



Technical Report

Visualizing NetApp HCI Performance

Using Grafana, Docker, Trident and Graphite

Aaron Patten, NetApp
May 2018 | TR-XXXX

Enabled by



Abstract

Grafana is a powerful tool for visualizing time series performance data. This TR describes how to build a fully customizable Grafana instance for visualizing performance statistics for SolidFire, VMware and NetApp HCI.

This solution is completely open source and leverages Grafana for graphing performance data, Docker for containerizing the applications, Trident plugin for Docker for persistent storage of metrics and container state and Graphite for storing the time series data.

TABLE OF CONTENTS

1	Overview	3
1.1	Components.....	4
1.2	How it All Works Together.....	5
1.3	Primary Use Cases	6
2	Installation	6
2.1	Preparation	6
2.2	Collector Installation - Scripted	7
2.3	Collector Installation – Manual	8
2.4	Grafana Configuration.....	12
2.5	Graph Conventions Used.....	15
	Appendix A: Troubleshooting	17
	Validating Metrics in the Graphite Database	17
	Checking Collector Logs.....	18
	Rebuilding a Container	18
	Removing Stale Metrics.....	18
	Appendix B	19
	Where to Find Additional Information	26
	Version History	26

LIST OF TABLES

Table 1) Trident prerequisites.....	Error! Bookmark not defined.
-------------------------------------	-------------------------------------

LIST OF FIGURES

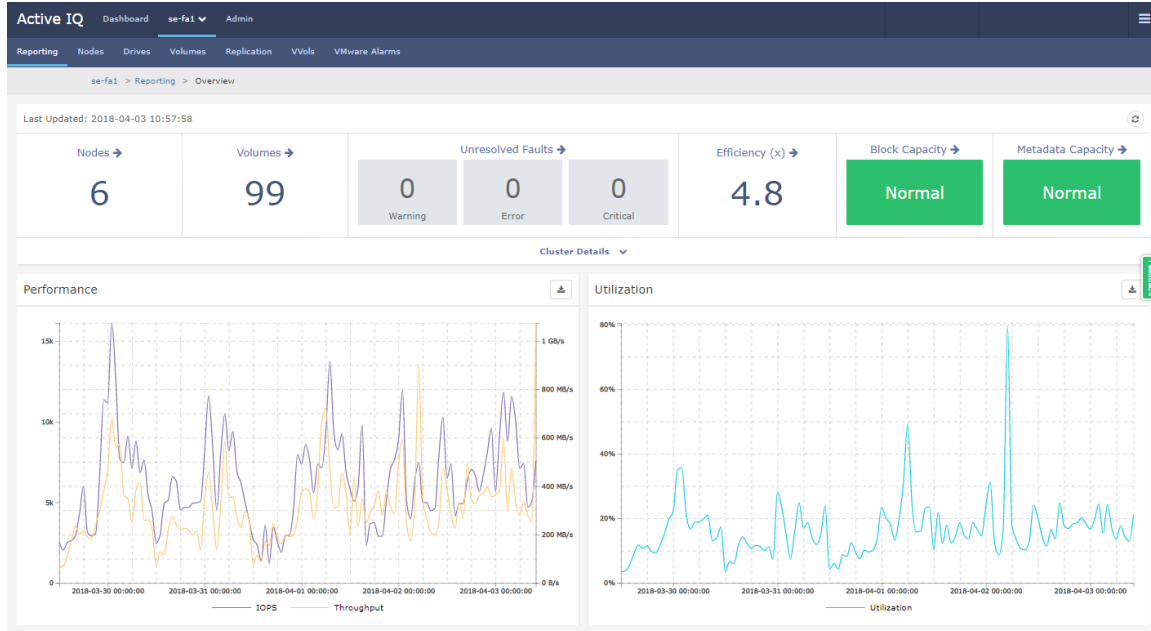
Figure 1) AIQ dashboard.....	3
Figure 2) AIQ capacity forecast.....	4
Figure 3) HCI collector components.....	5
Figure 4) Scripted installation.....	8
Figure 5) Graphite data source.....	13
Figure 6) Grafana dashboards.....	14
Figure 7) Sample dashboard.....	15
Figure 8) keepLastValue not set.....	16
Figure 9) keepLastValue is set.....	16
Figure 10) Using the render API for Graphite.....	17
Figure 11) Viewing stats from the last hour.....	17

1 Overview

Effective monitoring of critical infrastructure is the key stone in maintaining operational readiness and a key enabler of risk mitigation. Monitoring critical infrastructure for atypical workloads and events can help prevent small issues from becoming outages.

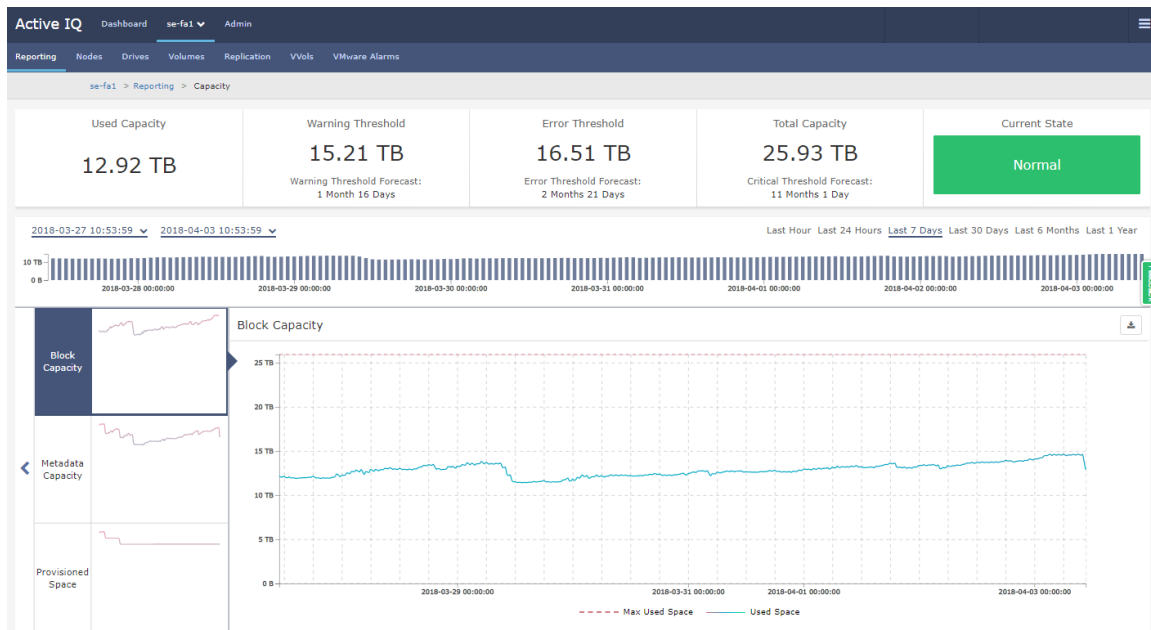
NetApp HCI includes access to the ActiveIQ (AIQ) platform which is NetApp's cloud based performance monitoring and alerting platform. AIQ provides a rich set of pre-configured dashboards that can show real time cluster performance and alerts as well as a view of historical data.

Figure 1) AIQ dashboard.



AIQ also allows for capacity modeling and forecasts when additional capacity should be added to the cluster to aid with future planning and budgeting.

Figure 2) AIQ capacity forecast.



AIQ is the preferred method for monitoring and alerting for NetApp HCI and SolidFire systems. However, for AIQ to function it requires outbound connectivity from the customer site to the cloud. For sites that do not allow outbound connections (dark sites) AIQ is not an available option.

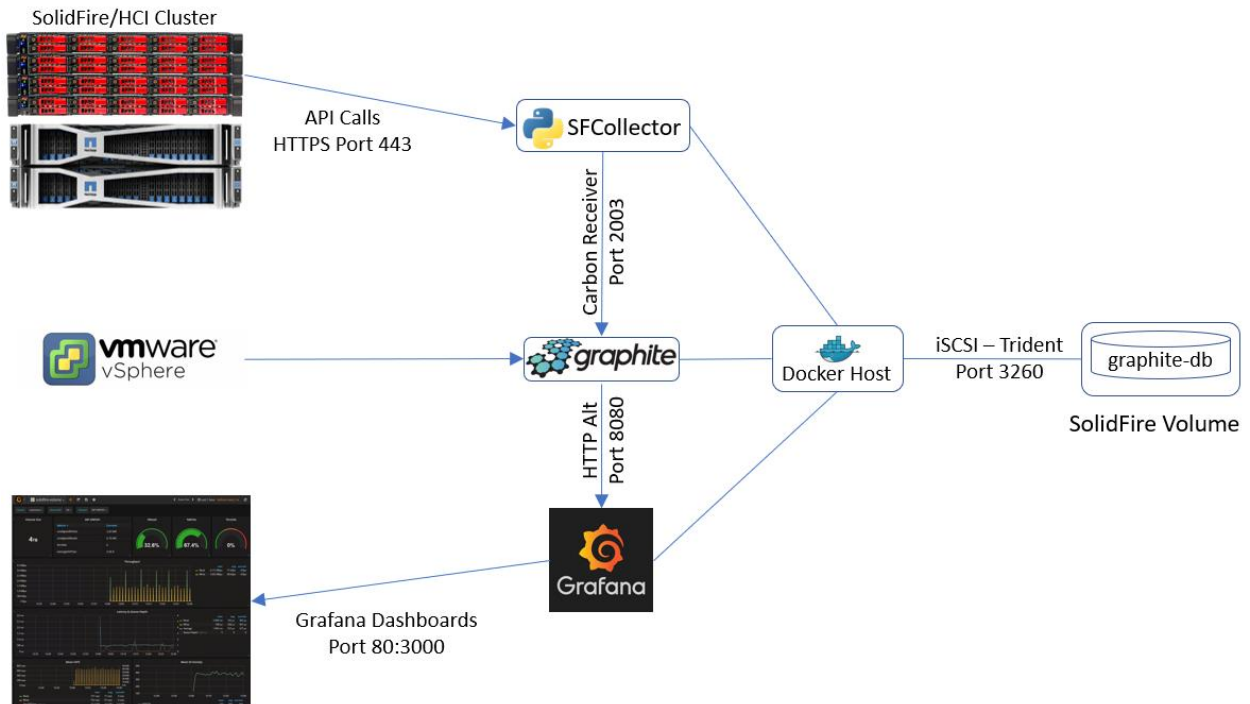
The HCI Collector is a community open source project that replicates a subset of the AIQ functionality in a collection of Docker containers that can be run on local infrastructure, removing the need for external Internet connectivity. The HCI collector assembles time series data from both SolidFire and HCI components, including vSphere, and presents those metrics through a collection of pre-configured Grafana dashboards.

1.1 HCI Collector Components

The HCI collector is composed of a number of individual components:

- **SFCollector** – Docker container hosting a python script that collects performance data from SolidFire systems or HCI storage nodes and sends the data to a Graphite time series database.
- **Graphite** – Docker container hosting the time series database for holding performance data collected from storage and compute hosts.
- **VMwCollector** – Docker container hosting a statistics collector for VMware components, written in Golang.
- **Grafana** – Docker container hosting the graphing front end used to visualize the time series data in the Graphite database.
- **Docker Host** – An Ubuntu 16.04 LTS VM hosting the HCI collector containers.
- **Trident** – An (optional) Docker volume plugin running in the Docker host that automates creating and presenting persistent storage volumes for the containers that make up the HCI collector. Local host volumes can be used as well, but their configuration is not covered in this guide.

Figure 3) HCI collector components.



1.2 How it All Works Together

As implemented in this guide, the HCI collector will function as follows:

- Some initial setup is required to provide IPs and credentials to the components of the collector. This process is partially automated by the included `install_hcicollector.sh` shell script. Note that this configuration expects and requires DNS resolution of vCenter and ESXi hosts.
- The Trident for Docker plugin will create an iSCSI volume on a SolidFire system for storing the Whisper database files for Graphite. Using an external datasource allows portability of the collected data as well as protection through snapshots, clones or replication of the data to another SolidFire or NetApp FAS system.
- The Docker host mounts the volume provided by Trident and passes this volume through to the Graphite container for persistent storage of collected metrics. Container management is handled by docker-compose. This handles provisioning the container configuration as well as networks and external volumes used by the containers.
- The SFCollector connects to one or more SolidFire storage backends and collects stats every 60 seconds and sends them to Graphite under the “netapp.solidfire” namespace. A list of stats collected is available in Appendix B.
- The VMwCollector connects to one or more vCenter instances and collects stats every 60 seconds and sends them to Graphite under the “vsphere” namespace. A list of stats collected is available by looking at the `./vmwcollector/vsphere-graphite.json` file in the “Metrics” section.
- Grafana pulls data from the Graphite database and uses it to draw a set of preconfigured dashboards. To aid in setup, the dashboards and datasource are automatically configured and populated via the provisioning functionality introduced in Grafana 5.

1.3 Primary Use Cases

The HCI Collector can be used in isolation, but the primary use case is to augment the data provided by AIQ. The HCI collector can be leveraged in the following ways:

- Creating custom dashboards for visualizing data not present in AIQ.
- Creating multi-system reporting dashboards. Currently, AIQ reporting is done per system.
- Visualizing more complete VMware statistics.
- Extending the collectors to capture additional data from switches or other infrastructure that is of interest.

2 Installation

The following section describes how to deploy the HCI collector. It is assumed that a Docker host is available. These instructions use an Ubuntu 16.04 LTS VM as an example. It is also assumed that Docker CE 17.03+ and docker-compose have been installed.

Instructions for installing Docker components are available on <https://docs.docker.com>

Note: Docker version 18.03.0 breaks plugins. This is resolved in 18.03.1. Earlier versions are also unaffected. Docker 17.12.1 is used in this guide.

2.1 Preparation

You will want to make the following preparations before continuing further.

- Verify you have a suitable Docker host machine available and that machine has a supported version of Docker installed as well as any support tools you may require.
- DNS records exist for the equipment you will be monitoring.
- The following environment information
 - vCenter hostname, FQDN, username and password.
 - SolidFire management IP (MVIP), storage virtual IP (SVIP), username, password.
 - Docker host IP and login information
- The <https://github.com/jedimt/hcicollector> repo has been cloned into /opt/github/hcicollector on the Docker host.
- The Docker host must be able to connect to the SolidFire SVIP if using Trident.

After cloning the repo, the following high level directory structure should be in place:

```
root@sfps-grafana-dev:/opt/github/hcicollector# tree -d -L 3
```

```
├── sfcollector          #Container configuration for the SFCollector components
├── grafana             #Container configuration for the Grafana components
│   ├── dashboards      #Contains all the preconfigured dashboards
│   └── provisioning     #Configuration files for Grafana dashboard and datasource provisioning
│       ├── dashboards  #Dashboard provisioning configuration
│       └── datasources #Datasource provisioning configuration
├── graphite           #Container configuration for Graphite database
└── vmwcollector        #Contain configuration for the vSphere-Graphite collector
```

Note: This tree view is abbreviated to show just the directories of interest.

2.2 Collector Installation - Scripted

The collector includes a rudimentary bash install script (`install_hcicollector.sh`) which does the following tasks:

- Prompts the user for required information.
- Writes out the Trident configuration file to `/etc/netappdvp/config.json` and installs Trident 18.04
- Creates the docker volume for the Graphite database to mount
- Writes out the following configuration files:
 - `./docker-compose.yml`
 - `./sfcollector/wrapper.sh` file for the SolidFire collector
 - `./vmwcollector/vsphere-graphite.json`
- Sets the correct datasource for the included dashboards in the `./grafana/dashboards` directory

When using the script to drive the installation the work flow would be:

1. Create the directory to house the github repository, for example `/opt/github/hcicollector`
2. Clone the <https://github.com/jedimthcicollector> Github repo into the desired directory
3. Execute the `install_hcicollector.sh` script and provide the requested inputs
4. Start the collector by running `docker-compose up -d` from the `/opt/github/hcicollector` directory

Starting the containers the first time will require roughly 10 minutes.

Figure 4) Scripted install with install_hcicollector.sh script.

```
root@sfps-grafana-devtemp:/opt/github/hcicollector# ./install_hcicollector.sh
#####
Enter the SolidFire management virtual IP (MVIP):
10.193.136.240
Enter the SolidFire storage virtual IP (SVIP):
10.193.139.44
Enter the SolidFire username (case sensitive):
netapp
Enter the Solidfire password:
Enter the tenant account to use for Trident:
docker-dev
Enter the volume name to create for Graphite
graphite-db-dev
Enter the password to use for the Grafana admin account:
Enter the vCenter username:
administrator@vsphere.local
Enter the vCenter password:
Enter the vCenter hostname. Ex. vcса:
sfps-prototype-vcса
Enter the vCenter domain. Ex. rtp.openenglab.netapp.com:
rtp.openenglab.netapp.com
Enter the IP address of this Docker host:
10.193.136.230
Beginning Install
Installing Trident and creating the volume
18.01: Pulling from netapp/trident-plugin
88c0023f068a: Download complete
Digest: sha256:306c6dcb4c6e822f2db85c7ed2f73bb5254e4cdb4f14fa36e629451f70a35055
Status: Downloaded newer image for netapp/trident-plugin:18.01
Installed plugin netapp/trident-plugin:18.01
graphite-db-dev
Creating the docker-compose.yml file
Creating the SolidFire collector wrapper.sh script
Marking wrapper.sh as executable
Creating the storage-schemas.conf file
Creating the vsphere-graphite.json file
Modifying the default 'datasource' values in the pre-packaged dashboards
```

If the collector will be customized, then the `install_hcicollector.sh` script can be modified. Alternatively the manual setup instructions will also be covered in the Collector Installation – Manual section.

2.3 Collector Installation – Manual

The following detailed setup steps can be used to manually install the collector on a Docker host.

If multipathing will be used, please follow the steps outlined here:

https://netapp-trident.readthedocs.io/en/stable-v18.04/docker/install/host_config.html#host-configuration

Trident Installation and Configuration

First clone the Github repo into `/opt/github`

```
mkdir -p /opt/github
```



```
git clone https://github.com/jedimt/hcicollector /opt/github/hcicollector
```

Create a location to store the NDVP configuration files

```
sudo mkdir -p /etc/netappdvp
```

Create the configuration file for SolidFire

```
cat << EOF > /etc/netappdvp/config.json
{
  "version": 1,
  "storageDriverName": "solidfire-san",
  "Endpoint": "https://admin:solidfire@10.193.136.240/json-rpc/9.0",
  "SVIP": "10.193.137.240:3260",
  "TenantName": "docker",
  "InitiatorIFace": "default",
  "Types": [
    {
      "Type": "docker-default",
      "Qos": {
        "minIOPS": 1000,
        "maxIOPS": 2000,
        "burstIOPS": 4000
      }
    },
    {
      "Type": "docker-app",
      "Qos": {
        "minIOPS": 4000,
        "maxIOPS": 6000,
        "burstIOPS": 8000
      }
    },
    {
      "Type": "docker-db",
      "Qos": {
        "minIOPS": 6000,
        "maxIOPS": 8000,
        "burstIOPS": 10000
      }
    }
  ]
}
EOF
```

Install the Trident plugin

```
docker plugin install --grant-all-permissions --alias netapp netapp/trident-plugin:18.04
config=config.json
```

Verify the plugin installed and is enabled

ID	NAME	DESCRIPTION	ENABLED
047ac2d0663f	netapp:latest	Trident - NetApp Docker Volume Plugin	true

Create docker volumes to be used for Graphite persistent storage.

```
# Create Graphite docker volume
docker volume create -d netapp --name graphite-db -o type=docker-db -o size=100G

#show volume
docker volume list --filter 'driver=netapp:latest'
DRIVER          VOLUME NAME
netapp:latest    graphite-db
```

Container Setup and Configuration

Install docker-compose if it is not installed

```
apt install docker-compose
```

Create the docker-compose.yml as shown below specifying the persistent data volumes to use for Graphite and Grafana as well as the password to secure the Grafana web interface.

```
cat << EOF > /opt/github/hcicollector/docker-compose.yml
version: "2"
services:
  graphite:
    build: ./graphite
    restart: always
    ports:
      - "8080:80"
      - "8125:8125/udp"
      - "8126:8126"
      - "2003:2003"
      - "2004:2004"
    volumes: #Trident or local volumes for persistent storage
      - graphite-db:/opt/graphite/storage/whisper
    networks:
      - net_hcicollector

  grafana:
    build: ./grafana
    restart: always
    ports:
      - "80:3000"
    networks:
      - net_hcicollector
    environment:
      #Set password for Grafana web interface
      - GF_SECURITY_ADMIN_PASSWORD=<your password>
      #Optional SMTP configuration for alert queries
      - GF_SMTP_ENABLED=true
      - GF_SMTP_HOST=smtp.gmail.com:465
      - GF_SMTP_USER=<email address>
      - GF_SMTP_PASSWORD=<email password>
      - GF_SMTP_SKIP_VERIFY=true

  sfcollector:
    build: ./sfcollector
    restart: always
    networks:
      - net_hcicollector

  vmwcollector:
    build: ./vsphere-graphite
    restart: always
    networks:
      - net_hcicollector
    depends_on:
      - graphite

networks:
  net_hcicollector:
    driver: bridge

volumes:
  graphite-db:
    external: true
EOF
```

Create the /opt/github/hcicollector/sfcollector/wrapper.sh script with the appropriate SolidFire cluster MVIP, username and password. If the graphite container name was changed, also specify the new hostname using the “-g” option.

```
cat << EOF > /opt/github/hcicollector/sfcollector/wrapper.sh
#!/usr/bin/env bash
while true
do
/usr/bin/python /solidfire_graphite_collector.py -s 10.193.136.39 -u admin -p <yourpassword> -g
graphite &
sleep 60
done
EOF
```

Edit the `/opt/github/hcicollector/graphite/storage-schemas.conf` file to adjust the retention period for the NetApp stats if desired. By default the following retention is set which keeps 1 minute stats for 7 days, 5 minute stats for 28 days and 10 minute stats for a year. If you are also collecting stats from vCenter, add the [vsphere] section as well.

```
[netapp]
pattern = ^netapp\.
retentions = 1m:7d,5m:28d,10m:1y

[vsphere]
pattern = ^vsphere\.
retentions = 1m:7d,5m:28d,10m:1y
```

Create the `/opt/github/hcicollector/vmwcollector/vsphere-graphite.json` file and add your vCenter details.

```
cat << EOF > /opt/github/hcicollector/vmwcollector/vsphere-graphite.json
{
  "Domain": "<yourdomain>",
  "Interval": 60,
  "FlushSize": 100,
  "VCenters": [
    { "Username": "administrator@vsphere.local", "Password": "<yourpassword>", "Hostname": "sfps-
prototype-vcsa" }
  ],
  "Backend": {
    "Type": "graphite",
    "Hostname": "graphite",
    "Port": 2003
  },
  "Metrics": [
    {
      "ObjectType": [ "VirtualMachine", "HostSystem" ],
      "Definition": [
        { "Metric": "cpu.usage.average", "Instances": "" },
        { "Metric": "cpu.usage.maximum", "Instances": "" },
        { "Metric": "cpu.usagemhz.average", "Instances": "" },
        { "Metric": "cpu.usagemhz.maximum", "Instances": "" },
        { "Metric": "cpu.totalCapacity.average", "Instances": "" },
        { "Metric": "cpu.ready.summation", "Instances": "" },
        { "Metric": "mem.usage.average", "Instances": "" },
        { "Metric": "mem.usage.maximum", "Instances": "" },
        { "Metric": "mem.consumed.average", "Instances": "" },
        { "Metric": "mem.consumed.maximum", "Instances": "" },
        { "Metric": "mem.active.average", "Instances": "" },
        { "Metric": "mem.active.maximum", "Instances": "" },
        { "Metric": "mem.vmmemctl.average", "Instances": "" },
        { "Metric": "mem.vmmemctl.maximum", "Instances": "" },
        { "Metric": "disk.commandsAveraged.average", "Instances": "*" },
        { "Metric": "mem.totalCapacity.average", "Instances": "" }
      ]
    },
    {
      "ObjectType": [ "VirtualMachine" ],
      "Definition": [
        { "Metric": "virtualDisk.totalWriteLatency.average", "Instances": "*" },
        { "Metric": "virtualDisk.totalReadLatency.average", "Instances": "*" },
        { "Metric": "virtualDisk.numberReadAveraged.average", "Instances": "*" }
      ]
    }
  ]
}
```

```

        { "Metric": "virtualDisk.numberWriteAveraged.average", "Instances": "*" },
        { "Metric": "cpu.ready.summation", "Instance": ""}
    ],
    },
    {
        "ObjectType": [ "HostSystem" ],
        "Definition": [
            { "Metric": "disk.maxTotalLatency.latest", "Instances": "" },
            { "Metric": "disk.numberReadAveraged.average", "Instances": "*" },
            { "Metric": "disk.numberWriteAveraged.average", "Instances": "*" },
            { "Metric": "disk.deviceLatency.average", "Instances": "*" },
            { "Metric": "disk.deviceReadLatency.average", "Instances": "*" },
            { "Metric": "disk.deviceWriteLatency.average", "Instances": "*" },
            { "Metric": "disk.kernelLatency.average", "Instances": "*" },
            { "Metric": "disk.queueLatency.average", "Instances": "*" },
            { "Metric": "datastore.datastoreIops.average", "Instances": "*" },
            { "Metric": "datastore.datastoreMaxQueueDepth.latest", "Instances": "*" },
            { "Metric": "datastore.datastoreReadBytes.latest", "Instances": "*" },
            { "Metric": "datastore.datastoreReadIops.latest", "Instances": "*" },
            { "Metric": "datastore.datastoreWriteBytes.latest", "Instances": "*" },
            { "Metric": "datastore.datastoreWriteIops.latest", "Instances": "*" },
            { "Metric": "datastore.numberReadAveraged.average", "Instances": "*" },
            { "Metric": "datastore.numberWriteAveraged.average", "Instances": "*" },
            { "Metric": "datastore.read.average", "Instances": "*" },
            { "Metric": "datastore.totalReadLatency.average", "Instances": "*" },
            { "Metric": "datastore.totalWriteLatency.average", "Instances": "*" },
            { "Metric": "datastore.write.average", "Instances": "*" },
            { "Metric": "mem.state.latest", "Instances": "" }
        ]
    }
]
}
EOF

```

Bring up the containers using docker-compose. This will take several moments to complete.

Note: The containers can be brought up the first time with the (-d) flag omitted so the logs can be viewed.

```
docker-compose -f /opt/github/hcicollector/docker-compose.yml up -d
```


2.4 Grafana Configuration

When the compose process finishes, launch a web browser to <http://<Docker VM IP Addr>>. The Grafana web interface should appear. Log in with user “admin” and the password configured in the docker-compose.yml file.

Note: If you open the Grafana interface before any metrics have been collected, you may see all dashboards with “NA” values for all counters. Wait two minutes and reload the dashboard and the issue should resolve itself.

Verify the Graphite data source has been automatically provisioned.

Figure 5) Graphite data source.



Data Sources / graphite-db

Type: Graphite

Settings

Dashboards

Name

graphite-db

Default

☒

Type

Graphite

HTTP

URL

http://10.193.136.36:8080

Access

proxy

Auth

Basic Auth

☐

With Credentials

☐

TLS Client Auth

☐

With CA Cert

☐

Skip TLS Verification (Insecure)

☐

Advanced HTTP Settings

Whitelisted Cookies

Add Name

Graphite details

Version

Save & Test

Delete

Back

Verify the dashboards were automatically provisioned and functional.

Figure 6) Grafana dashboards.

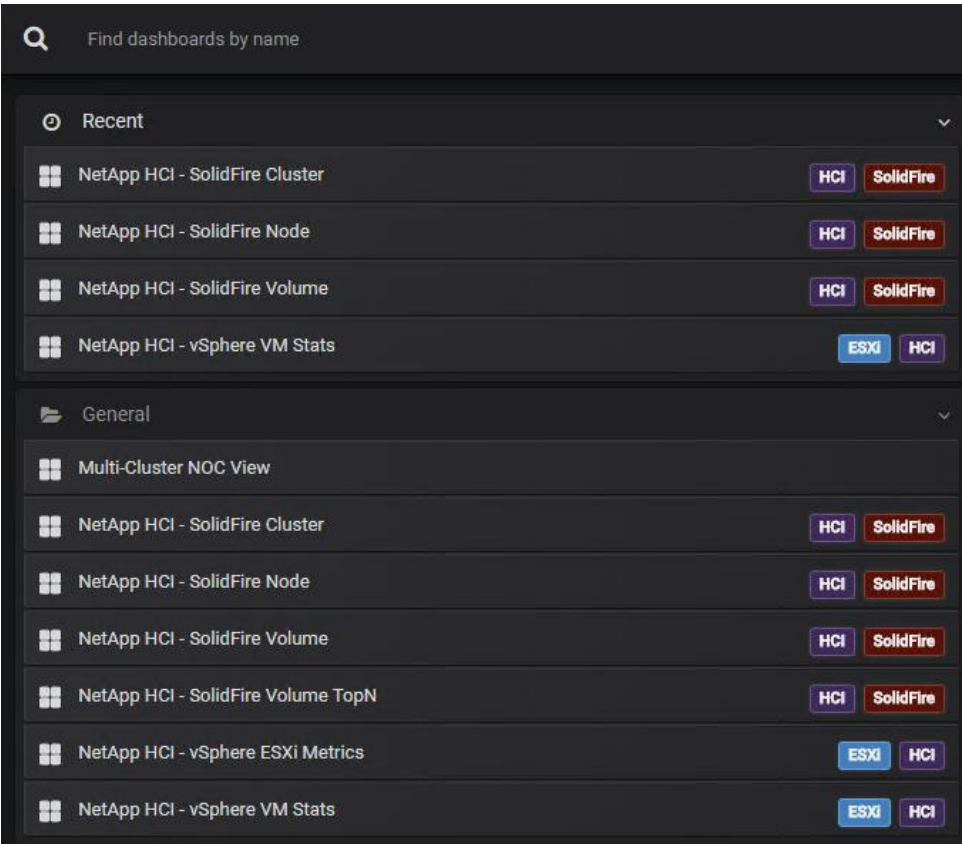
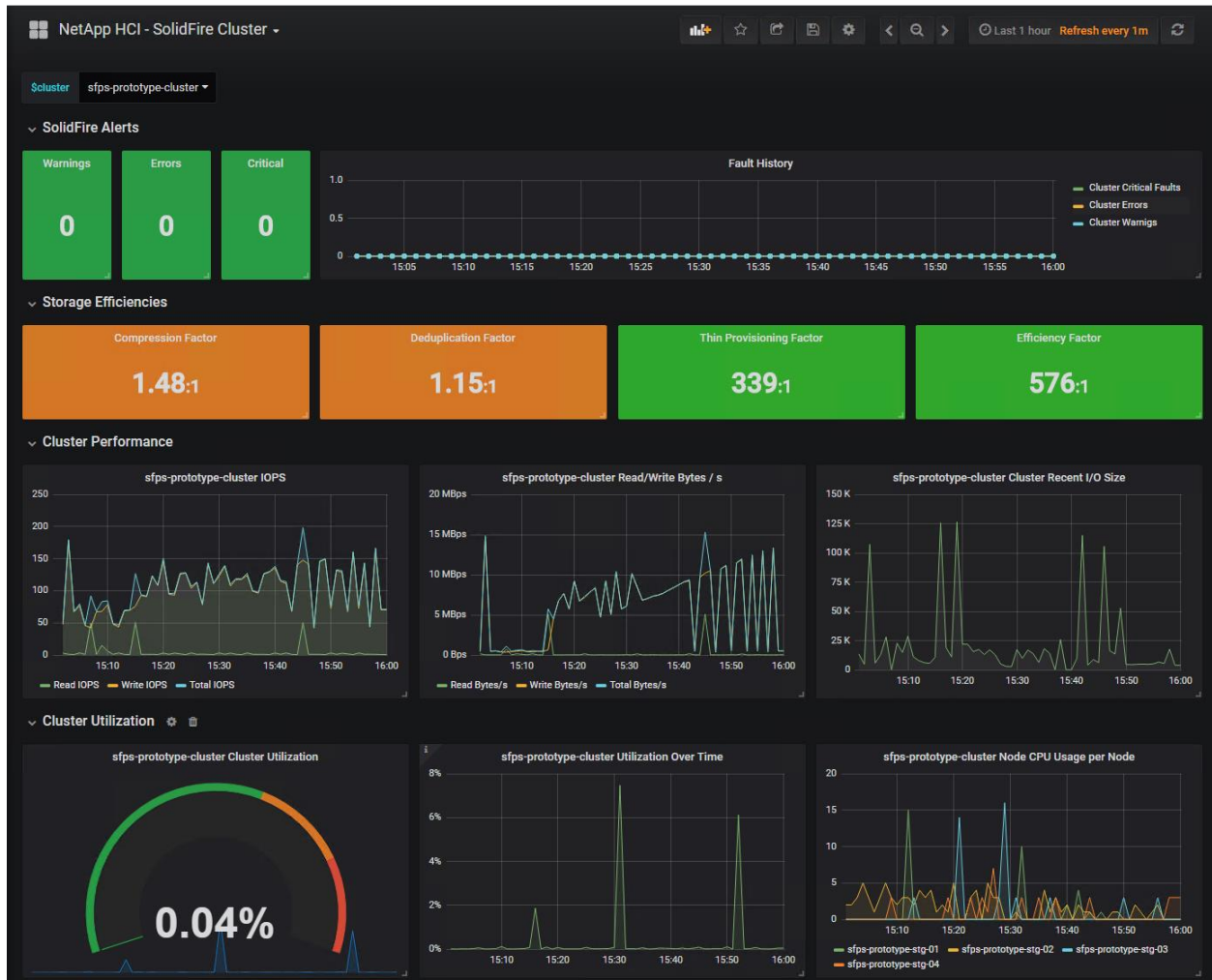


Figure 7) Sample dashboard.



2.5 Graph Conventions Used

The following conventions are used in the system graphs.

- Dashboards expect the host objects (vcenter, ESXi hosts, etc) to be pulled in by their fully qualified domain name. Since Graphite uses "." to delimit metrics, pulling in an IP address will break the dashboard templating. Alternatively, the dashboard templating could be changed to account for IP addresses.
- Null values are shown as null for most graphs. This allows spotting objects that fail to report stats. It is augmented by keepLastValue.
- keepLastValue(5) - Continues the line with the last received value when gaps ('None' values) appear in your data, rather than breaking your line. If there are more than (5) consecutive missed reporting periods, a break will show in the graph. Removing this option will show a graph with breaks for any object that has no stats for the reporting period. Five minutes was chosen as that is the break point for evicting a node from the cluster. Note the difference in the following screen shots.

Figure 8) keepLastValue not set.

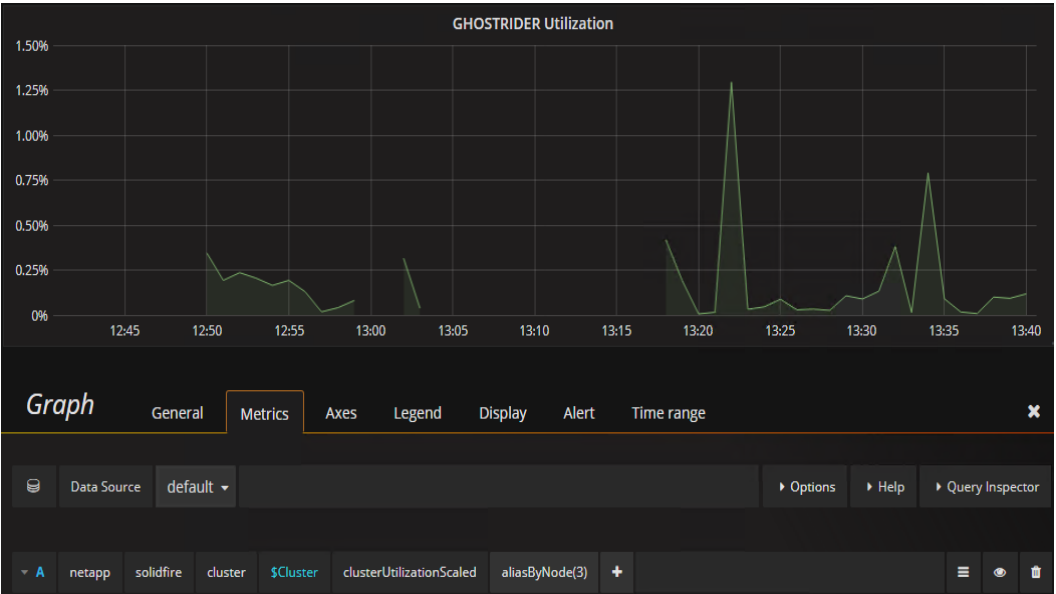
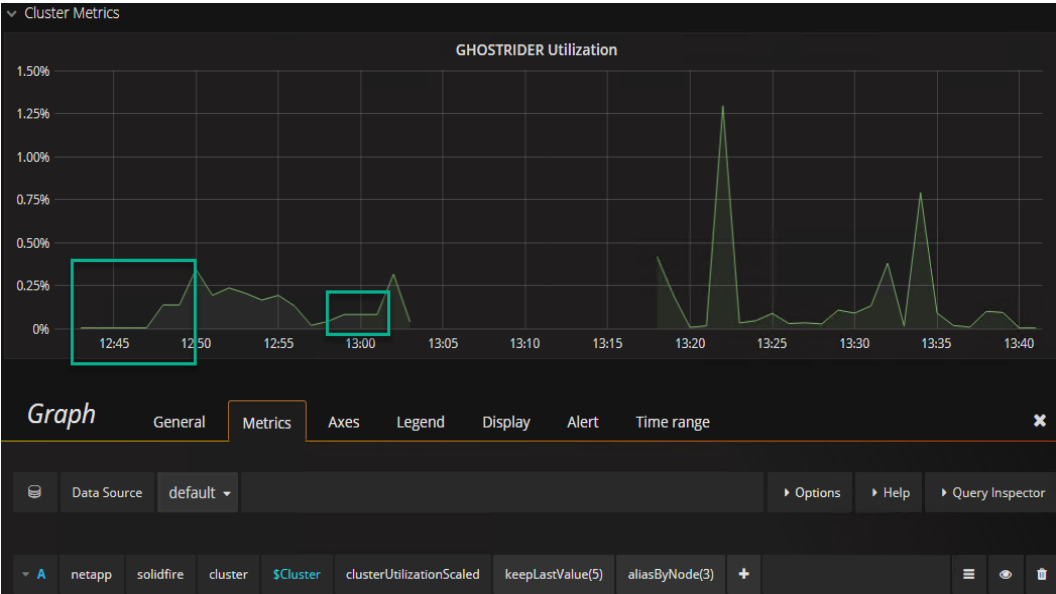


Figure 9) keepLastValue is set to '5'.



Appendix A: Troubleshooting

This section includes some troubleshooting steps that can be taken when having issues with the configuration of the collector.

Validating Metrics in the Graphite Database

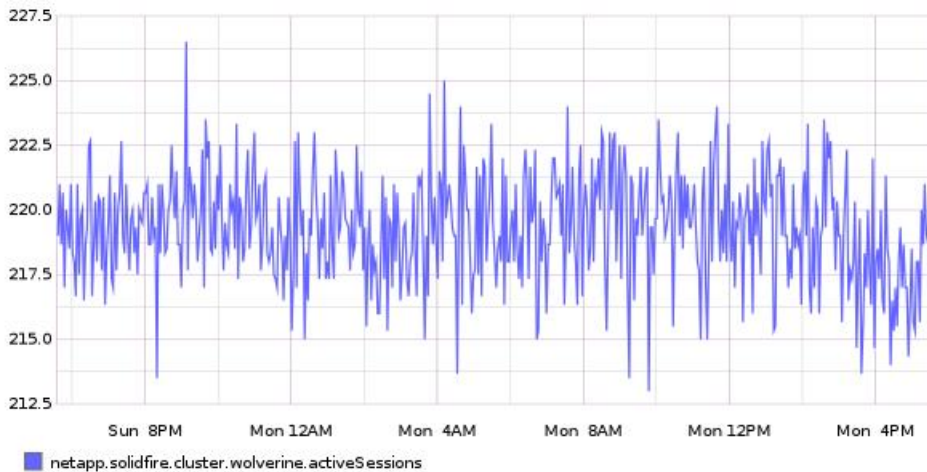
The Graphite API used in this project does not include the graphical front end for Graphite so the render API for Graphite can be used to verify that metrics are being pushed into the graphite database. The format for displaying cluster metrics is:

`http://<docker VM IP>:8080/render?target=netapp.solidfire.cluster.<cluster name>.<metric>`

For example, to see cluster activeSessions:

<http://172.27.96.14:8080/render?target=netapp.solidfire.cluster.wolverine.activeSessions>

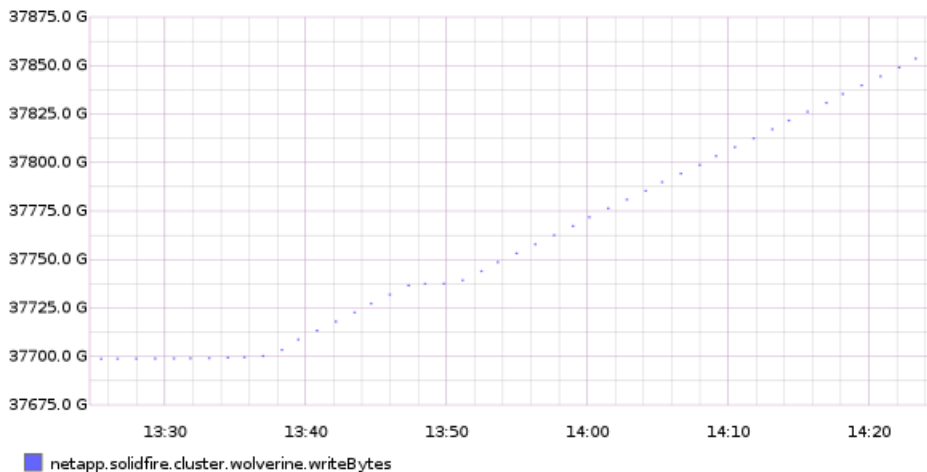
Figure 10) Using the render API for Graphite.



To display only the metrics from the last hour, add the `$from=-<time window>` argument

<http://172.27.96.26:8080/render?target=netapp.solidfire.cluster.wolverine.writeBytes&from=-1hour>

Figure 11) Viewing stats from the last hour.



Checking Collector Logs

If it becomes necessary to connect to the sfcollector to troubleshoot, the ENTRYPOINT for the container will need to be overridden

```
docker run --entrypoint "/bin/bash" -it sfcollector-v.6
```

Logs for the collector are stored in the /tmp directory of the container.

Rebuilding a Container

If you need to make a change to a single container in the docker-compose setup (for instance to change the collector wrapper script) that can be done without taking down all the containers.

```
#List the services
root@hci-grafana01:/opt/github/hcicollector# docker-compose config --services
graphite
grafana
sfcollector
vmwcollector

#Stop the service
docker-compose stop sfcollector #this is the service name
<make changes>

#Start the service
docker-compose up -d --no-deps --build sfcollector
```

Removing Stale Metrics

If there are stale metrics in the Whisper database the corresponding metric files have to be removed from the graphite container persistent storage. This can be done either from the perspective of the container (method 1) or from the docker host (method 2).

For example, to remove all the metrics for the “ultron” cluster from Graphite, do the following.

Method 1 – From Container’s Perspective

```
#Using Trident - Deleting stale stats from the container’s perspective
#Stop the sfcollector container
root@vmgrafana01-0:/opt/github/hcicollector/# docker-compose down graphite
Stopping graphite-v.6 ... done
Removing graphite-v.6 ... done

#Start the graphite container interactively with persistent storage
docker run --rm -it --entrypoint "/bin/bash" --volume graphite-db:/opt/graphite/storage/whisper
graphite-v.6

#Remove old stats for the ultron cluster
root@ed7dbf28f424:/# ls /opt/graphite/storage/whisper/netapp/solidfire/cluster/
ultron  wolverine

root@ed7dbf28f424:/# rm -rf /opt/graphite/storage/whisper/netapp/solidfire/cluster/ultron
```

Method 2 – From the Docker Host Perspective

```
#Using Trident - Deleting stats from the Docker host’s perspective
root@vmgrafana01-0:/opt/github/hcicollector/collector# docker-compose down graphite
Stopping graphite-v.6 ... done
Removing graphite-v.6 ... done

#Find the “Id” of the Trident plugin
root@sfps-grafana01:/opt/github/hcicollector# docker plugin list
ID                NAME                DESCRIPTION                ENABLED
047ac2d0663f      netapp:latest        Trident - NetApp Docker Volume Plugin    true
```

```

root@sfps-grafana01:/opt/github/hcicollector# docker plugin inspect 047ac2d0663f | grep Id
    "Id": "047ac2d0663f75405b42cb0343ab60ef92da3af8d13c9af751f0a1alada0bdec",

#variable for long pathname
graphitedb=/var/lib/docker/plugins/047ac2d0663f75405b42cb0343ab60ef92da3af8d13c9af751f0a1alada0bdec/rootfs/var/lib/docker-volumes/netapp/graphite-db/

#Navigate to the file system location
root@sfps-grafana01:/opt/github/hcicollector# cd $graphitedb

#Remove the old stats (old ESXi servers in ./vsphere/sfps-vcsa/hostsystem)
root@sfps-grafana01:/var/lib/docker/plugins/047ac2d0663f75405b42cb0343ab60ef92da3af8d13c9af751f0a1alada0bdec/rootfs/var/lib/docker-volumes/netapp/graphite-db/vsphere/sfps-vcsa/hostsystem# ls
sfps-cmp-12  sfps-cmp-13  sfps-cmp-14  solutions-infra-esxi01-mgt1  solutions-infra-esxi02-mgt1  solutions-infra-esxi03-mgt1

root@sfps-grafana01:/var/lib/docker/plugins/047ac2d0663f75405b42cb0343ab60ef92da3af8d13c9af751f0a1alada0bdec/rootfs/var/lib/docker-volumes/netapp/graphite-db/vsphere/sfps-vcsa/hostsystem# rm -rf
solutions*
root@sfps-grafana01:/var/lib/docker/plugins/047ac2d0663f75405b42cb0343ab60ef92da3af8d13c9af751f0a1alada0bdec/rootfs/var/lib/docker-volumes/netapp/graphite-db/vsphere/sfps-vcsa/hostsystem# ls
sfps-cmp-12  sfps-cmp-13  sfps-cmp-14

```

Automatically Purging Stale Data

To automate the removal of stale data from the Graphite database you can use a cron job to run a cleanup script at a set period. For example:

```

#Add the following cron job (crontab -e)
@daily /root/graphite-whisper-cleanup.sh

root@sfps-grafana01:~# cat graphite-whisper-cleanup.sh
#variable for long pathname
graphitedb=/var/lib/docker/plugins/50cf6ba66948f4c7e329be406d070c25e2bee103b49f36382e768848e807c8a1/rootfs/var/lib/docker-volumes/netapp/graphite-db/

# how much space to reclaim if we delete files not updated in last 30 days?
# find $graphitedb -name "*wsp" -mtime +30 -exec echo -n -e {} "\0" \; | du -hc --files0-from=-

# delete the files!
find $graphitedb -type f -mtime +30 -name "*wsp" -exec rm '{}' \;

# delete empty directories
find $graphitedb -type d -empty -delete

```

Appendix B: SFCollector Statistics

This is a (potentially non-exhaustive) list of statistics used by sfcollector.

API	Stat Name	Description	Calc	Type	Ver
clusterStats	actualIOPS	Current actual IOPS for the entire cluster in the last 500 milliseconds	Point in Time	Integer	9,10
	clientQueueDepth	The number of outstanding read and write operations to the cluster	NA	Integer	9,10
	clusterUtilization	Cluster capacity being utilized	NA	Float	9,10

API	Stat Name	Description	Calc	Type	Ver
	latencyUsec	The average time, in microseconds to complete operations to a cluster in the last 500 milliseconds	Point in Time	Integer	9,10
	normalizedIOPS	Average number of IOPS for the entire cluster in the last 500 milliseconds	Point in Time	Integer	10
	readBytes	The total cumulative bytes read from the cluster since the creation of the cluster	Monotonic	Integer	9,10
	readBytesLastSample	The total number of bytes read from the cluster during the last sample period	Point in Time	Integer	9,10
	readLatencyUsec	The average time, in microseconds, to complete read operations to the cluster in the last 500 milliseconds.	Point in Time	Integer	9,10
	readOps	The total cumulative read operations to the cluster since the creation of the cluster	Monotonic	Integer	9,10
	readOpsLastSample	The total number of read operations during the last sample period.	Point in Time	Integer	9,10
	unalignedReads	The total cumulative unaligned read operations to a cluster since the creation of the cluster.	Monotonic	Integer	9,10
	unalignedWrites	The total cumulative unaligned write operations to a cluster since the creation of the cluster.	Monotonic	Integer	9,10
	writeLatencyUsec	The average time, in microseconds, to complete write operations to the cluster in the last 500 milliseconds.	Point in Time	Integer	9,10
	writeOps	The total cumulative write operations to the cluster since the creation of the cluster	Monotonic	Integer	9,10
	writeOpsLastSample	The total number of write operations during the last sample period.	Point in Time	Integer	9,10

API	Stat Name	Description	Calc	Type	Ver
	writeBytes	The total cumulative bytes write from the cluster since the creation of the cluster	Monotonic	Integer	9,10
	writeBytesLastSample	The total number of bytes write from the cluster during the last sample period	Point in Time	Integer	9,10
clusterCapacity	activeBlockSpace	The amount of space on the block drives. This includes additional information such as metadata entries and space which can be cleaned up.	NA	Integer	9,10
	activeSessions	The number of active iSCSI sessions	NA	Integer	9,10
	averageIOPS	The average IOPS for the cluster since midnight UTC	NA	Integer	9,10
	ClusterRecentIOSize	The average size of IOPS to all volumes in the cluster	NA	Integer	9,10
	currentIOPS	The average IOPS for all volumes in the cluster over the last five seconds	NA	Integer	9,10
	maxIOPS	The estimated maximum IOPS capability of the current cluster	NA	Integer	9,10
	maxOverProvisionableSpace	The maximum amount of provisionable space.	NA	Integer	9,10
	maxProvisionedSpace	The total amount of provisionable space if all volumes are filled to 100%	NA	Integer	9,10
	maxUsedMetadataSpace	The number of bytes on volume drives used to store metadata	NA	Integer	9,10
	maxUsedSpace	The total amount of space on all active block drives	NA	Integer	9,10
	nonZeroBlock	The total number of 4KiB blocks that contain data after the last garbage collection.	NA	Integer	9,10
	peakActiveSessions	The peak number of iSCSI connections since midnight UTC	NA	Integer	9,10
	peakIOPS	The highest value for currentIOPS since midnight UTC	NA	Integer	9,10

API	Stat Name	Description	Calc	Type	Ver
	provisionedSpace	The total amount of space provisioned in all volumes on the cluster	NA	Integer	9,10
	totalOps	The total number of I/O operations performed throughout the lifetime of the cluster	NA	Integer	9,10
	uniqueBlocks	The total number of blocks stored on the block drives, including replicated blocks.	NA	Integer	9,10
	uniqueBlocksUsedSpace	The total amount of data the uniqueBlocks take up on the block drives.	NA	Integer	9,10
	usedMetadataSpace	The total number of bytes on volume drives used to store metadata	NA	Integer	9,10
	usedMetadataSpaceInSnapshots	The number of bytes on volume drives used for storing unique data in snapshots. This provides an estimate of how much metadata space would be regained by deleting all snapshots on the system	NA	Integer	9,10
	usedSpace	The total amount of space used by all block drives in the system	NA	Integer	9,10
	zeroBlocks	The total number of empty 4KiB blocks without data after the last round of garbage collection.	NA	Integer	9,10
nodeStats	cpu	CPU usage in %	NA	Integer	9,10
	cBytesIn	Bytes in on the cluster interface	NA	Integer	9,10
	cBytesOut	Bytes out on the cluster interface	NA	Integer	9,10
	sBytesIn	Bytes in on the storage interface	NA	Integer	9,10
	sBytesOut	Bytes out on the storage interface	NA	Integer	9,10
	mBytesIn	Bytes in on the management interface	NA	Integer	9,10

API	Stat Name	Description	Calc	Type	Ver
	mBytesOut	Bytes out on the management interface	NA	Integer	9,10
	networkUtilizationCluster	Network interface utilization (%) for the cluster network interface	NA	Integer	9,10
	networkUtilizationStorage	Network interface utilization (%) for the storage network interface	NA	Integer	9,10
	readOps	Monotonically increasing value of the total read operations to a node	NA	Integer	10
	usedMemory	Total usage in bytes	NA	Integer	9,10
	writeOps	Monotonically increasing value of the total write operations to a node	NA	Integer	10
volumeStats	accountID	The ID of the account of the volume owner	NA	Integer	9,10
	actualIOPS	The current actual IOPS to the volume in the last 500 milliseconds	Point in Time	Integer	9,10
	averageIOPSize	The average size in bytes of the recent IO to the volume in the last 500 milliseconds	Point in Time	Integer	9,10
	burstIOPSCredit	The total number of IOP credits available.	NA	Integer	9,10
	clientQueueDepth	The number of outstanding read and write operations to the volume	NA	Integer	9,10
	latencyUsec	The average time, in microseconds, to complete operations to the volume in the last 500 milliseconds.	Point in Time	Integer	9,10
	readBytes	The total cumulative bytes read from the volume since the creation of the volume	Monotonic	Integer	9,10
	readBytesLastSample	The total number of bytes read from the volume during the last sample period	Point in Time	Integer	9,10

API	Stat Name	Description	Calc	Type	Ver
	readLatencyUsec	The average time, in microseconds, to complete read operations to the volume in the last 500 milliseconds.	Point in Time	Integer	9,10
	readOps	The total cumulative read operations to the volume since the creation of the volume	Monotonic	Integer	9,10
	readOpsLastSample	The total number of read operations during the last sample period.	Point in Time	Integer	9,10
	throttle	A floating value between 0 and 1 that represents how much the system is throttling clients below their maxIOPS because of replication of data, transient errors and snapshots taken.	NA	Float	9,10
	unalignedReads	The total cumulative unaligned read operations to a volume since the creation of the volume.	Monotonic	Integer	9,10
	unalignedWrites	The total cumulative unaligned write operations to a volume since the creation of the volume.	Monotonic	Integer	9,10
	volumeUtilization	A floating value that describes how much the client is using the volume. 0 = the client is not using the volume; 1 = the client is using their max; >1 = the client is using their burst	NA	Float	9,10
	writeLatencyUsec	The average time, in microseconds, to complete write operations to the volume in the last 500 milliseconds.	Point in Time	Integer	9,10
	writeOps	The total cumulative write operations to the volume since the creation of the volume	Monotonic	Integer	9,10
	writeOpsLastSample	The total number of write operations during the last sample period.	Point in Time	Integer	9,10

API	Stat Name	Description	Calc	Type	Ver
	writeBytes	The total cumulative bytes write from the volume since the creation of the volume	Monotonic	Integer	9,10
	writeBytesLastSample	The total number of bytes write from the volume during the last sample period	Point in Time	Integer	9,10
	zeroBlocks	The total number of empty 4KiB blocks without data after the last round of garbage collection.	NA	Integer	9,10

Where to Find Additional Information

To learn more about the information described in this document, refer to the following documents and/or websites:

- This blog has some excellent troubleshooting steps for a graphite+Grafana configuration.
<http://dieter.plaetinck.be/post/25-graphite-grafana-statsd-gotchass>
- Install Docker
<https://docs.docker.com/engine/installation/linux/ubuntu>
- Trident Quick Start
<https://netapp-trident.readthedocs.io/en/stable-v18.04/docker/index.html>
- Composing Graphite server w/Docker
<https://thepracticalsysadmin.com/composing-a-graphite-server-with-docker/>
- SolidFire Collector for Graphite
<https://github.com/cbiebers/solidfire-graphite-collector>
- #vSphere Graphite collector
<https://github.com/cblomart/vsphere-graphite>
- VMware SDK – PerformanceManager
<http://pubs.vmware.com/vsphere-6-5/topic/com.vmware.wssdk.apiref.doc/vim.PerformanceManager.html>

Version History

Version	Date	Document Version History
Version 1.0	April 2018	Initial release

Refer to the [Interoperability Matrix Tool \(IMT\)](#) on the NetApp Support site to validate that the exact product and feature versions described in this document are supported for your specific environment. The NetApp IMT defines the product components and versions that can be used to construct configurations that are supported by NetApp. Specific results depend on each customer's installation in accordance with published specifications.

Copyright Information

Copyright © 1994–2018 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as

expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

Trademark Information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.