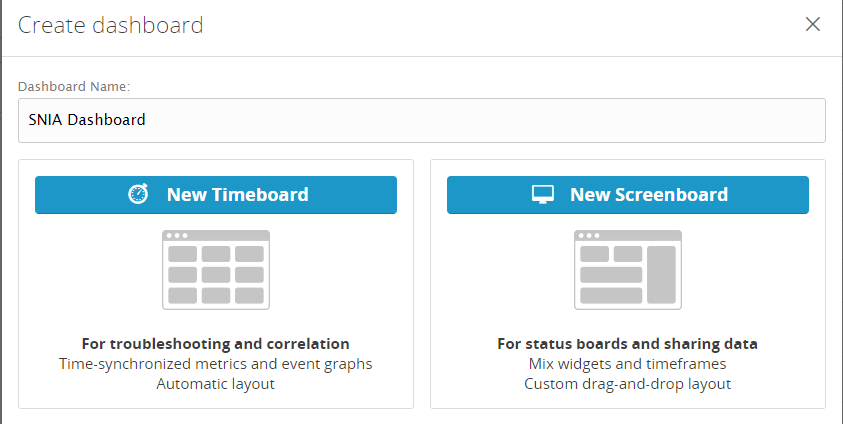


# Creation of Datadog dashboard:

* + To create a dashboard, go to dashboards tab at left panel.
  + Select New dashboard button at top right.



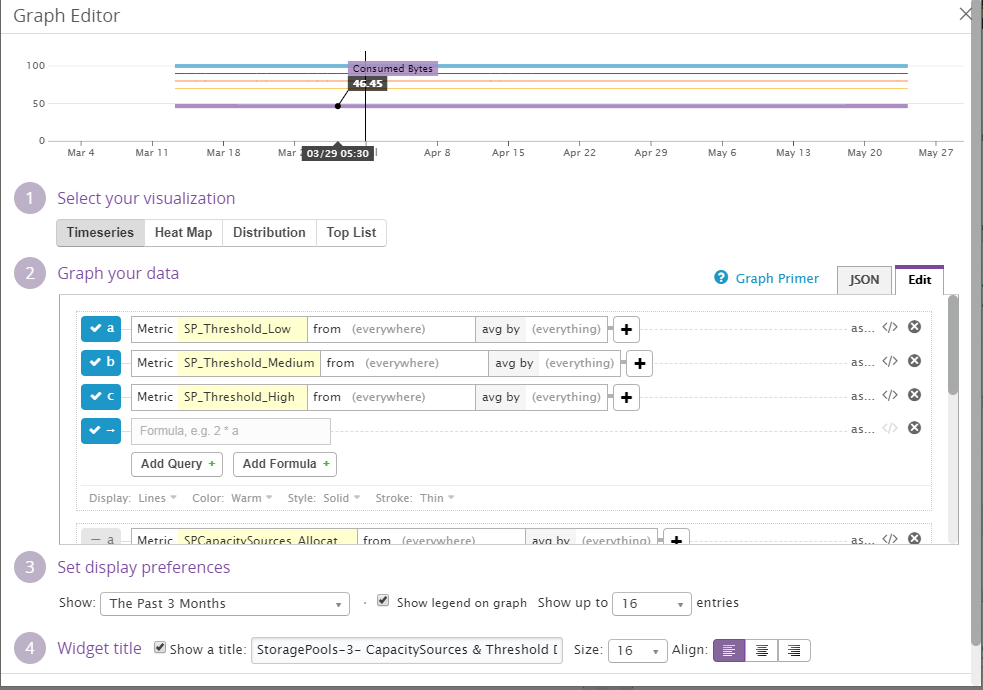
* The below window will open so we have to select any of the dashboard type.
* Developer should give a dashboard name and select dashboard.



* New Dashboard window will open as shown in below figure.
* Now Developer can select different types of graphs for visualization.

## Time Board:

In Timeboard developer selects different type of graphs where just drag and drop time series widget onto dashboard to add it. Timeboard have very specific structure, developers are able to add it to grid of graphs. If developers want to show more flexibility with placement of graphs, can go for screenboards.



Line graph will displays a line for each graph of data. By default, all hosts that reports the metrics are combined into single line. Developer can choose one metrics collected by organization here, as long as metrics have seen new data in last 24 hrs.

Next dropdown will select the subset of hosts from which to collect data. We have to mension the tags here which are created before. Here you could choose that same tag and see metrics only from the hosts running a database. Next you can choose whether you want to show avg, max, min and sum of values in a metrics.

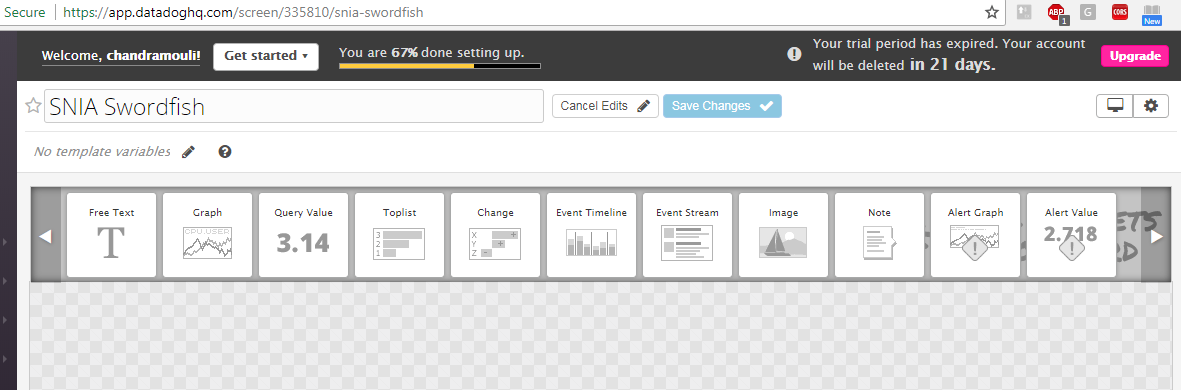
If you click the advanced link to the right, you can combine metrics using variety of mathematical functions.

After all the above changes click on save and create next gauge if any.

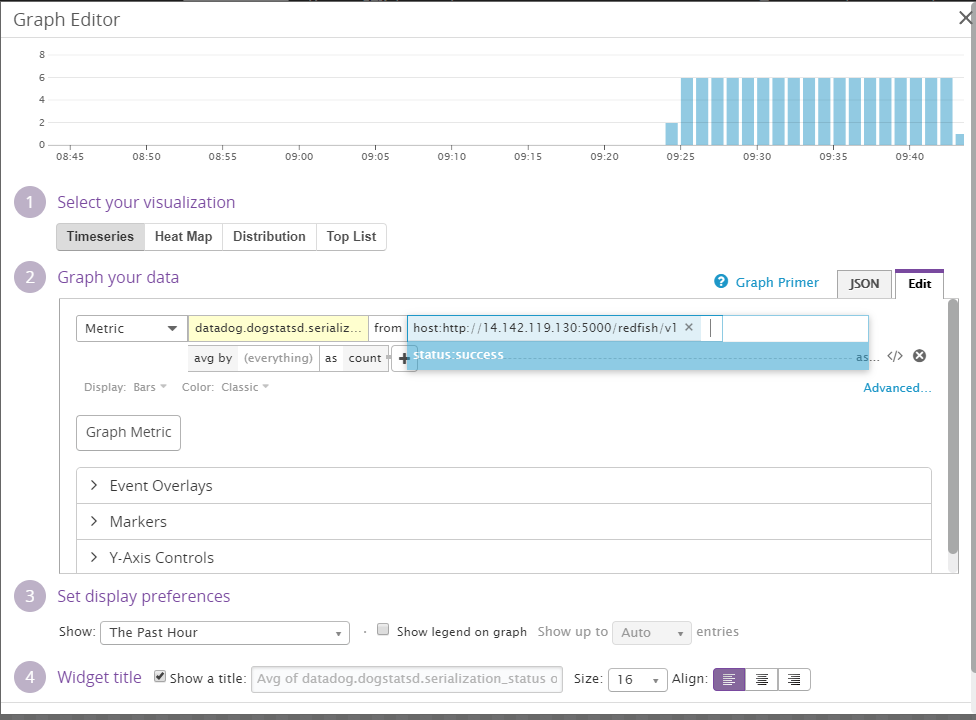
There are few visualizations available in ScreenBoards that are not in Timeboards.

ScreenBoard**:**

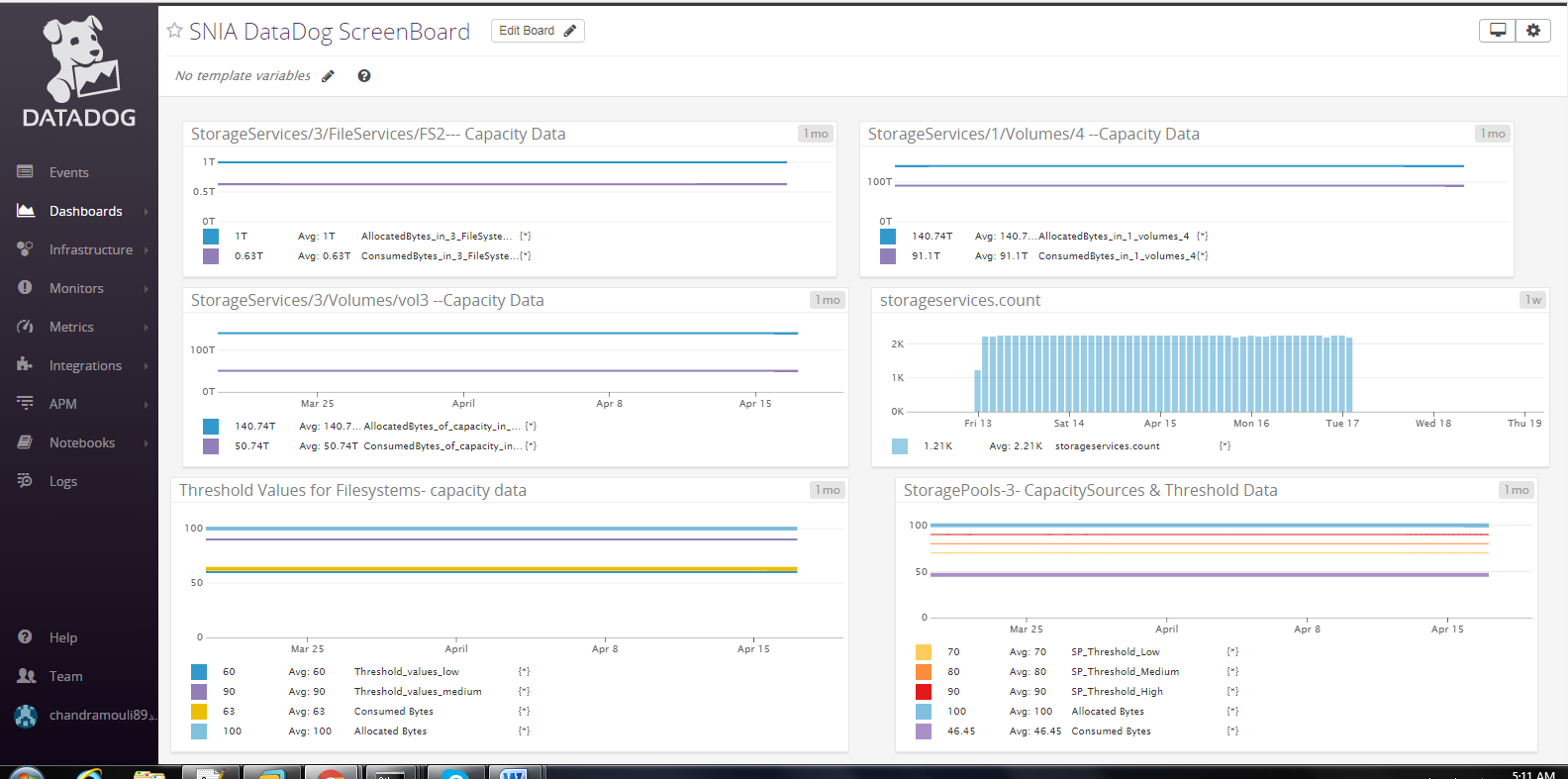
To create a ScreenBoard, just create a new dashboard and choose Screenboard. In Screenboard we can place and size the graphs anywhere you like and you can control the period shown on per graph basis.There are few visualizations availabel onlyin screenboard.



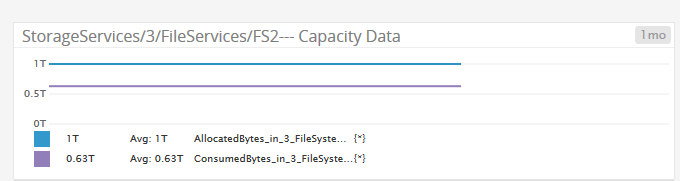
* If Developer tries to select any graph, a window will open as shown in below figure.
* Developer should edit the graph data and select visualization type to create a gauge.



* So , these are the different types of graphs in a dashboard as shown in below figure.
* If Developer want to modify data , we have to click on edit board button.



* The below Dashboard shows the Consumed bytes and Allocated bytes data.
* Select the graph from the top menu by editing the menu.



* If a Developer enters in to graph editor, as shown in below figure.
* Developer can have options to select visualization and add metrics data.

